

# **USB Type-C™ Bridge Spec**

## **USB Command Verifier Compliance Test Specification**

**Revision 1.02**

**Date:** July 30, 2019

**Revision:** 1.02

***Intellectual Property Disclaimer***

**Revision History**

<b>Revision</b>	<b>Issue Date</b>	<b>Comments</b>
0.90		Initial release
1.00	Jan 11, 2018	Completed tests
1.02	Jul 30, 2019	Removed consistency checks. Defined PD Support check.

**Significant Contributors:**

---

Diane Lenox  
Soren Petersen

SpecWerkz, LLC  
SpecWerkz, LLC

***Please send comments via electronic mail to: [techadmin@usb.org](mailto:techadmin@usb.org) or [ssusbcompliance@usb.org](mailto:ssusbcompliance@usb.org)***

## Contents

1	Introduction .....	5
2	Assertions for USB Type-C™ Bridge Class 1.0 Specification.....	5
	General Requirements: .....	5
	Assertion Table.....	5
3	Test Descriptions for USB Type-C Bridge 1.0.....	44
	Test Results.....	44
	General Test Initialization .....	44
	General USB Type-C Bridge Test Definitions Used in this Test Specification.....	44
	USB Type-C Interface Validation.....	45
	General PPM Initialization .....	45
	General PPM Notification Validation .....	46
	Ensure Connector has/does not have Connector Partner .....	46
	Determine whether Power Delivery Is Supported .....	47
	TD 3.1 Standard Descriptors Test.....	47
	TD 3.2 Bridge Capability Descriptor Test.....	48
	TD 4.1 PPM RESET Command Test .....	50
	TD 4.2 CANCEL Command Test.....	51
	TD 4.3 ACK_PCR_PAN Command Test.....	53
	TD 4.4 Set Notification Enable Command Test.....	55
	TD 4.5 Get Capability Command Test .....	56
	TD 4.6 Get Connector Capability Command Test.....	58
	TD 4.7 Get Connector Status Command - No Device Attached Test .....	61
	TD 4.8 Get PDOs Command Test.....	63
	TD 4.9 Get Error Status Command Test .....	67
	TD 4.10 Connect/Disconnect Notification Test.....	71
	TD 4.11 Default Notifications Test.....	73
	TD 4.12 PD Hard Reset Notification Test.....	74
	TD 5.1 Type-C Authentication Test .....	76
	TD 5.2 Remote Wake Test.....	78
	Appendix .....	81
	Changes in Revision 0.90.....	81
	Changes in Revision 1.00.....	81
	Changes in Revision 1.01.....	81
	Changes in Revision 1.02.....	81

# 1 Introduction

## 2 Assertions for USB Type-C™ Bridge Class 1.0 Specification.

### ***General Requirements:***

- All Reserved fields shall be set to 0.

### ***Assertion Table***

Assertion Number	Assertion Description	Test #
<b>Subsection reference: 1 Introduction</b>		
<b>Subsection reference: 1.4 Terms and Abbreviations</b>		
1.4#1	A Billboard Only USB Type-C™ Bridge shall have no Connectors.	TD 3.2
<b>Subsection reference: 2 Management Overview</b>		
2#1	A USB Type-C™ Bridge shall only be implemented as an integral part of a PDUSB Hub or as part of a PDUSB Device.	Untestable
<b>Subsection reference: 3 USB Descriptors and Requests</b>		
<b>Subsection reference: 3.1 Descriptors</b>		
3.1#1	A USB Type-C™ Bridge shall return and respond to all standard USB commands as defined in Chapter 9 of the USB 2.0 specification or Chapter 9 of the USB 3.1 specification.	USB Chapter 9 (See [USB2.0] and [USB3.1])
3.1#2	A USB Type-C™ Bridge shall return a Class value of 12h in its Interface descriptor.	TD 3.1
3.1#3	A USB Type-C™ Bridge shall return a Subclass value of 00h in its Interface Descriptor.	TD 3.1
3.1#4	A USB Type-C™ Bridge shall return a Protocol value of 00h in its Interface Descriptor.	TD 3.1
3.1#5	The USB Type-C™ Bridge Function shall be encapsulated in a single interface and shall not be part of any other Function.	TD 3.1 TD 3.2
3.1#6	A USB Type-C™ Bridge interface shall contain one Interrupt IN endpoint.	TD 3.1
3.1#7	The Bridge Capability Descriptor shall immediately follow the Type-C™ Bridge Interface Descriptor in the list of descriptors returned by the USB Type-C Bridge.	TD 3.2
3.1#8	The bLength field of a Bridge Capability Descriptor shall be 07h.	TD 3.2
3.1#9	The bDescriptorType field of a Bridge Capability Descriptor shall be 21h (BRIDGE_CAPABILITY).	TD 3.2

Assertion Number	Assertion Description	Test #
3.1#10	The bcdVersion field of a Bridge Capability Descriptor shall be 0100h.	TD 3.2
3.1#11	The bmConnectorMask field of a Bridge Capability Descriptor shall indicate the connectors a bridge can communicate with, where if a bit is set to one, then the USB Type-C™ Bridge shall be able to communicate with that Connector.	TD 3.2
3.1#12	The bmConnectorMask field of a Bridge Capability Descriptor shall not have more bits set than bNumConnectors field returned from Get Capability PPM Command.	TD 3.2
3.1#13	Bit 0 of the bmConnectorMask field of a Bridge Capability Descriptor is Reserved and shall be set to 0.	TD 3.2
3.1#14	Bit 0 of the bmAttributes field of a Bridge Capability Descriptor shall be set to 1 if the USB Type-C™ Bridge supports PPM functionality.	TD 3.2
3.1#15	Bit 1 of the bmAttributes field of a Bridge Capability Descriptor shall be set to 1 if the USB Type-C™ Bridge supports PAM functionality.	TD 3.2
3.1#16	Bit 2 of the bmAttributes field of a Bridge Capability Descriptor shall be set to 1 if the USB Type-C™ Bridge supports PFUM functionality.	TD 3.2
3.1#17	Bit 3 of the bmAttributes field of a Bridge Capability Descriptor shall be set to 1 if the USB Type-C™ Bridge supports Billboard functionality.	TD 3.2
3.1#18	Bits 4-7 of the bmAttributes field of a Bridge Capability Descriptor are Reserved and shall be set to 0.	TD 3.2
3.1#19	The Notification Endpoint of a USB Type-C™ Bridge shall be an Interrupt Endpoint (bmAttributes 3h).	TD 3.1
3.1#20	The Notification Endpoint of a USB Type-C™ Bridge shall have a wMaxPacketSize of 40h.	TD 3.1
3.1#21	The Notification Endpoint of a USB Type-C™ Bridge shall have a bInterval of 04h.	TD 3.1
3.1#22	The bNotificationType field of the data structure that a USB Type-C™ Bridge shall return in a Notification Endpoint shall identify the notification type. Valid values for the notification type are in the range 1h through 3h.	Implicit in PPM Command and Asynchronous Notification tests.
3.1#23	The data structure that a USB Type-C™ Bridge shall return in a Notification Endpoint shall be at least 4 bytes.	Implicit in PPM Command and Asynchronous Notification tests.
<b>Subsection reference: 3.2 USB Class-Specific Requests</b>		
<b>Subsection reference: 3.2.1 PPM Functionality</b>		
3.2.1#1	A USB Type-C™ Bridge that is not a Billboard Only USB Type-C Bridge shall support PPM functionality.	TD 3.2

Assertion Number	Assertion Description	Test #
3.2.1#2	A Billboard Only USB Type-C™ Bridge shall not support PPM functionality.	TD 3.2
3.2.1#3	A USB Type-C™ Bridge shall only use the PPM Commands and PPM Command Responses/PPM Asynchronous Notifications as defined in Section 4 of the USB Type-C Bridge specification.	Not tested
<b>Subsection reference: 3.2.1.1 Send PPM Command Request</b>		
3.2.1.1#1	The bRequest field of a Send PPM Command request shall be set to 01h (SEND_PPM_COMMAND).	N/A (Host)
3.2.1.1#2	The wValue field of a SEND_PPM_COMMAND request shall be set to 00h.	N/A (Host)
3.2.1.1#3	The lower byte of the wIndex field of a SEND_PPM_COMMAND request shall be set to the interface to which the Request is to be sent.	N/A (Host)
3.2.1.1#4	The upper byte of the wIndex field of a SEND_PPM_COMMAND request shall be set to 00h.	N/A (Host)
3.2.1.1#5	The USB Type-C™ Bridge Class Driver shall set the wLength field to the number of bytes in the Data field.	N/A (Host)
<b>Subsection reference: 3.2.1.2 USB Type-C Bridge PPM Notification</b>		
3.2.1.2#1	The bNotificationType field of the USB Type-C™ Bridge PPM Notification sent back to the USB Type-C Bridge Class Driver shall be set to 01h (PPM_NOTIFICATION).	See individual USB Request or PPM Command Tests.
3.2.1.2#2	The bParam1 field of the USB Type-C™ Bridge PPM Notification sent back to the USB Type-C Bridge Class Driver shall be set to 00h.	See individual USB Request or PPM Command Tests.
3.2.1.2#3	The bParam2 field of the USB Type-C™ Bridge PPM Notification sent back to the USB Type-C Bridge Class Driver shall be set to 00h.	See individual USB Request or PPM Command Tests.
3.2.1.2#4	The bParam3 field of the USB Type-C™ Bridge PPM Notification sent back to the USB Type-C Bridge Class Driver shall be set to 00h.	See individual USB Request or PPM Command Tests.
3.2.1.2#5	The Payload fields of the USB Type-C™ Bridge PPM Notification sent back to the USB Type-C Bridge Class Driver shall be Set to the PPM Command Response or PPM Asynchronous Notification. If there is any additional data associated with the PPM Command Response, that data shall be appended to the end of the PPM Command Response.	See individual USB Request or PPM Command Tests.
<b>Subsection reference: 3.2.2 PAM Functionality</b>		
3.2.2#1	Authentication Requests and Responses shall be as defined in the USB Type-C™ Authentication Specification Revision 1.0.	See individual USB Request or PAM Command Tests
3.2.2#2	A Billboard Only USB Type-C™ Bridge shall not support PAM functionality.	TD 3.2
<b>Subsection reference: 3.2.2.1 Send Authentication Data Request</b>		

Assertion Number	Assertion Description	Test #
3.2.2.1#1	The USB Type-C™ Bridge Class driver shall encapsulate either an Authentication Request or Authentication Response in the Data field of a SEND_PD_DATA request.	N/A (Host)
3.2.2.1#2	The bRequest field of an Authentication Request or Authentication Response request shall be set to 02h (SEND_PD_DATA).	N/A (Host)
3.2.2.1#3	The low order byte of the wValue field of a SEND_PD_DATA request shall be set to 01h (AUTH).	N/A (Host)
3.2.2.1#4	If the Target of a SEND_PD_DATA request is SOP, then the high order byte of the wValue field shall be set to 0.	N/A (Host)
3.2.2.1#5	If the Target of a SEND_PD_DATA request is SOP', then the high order byte of the wValue field shall be set to 1.	N/A (Host)
3.2.2.1#6	If the Target of a SEND_PD_DATA request is SOP'', then the high order byte of the wValue field shall be set to 2.	N/A (Host)
3.2.2.1#7	The low order byte of the wIndex field shall be set to the Interface to which the Request is to be sent.	N/A (Host)
3.2.2.1#8	The high order byte of the wIndex field of a SEND_PD_DATA request shall be set to the Connector Number of the Connector being addressed.	N/A (Host)
3.2.2.1#9	The USB Type-C™ Bridge Class Driver shall set the wLength field of a SET_PD_DATA request to the number of bytes in the Data field.	N/A (Host)
<b>Subsection reference: 3.2.2.2 Get Authentication Data Request</b>		
3.2.2.2#1	The USB Type-C™ Bridge shall send any received Authentication Request or Authentication Response in the Data that is sent in response to the GET_PD_DATA request.	TD 5.1
3.2.2.2#2	The bRequest field of an Authentication Request or Authentication Response request shall be set to 03h (GET_PD_DATA).	N/A (Host)
3.2.2.2#3	The low order byte of the wValue field of a GET_PD_DATA request shall be set to 01h (AUTH).	N/A (Host)
3.2.2.2#4	If the Target of a GET_PD_DATA request is SOP, then the high order byte of the wValue field shall be set to 0.	N/A (Host)
3.2.2.2#5	If the Target of a GET_PD_DATA request is SOP', then the high order byte of the wValue field shall be set to 1.	N/A (Host)
3.2.2.2#6	If the Target of a GET_PD_DATA request is SOP'', then the high order byte of the wValue field shall be set to 2.	N/A (Host)
3.2.2.2#7	The low order byte of the wIndex field shall be set to the Interface to which the Request is to be sent.	N/A (Host)
3.2.2.2#8	The high order byte of the wIndex field of a GET_PD_DATA request shall be set to the Connector Number of the Connector being addressed.	N/A (Host)



Assertion Number	Assertion Description	Test #
3.2.2.2#9	The USB Type-C™ Bridge Class Driver shall set the wLength field of a GET_PD_DATA request to the maximum number of bytes the USB Type-C Bridge may return in response to this Request.	N/A (Host)
<b>Subsection reference: 3.2.2.3 Authentication Notification</b>		
3.2.2.3#1	The Authentication Notification shall be sent by the USB Type-C™ Bridge to indicate to the OSAM that it has either received Authentication-related data from a PD Product attached to a Connector on the Device Container or failed to transmit an Authentication Request or Response to a PD Product attached to a Connector on the Device Container.	TD 5.1
3.2.2.3#2	The bNotificationType field of the Authentication Notification sent by the USB Type-C™ Bridge shall be set to 02h (PD_NOTIFICATION).	TD 5.1
3.2.2.3#3	The bParam1 field of the Authentication Notification sent by the USB Type-C™ Bridge shall be set to the Connector Number of the Connector being authenticated.	TD 5.1
3.2.2.3#4	The bParam2 field of the Authentication Notification sent by the USB Type-C™ Bridge shall be set to 01h (AUTH).	TD 5.1
3.2.2.3#5	The bParam3 field of the Authentication Notification sent by the USB Type-C™ Bridge shall be set to 00h (Success) if the Request or Response was transmitted successfully and there is associated data for the OSAM to retrieve.	TD 5.1
3.2.2.3#6	The bParam3 field of the Authentication Notification sent by the USB Type-C™ Bridge shall be set to 80h (Error) if the USB Type-C Bridge encountered an error in transmitting a Request or Response and there is no data for the OSAM to retrieve.	Not tested
3.2.2.3#7	The bParam3 field of the Authentication Notification sent by the USB Type-C™ Bridge shall be set to 01h (Not Supported) if the Request or Response was transmitted successfully and a Not_Supported PD Message was received in response.	TD 5.1
3.2.2.3#8	There shall be no Payload in the Authentication Notification sent back to the USB Type-C™ Bridge Class Driver.	5.1
<b>Subsection reference: 3.2.3 PFUM Functionality</b>		
3.2.3#1	Firmware Requests and Responses shall be as defined in the USB Power Delivery Firmware Update Specification.	Not tested
3.2.3#2	A Billboard Only USB Type-C™ Bridge shall not support PFUM functionality.	TD 3.1
<b>Subsection reference: 3.2.3.1 Send Firmware Update Data Request</b>		
3.2.3.1#1	The Send Firmware Update Data Request shall be used by the USB Type-C™ Bridge to either transmit a FW Update Request to a PD Product attached to a Connector on a Device Container or to notify the OSFUM that a Firmware Update Request could not be transmitted.	N/A (Host)

Assertion Number	Assertion Description	Test #
3.2.3.1#2	The bRequest field of a Firmware Update Request shall be set to 02h (SEND_PD_DATA).	N/A (Host)
3.2.3.1#3	The low order byte of the wValue field of a SEND_PD_DATA request shall be set to 02h (FWU).	N/A (Host)
3.2.3.1#4	If the Target of a SEND_PD_DATA request is SOP, then the high order byte of the wValue field shall be set to 0.	N/A (Host)
3.2.3.1#5	If the Target of a SEND_PD_DATA request is SOP', then the high order byte of the wValue field shall be set to 1.	N/A (Host)
3.2.3.1#6	If the Target of a SEND_PD_DATA request is SOP'', then the high order byte of the wValue field shall be set to 2.	N/A (Host)
3.2.3.1#7	The low order byte of the wIndex field shall be set to the Interface to which the Request is to be sent.	N/A (Host)
3.2.3.1#8	The high order byte of the wIndex field of a SEND_PD_DATA request shall be set to the Connector Number of the Connector being addressed.	N/A (Host)
3.2.3.1#9	The USB Type-C™ Bridge Class Driver shall set the wLength field of a SET_PD_DATA request to the number of bytes in the Data field.	N/A (Host)
<b>Subsection reference: 3.2.3.2 Firmware Update Notification</b>		
3.2.3.2#1	The Firmware Update Notification shall be used by the USB Type-C™ Bridge to transmit a Firmware Update Response to the OSFUM from a PD Product attached to a Connector on a Device Container.	Not Tested
3.2.3.2#2	The bNotificationType field of the Firmware Update Notification sent back to the USB Type-C™ Bridge Class Driver shall be set to 02h (PD_NOTIFICATION).	Not Tested
3.2.3.2#3	The bParam1 field of the Firmware Update Notification sent back to the USB Type-C™ Bridge Class Driver shall be set to the Connector Number of the Connector to which the PD Product whose firmware is being updated is attached.	Not Tested
3.2.3.2#4	The bParam2 field of the Firmware Update Notification sent back to the USB Type-C™ Bridge Class Driver shall be set to 02h (FWU).	Not Tested
3.2.3.2#5	The bParam3 field of the Firmware Update Notification shall be set to 00h (Success) if the Firmware Update Request was transmitted successfully and the associated Firmware Update Response is contained in the Payload of the Notification.	Not Tested
3.2.3.2#6	The bParam3 field of the Firmware Update Notification shall be set to 80h (Error) if the USB Type-C™ Bridge encountered an error in transmitting the Firmware Update Request.	Not Tested
3.2.3.2#7	The bParam3 field of the Firmware Update Notification shall be set to 01h (Not_Supported) if the Firmware Update Request was transmitted successfully and a Not_Supported PD Message was received in response.	Not Tested

Assertion Number	Assertion Description	Test #
3.2.3.2#8	The Payload field of the Firmware Update Notification shall contain the Firmware Update Response. (See the USB Power Delivery Firmware Update Specification.)	Not Tested
<b>Subsection reference: 3.2.4 Billboard Functionality</b>		
3.2.4#1	The USB Type-C™ Bridge shall optionally support Billboard functionality.	TD 3.2
<b>Subsection reference: 3.2.4.1 Enable/Disable Billboard Notification Request</b>		
3.2.4.1#1	The Enable/Disable Billboard Notification Request shall be used by the USB Type-C™ Bridge Class Driver to enable or disable Billboard Notifications from the USB Type-C Bridge.	
3.2.4.1#2	The bRequest field of an Enable/Disable Billboard Request shall be set to 04h (ENABLE_BB_NOTIFICATION).	
3.2.4.1#3	The wValue field of an ENABLE_BB_NOTIFICATION request shall be set to Enable (1) when it wants to enable Billboard Notifications from the USB Type-C™ Bridge.	
3.2.4.1#4	The wValue field of an ENABLE_BB_NOTIFICATION request shall be set to Disable (0) when it wants to disable Billboard Notifications from the USB Type-C™ Bridge.	
3.2.4.1#5	The USB Type-C™ Bridge Class Driver shall set the wIndex field of an ENABLE_BB_NOTIFICATION request to the Interface to which the Request is to be sent.	
3.2.4.1#6	The USB Type-C™ Bridge Class Driver shall set the wLength field of an ENABLE_BB_NOTIFICATION request to 00h.	
<b>Subsection reference: 3.2.4.2 Billboard Notification</b>		
3.2.4.2#1	The Billboard Notification shall be sent by the USB Type-C™ Bridge to indicate that there is a change to the Billboard Capability in the Device Container.	
3.2.4.2#2	The bNotificationType field of the Firmware Update Notification sent back to the USB Type-C™ Bridge Class Driver shall be set to 03h (BILLBOARD_NOTIFICATION).	
3.2.4.2#3	The bParam1 field of Billboard Notification sent back to the USB Type-C™ Bridge Class Driver shall be set to 00h.	
3.2.4.2#4	The bParam2 field of Billboard Notification sent back to the USB Type-C™ Bridge Class Driver shall be set to 00h.	
3.2.4.2#5	The bParam3 field of the Billboard Notification sent back to the USB Type-C™ Bridge Class Driver shall be set to 00h.	
3.2.4.2#6	There shall be no Payload in the Billboard Notification sent back to the USB Type-C™ Bridge Class Driver.	
<b>Subsection reference: 4 PPM Commands, PPM Command Responses and PPM Asynchronous Notifications</b>		
4#1	An OSPM shall only use data structures and PPM Commands that are defined in Chapter 4 of the Universal Serial Bus Type-C™ Bridge Specification.	N/A (Host)

Assertion Number	Assertion Description	Test #
<b>Subsection reference: 4.1 Data Structures</b>		
<b>Subsection reference: 4.1.1 COMMAND</b>		
4.1.1#1	The COMMAND Data Structure is sent as the Data of a SEND_PPM_COMMAND Request, and defines the PPM Command that shall be executed by the PPM in the USB Type-C™ Bridge.	N/A (Host)
4.1.1#2	The Command field of the COMMAND data structure shall be set to the command that the OSPM wishes the PPM to execute.	N/A (Host)
4.1.1#3	The Reserved field of the COMMAND data structure is reserved and shall be set to 00h.	N/A (Host)
<b>Subsection reference: 4.1.2 PPM Command Response or PPM Asynchronous Notification</b>		
4.1.2#1	When the PPM completes a PPM Command or when it detects an Asynchronous Event on one of its Connectors it shall send a PPM Command Response or a PPM Asynchronous Notification respectively to the OSPM.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
4.1.2#2	All Reserved fields of the PPM Command Response or PPM Asynchronous Notification Data Structure are reserved and shall be set to 0.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
4.1.2#3	The Connect Change Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be set to the Connector Number that a change occurred on.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
4.1.2#4	The Connect Change Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be less than or equal to the maximum number of connectors supported on the platform.	TD 4.4 TD 4.10 TD 4.12

Assertion Number	Assertion Description	Test #
4.1.2#5	If no changes occurred on any Connector, the Connect Change Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be set to 0.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
4.1.2#6	The Data Length field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be less than or equal to 38h (MAX_DATA_LENGTH).	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
4.1.2#7	The Data Length field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be set to 0 for an Asynchronous PPM Notification.	TD 4.4 TD 4.10 TD 4.12
4.1.2#8	The Not Supported Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be set to 1 by the PPM if it does not currently support a PPM command.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
4.1.2#9	The PPM shall set the Cancel Completed Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure to 1 after it has completed a CANCEL command.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9

Assertion Number	Assertion Description	Test #
4.1.2#10	The PPM shall set the Reset Completed Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure to 1 after it has completed a PPM_RESET command.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
4.1.2#11	If the Reset Completed Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure is set to 1, then all other fields in the Structure shall be set to 0.	TD 4.1
4.1.2#12	The PPM shall clear the Reset Completed Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure upon receiving the next PPM command, unless that command is PPM_RESET.	TD 4.1
4.1.2#13	The Busy Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be ignored.	Not tested
4.1.2#14	The Acknowledge Command Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be to one when it completes the ACK_PCR_PAN (Acknowledge PPM Command Response and/or PPM Asynchronous Notification) Command.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
4.1.2#15	The PPM shall automatically reset the Acknowledge Command Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure when it receives the next PPM Command from the OSPM.	TD 4.3
4.1.2#16	If the Acknowledge Command Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure is set to 1, then all other fields other than the Connector Change Indicator shall be set to 0.	TD 4.3
4.1.2#17	PPM shall set the Error Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure to 1 if it encounters an error when executing the PPM Command sent to it by the OSPM. The Error Indicator field shall only be valid when the Command Completed Indicator field is set to one.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9

Assertion Number	Assertion Description	Test #
4.1.2#18	The Command Completed Indicator field of the PPM Command Response or PPM Asynchronous Notification Data Structure shall be set to 1 upon completion of a PPM Command sent to it by the OSPM.	TD 4.1 TD 4.2 TD 4.3 TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
<b>Subsection reference: 4.2 Commands</b>		
<b>Subsection reference: 4.2.1 Command Codes</b>		
4.2.1#1	The Command field of the COMMAND data structure shall only be set to one of the following values: 01h (PPM_RESET), 02h (CANCEL), 03h (CONNECTOR_RESET), 04h (ACK_PCR_PAN), 05h (SET_NOTIFICATION_ENABLE), 06h (GET_CAPABILITY), 07h (GET_CONNECTOR_CAPABILITY), 10h (GET_PDOS), 12h (GET_CONNECTOR_STATUS), 13h (GET_ERROR_STATUS).	N/A (Host)
<b>Subsection reference: 4.2.2 PPM RESET</b>		
4.2.2#1	The PPM_RESET Command shall be used to reset the PPM.	Not tested.
4.2.2#2	The Command field of a PPM RESET Command shall be set to 01h (PPM_RESET).	N/A (Host)
4.2.2#3	The Data Length field of a PPM_RESET Command shall be set to 00h.	N/A (Host)
4.2.2#4	The Reserved field of a PPM_RESET Command is reserved and shall be set to 0.	N/A (Host)
4.2.2#5	On successful completion of the PPM_RESET command the PPM shall return a PPM command Response.	TD 4.1
4.2.2#6	The reserved fields of a PPM_RESET Response are reserved and shall be set to 0.	TD 4.1
4.2.2#7	The Connector Change Indicator field of a PPM_RESET Response shall be set to 00h.	TD 4.1
4.2.2#8	The Data Length field of a PPM_RESET Response shall be set to 00h.	TD 4.1
4.2.2#9	The Not Supported Indicator field of a PPM_RESET Response shall be set to 0.	TD 4.1
4.2.2#10	The Cancel Completed Indicator field of a PPM_RESET Response shall be set to 0.	TD 4.1
4.2.2#11	The Reset Completed Indicator field of a PPM_RESET Response shall be set to 1.	TD 4.1
4.2.2#12	The Reset Completed Indicator field of a PPM_RESET Response shall be cleared on reception of the next PPM command from the OSPM.	TD 4.1

Assertion Number	Assertion Description	Test #
4.2.2#13	The Busy Indicator field of a PPM_RESET Response shall be set to 0.	TD 4.1
4.2.2#14	The Acknowledge Command Indicator field of a PPM_RESET Response shall be set to 0.	TD 4.1
4.2.2#15	The Error Indicator field of a PPM_RESET Response shall be set to 0.	TD 4.1
4.2.2#16	The Command Completed Indicator field of a PPM_RESET Response shall be set to 0.	TD 4.1
<b>Subsection reference: 4.2.3 PPM CANCEL</b>		
4.2.3#1	The CANCEL Command shall be used to cancel a PPM Command previously sent to the PPM.	TD 4.2
4.2.3#2	The Command field of a CANCEL Command shall be set to 02h (CANCEL).	N/A (Host)
4.2.3#3	The Data Length field of a CANCEL Command shall be set to 00h.	N/A (Host)
4.2.3#4	The Reserved field of a CANCEL Command is reserved and shall be set to 0.	N/A (Host)
4.2.3#5	On successful completion of the CANCEL Command the PPM shall return a PPM Command Response.	TD 4.2
4.2.3#6	The reserved fields of a CANCEL Response are reserved and shall be set to 0.	TD 4.2
4.2.3#7	If an Asynchronous Event occurred on a Connector, then the Connector Change Indicator field of a CANCEL Response shall be set to the Connector Number on which the change occurred.	TD 4.2
4.2.3#8	The Data Length field of a CANCEL Response shall be set to 00h.	TD 4.2
4.2.3#9	The Not Supported Indicator field of a CANCEL Response shall be set to 0.	TD 4.2
4.2.3#10	The Cancel Completed Indicator field of a CANCEL Response shall be set to 1.	TD 4.2
4.2.3#11	The Reset Completed Indicator field of a CANCEL Response shall be set to 0.	TD 4.2
4.2.3#12	The Busy Indicator field of a CANCEL Response shall be set to 0.	TD 4.2
4.2.3#13	The Acknowledge Command Indicator field of a CANCEL Response shall be set to 0.	TD 4.2
4.2.3#14	The Error Indicator field of a CANCEL Response shall be set to 0.	TD 4.2
4.2.3#15	The Command Completed Indicator field of a CANCEL Response shall be set to 1.	TD 4.2
<b>Subsection reference: 4.2.4 Connector Reset</b>		
4.2.4#1	This CONNECTOR_RESET Command is reserved in this version of the specification.	Not tested.



Assertion Number	Assertion Description	Test #
<b>Subsection reference: 4.2.5 Acknowledge PPM Command Response and/or PPM Asynchronous Notification</b>		
4.2.5#1	The ACK_PCR_PAN Command shall be used to acknowledge to the PPM that the OSPM received and processed a PPM Command Response or PPM Asynchronous Notification.	N/A (Host)
4.2.5#2	The Command field of an ACK_PCR_PAN Command shall be set to 04h (ACK_PCR_PAN).	N/A (Host)
4.2.5#3	The Data Length field of an ACK_PCR_PAN Command shall be set to 00h.	N/A (Host)
4.2.5#4	The Connector Change Acknowledge field of an ACK_PCR_PAN Command shall be set to 1 to acknowledge a connector change that occurred on the Connector indicated by the PPM in the CCI Data Structure.	N/A (Host)
4.2.5#5	The Command Completed Acknowledge field of an ACK_PCR_PAN Command shall be set to 1 to acknowledge that a PPM Command completed.	N/A (Host)
4.2.5#6	The Reserved field of an ACK_PCR_PAN Command is reserved and shall be set to 0.	N/A (Host)
4.2.5#7	On successful completion of the ACK_PCR_PAN command the PPM shall return a PPM Command Response	TD 4.3
4.2.5#8	The reserved fields of an ACK_PCR_PAN Response are reserved and shall be set to 0.	TD 4.3
4.2.5#9	The Connector Change Indicator field of an ACK_PCR_PAN Response shall be set to 00h.	TD 4.3
4.2.5#10	The Data Length field of an ACK_PCR_PAN Response shall be set to 00h.	TD 4.3
4.2.5#11	The Not Supported Indicator field of an ACK_PCR_PAN Response shall be set to 0.	TD 4.3
4.2.5#12	The Cancel Completed Indicator field of an ACK_PCR_PAN Response shall be set to 0.	TD 4.3
4.2.5#13	The Reset Completed Indicator field of an ACK_PCR_PAN Response shall be set to 0.	TD 4.3
4.2.5#14	The Busy Indicator field of an ACK_PCR_PAN Response shall be set to 0.	TD 4.3
4.2.5#15	The Acknowledge Command Indicator field of an ACK_PCR_PAN Response shall be set to 1.	TD 4.3
4.2.5#16	The Error Indicator field of an ACK_PCR_PAN Response shall be set to 0.	TD 4.3
4.2.5#17	The Command Completed Indicator field of an ACK_PCR_PAN Response shall be set to 0.	TD 4.3
<b>Subsection reference: 4.2.6 Set Notification Enable</b>		
4.2.6#1	The SET_NOTIFICATION_ENABLE Command shall be used to set the list of Events that the PPM may send PPM Notifications about to the OSPM.	N/A (Host)

Assertion Number	Assertion Description	Test #
4.2.6#2	The values in the Notification Enable field of SET_NOTIFICATION_ENABLE command shall overwrite any notifications enabled/disabled by a prior SET_NOTIFICATION_ENABLE Command.	TD 4.4
4.2.6#3	If the OSPM enables any PPM Asynchronous Notification, it shall also enable the Command Completed Notification.	TD 4.4
4.2.6#4	The Command field of a SET_NOTIFICATION_ENABLE Command shall be set to 05h (SET_NOTIFICATION_ENABLE).	N/A (Host)
4.2.6#5	The Data Length field of a SET_NOTIFICATION_ENABLE Command shall be set to 00h.	N/A (Host)
4.2.6#6	Bits 1, 3, 4, 8-10, and 13 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command are reserved and shall be set to 0 by the OSPM.	N/A (Host)
4.2.6#7	Command Completion notifications are enabled when bit 0 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	Inherent in PPM Command tests
4.2.6#8	Power Operation Mode Change notifications are enabled when bit 2 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	
4.2.6#9	Supported Provider Capabilities Change notifications are enabled when bit 5 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	
4.2.6#10	Negotiated Power Level Change notifications are enabled when bit 6 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	
4.2.6#11	PD Reset Complete notifications are enabled when bit 7 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	TD 4.12
4.2.6#12	Connector Partner Change notifications are enabled when bit 11 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	
4.2.6#13	Power Direction Change notifications are enabled when bit 12 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	TD 4.11
4.2.6#14	Connect Change notifications are enabled when bit 14 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	TD 4.10
4.2.6#15	Error notifications are enabled when bit 15 of the Notification Enable field of a SET_NOTIFICATION_ENABLE Command is set to 1.	
4.2.6#16	The Reserved field of a SET_NOTIFICATION_ENABLE Command is reserved and shall be set to 0.	N/A (Host)
4.2.6#17	On successful completion of the SET_NOTIFICATION_ENABLE Command the PPM shall return a PPM Command Response.	TD 4.4

Assertion Number	Assertion Description	Test #
4.2.6#18	The reserved fields of a SET_NOTIFICATION_ENABLE Response are reserved and shall be set to 0.	TD 4.4
4.2.6#19	If an Asynchronous Event occurred on a Connector, then the Connector Change Indicator field of a SET_NOTIFICATION_ENABLE Response shall be set to the Connector Number on which the change occurred.	TD 4.4
4.2.6#20	The Data Length field of a SET_NOTIFICATION_ENABLE Response shall be set to 00h.	TD 4.4
4.2.6#21	The Not Supported Indicator field of a SET_NOTIFICATION_ENABLE Response shall be set to 0.	TD 4.4
4.2.6#22	The Cancel Completed Indicator field of a SET_NOTIFICATION_ENABLE Response shall be set to 0.	TD 4.4
4.2.6#23	The Reset Completed Indicator field of a SET_NOTIFICATION_ENABLE Response shall be set to 0.	TD 4.4
4.2.6#24	The Busy Indicator field of a SET_NOTIFICATION_ENABLE Response shall be set to 0.	TD 4.4
4.2.6#25	The Acknowledge Command Indicator field of a SET_NOTIFICATION_ENABLE Response shall be set to 0.	TD 4.4
4.2.6#26	If the PPM Command was not successfully completed the PPM shall set the Error Indicator field of the SET_NOTIFICATION_ENABLE Response to 1.	TD 4.4
4.2.6#27	The Command Completed Indicator field of a SET_NOTIFICATION_ENABLE Response shall be set to 1.	TD 4.4
<b>Subsection reference: 4.2.7 Get Capability</b>		
4.2.7#1	The GET_CAPABILITY Command shall be used to get the PPM capabilities.	N/A (Host)
4.2.7#2	The Command field of a GET_CAPABILITY Command shall be set to 06h (GET_CAPABILITY).	N/A (Host)
4.2.7#3	The Data Length field of a GET_CAPABILITY Command shall be set to 00h.	N/A (Host)
4.2.7#4	The Reserved field of a GET_CAPABILITY Command is reserved and shall be set to 0.	N/A (Host)
4.2.7#5	On successful completion of the GET_CAPABILITY Command the PPM shall return a PPM Command Response.	TD 4.5
4.2.7#6	The reserved fields of a GET_CAPABILITY Response are reserved and shall be set to 0.	TD 4.5
4.2.7#7	If an Asynchronous Event occurred on a Connector, then the Connector Change Indicator field of a GET_CAPABILITY Response shall be set to the Connector Number on which the change occurred.	TD 4.5
4.2.7#8	If the GET_CAPABILITY command is successful, then the Data Length field of a GET_CAPABILITY Command shall be set to 10h.	TD 4.5

Assertion Number	Assertion Description	Test #
4.2.7#9	If the GET_CAPABILITY command is not successful, then the Data Length field of a GET_CAPABILITY Command shall be set to 00h.	TD 4.5
4.2.7#10	The Not Supported Indicator field of a GET_CAPABILITY Response shall be set to 0.	TD 4.5
4.2.7#11	The Cancel Completed Indicator field of a GET_CAPABILITY Response shall be set to 0.	TD 4.5
4.2.7#12	The Reset Completed Indicator field of a GET_CAPABILITY Response shall be set to 0.	TD 4.5
4.2.7#13	The Busy Indicator field of a GET_CAPABILITY Response shall be set to 0.	TD 4.5
4.2.7#14	The Acknowledge Command Indicator field of a GET_CAPABILITY Response shall be set to 0.	TD 4.5
4.2.7#15	If the PPM Command was not successfully completed the PPM shall set the Error Indicator field of the GET_CAPABILITY Response to 1.	TD 4.5
4.2.7#16	The Command Completed Indicator field of a GET_CAPABILITY Response shall be set to 1.	TD 4.5
4.2.7#17	If the GET_CAPABILITY Command completed successfully then the PPM shall return additional GET_CAPABILITY Data following the GET_CAPABILITY Response.	TD 4.5
4.2.7#18	The Reserved fields of the GET_CAPABILITY Data structure are reserved and shall be set to 0.	TD 4.5
4.2.7#19	The bmAttributes field of the GET_CAPABILITY Data structure contains a bitmap encoding of supported PPM features.	TD 4.5
4.2.7#20	Bits 3-7, 9, 11-13, and 16-31 of the bmAttributes field of the GET_CAPABILITY Data structure are reserved and shall be set to 0.	TD 4.5
4.2.7#21	Bit 0 (Disabled State Support) of the bmAttributes field of the GET_CAPABILITY Data structure shall be set to 1 to indicate this platform supports the Disabled State as defined in Section 4.5.2.2.1 in the USB Type-C™ Cable and Connector Specification Revision 1.2.	TD 4.5
4.2.7#22	Bit 1 (Battery Charging) of the bmAttributes field of the GET_CAPABILITY Data structure shall be set to 1 to indicate this platform supports the Battery Charging Specification as per the value reported in the bcdBCVersion field of this GET_CAPABILITY Data structure.	TD 4.5
4.2.7#23	Bit 2 (USB Power Delivery) of the bmAttributes field of the GET_CAPABILITY Data structure shall be set to 1 to indicate this platform the USB Power Delivery Specification as per the value reported in the bcdPDVersion field of this GET_CAPABILITY Data structure.	TD 4.5

Assertion Number	Assertion Description	Test #
4.2.7#24	If this platform supports AC Supply as a power source, then bit 8 (AC Supply) of the bmAttributes field of the GET_CAPABILITY Data structure shall be set to 1.	TD 4.5
4.2.7#25	If this platform Uses VBUS as a power source, then bit 14 (Uses VBUS) of the bmAttributes field of the GET_CAPABILITY Data structure shall be set to 1.	TD 4.5
4.2.7#26	If this platform is a Charge-through Connector, then bit 15 (Charge-through Connector) of the bmAttributes field of the GET_CAPABILITY Data structure shall be set to 1.	TD 4.5
4.2.7#27	At least one of bits 8 (AC Supply), 10 (Other) and 14 (Uses Vbus) of the bmAttributes field of the GET_CAPABILITY Data structure shall be set 1.	TD 4.5
4.2.7#28	The bNumConnectors field of the GET_CAPABILITY Data structure shall not be set to 0.	TD 4.5
4.2.7#29	The bmOptionalFeatures field of the GET_CAPABILITY Data structure shall be set to 0.	TD 4.5
4.2.7#30	The bNumAltModes field of the GET_CAPABILITY Data structure shall be set to 0.	TD 4.5
4.2.7#31	The bcdBCVersion field of the GET_CAPABILITY Data structure shall be set to the Battery Charging Specification Release Number in Binary-Coded decimal.	TD 4.5
4.2.7#32	If the device indicates that it does not support battery charging in the bmAttributes field of this GET_CAPABILITY Data structure, the bcdBCVersion field of the GET_CAPABILITY Data structure shall be set to 0.	TD 4.5
4.2.7#33	The bcdPDVersion field of the GET_CAPABILITY Data structure shall be set to the Power Delivery Specification Release Number in Binary-Coded decimal.	TD 4.5
4.2.7#34	If the device indicates that it does not support power delivery in the bmAttributes field of this GET_CAPABILITY Data structure, the bcdPDVersion field of the GET_CAPABILITY Data structure shall be set to 0.	TD 4.5
4.2.7#35	The bcdUSBTypeCVersion field of the GET_CAPABILITY Data structure shall be set to the USB Type-C™ Specification Revision Number in Binary-Coded Decimal.	TD 4.5
<b>Subsection reference: 4.2.8 Get Connector Capability</b>		
4.2.8#1	The GET_CONNECTOR_CAPABILITY Command shall be used to get the capabilities of a Connector.	N/A (Host)
4.2.8#2	The Command field of a GET_CONNECTOR_CAPABILITY Command shall be set to 07h (GET_CONNECTOR_CAPABILITY).	N/A (Host)
4.2.8#3	The Connector Number field of a GET_CONNECTOR_CAPABILITY Command shall not be set to 0.	N/A (Host)

Assertion Number	Assertion Description	Test #
4.2.8#4	The Data Length field of a GET_CONNECTOR_CAPABILITY Command shall be set to 00h.	N/A (Host)
4.2.8#5	The Connector Number field of a GET_CONNECTOR_CAPABILITY Command shall be set to the Connector whose capabilities are to be retrieved.	N/A (Host)
4.2.8#6	The Reserved field of a GET_CONNECTOR_CAPABILITY Command is reserved and shall be set to 0.	N/A (Host)
4.2.8#7	On successful completion of the GET_CONNECTOR_CAPABILITY Command the PPM shall return a PPM Command Response.	TD 4.6
4.2.8#8	The reserved fields of a GET_CONNECTOR_CAPABILITY Response are reserved and shall be set to 0.	TD 4.6
4.2.8#9	If an Asynchronous Event occurred on a Connector, then the Connector Change Indicator field of a GET_CONNECTOR_CAPABILITY Response shall be set to the Connector Number on which the change occurred.	TD 4.6
4.2.8#10	If the GET_CONNECTOR_CAPABILITY command is successful, then the Data Length field of a GET_CONNECTOR_CAPABILITY Command shall be set to 02h.	TD 4.6
4.2.8#11	If the GET_CONNECTOR_CAPABILITY command is not successful, then the Data Length field of a GET_CONNECTOR_CAPABILITY Command shall be set to 00h.	TD 4.6
4.2.8#12	The Not Supported Indicator field of a GET_CONNECTOR_CAPABILITY Response shall be set to 0.	TD 4.6
4.2.8#13	The Cancel Completed Indicator field of a GET_CONNECTOR_CAPABILITY Response shall be set to 0.	TD 4.6
4.2.8#14	The Reset Completed Indicator field of a GET_CONNECTOR_CAPABILITY Response shall be set to 0.	TD 4.6
4.2.8#15	The Busy Indicator field of a GET_CONNECTOR_CAPABILITY Response shall be set to 0.	TD 4.6
4.2.8#16	The Acknowledge Command Indicator field of a GET_CONNECTOR_CAPABILITY Response shall be set to 0.	TD 4.6
4.2.8#17	If the PPM Command was not successfully completed the PPM shall set the Error Indicator field of the GET_CONNECTOR_CAPABILITY Response to 1.	TD 4.6
4.2.8#18	The Command Completed Indicator field of a GET_CONNECTOR_CAPABILITY Response shall be set to 1.	TD 4.6
4.2.8#19	If the GET_CONNECTOR_CAPABILITY Command completed successfully then the PPM shall return additional GET_CONNECTOR_CAPABILITY Data following the GET_CONNECTOR_CAPABILITY Response.	TD 4.6
4.2.8#20	The Reserved field of the GET_CONNECTOR_CAPABILITY Data structure is reserved and shall be set to 0.	TD 4.6

Assertion Number	Assertion Description	Test #
4.2.8#21	If the connector supports Rp only, then bit 0 of the Operation Mode Capability field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1.	TD 4.6
4.2.8#22	If the connector supports Rd only, then bit 1 of the Operation Mode Capability field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1.	TD 4.6
4.2.8#23	If the connector supports (Rp/Rd), then bit 2 of the Operation Mode Capability field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1.	TD 4.6
4.2.8#24	If the connector supports Analog Audio Accessory Mode, then bit 3 of the Operation Mode Capability field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1.	TD 4.6
4.2.8#25	If the connector supports Debug Accessory Mode, then bit 4 of the Operation Mode Capability field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1.	TD 4.6
4.2.8#26	If the connector supports USB 2.0, then bit 5 of the Operation Mode Capability field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1.	TD 4.6
4.2.8#27	If the connector supports USB 3.2, then bit 6 of the Operation Mode Capability field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1.	TD 4.6
4.2.8#28	If the connector supports Alternate Mode, then bit 7 of the Operation Mode Capability field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1.	TD 4.6
4.2.8#29	If the Operation Mode Capability field has bits 0 or 2 set, then the Provider field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1 if the Connector is capable of providing power on this Connector (either BC 1.2, USB Type-C™ Current or PD). Otherwise the bit shall be set to 0.	TD 4.6
4.2.8#30	If the Operation Mode Capability field has bits 1 or 2 set, then the Consumer field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1 if the Connector is capable of consuming power on this Connector (either BC 1.2, USB Type-C™ Current or PD). Otherwise the bit shall be set to 0.	TD 4.6
4.2.8#31	If the Operation Mode Capability field has bits 0, 1 or 2 set, then the Swap to DFP field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1 if the Connector is capable of accepting swap to DFP.	TD 4.6
4.2.8#32	If the Operation Mode Capability field has bits 0, 1 or 2 set, then the Swap to UFP field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1 if the Connector is capable of accepting swap to UFP.	TD 4.6



Assertion Number	Assertion Description	Test #
4.2.8#33	If the Operation Mode Capability field has bit 2 set, then the Swap to SRC field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1 if the Connector is capable of accepting swap to SRC.	TD 4.6
4.2.8#34	If the Operation Mode Capability field has bit 2 set, then the Swap to SNK field of the GET_CONNECTOR_CAPABILITY Data structure shall be set to 1 if the Connector is capable of accepting swap to SNK.	TD 4.6
<b>Subsection reference: 4.2.9 Get PDOs</b>		
4.2.9#1	The GET_PDOS Command shall be used to get the Sink or Source PDOs associated with the Connector identified with the command.	
4.2.9#2	For the Connector associated with the GET_PDOS Command, the GET_PDOS may be used to retrieve the Maximum Supported Source PDOs, the Maximum Provider Capabilities that the Source can support, the Provider Capabilities that the Source currently supports, the Advertised Source Capabilities, and the Provider Capabilities that are advertised by the Source during PD contract negotiation.	
4.2.9#3	The GET_PDOS Command may be used to return the Sink or Source PDOs of the device that is connected to this Connector.	
4.2.9#4	The Command field of a GET_PDOS Command shall be set to 10h (GET_PDOS).	N/A (Host)
4.2.9#5	The Data Length field of a GET_PDOS Command shall be set to 00h.	N/A (Host)
4.2.9#6	The Connector Number field of a GET_PDOS Command shall be set to the Connector whose status is being queried.	
4.2.9#7	The Connector Number field of a GET_PDOS Command shall not be set to 0.	N/A (Host)
4.2.9#8	If the Partner PDO field is set to 0 then the PPM shall return the PDOs of the Connector associated with the GET_PDOS Command.	
4.2.9#9	If the Partner PDO field is set to 1 then the PPM shall return the PDOs of the device connected to the Connector associated with the GET_PDOS Command.	
4.2.9#10	The PPM shall return PDOs starting at the offset indicated by the PDO Offset field.	
4.2.9#11	The OSPM shall not set the PDO Offset field to any value greater than 6.	N/A (Host)
4.2.9#12	If the Source or Sink PDOs field is set to 0 then the PPM shall return Sink PDOs.	
4.2.9#13	If the Source or Sink PDOs field is set to 1 then the PPM shall return Source PDOs.	
4.2.9#14	The Source Capabilities field is only valid if the Partner PDO field is set to 0 and the Source or Sink PDOs field is set to 1.	



Assertion Number	Assertion Description	Test #
4.2.9#15	If the Partner PDO field is set to 0 and the Source or Sink PDOs field is set to 1, then the PPM shall return the Current Supported Source Capabilities when the Source Capabilities field is set to 0.	
4.2.9#16	If the Partner PDO field is set to 0 and the Source or Sink PDOs field is set to 1, then the PPM shall return the Advertised Capabilities when the Source Capabilities field is set to 1.	
4.2.9#17	If the Partner PDO field is set to 0 and the Source or Sink PDOs field is set to 1, then the PPM shall return the Maximum Supported Source Capabilities when the Source Capabilities field is set to 2.	
4.2.9#18	The OSPM shall not set the Source Capabilities Type field to 3.	N/A (Host)
4.2.9#19	The PPM shall return the number of PDOs indicated by the value of the Number of PDOs field plus 1.	
4.2.9#20	The OSPM shall not set the Number of PDOs field to any value greater than 6.	N/A (Host)
4.2.9#21	The Reserved fields of the GET_PDOS Command structure are reserved and shall be set to 0.	N/A (Host)
4.2.9#22	On successful completion of GET_PDOS Command the PPM shall return a PPM Command Response	TD 4.8
4.2.9#23	The reserved fields of a GET_PDOS Response are reserved and shall be set to 0.	TD 4.8
4.2.9#24	If an Asynchronous Event occurred on a Connector, then the Connector Change Indicator field of a GET_PDOS Response shall be set to the Connector Number on which the change occurred.	TD 4.8
4.2.9#25	If the GET_PDOS command is successful, then the Data Length field of a GET_PDOS Command shall be set to 4 times the number of PDOs returned up to a maximum of MAX_DATA_LENGTH bytes.	TD 4.8
4.2.9#26	If the GET_PDOS command is not successful, then the Data Length field of a GET_PDOS Command shall be set to 00h.	TD 4.8
4.2.9#27	The Not Supported Indicator field of a GET_PDOS Response shall be set to 0.	TD 4.8
4.2.9#28	The Cancel Completed Indicator field of a GET_PDOS Response shall be set to 0.	TD 4.8
4.2.9#29	The Reset Completed Indicator field of a GET_PDOS Response shall be set to 0.	TD 4.8
4.2.9#30	The Busy Indicator field of a GET_PDOS Response shall be set to 0.	TD 4.8
4.2.9#31	The Acknowledge Command Indicator field of a GET_PDOS Response shall be set to 0.	TD 4.8
4.2.9#32	If the PPM Command was not successfully completed the PPM shall set the Error Indicator field of the GET_PDOS Response to 1.	TD 4.8

Assertion Number	Assertion Description	Test #
4.2.9#33	The Command Completed Indicator field of a GET_PDOS Response shall be set to 1.	TD 4.8
4.2.9#34	If the GET_PDOS Command completed successfully then the PPM shall return zero or more PDOs as specified by the OSPM in the GET_PDOS Command, and by the PPM as determined by the Command target's capability.	TD 4.8
4.2.9#35	If the PPM receives an otherwise valid GET_PDOS Command for which the target cannot provide any PDOs, it shall set the Error Indicator to 0b in the GET_PDOS Status and shall set the Data Length field to 0. The PPM shall not return any GET_PDO data.	TD 4.8
4.2.9#36	If the PPM receives a GET_PDOS Command in which the sum of the PDO Offset field and the Number of PDOs field is greater than 6, it shall set the Error Indicator to 1b in the GET_PDOS Status and shall set the Invalid Command Specific Parameters bit to 1b in the GET_ERROR_STATUS Data.	TD 4.8
4.2.9#37	If the PPM receives a GET_PDOS Command with the <i>Partner PDO</i> field set to 1 when no PD device is attached, it shall set the Error Indicator to 1b in the GET_PDOS Status and shall set the Incompatible Connector Partner bit to 1b in the GET_ERROR_STATUS Data.	TD 4.8
4.2.9#38	If the PPM receives a GET_PDOS command with the Partner PDO field set to 1 when no Connector partner is present, it shall set the Error Indicator to 1b in the GET_PDOS Status and shall set the CC Communication Error bit to 1b in the GET_ERROR_STATUS Data.	TD 4.8
4.2.9#39	If the PPM receives a GET_PDOS Command with the Partner PDO field set to 0 that is inappropriate for the target, it shall set the Error Indicator to 1b in the GET_PDOS Status and shall set the Invalid Command Specific Parameters bit to 1b in the GET_ERROR_STATUS Data.	TD 4.8
4.2.9#40	If the PPM receives a GET_PDOS Command with the Partner PDO field set to 1 that is inappropriate for the target, it shall set the Error Indicator to 1b in the GET_PDOS Status and shall set the Incompatible Connector Partner bit to 1b in the GET_ERROR_STATUS Data.	TD 4.8
<b>Subsection reference: 4.2.10 Get Connector Status</b>		
4.2.10#1	The GET_CONNECTOR_STATUS Command shall be used to get the current status of a Connector.	N/A (Host)
4.2.10#2	The Command field of a GET_CONNECTOR_STATUS Command shall be set to 12h (GET_CONNECTOR_STATUS).	N/A (Host)
4.2.10#3	The Data Length field of a GET_CONNECTOR_STATUS Command shall be set to 00h.	N/A (Host)
4.2.10#4	The Connector Number field of a GET_CONNECTOR_STATUS Command shall be set to the Connector whose status is being queried.	N/A (Host)

Assertion Number	Assertion Description	Test #
4.2.10#5	The Connector Number field of a GET_CONNECTOR_STATUS Command shall not be set to 0.	N/A (Host)
4.2.10#6	The Reserved field of a GET_CONNECTOR_STATUS Command is reserved and shall be set to 0.	N/A (Host)
4.2.10#7	On successful completion of the GET_CONNECTOR_STATUS Command the PPM shall return a PPM Command Response.	TD 4.7
4.2.10#8	The reserved fields of a GET_CONNECTOR_STATUS Response are reserved and shall be set to 0.	TD 4.7
4.2.10#9	If an Asynchronous Event occurred on a Connector, then the Connector Change Indicator field of a GET_CONNECTOR_STATUS Response shall be set to the Connector Number on which the change occurred.	TD 4.7
4.2.10#10	If the GET_CONNECTOR_STATUS command is successful, then the Data Length field of a GET_CONNECTOR_STATUS Command shall be set to 09h.	TD 4.7
4.2.10#11	If the GET_CONNECTOR_STATUS command is not successful, then the Data Length field of a GET_CONNECTOR_STATUS Command shall be set to 00h.	TD 4.7
4.2.10#12	The Not Supported Indicator field of a GET_CONNECTOR_STATUS Response shall be set to 0.	TD 4.7
4.2.10#13	The Cancel Completed Indicator field of a GET_CONNECTOR_STATUS Response shall be set to 0.	TD 4.7
4.2.10#14	The Reset Completed Indicator field of a GET_CONNECTOR_STATUS Response shall be set to 0.	TD 4.7
4.2.10#15	The Busy Indicator field of a GET_CONNECTOR_STATUS Response shall be set to 0.	TD 4.7
4.2.10#16	The Acknowledge Command Indicator field of a GET_CONNECTOR_STATUS Response shall be set to 0.	TD 4.7
4.2.10#17	If the PPM Command was not successfully completed the PPM shall set the Error Indicator field of the GET_CONNECTOR_STATUS Response to 1.	TD 4.7
4.2.10#18	The Command Completed Indicator field of a GET_CONNECTOR_STATUS Response shall be set to 1.	TD 4.7
4.2.10#19	If the GET_CONNECTOR_STATUS Command completed successfully then the PPM shall return additional GET_CONNECTOR_STATUS Data following the GET_CONNECTOR_STATUS Response.	TD 4.7
4.2.10#20	The Reserved fields of the GET_CONNECTOR_STATUS Data structure are reserved and shall be set to 0.	TD 4.7
4.2.10#21	Bits 0, 1, 3, 4, 8-10 and 13 of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure are reserved and shall be set to 0.	TD 4.7

Assertion Number	Assertion Description	Test #
4.2.10#22	When bit 2 (Power Operation Mode Change) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure is set, the Power Operation Mode shall indicate the current power operational mode of the Connector.	
4.2.10#23	When bit 5 (Supported Provider Capabilities Change) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure is set, the OSPM shall get the updated Power Data Objects by using the GET_PDOS Command.	
4.2.10#24	When bit 5 (Supported Provider Capabilities Change) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure is set, the Supported Provider Capabilities Limited Reason field shall indicate the reason if the provider capabilities are limited.	
4.2.10#25	When bit 6 (Negotiated Power Level Change) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure is set, the Request Data Object field shall indicate the newly negotiated power level.	
4.2.10#26	Bit 7 (PD Reset Complete) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure shall be set when the PPM completes a PD Hard Reset requested by the connector partner.	TD 4.12
4.2.10#27	Bit 11 (Connector Partner Changed) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure shall be set when either the Connector Partner Type field or the Connector Partner Flags field changes.	
4.2.10#28	Bit 12 (Power Direction Changed) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure shall be set when the PPM completes a Power Role Swap requested by the connector partner or due to the PPM autonomously performing a Power Role Swap.	
4.2.10#29	When bit 12 (Power Direction Changed) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure is set, the Power Direction field shall indicate the new Power Role.	
4.2.10#30	When bit 14 (Connect Change) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure is set, the Connection Status field shall indicate whether a device is attached to the Connector.	TD 4.10
4.2.10#31	When bit 14 (Connect Change) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure is set, the Operation Mode field shall indicate the current operational mode of the Connector.	

Assertion Number	Assertion Description	Test #
4.2.10#32	When bit 15 (Error) of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure is set, an unknown error has occurred on the Connector.	
4.2.10#33	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 0, the Power Operation Mode field of the Connector Status Change field is reserved and set to 0.	TD 4.7
4.2.10#34	Bits 6-7 of the Power Operation Mode field of the GET_CONNECTOR_STATUS Data structure are reserved and shall be set to 0.	
4.2.10#35	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1, bit 0 of the Power Operation Mode field shall be set to 1 if the power operation mode is no consumer.	
4.2.10#36	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1, bit 1 of the Power Operation Mode field shall be set to 1 if the power operation mode is USB Default Operation/Type-C™ - default Current.	
4.2.10#37	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1, bit 2 of the Power Operation Mode field shall be set to 1 if the power operation mode is Battery Charging.	
4.2.10#38	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1, bit 3 of the Power Operation Mode field shall be set to 1 if the power operation mode is USB Power Delivery.	
4.2.10#39	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1, bit 1 of the Power Operation Mode field shall be set to 1 if the power operation mode is Type-C™ Current - 1.5A.	
4.2.10#40	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1, bit 1 of the Power Operation Mode field shall be set to 1 if the power operation mode is Type-C™ Current - 3A.	
4.2.10#41	The Connection Status field of the GET_CONNECTOR_STATUS Data structure shall be set to 1 when a device is attached to this Connector. It shall be set to 0 otherwise.	TD 4.7 TD 4.12 TD 4.10
4.2.10#42	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and the Connector is acting as a Consumer, then the Power Direction field shall be set to 0.	

Assertion Number	Assertion Description	Test #
4.2.10#43	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and the Connector is acting as a Provider, then the Power Direction field shall be set to 1.	
4.2.10#44	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 0, the Connector Partner Flags field of the Connector Status Change field is reserved and set to 0.	TD 4.7
4.2.10#45	Bits 2-7 of the Connector Partner Flags field of the GET_CONNECTOR_STATUS Data structure are reserved and shall be set to 0.	TD 4.7
4.2.10#46	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and the Connector is operating in USB mode, then bit 0 of the Connector Partner Flags field shall be set to 1.	
4.2.10#47	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and the Connector is operating in Alternate mode, then bit 1 of the Connector Partner Flags field shall be set to 1.	
4.2.10#48	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 0, the Connector Partner Type field of the Connector Status Change field is reserved and set to 0.	TD 4.7
4.2.10#49	Bits 0 and 7 of the Connector Partner Type field of the GET_CONNECTOR_STATUS Data structure are reserved and shall be set to 0.	
4.2.10#50	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and connector partner is a DFP, then bit 1 of the Connector Partner Type field shall be set to 1.	
4.2.10#51	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and connector partner is an UFP, then bit 2 of the Connector Partner Type field shall be set to 1.	
4.2.10#52	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and connector partner is a Powered cable with no UFP, then bit 3 of the Connector Partner Type field shall be set to 1.	
4.2.10#53	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and connector partner is a Powered cable with an UFP, then bit 4 of the Connector Partner Type field shall be set to 1.	
4.2.10#54	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and connector partner is a Debug Accessory, then bit 5 of the Connector Partner Type field shall be set to 1.	

Assertion Number	Assertion Description	Test #
4.2.10#55	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and connector partner is an Audio Adapter Accessory, then bit 6 of the Connector Partner Type field shall be set to 1.	
4.2.10#56	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 0 or the Power Operation Mode field is not set to PD, the Request Data Object field of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure shall be reserved and set to 0.	TD 4.7
4.2.10#57	When the Connection Status field of the GET_CONNECTOR_STATUS Data structure is set to 1 and the Power Operation Mode field is set to PD, the Request Data Object field of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure shall return the currently negotiated power level as specified by the Request Message during power negotiation.	
4.2.10#58	When the Connector is not acting as a provider, the Provider Capabilities Limited Reason field of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure shall be reserved and set to 0.	TD 4.7 TD 4.11
4.2.10#59	If the Provider Capabilities change and are no longer limited, the PPM shall clear the appropriate bits of the Provider Capabilities Limited Reason field of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure.	
4.2.10#60	Bits 2 and 3 of the Provider Capabilities Limited Reason field of the GET_CONNECTOR_STATUS Data structure are reserved and shall be set to 0.	TD 4.7
4.2.10#61	When the Connector is acting as a provider and the Power Budget for the PPM has been lowered due to it being unplugged from an External Supply then bit 0 of the Provider Capabilities Limited Reason field of the Connector Status Change field of the GET_CONNECTOR_STATUS Data structure shall be set to 1.	
4.2.10#62	When the Connector is acting as a provider and the PPM is reaching the Power Budget Limit due to too many attached PD Sinks then bit 1 of the Provider Capabilities Limited Reason field of the Connector Status Change Field of the GET_CONNECTOR_STATUS Data structure shall be set to 1.	
<b>Subsection reference: 4.2.11 Get Error Status</b>		
4.2.11#1	The GET_ERROR_STATUS Command shall be used to get details about an error, if one is reported by the PPM.	N/A (Host)
4.2.11#2	The PPM shall clear the error status data after the OSPM has acknowledged the PPM Command Response or PPM Asynchronous Notification for the GET_ERROR_STATUS Command.	TD 4.9



Assertion Number	Assertion Description	Test #
4.2.11#3	The PPM shall clear the error status data after receiving a PPM RESET Command.	TD 4.9
4.2.11#4	The PPM shall not clear the error status data unless it has received a PPM RESET or an ACK_PCR_PAN acknowledging a Get Error Status Command.	TD 4.9
4.2.11#5	The Command field of a GET_ERROR_STATUS Command shall be set to 13h (GET_ERROR_STATUS).	N/A (Host)
4.2.11#6	The Data Length field of a GET_ERROR_STATUS Command shall be set to 00h.	N/A (Host)
4.2.11#7	The Reserved field of a GET_ERROR_STATUS Command is reserved and shall be set to 0.	N/A (Host)
4.2.11#8	On successful completion of the GET_ERROR_STATUS Command the PPM shall return a PPM Command Response.	TD 4.9
4.2.11#9	The reserved fields of a GET_ERROR_STATUS Response are reserved and shall be set to 0.	TD 4.9
4.2.11#10	If an Asynchronous Event occurred on a Connector, then the Connector Change Indicator field of a GET_ERROR_STATUS Response shall be set to the Connector Number on which the change occurred.	TD 4.9
4.2.11#11	The Data Length field of a GET_ERROR_STATUS Command shall be set to 10h.	TD 4.9
4.2.11#12	The Not Supported Indicator field of a GET_ERROR_STATUS Response shall be set to 0.	TD 4.9
4.2.11#13	The Cancel Completed Indicator field of a GET_ERROR_STATUS Response shall be set to 0.	TD 4.9
4.2.11#14	The Reset Completed Indicator field of a GET_ERROR_STATUS Response shall be set to 0.	TD 4.9
4.2.11#15	The Busy Indicator field of a GET_ERROR_STATUS Response shall be set to 0.	TD 4.9
4.2.11#16	The Acknowledge Command Indicator field of a GET_ERROR_STATUS Response shall be set to 0.	TD 4.9
4.2.11#17	The Error Indicator field of a GET_ERROR_STATUS Response shall be set to 0.	TD 4.9
4.2.11#18	The Command Completed Indicator field of a GET_ERROR_STATUS Response shall be set to 1.	TD 4.9
4.2.11#19	If the GET_ERROR_STATUS Command completed successfully then the PPM shall return additional GET_ERROR_STATUS Data following the GET_ERROR_STATUS Response.	TD 4.9
4.2.11#20	Bits 13-15 of the Error Information field of the GET_ERROR_STATUS Data structure are reserved and shall be set to 0.	TD 4.9
4.2.11#21	If the PPM receives an unrecognized command, then bit 0 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	TD 4.9



Assertion Number	Assertion Description	Test #
4.2.11#22	If the PPM receives the Connector Number for a non-existing Connector, then bit 1 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	TD 4.9
4.2.11#23	If the PPM receives invalid PPM Command specific parameters, then bit 2 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	TD 4.9
4.2.11#24	If the connector partner is incompatible, then bit 3 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#25	If there is a CC communication failure, then bit 4 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#26	If the PPM Command is unsuccessful due to dead battery condition, then bit 5 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#27	If there is a contract negotiation failure, then bit 6 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#28	If there is an overcurrent condition, then bit 7 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#29	If there is an undefined/unknown error, then bit 8 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#30	If the port partner rejected a swap, then bit 9 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#31	If the partner sent a Hard Reset, then bit 10 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	TD 4.12
4.2.11#32	If there is a PPM Policy Conflict (e.g., the OSPM sends a command which violates the PPM Policy), then bit 11 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#33	If a swap is rejected, then bit 12 of the Error Information field of the GET_ERROR_STATUS Data structure shall be set to 1.	
4.2.11#34	The contents of the Vendor Defined field of the GET_ERROR_STATUS Data structure are vendor specific.	Not tested.
<b>Subsection reference: 5 Operational Model</b>		
5#1	There shall only be one USB Type-C™ Bridge per Device Container.	TD 3.1 TD 3.2

Assertion Number	Assertion Description	Test #
5#2	If a USB Type-C™ Bridge Device has multiple Configurations, then the USB Type-C Bridge function shall be exposed in all of them.	TD 3.1 TD 3.2
5#3	A device container that supports USB Type-C™ Bridge shall have at least 2 independent PD ports.	
5#4	In addition to the default Control endpoint, a USB Type-C™ Bridge shall support a Notification endpoint.	TD 3.1
5#5	The default Control and Notification endpoints shall be used by a USB Type-C™ Bridge Class Driver in a PDUSB Host to send and receive PPM, Authentication and Firmware Update Related Requests and Notifications/Responses respectively.	Inherent in tests
5#6	A USB Type-C™ Bridge shall not send any notifications or responses that are not defined in this specification.	
5#7	A USB Type-C™ Bridge shall ignore any received requests or responses that are not defined in this specification.	Not tested
5#8	The USB Type-C™ Bridge shall respond to USB Type-C Bridge class-specific USB Requests with a request error if it is not in the Configured USB State.	
5#9	The USB Type-C™ Bridge Class Driver shall only have one outstanding USB Type-C Bridge class-specific USB Request for a given USB Type-C Bridge active at any given time.	N/A (Host)
5#10	The USB Type-C™ Bridge having sent a USB Type-C Bridge notification shall not send another notification until the previous notification has been acknowledged.	
5#11	All requests to or notifications from the USB Type-C™ Bridge shall reference the Connector Number of a Connector on the Device Container, if a Connector Number is part of that request or notification.	
5#12	The Connector Number used to address the USB Type-C™ Connectors that the USB Type-C Bridge Class Driver can communicate with shall be determined by the position of the bits set to one in the bmConnectorMask field of the Bridge Capability descriptor.	N/A (Host)
<b>Subsection reference: 5.1 Operating Speed</b>		
5.1#1	A High Speed Bridge Device shall also support Bridge functionality at Full Speed.	TD 3.1
5.1#2	All Device Containers that support the USB Type-C™ Bridge class shall only operate at USB 2.0 unless the Device Container already supports some form of USB functionality.	TD 3.1
5.1#3	If the Device Container includes a PDUSB Hub, then the USB Type-C™ Bridge shall be connected to a USB 2.0 downstream port on the PDUSB Hub.	TD 3.1

Assertion Number	Assertion Description	Test #
5.1#4	The downstream port of the PDUSB Hub to which the USB Type-C™ Bridge is connected shall be marked as an Internal/Non-removable port.	Untestable
5.1#5	If the Device Container includes more than one PDUSB Hub, each PDUSB Hub shall have a USB Type-C™ Bridge Device.	
5.1#6	If the Device Container does not include a PDUSB Hub but includes some other form of USB functionality, then that Device Container shall expose the USB Type-C™ Bridge as a separate function within that Device Container.	TD 3.1
5.1#7	The Device Container shall support Bridge functionality at all speeds the other USB Device functionality supports.	
<b>Subsection reference: 5.2 USB TYPE-C Bridge and PPM</b>		
5.2#1	A PPM shall always respond to a PPM_RESET Command.	TD 4.1
5.2#2	The PPM shall not send PPM Notifications for any Command other than a PPM_RESET until the OSPM enables one or more notifications via the Set Notification Enable PPM Command.	TD 4.11
5.2#3	The PPM shall process PPM Commands regardless of whether or not Command Completed Notifications are enabled.	TD 4.4
5.2#4	The PPM shall not send the OSPM any asynchronous notifications unless the corresponding notification type has been enabled by a SET_NOTIFICATION_ENABLE Command.	TD 4.12 TD 4.10
5.2#5	The USB Type-C™ Bridge Class Driver shall use the default Control endpoint to send PPM Commands to the USB Type-C Bridge.	N/A (Host)
5.2#6	The USB Type-C™ Bridge Class Driver shall receive the PPM Asynchronous Notifications and PPM Command Responses via the Notification endpoint.	N/A (Host)
5.2#7	The OSPM shall send at most one PPM Command at a time to the PPM.	N/A (Host)
5.2#8	With the exception of the CANCEL and PPM_RESET Commands the OSPM shall wait until the PPM completes the current PPM Command before sending the next PPM Command.	N/A (Host)
5.2#9	The OSPM shall consider a PPM_RESET Command to be complete when it receives a PPM Response with the Reset Complete Indicator bit set to 1b.	N/A (Host)
5.2#10	The OSPM shall consider a CANCEL Command complete when the OSPM receives a PPM Response with the Cancel Command Indicator bit set to 1b.	N/A (Host)
5.2#11	The OSPM shall consider a command that has not been cancelled and is not a PPM_RESET to be complete when the OSPM receives a PPM Response with the Acknowledge Command Indicator bit set to 1b.	N/A (Host)

Assertion Number	Assertion Description	Test #
5.2#12	A PPM_RESET Command may be sent by the OSPM at any time.	N/A (Host)
5.2#13	If the OSPM receives a PPM Command Response or an Asynchronous Notification after sending a PPM_RESET Command, but before receiving a PPM Response with the Reset Complete Indicator bit set to 1b, it shall not acknowledge it.	N/A (Host)
5.2#14	An OSPM may cancel a PPM command by sending a CANCEL Command before acknowledging the Command Response.	N/A (Host)
5.2#15	An OSPM shall not cancel a PPM Command after it has acknowledged the Command Response.	N/A (Host)
5.2#16	An OSPM shall not cancel a PPM_RESET Command, a CANCEL Command, or an ACK_PCR_PAN Command.	N/A (Host)
5.2#17	After the OSPM sends a CANCEL Command, it shall not initiate a new PPM Command other than a PPM_RESET until it receives a PPM Command Response for the CANCEL Command.	N/A (Host)
5.2#18	Once the OSPM receives a PPM Command Response for the CANCEL Command, it shall consider the previously pending PPM Command to be complete regardless of whether or not it received a PPM Command Response for the cancelled PPM Command.	N/A (Host)
5.2#19	An OSPM shall not send an ACK_PCR_PAN Command for a cancelled PPM Command, even if it received a PPM Command Response for the cancelled PPM Command.	N/A (Host)
5.2#20	The PPM shall send at most one PPM Asynchronous Notification at a time to the OSPM.	
5.2#21	The PPM shall wait until the OSPM Acknowledges the PPM Asynchronous Notification before sending the next PPM Asynchronous Notification.	
5.2#22	A PPM shall not set the Connector Change Indicator if the change on the Connector occurred as a direct result of a PPM Command sent to that Connector unless the PPM will require more time to process the command.	
<b>Subsection reference: 5.2.1 PPM Command (other than CANCEL or PPM_RESET) Reception</b>		
5.2.1#1	If the PPM is currently processing a different PPM Command, it shall complete the newly received PPM Command USB Request with Request Error.	TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9

Assertion Number	Assertion Description	Test #
5.2.1#2	If the PPM is not currently processing a different PPM Command, it shall execute the Command.	TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
5.2.1#3	Once it has finished executing a PPM Command, the PPM shall set the CCI Data Structure and any additional Data associated with the PPM Command Response as detailed in the sections for each command.	TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
5.2.1#4	Once the PPM has populated the CCI Data Structure and any additional Data, if the Command Completed notification is enabled, the PPM shall send a PPM Command Response to the OSPM.	TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
<b>Subsection reference: 5.2.2 Cancel Command Reception</b>		
5.2.2#1	Upon receiving a CANCEL Command, if the last received PPM Command was an ACK_PCR_PAN Command, the PPM shall complete the newly received CANCEL Command USB Request with a Request Error.	TD 4.2
5.2.2#2	Upon receiving a CANCEL Command, if the last received PPM Command was a CANCEL Command, the PPM shall complete the newly received CANCEL Command USB Request with a Request Error.	TD 4.2
5.2.2#3	Upon receiving a CANCEL Command, if the last received PPM Command was a PPM_RESET Command, the PPM shall complete the newly received CANCEL Command USB Request with a Request Error.	TD 4.2
5.2.2#4	Upon receiving a CANCEL Command, if the last received PPM Command hasn't started executing yet, the PPM shall not execute the PPM Command.	Not Tested
5.2.2#5	Upon receiving a CANCEL Command, if the last received PPM Command is currently in process, the PPM shall stop processing the PPM Command.	Not Tested
5.2.2#6	Upon receiving a CANCEL Command, if the last received PPM Command has completed but PPM has not yet sent a PPM Command Response, it shall drop the Response.	Not Tested
5.2.2#7	Upon receiving a CANCEL Command, the PPM shall not send a PPM Command Response for the cancelled PPM Command.	TD 4.2

Assertion Number	Assertion Description	Test #
5.2.2#8	Upon completion of a CANCEL Command, the PPM shall update the CCI Data Structure with the Cancel Completed Indicator set to one.	TD 4.2
5.2.2#9	If the Command Completed notification is enabled, the PPM shall, upon populating the CCI Data Structure, send it in a PPM Command Response to the OSPM.	TD 4.2
<b>Subsection reference: 5.2.3 PPM_RESET Command Reception</b>		
5.2.3#1	Upon reception of a PPM_RESET Command, the PPM shall disable all notifications.	
5.2.3#2	Upon reception of a PPM_RESET Command, the PPM shall reset itself.	Not tested
5.2.3#3	Upon reception of a PPM_RESET command, the PPM shall populate the CCI Data Structure with the Reset Completed Indicator set to one.	TD 4.1
5.2.3#4	Upon reception of a PPM_RESET command, the PPM shall send, a PPM Command Response to the OSPM.	TD 4.1
<b>Subsection reference: 5.2.4 Asynchronous Events</b>		
5.2.4#1	On the occurrence of an Asynchronous Event on one or more Connectors, the PPM shall set the CCI Data Structure and optionally any data associated with the PPM Asynchronous Notification.	TD 4.4 TD 4.10 TD 4.12
5.2.4#2	On the occurrence of an Asynchronous Event on one or more Connectors, the PPM shall send an Asynchronous Notification with that CCI Data Structure to the OSPM.	TD 4.4 TD 4.10 TD 4.12
<b>Subsection reference: 5.2.5 PPM Command Response or PPM Asynchronous Notification Reception</b>		
5.2.5#1	Upon reception of a Notification from a PPM, the OSPM shall decode the CCI and any additional Data.	N/A (Host)
5.2.5#2	If the Notification was a PPM Asynchronous Notification, the OSPM shall send any other PPM Commands it needs in order to get details of the PPM Asynchronous Notification.	N/A (Host)
5.2.5#3	For all PPM Commands other than PPM_RESET and ACK_PCR_PAN, the OSPM shall acknowledge the Notification from the PPM with an ACK_PCR_PAN Command.	N/A (Host)
5.2.5#4	The OSPM shall not acknowledge the response to a PPM_RESET Command.	N/A (Host)
5.2.5#5	The OSPM shall not acknowledge the response to an ACK_PCR_PAN Command.	N/A (Host)
<b>Subsection reference: 5.3 USB TYPE-C Bridge and PAM</b>		
5.3#1	The USB Type-C™ Bridge Class Driver shall use a Send Authentication Data Request to send an Authentication Message from the OSAM to the USB Type-C Bridge.	N/A (Host)

Assertion Number	Assertion Description	Test #
5.3#2	A USB Type-C™ Bridge shall send an Authentication Notification when it receives an Authentication Message from a PD Product attached to a Connector on a Device Container with bParam3 set to Success (00h).	
5.3#3	A USB Type-C™ Bridge shall send an Authentication Notification when it receives a Not_Supported PD Message from a PD Product attached to a Connector on a Device Container with bParam3 set to Not Supported (01h).	
5.3#4	A USB Type-C™ Bridge shall send an Authentication Notification when it is unable to send an Authentication Message to a PD Product attached to a Connector with bParam3 set to Error (80h).	
5.3#5	On reception of an Authentication Request/Response from a PD Product attached to a Connector on a Device Container, the USB Type-C™ Bridge shall send an Authentication Notification and the USB Type-C Bridge Class Driver shall retrieve the Authentication Request/Response by using the Get Authentication Data Request.	N/A (Host)
5.3#6	All Authentication related requests shall be deemed complete when the USB Type-C™ Bridge sends a USB Type-C Bridge Authentication Notification for that request or the when the OSAM times out the request.	
5.3#7	The USB Type-C™ Bridge Authentication Notifications shall be acknowledged implicitly by successfully transferring the Authentication Notification to the USB Host over the USB Notification endpoint.	Not tested
<b>Subsection reference: 5.3.1 Host-Initiated Authentication</b>		
5.3.1#1	To initiate Authentication of a PD Product attached to a Connector on the Device Container, a USB Host shall Send an Authentication Request in the Data field of a Send Authentication Data Request.	N/A (Host)
5.3.1#2	The USB Type-C™ Bridge shall send and receive Authentication Messages to/from a PD Product attached to a Connector on the Device Container as described in the USB Power Delivery specification.	
5.3.1#3	When the USB Type-C™ Bridge receives an Authentication Response from the PD Product, it shall notify the Host using an Authentication Notification.	
5.3.1#4	Upon receiving the Authentication Notification, a Host shall use a Get Authentication Data Request to retrieve the Authentication Message from the Bridge.	N/A (Host)
<b>Subsection reference: 5.3.2 PD Product-Initiated Authentication</b>		
5.3.2#1	Upon receiving an Authentication Request from a PD Product attached to a Connector on a Device Container, a USB Type-C™ Bridge shall notify a USB Host by sending an Authentication Notification.	



Assertion Number	Assertion Description	Test #
5.3.2#2	Upon receiving the Authentication Notification, a Host shall use a Get Authentication Data Request to retrieve the Authentication Request from the Bridge.	N/A (Host)
5.3.2#3	When the Bridge receives a Send Authentication Data Request from the Host, it shall extract the Authentication Message from the Data field of the received Send Authentication Data Request and shall send it to the PD Product as described in the USB Power Delivery specification.	Tested in all PAM tests.
<b>Subsection reference: 5.4 USB TYPE-C Bridge and PFUM</b>		
5.4#1	The USB Type-C™ Bridge Class Driver shall use the default Control endpoint to Send Firmware Update Requests.	N/A (Host)
5.4#2	The USB Type-C™ Bridge notifies the USB Type-C Bridge Class Driver of the completion of the same by sending back a Firmware Update Notification with the bParam3 field set to either Success or Error.	Not tested
5.4#3	On reception of a Firmware Update Response from a PD Product attached to a Connector on the Device Container, the USB Type-C™ Bridge shall send a Firmware Update Notification containing the response data.	Not tested
5.4#4	All Firmware Update related requests are deemed completed when the USB Type-C™ Bridge sends a USB Type-C Bridge Firmware Update Notification for that command.	Untestable
5.4#5	The USB Type-C™ Bridge Firmware Update Notifications shall be acknowledged implicitly by successfully transferring the Firmware Update Notification to the System over the USB Notification endpoint.	Untestable
<b>Subsection reference: 5.4.1 Host-Initiated Firmware Update</b>		
5.4.1#1	To initiate Firmware Update of a PD Product attached to a Connector on a Device Container, a USB Host shall Send a Firmware Update Request in the Data field of a Send Firmware Update Data Request.	N/A (Host)
<b>Subsection reference: 5.5 USB TYPE-C Bridge and Billboard</b>		
5.5#1	When a USB Type-C™ Bridge supports the Billboard functionality, the Billboard Capability shall be as defined in the USB Billboard specification.	TD 3.2
5.5#2	The USB Type-C™ Bridge shall use the mechanisms defined in the USB Billboard specification to notify System Software of a change in its Billboard Capability descriptor.	
5.5#3	If a USB Type-C™ Bridge supports the Billboard functionality it shall also support the Billboard Notification.	TD 3.2



Assertion Number	Assertion Description	Test #
5.5#4	On Power up or after a USB Reset the USB Type-C™ Bridge shall not send any Billboard Notifications and it shall use the mechanisms defined in the USB Billboard specification to notify System Software of any Billboard Capability changes.	
5.5#5	If the USB Type-C™ Bridge supports the Billboard functionality as specified in the Bridge Capability descriptor and System Software Enables the Billboard Notification via the Enable/Disable Billboard Notification Request, then the USB Type-C Bridge shall transition to notifying System Software of changes in its Billboard Capability descriptor via the Notification endpoint.	
5.5#6	The USB Type-C™ Bridge Class Driver shall use the standard Get BOS Descriptor Request, to retrieve the Billboard Capability descriptor, when the USB Type-C Bridge sends a Billboard Notification to the System Software.	N/A (Host)
5.5#7	The USB Type-C™ Bridge Billboard Notifications shall be acknowledged implicitly by successfully transferring the Billboard Notification to the System over the USB Notification endpoint.	Untestable
<b>Subsection reference: 5.6 USB TYPE-C Bridge Initialization</b>		
5.6#1	The USB Type-C™ Bridge shall enumerate and initialize as a standard USB Device.	USB Chapter 9 (See [USB2.0] and [USB3.1])
5.6#2	During Type-C™ Bridge initialization, System Software should scan all the Connectors on the Device Container.	N/A (Host)
<b>Subsection reference: 5.7 USB TYPE-C Bridge Operational State Diagram</b>		
5.7#1	If the USB Type-C™ Bridge is not in the Configured state, it shall return a STALL for all USB Type-C Bridge Requests.	
5.7#2	If the USB Type-C™ Bridge is not in the Configured state, it shall discard all Asynchronous Events.	
5.7#3	If an Asynchronous PPM Event is received and the appropriate PPM Notification is not Enabled, then the Event shall be discarded.	
5.7#4	Upon receiving a CANCEL Command, if the PPM does not have an active Command, the PPM shall complete the newly received CANCEL Command USB Request with a Request Error.	TD 4.2
5.7#5	If the PPM receives a Command (other than a PPM Reset or CANCEL) while it has an active Command, it shall continue to process the active Command, as if the new Command had not been received.	TD 4.4 TD 4.5 TD 4.6 TD 4.7 TD 4.8 TD 4.9
5.7#6	Upon receiving a CANCEL Command, if the active Command is one of PPM Reset, CANCEL, or ACK_PCR_PAN, the PPM shall continue to process the active Command, as if the new CANCEL Command had not been received.	TD 4.2

Assertion Number	Assertion Description	Test #
5.7#7	If Command Completed Notifications are not enabled, then the PPM shall not send a PPM Response to any Command other than PPM Reset.	TD 4.4
5.7#8	If the PPM receives an ACK_PCR_PAN acknowledging a previous ACK_PCR_PAN command, it shall respond with a request error.	TD 4.3
5.7#9	If the PPM receives an ACK_PCR_PAN acknowledging a PPM Reset command, it shall respond with a request error.	TD 4.3
<b>Subsection reference: 5.8 USB TYPE-C Bridge Suspend/Resume</b>		
<b>Subsection reference: 5.8.1 USB TYPE-C Bridge Suspend</b>		
5.8.1#1	The USB Type-C™ Bridge shall support Remote Wake.	TD 3.1 TD 5.2
5.8.1#2	When enabled for Remote Wake, the USB Type-C™ Bridge shall attempt to wake the System up whenever it detects an Asynchronous Event that would have caused it to send a notification.	
5.8.1#3	System Software shall enable the appropriate notifications such that an Asynchronous Event is translated into a Resume Signal.	N/A (Host)
5.8.1#4	Only PPM Notifications can cause a Remote Wake event to be generated by the USB Type-C™ Bridge.	
5.8.1#5	In order to enable Remote Wake on a USB Type-C™ bridge, System Software shall first enable the appropriate PPM Asynchronous Notifications.	N/A (Host)
5.8.1#6	After enabling the appropriate PPM Asynchronous Notifications, System Software shall enable Remote Wake on the USB Type-C™ Bridge.	N/A (Host)
5.8.1#7	After enabling Remote Wake on the USB Type-C™ Bridge, System Software shall suspend the port on which the USB Type-C Bridge is connected.	N/A (Host)
5.8.1#8	If the Device Container contains a PDUSB Hub, then after suspending the port on which the USB Type-C™ Bridge is connected, System Software shall enable Remote Wake on the PDUSB Hub.	N/A (Host)
5.8.1#9	If the Device Container contains a PDUSB Hub, then after enabling Remote Wake on the PDUSB Hub, System Software shall suspend on the PDUSB Hub.	N/A (Host)
<b>Subsection reference: 5.8.2 USB Type-C Bridge Resumed by Host</b>		
5.8.2#1	If the Device Container is resumed by Host-Initiated Wake, System Software shall drive Resume signaling on Device Container upstream port.	N/A (Host)

Assertion Number	Assertion Description	Test #
5.8.2#2	If the Device Container contains a PDUSB Hub, then after driving Resume signaling on Device Container upstream port, System Software shall resume the port on which the USB Type-C™ Bridge is connected.	N/A (Host)
5.8.2#3	After driving Resume signaling on the Device Container upstream port and, if the Device Container contains a PDUSB Hub, resuming the port on which the USB Type-C™ Bridge is connected, System Software shall scan all the Connectors on the Device Container.	N/A (Host)
<b>Subsection reference: 5.8.3 USB Type-C Bridge Resumed by USB Type-C Bridge</b>		
5.8.3#1	If the Device Container initiates the Wake, the USB Type-C™ Bridge shall initiate Resume signaling.	
5.8.3#2	After the USB Type-C™ Bridge initiates Resume signaling, System Software shall acknowledge resume signaling and clear the port suspend on the downstream port on which the Device Container is connected.	N/A (Host)
5.8.3#3	After clearing the port suspend, if the Device Container includes a PDUSB Hub, System Software shall resume the downstream port on which the USB Type-C™ Bridge is attached.	N/A (Host)
5.8.3#4	After clearing the port suspend on the downstream port on which the Device Container and, if the Device Container includes a PDUSB Hub, resuming the downstream port on which the USB Type-C™ Bridge is attached, the USB Type-C Bridge shall send the PPM Notification to System Software.	
5.8.3#5	After the USB Type-C™ Bridge sends the PPM Notification to System Software, System Software shall scan all the Connectors on the Device Container.	N/A (Host)

### 3 Test Descriptions for USB Type-C Bridge 1.0

#### **Test Results**

A test is considered to **pass** if it completes with no *Failures* and no *Aborts*.

- A **Failure** shall be logged (alternately, **the test fails**) when the Device Under Test does not act as required by the USB Type-C Specification.
- An **Abort** shall be logged when one or more steps of the test cannot be run. A test fails if it logs one or more Aborts.

#### **General Test Initialization**

All USB Type-C Bridge Device tests follow the same initialization procedure. At the beginning of a test run, the following steps are taken:

1. The host controller is reset and devices attached to the host are enumerated.
2. If attached devices fail initialization assertions, the test tool will not be able to run the tests. Record an abort and exit the test suite. (Note that the host controller is not necessarily reset in between tests when more than one test is selected.)
3. Prompt the tester to select the Connector Numbers of the Connectors on the Device Under Test.
4. Prompt the tester to select the Connector Numbers of the **Accessible Connectors** (that is, those that to which a user can physically attach and detach a USB Type-C Device or Fixture) from the list of Connectors.
5. Note that the host controller is not reset in between tests if more than 1 test is selected.

#### **Assertions Used in Test Initialization**

*Chapter 9 Assertions:* 9.1.1#1, 9.1.1#2, 9.1.1.3#1, 9.1.1.4#1, 9.1.1.5#1, 9.2.6.1#1, 9.2.6.3#1, 9.2.6.4#1, 9.3#1, 9.4.3#5, 9.4.3#6, 9.4.3#10, 9.4.3#10, 9.4.3#11

#### **General USB Type-C Bridge Test Definitions Used in this Test Specification**

**Supports PPM functionality** – If bit 0 of the *bmAttributes* field of the Bridge Capability Descriptor is set to 1 then the device is considered to support PPM Requests, Commands and data structures and is tested as such.

**Supports PAM functionality** – If bit 1 of the *bmAttributes* field of the Bridge Capability Descriptor is set to 1 then the device is considered to support PAM Requests, Commands and data structures and is tested as such

**Supports PFUM functionality** – If bit 2 of the *bmAttributes* field of the Bridge Capability Descriptor is set to 1 then the device is considered to support PFUM Requests, Commands and data structures and is tested as such

**Supports Billboard functionality** – If bit 3 of the *bmAttributes* field of the Bridge Capability Descriptor is set to 1 then the device is considered to support Billboard Requests, Commands and data structures and is tested as such

**Billboard only** – A device is **billboard only** if it supports Billboard functionality but doesn't support any other USB Type-C functionality. If bits 0-2 of the *bmAttributes* field of the Bridge Capability Descriptor are set to 0, and bit 3 is set to 1, then that device is to be tested as **billboard only**.

**Function** – A collection of one or more interfaces that perform a specific task. In a full Configuration Descriptor, a **function** is represented as either an Interface Association Descriptor and its associated Interfaces (an Interface Association), or a single Interface Descriptor that does not belong to an Interface Association.

**PPM Command failure** – In addition to failure modes defined in the USB Type-C Bridge specification, a PPM Command will be considered to have failed if a PPM Response is not received within 5 seconds.

**Standalone** – The Device Under Test is a **standalone** USB Type-C Bridge if it contains no additional USB functions.

**USB Type-C Bridge Interface** – The interface for a USB Type-C **function**: in the context of descriptors, the **USB Type-C Bridge Interface Descriptor** and all its child descriptors.

**USB Type-C Bridge Interface Descriptor** – The descriptor that represents a USB Type-C Bridge. It is identified by its *bInterface* field which must be set to 12h.

## USB Type-C Interface Validation

For all tests in which Type-C Bridge class-specific requests are sent to the Device Under Test, the **USB Type-C Bridge Interface** must be validated.

1. Retrieve the full Configuration Descriptor of the Device Under Test by first sending a GetDescriptor(Configuration) with a length of 9, and then sending a GetDescriptor(Configuration) with a length of the wTotalLength field of the Configuration Descriptor.
  - a. Test records an abort and ceases validation if either request fails.
2. Parse the full Configuration Descriptor, saving all **USB Type-C interfaces** for future reference:
  - a. A USB Type-C interface consists of a single USB **function**, consisting of the following descriptors in order:
    - i. A **USB Type-C Bridge Interface Descriptor**. See TD 3.1 for further requirements for this descriptor.
    - ii. A Bridge Capability Descriptor. See TD 3.2 for further requirements for this descriptor.
    - iii. An Endpoint Descriptor. See TD 3.1 for further requirements for this descriptor.
3. If the full Configuration descriptor does not have a **USB Type-C Bridge Interface**, or if it has more than one, then test records failure. (5#1, 5#2)

## General PPM Initialization

There are a number of tests, involving PPM Commands other than PPM RESET, which are initialized the same way. For those tests, the following is performed after USB enumeration, but prior to any other PPM Commands

1. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification.
  - a. Record an abort and exit **General PPM Initialization** if this fails for any reason.
2. Send a Set Notification Enable PPM command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the Device Under Test. In this command, if not otherwise specified, set bit 0 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1.
  - a. Record an abort and go to step 4 if command fails.
3. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.

- i. Record an abort and go to step 4 if command fails.
4. If the Set Notification Enable PPM Command in step 2 or the ACK\_PCR\_PAN in step 3 failed for any reason:
  - a. Record an abort.
  - b. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification.
    - i. Record an abort and exit **General PPM Initialization** if this fails for any reason.

## General PPM Notification Validation

1. Parse the Type-C Bridge PPM Notification returned to the Notification endpoint.
  - a. Test fails if the *bNotificationType* field is not set to 01h (PPM\_NOTIFICATION). (3.2.1.2#1)
  - b. Test fails if the *bParam1* field is not set to 00h. (3.2.1.2#2)
  - c. Test fails if the *bParam2* field is not set to 00h. (3.2.1.2#3)
  - d. Test fails if the *bParam3* field is not set to 00h. (3.2.1.2#4)

## Ensure Connector has/does not have Connector Partner

There are several tests which need to determine whether or not anything is attached the Device Under Test and based on that request that the user attach items to or detach items from the Device Under Test as appropriate.

Upon success, the test that executes these steps shall have access to up-to-date Get Connector Status Response data.

For these tests, perform the following steps. Unless otherwise specified, if any step fails for any reason, record an abort and skip the remaining steps.

1. Send a Get Connector Status Command as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the number of the desired Connector.
2. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
3. Determine whether a device is attached to the desired Connector, by checking the *Connection Status* field of the Get Connector Status Data structure.
  - a. Retain the Get Connector Status Response data for future use.
  - b. If there is a Connector Partner, and that is what is desired, then skip all remaining steps.
  - c. If there is no Connector Partner, and that is what is desired, then skip all remaining steps.
4. If the desired Connector is not **Accessible**, go to step 7.
5. Prompt user to attach/detach Connector Partner as appropriate.
6. Wait for the Device Under Test to send an Asynchronous Notification.
  - a. If no Asynchronous Notification arrives, then allow user to stop waiting, test records an abort and skips all remaining steps.
  - b. If the Asynchronous Notification is not a PPM notification, then test records an abort and skips all remaining steps.
  - c. If the Asynchronous Notification is for a different Connector than the desired Connector, then test records an abort and skips all remaining steps.

7. Send a Get Connector Status Command as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the number of the desired Connector.
  - a. Retain the Get Connector Status Response data for future use. (*Note: if the Connection Status field of the Response data is set to 1, then a device is connected*)
8. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
9. If the desired Connector is **Accessible**, send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 1 and the *Command Completed Acknowledge* field set to 0.

### ***Determine whether Power Delivery Is Supported***

Certain tests are only applicable when the Device Under Test supports PD. To determine this, perform the following steps:

1. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Retain the Get Capability Response data for future use.
2. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
3. If bit 2 of the *bmAttributes* field is not set, then the Device Under Test **does not support** PD. Skip all remaining steps.
4. If bit 2 of the *bmAttributes* field is set, and the test is not concerned with a specific connector, then the Device Under Test **supports** PD. Skip all remaining steps.
5. If bit 2 of the *bmAttributes* field is set, and the test concerns a specific Connector:
  - a. Send a Get Connector Capability Command as defined in sections 3.2.1.1 and 4.2.8 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the number of the desired Connector.
    - i. Retain the Get Connector Capability Response data for future use.
  - b. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - c. If bits 0-2 of the *Operation Mode Capability* field are all set to 0, then the Device Under Test **does not support** PD on that Connector. Skip all remaining steps.
  - d. If at least one of bits 0-2 of the *Operation Mode Capability* field are set to 1, and both the *Provider* field and the *Consumer* field are set to 0, then the Device Under Test **does not support** PD on that Connector. Skip all remaining steps.
  - e. If at least one of bits 0-2 of the *Operation Mode Capability* field are set to 1, and at least one of the *Provider* field and the *Consumer* field are set to 0, then the Device Under Test **supports** PD on that Connector.

### ***TD 3.1 Standard Descriptors Test***

#### **Device States for Test**

This test is run with the device in the configured state.

#### **Repetitions**

Run this test for all Configurations the Device Under Test supports.



1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Retrieve the Device Descriptor of the Device Under Test by sending a GetDescriptor(Device) request.
  - a. Test records an abort and exits if request fails.
  - b. Test fails if Device Under Test is a Hub (*bDeviceType* = 9) (5.1#3)
3. Retrieve the full Configuration Descriptor of the Device Under Test by first sending a GetDescriptor(Configuration) with a length of 9, and then sending a GetDescriptor(Configuration) with a length of the *wTotalLength* field of the Configuration Descriptor.
  - a. Test records an abort and exits if either request fails.
  - b. Test fails if bit 5 (Remote Wakeup) of the *bmAttributes* field of the Configuration Descriptor is not set to 1. (5.8.1#1)
4. Determine whether the Device Under Test is a **standalone** USB Type-C Bridge
  - a. Test fails if **standalone** Device Under Test is not operating at a USB 2.0 speed. (5.1#2)
5. Parse the full Configuration Descriptor, retrieving all Interface Descriptors. Examine each Descriptor:
  - a. Test fails if there isn't exactly one **USB Type-C Bridge Interface Descriptor** (3.1#5, 5#1, 5#2). That is, one with the following properties:
    - i. *bInterfaceClass* = 12h (3.1#2)
    - ii. *bInterfaceSubClass* = 0 (3.1#3)
    - iii. *bInterfaceProtocol* = 0 (3.1#4)
  - b. Repeat step 5a for all **USB Type-C Bridge Interface Descriptors**.
6. Parse the full **USB Type-C Interface Descriptor**, retrieving all Endpoint Descriptors.
  - a. Test fails if there isn't exactly one Endpoint Descriptor. (3.1#6)
    - i. If there are no Endpoint Descriptors, record assertion 5#4.
  - b. For all Endpoint Descriptors in **USB Type-C interfaces**, test fails if the descriptor is not as follows:
    - i. *bmAttributes* = 3. ((3.1#19)
    - ii. *wMaxPacketSize* = 40h (3.1#20)
    - iii. *bInterval* = 04h (3.1#21)
  - c. Repeat step 6a - 6b for all Endpoint Descriptors in the full **USB Type-C Interface Descriptor**.
7. Repeat step 6 for all full **USB Type-C Interface Descriptors**.
8. Parse the full Configuration Descriptor, dividing it into **functions**
  - a. Test fails if any full **USB Type-C Interface** is not the sole interface within its **function**. (3.1#5, 5.1#6)

## TD 3.2 Bridge Capability Descriptor Test

### Device States for Test

This test is run with the device in the configured state.

### Repetitions

Run this test for all Configurations the Device Under Test supports.

### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.



2. Retrieve the full Configuration Descriptor of the Device Under Test by first sending a GetDescriptor(Configuration) with a length of 9, and then sending a GetDescriptor(Configuration) with a length of the *wTotalLength* field of the Configuration Descriptor.
  - a. Test records an abort and exits if either request fails.

#### [Existence and Uniqueness]

3. Parse the full Configuration Descriptor, retrieving all Interface Descriptors and all Bridge Capability Descriptors. Examine each Descriptor:
  - a. Test fails if there isn't exactly one Bridge Capability Descriptor. (3.1#5, 5#1)
  - b. Test fails if each **USB Type-C Bridge Interface Descriptor** is not immediately followed by a Bridge Capability Descriptor. (3.1#7)
  - c. Test fails if Bridge Capability Descriptor appears in any location other than immediately following a **USB Type-C Bridge Interface Descriptor**. (3.1#7)

#### [The Bridge Capability Descriptor]

4. For the Bridge Capability Descriptor:
  - a. Test fails if *bLength* is not equal to 7. (3.1#8)
  - b. Test fails if *bDescriptorType* is not equal to 21h. (3.1#9)
  - c. Test fails if *bcdVersion* is not equal to 0100h. (3.1#10)
  - d. Test fails if bit 0 of *bmConnectorMask* is not set to 0. (3.1#13)
  - e. If bits 0-2 of *bmAttributes* are all set to 0 and bit 3 is set to 1, then the Device Under Test is referred to as **Billboard Only**.
    - i. Test fails if Device Under Test is Billboard Only and *bmConnectorMask* is not equal to 0. (1.4#1)
    - ii. Test fails if Device is not Billboard only and bit 0 is set to 0. (3.2.1#1)
  - f. Test fails if bits 4-7 of *bmAttributes* is not set to 0 (3.1#18)

#### [Billboard]

5. If bit 3 of *bmAttributes* is set to 1, then get the full BOS descriptor by first sending a GetDescriptor(BOS) with a length of 5, and then sending a GetDescriptor(BOS) with a length of the *wTotalLength* field of the BOS Descriptor.
  - a. Record an abort and go to step 7 if either GetDescriptor fails.
6. If bit 3 of *bmAttributes* is set to 1, Parse the full BOS descriptor, retrieving all *DEVICE\_CAPABILITY* descriptors in which the *bDevCapabilityType* field is set to 0Dh (13).
  - a. Test fails if there isn't one. (3.1#17, 5.5#1)

#### [Connectors and *bmConnectorMask*]

7. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits test if validation fails.
8. If bit 0 of *bmAttributes* of the Bridge Capability descriptor is set to 0, then exit test.
9. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 13.
10. Send a Get Capability PPM Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Record an abort and go to step 11 if command fails.
  - b. Test fails if value of *bNumConnectors* is less than the number of bits set in the *bmConnectorMask* field of the Bridge Capability Descriptor. (3.1#12)

- c. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
11. For each bit that is set in the *bmConnectorMask*, send a Get Connector Status command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the corresponding Connector.
  - a. Test fails if command fails. (3.1#11)
  - b. Send an Acknowledge PPM Command Response as defined in sections 3.2.1.1 and 4.2.5 to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
12. If any PPM Command fails, then record an abort and go to step 13.
13. Send a PPM RESET Command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. If Command fails for any reason, record an abort and exit the test.

## TD 4.1 PPM RESET Command Test

### Device States for Test

This test is run with the device in the configured state.

### Repetitions

Run this test for all Configurations the Device Under Test supports.

#### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.

#### [PPM RESET]

4. Send a PPM RESET Command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test fails if PPM Response is not received within 5 seconds. (5.2#1, 5.2.3#4)
  - b. Test records an abort and exits if this fails for any reason.
5. Perform **General PPM Notification Validation**.
6. Parse the PPM RESET Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.2#6)
  - b. Test fails if the *Connect Change Indicator* field is not set to 0h. (4.1.2#3, 4.2.2#7)
  - c. Test fails if the *Data Length* field is not set to 0h. (4.2.2#8)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0h. (4.1.2#8, 4.2.2#9)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0h. (4.1.2#9, 4.2.2#10)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 1h. (4.1.2#10, 4.2.2#11, 5.2.3#3)
  - h. Test fails if the *Busy Indicator* field is not set to 0h. (4.1.2#13, 4.2.2#13)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0h. (4.1.2#14, 4.2.2#14)
  - j. Test fails if the *Error Indicator* field is not set to 0h. (4.1.2#17, 4.2.2#15)
  - k. Test fails if the *Command Completed Indicator* field is not set to 0h. (4.1.2#18, 4.2.2#16)

- I. If the *Reset Completed Indicator* field is set to 1, then test fails if any other field in the PPM Reset Response is set to 1. (4.1.2#11)

**[Reset Completed Indicator is cleared after next Command.]**

7. Send a Set Notification Enable PPM command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the Device Under Test. In this command, set bit 0 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1.
  - a. Record an abort and go to step 8 if command fails.
8. Parse the Set Notification Enable Response.
  - a. Test fails if the *Reset Completed Indicator* field is not set to 0h. (4.2.2#12)

**[PPM\_RESET is responded to when Command is in process.]**

9. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. If Command fails for any reason, record an abort and exit the test.
10. Send a Set Notification Enable PPM command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the Device Under Test. In this command, set bit 0 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1.
  - a. Record an abort and go to step 13 if Command fails for any reason.
11. Without waiting for a PPM Response to the last Command, send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test fails if this does not succeed. (5.2#1)
  - b. Record an abort and go to step 13 if Command fails for any reason.
12. Parse any PPM Responses received. (*Note: it is possible that the Device Under Test sent a PPM Response to the Set Notification Enable Command prior to receiving the PPM\_RESET.*)
  - a. Test fails if Device Under Test does not send a PPM Response with the *Reset Completed Indicator* field set to 1h within 5 seconds. (5.2#1, 5.2.3#3, 5.2.3#4)
  - b. After receiving the PPM Response in the above step, wait for 5 seconds.
    - i. Test fails if Device Under Test sends a PPM Response within that time. (5.2.3#1)
13. Send a PPM RESET Command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. If Command fails for any reason, record an abort and exit the test.

## **TD 4.2 CANCEL Command Test**

### **Device States for Test**

This test is run with the device in the configured state.

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

**[Initialization]**

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.

4. Perform **General PPM Initialization**.
  - a. If any steps fail, then record an abort and exit the test.

**[Cancel active command]**

5. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
6. Without waiting for a response from the Device Under Test, immediately send a CANCEL Command as defined in sections 3.2.1.1 and 4.2.3 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 10 if this fails for any reason.
7. Parse any Response in the PPM Notification payload field. (*Note: it is possible that the Device Under Test sent a PPM Response to the Set Notification Enable Command prior to receiving the CANCEL.*)
  - a. Test fails if a PPM Response with the *Cancel Completed Indicator* is set to 1 does not arrive within 5 seconds. (5.2.2#7, 5.2.2#8)
    - i. If no such PPM Response arrives, then test records an abort and goes to step 10.
8. Examine the contents of the most recently arrived payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.3#6)
  - b. Test fails if the *Connector Change Indicator* field is not set to 0h. (4.1.2#3, 4.2.3#7)
  - c. Test fails if the *Data Length* field is not set to 00h. (4.2.3#8)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0h. (4.1.2#8, 4.2.3#9)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 1h. (4.1.2#9, 4.2.3#10)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0h. (4.1.2#10, 4.2.3#11)
  - h. Test fails if the *Busy Indicator* field is not set to 0h. (4.1.2#13, 4.2.3#12)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0h. (4.1.2#14, 4.2.3#13)
  - j. Test fails if the *Error Indicator* field is not set to 0h. (4.1.2#17, 4.2.3#14)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1h. (4.1.2#18, 4.2.3#15)
9. Do nothing for 5 seconds.
  - a. Test fails if PPM sends a Response. (5.2.2#4, 5.2.2#5, 5.2.2#6)

**[Cancel non-existent command]**

10. Perform **General PPM Initialization**.
  - a. If any steps fail, then record an abort and exit the test.
11. Send a CANCEL Command as defined in sections 3.2.1.1 and 4.2.3 to the Device Under Test.
  - a. Test fails if the USB SEND\_PPM\_COMMAND request doesn't STALL.

**[Cancel PPM\_RESET command]**

12. Perform **General PPM Initialization**.
  - a. If any steps fail, then record an abort and go to step 29.
13. Send a PPM\_RESET Command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 18 if this fails for any reason.
14. Without waiting for a PPM Response, send a CANCEL Command as defined in sections 3.2.1.1 and 4.2.3 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.2#3)
15. Wait for the PPM to send a PPM Response with the *Reset Complete Indicator* bit set.
  - a. Test fails if this doesn't happen within 5 seconds.
16. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort if this fails for any reason.

17. Send a Set Notification Enable Command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the Device Under Test. In this command, set bit 0 of the Notification Enable field of the SET\_NOTIFICATION\_ENABLE Command to 1.
  - a. Test records an abort and goes to step 9 if this fails for any reason.

**[Cancel CANCEL command]**

18. Perform **General PPM Initialization**.
  - a. If any steps fail, then record an abort and go to step 29.
19. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 24 if this fails for any reason.
20. Without waiting for a PPM Response to the previous Command, send a CANCEL Command as defined in sections 3.2.1.1 and 4.2.3 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 24 if this fails for any reason.
21. Without waiting for a PPM Response to the previous Command, send a CANCEL Command as defined in sections 3.2.1.1 and 4.2.3 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.2#2)
22. Wait for the PPM to send a PPM Response with the *Cancel Complete Indicator* bit set.
  - a. Test fails if this doesn't happen within 5 seconds.
23. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 24 if this fails for any reason.

**[Cancel ACK\_PCR\_PAN command]**

24. Perform **General PPM Initialization**.
  - a. If any steps fail, then record an abort and go to step 29.
25. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 29 if this fails for any reason.
26. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 29 if this fails for any reason.
27. Without waiting for a response to the previous Command, send a CANCEL Command as defined in sections 3.2.1.1 and 4.2.3 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.2#1)
28. Wait for the PPM to send a PPM Response with the *Acknowledge Command Indicator* bit set.
  - a. Test fails if this doesn't happen within 5 seconds.
29. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort if this fails for any reason.

## **TD 4.3 ACK\_PCR\_PAN Command Test**

### **Device States for Test**

This test is run with the device in the configured state.

## Repetitions

Run this test for all Configurations the Device Under Test supports.

### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
4. Perform **General PPM Initialization**.
  - a. If any steps fail, record an abort and go to step 14.

### [ACK\_PCR\_PAN]

5. Send a Get Capability command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 14 if this fails for any reason.
6. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 14 if this fails for any reason.
7. Perform **General PPM Notification Validation**.
8. Parse the ACK\_PCR\_PAN Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.5#8)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.5#9)
  - c. Test fails if the *Data Length* field is not set to 00h. (4.2.5#10)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.5#11)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.5#12)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.5#13)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.5#14)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 1. (4.1.2#14, 4.2.5#15)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.5#16)
  - k. Test fails if the *Command Completed Indicator* field is not set to 0. (4.1.2#18, 4.2.5#17)
  - l. Test fails if any field other than the *Connector Change Indicator* field and the *Acknowledge Command Indicator* field are not set to 0. (4.1.2#16)
9. Send a Get Capability command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 14 if this fails for any reason.
  - b. Test fails if *Acknowledge Command Indicator* is set to 1. (4.1.2#15)
10. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.

### [Acknowledge ACK\_PCR\_PAN]

11. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.



- a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.7#8)

**[Acknowledge PPM\_RESET]**

12. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and exit the test if this fails for any reason.
13. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.7#9)
14. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and exit the test if this fails for any reason.

## **TD 4.4 Set Notification Enable Command Test**

### **Device States for Test**

This test is run with the device in the configured state.

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

**[Initialization]**

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
4. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 19.

**[Set Notification Enable]**

5. Send a Set Notification Enable Command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the Device Under Test. In this command, set bit 0 of the Notification Enable field of the SET\_NOTIFICATION\_ENABLE Command to 1.
  - a. Test records an abort and goes to step 9 if this fails for any reason.
6. Perform **General PPM Notification Validation**.
7. Parse the Set Notification Enable Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.6#18)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.6#19)
  - c. Test fails if the *Data Length* field is not set to 00h. (4.2.6#20)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.6#21)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.6#22)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.6#23)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.6#24)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.6#25)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.6#26)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.6#27)

8. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 9 if this fails for any reason.
9. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort if this fails for any reason.

**[PPM Commands Processed when Notification not enabled.]**

10. If the Device Under Test does not have an **Accessible Connector**, then go to step 13.
11. Send a Set Notification Enable Command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the Device Under Test. In this command, set bit 14 of the Notification Enable field of the SET\_NOTIFICATION\_ENABLE Command to 1 and do nothing for 5 seconds.
  - a. Test fails if Device Under Test sends a PPM Response in that time. (5.2#2, 5.2#4)
12. Prompt user to attach or detach a Type-C device with an UFP to any **Accessible Connector** on the Device Under Test.
  - a. Test fails if Device Under Test does not send an Asynchronous Notification. (5.2#3)
  - b. Test fails if the Data Length field of the Asynchronous Notification is greater than 0. (4.1.2#7)
  - c. Test fails if the Connect Change Indicator field is greater than the maximum number of Connectors supported on the platform. (4.1.2#4)
    - i. *Note: this can be determined by counting the number of 1 bits in the bmConnectorMask field of the Bridge Capability Descriptor.*

**[STALL when prior PPM Command not complete.]**

13. Perform **General PPM Initialization**.
  - a. Test records an abort and goes to step 19 if any steps fail for any reason.
14. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 19 if this fails for any reason.
  - b. Save the Response data for later use.
15. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 19 if this fails for any reason.
16. Without waiting for a response, send a Set Notification Enable Command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the Device Under Test. In this Command, set bit 0 of the Notification Enable field to 1.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.1#1)
17. Wait to receive a PPM Response with *Command Completed Indicator* set to 1 and *Error Indicator* set to 0.
  - a. Test fails if this doesn't happen within 5 seconds.
18. Compare the Response data to the Response data received in step 14.
  - a. Test fails if the data doesn't match.
19. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and exits the test if this fails for any reason.

## **TD 4.5 Get Capability Command Test**

### **Device States for Test**



This test is run with the device in the configured state.

### Repetitions

Run this test for all Configurations the Device Under Test supports.

#### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
4. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 23.

#### [Get Capability – good command]

5. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 12 if this fails for any reason.
6. Perform **General PPM Notification Validation**.
7. Parse the Get Capability Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.7#6)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.7#7)
  - c. Test fails if the *Data Length* field is not set to 10h. (4.2.7#8)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.7#10)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.7#11)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.7#12)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.7#13)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.7#14)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.7#15)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.7#16)
8. If *Error Indicator* is set to 1 or *Data Length* is not set to 00h, then go to step 10.
9. Parse the Get Capability Data structure.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.2.7#18)
  - b. Test fails if any of bits 3-7, 9, 11-13 or 16-31 of the *bmAttributes* field are not set to 0. (4.2.7#20)
  - c. If bit 1 is not set, then test fails if *bcdBCVersion* is not set to 0. (4.2.7#32)
  - d. If bit 2 is not set, then test fails if *bcdPDVersion* is not set to 0. (4.2.7#34)
  - e. Test fails unless at least one of bits 8, 10, and 14 is set. (4.2.7#27).
  - f. Test fails if the *bNumConnectors* field is set to 0 and the Device Container is not a Billboard Only USB Type-C. Bridge. (4.2.7#28) (*Note: the Device Container will never be Billboard Only at this point, as Billboard only Device Containers do not support PPM commands.*)
  - g. Test fails if *bmOptionalFeatures* field is not set to 0. (4.2.7#29)
  - h. Test fails if *bNumAltModes* field is not set to 0. (4.2.7#30)
10. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
11. Test records an abort and goes to step 12 if this fails for any reason.

#### [Get Capability – bad command]

12. Perform **General PPM Initialization**.

- a. If any steps fail for any reason, then test records an abort and goes to step 23.
- 13. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test, with the *Data Length* field set to 01h.
  - a. Test records an abort and goes to step 17 if this fails for any reason.
- 14. Perform **General PPM Notification Validation**.
- 15. Parse the Get Capability Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.7#6)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.7#7)
  - c. Test fails if the *Data Length* field is not set to 00h. (4.2.7#9)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.7#10)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.7#11)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.7#12)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.7#13)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.7#14)
  - j. Test fails if the *Error Indicator* field is not set to 1. (4.1.2#17, 4.2.7#15)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.7#16)
- 16. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 16 if this fails for any reason.

**[STALL when prior PPM Command not complete.]**

- 17. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 23.
- 18. Send a Get Connector Capability Command as defined in sections 3.2.1.1 and 4.2.8 of the USB Type-C Bridge specification to the Device Under Test. Set the Connector Number field to the lowest numbered Connector on the Device Under Test.
  - a. Test records an abort and goes to step 23 if this fails for any reason.
  - b. Save the Response data for later use.
- 19. Send a Get Connector Capability Command as defined in sections 3.2.1.1 and 4.2.8 of the USB Type-C Bridge specification to the Device Under Test. Set the Connector Number field to the lowest numbered Connector on the Device Under Test.
  - a. Test records an abort and goes to step 23 if this fails for any reason.
- 20. Without waiting for a response, send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.1#1)
- 21. Wait to receive a PPM Response with *Command Completed Indicator* set to 1 and *Error Indicator* set to 0.
  - a. Test fails if this doesn't happen within 5 seconds.
- 22. Compare the Response data to the Response data received in step 18.
  - a. Test fails if the data doesn't match.
- 23. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and exits the test if this fails for any reason.

## **TD 4.6 Get Connector Capability Command Test**

### **Device States for Test**

This test is run with the device in the configured state.

## Repetitions

Run this test for all Configurations the Device Under Test supports.

Run test for each **Connector**.

### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
4. Save the *bmConnectorMask* field of the Bridge Capability Descriptor for later use.
5. If *bmConnectorMask* is set to 0, then test returns success and exits.
6. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 24.

### [Get Connector Capability – good command]

7. Send a Get Connector Capability Command as defined in sections 3.2.1.1 and 4.2.8 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the value corresponding with the lowest significant bit of the *bmConnectorMask* field that is set to 1.
  - a. Test records an abort and goes to step 13 if this fails to return a PPM Response for any reason.
8. Perform **General PPM Notification Validation**.
9. Parse the Get Connector Capability Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.8#8)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.8#9)
  - c. Test fails if the *Data Length* field is not set to 2h. (4.2.8#10)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.8#12)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.8#13)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.8#14)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.8#15)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.8#16)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.8#17)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.8#18)
10. If *Error Indicator* is set to 1 or *Data Length* is not set to 2h, then go to step 13.
11. Parse the Get Connector Capability data structure.
  - a. Test fails if the *Reserved* field is not set to 0. (4.2.8#20)
  - b. Ask user if Connector Under Test supports Rp only.
    - i. Test fails if answer does not match bit 0 of the *Operation Mode Capability* field. (4.2.8#21)
  - c. Ask user if Connector Under Test supports Rd only.
    - i. Test fails if answer does not match bit 1 of the *Operation Mode Capability* field. (4.2.8#22)
  - d. Ask user if Connector Under Test supports Rp/Rd.
    - i. Test fails if answer does not match bit 2 of the *Operation Mode Capability* field. (4.2.8#23)
  - e. Ask user if Connector Under Test supports Analog Audio Accessory Mode.
    - i. Test fails if answer does not match bit 3 of the *Operation Mode Capability* field. (4.2.8#24)
  - f. Ask user if Connector Under Test supports Debug Accessory Mode.

- i. Test fails if answer does not match bit 4 of the *Operation Mode Capability* field. (4.2.8#25)
- g. Ask user if Connector Under Test supports USB 2.
  - i. Test fails if answer does not match bit 5 of the *Operation Mode Capability* field. (4.2.8#26)
- h. Ask user if Connector Under Test supports USB 3.
  - i. Test fails if answer does not match bit 6 of the *Operation Mode Capability* field. (4.2.8#27)
- i. Ask user if Connector Under Test supports Alternate Mode.
  - i. Test fails if answer does not match bit 7 of the *Operation Mode Capability* field. (4.2.8#28)
- j. If either bit 0 or bit 2 of the *Operation Mode Capability* field is set to 1, ask user if Connector Under Test is capable of providing power.
  - i. Test fails if answer does not match *Provider* field. (4.2.8#29)
- k. If either bit 1 or bit 2 of the *Operation Mode Capability* field is set to 1, ask user if Connector Under Test is capable of consuming power.
  - i. Test fails if answer does not match *Consumer* field. (4.2.8#30)
- l. If any of bits 0, 1 or 2 of the *Operation Mode Capability* field are set to 1, ask user if Connector is capable of accepting swap to DFP.
  - i. Test fails if answer does not match *Swap to DFP* field. (4.2.8#31)
- m. If any of bits 0, 1 or 2 of the *Operation Mode Capability* field are set to 1, ask user if Connector is capable of accepting swap to UFP.
  - i. Test fails if answer does not match *Swap to UFP* field. (4.2.8#32)
- n. If any of bits 0, 1 or 2 of the *Operation Mode Capability* field are set to 1, ask user if Connector is capable of accepting swap to SRC.
  - i. Test fails if answer does not match *Swap to SRC* field. (4.2.8#33)
- o. If any of bits 0, 1 or 2 of the *Operation Mode Capability* field are set to 1, ask user if Connector is capable of accepting swap to SNK.
  - i. Test fails if answer does not match *Swap to SNK* field. (4.2.8#34)
- 12. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 13 if this fails for any reason.

**[Get Connector Capability – bad command]**

- 13. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 24.
- 14. Send a Get Connector Capability Command as defined in sections 3.2.1.1 and 4.2.8 of the USB Type-C Bridge specification to the Device Under Test, with the *Data Length* field set to 01h.
  - a. Test records an abort and goes to step 18 if this fails for any reason.
- 15. Perform **General PPM Notification Validation**.
- 16. Parse the Get Connector Capability Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.8#8)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.8#9)
  - c. Test fails if the *Data Length* field is not set to 00h. (4.2.8#11)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.8#12)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.8#13)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.8#14)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.8#15)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.8#16)

- j. Test fails if the *Error Indicator* field is not set to 1. (4.1.2#17, 4.2.8#17)
- k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.8#18)
- 17. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 18 if this fails for any reason.

**[STALL when prior PPM Command not complete.]**

- 18. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason then test records an abort and goes to step 24.
- 19. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 24 if this fails for any reason.
  - b. Save the Response data for later use.
- 20. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 24 if this fails for any reason.
- 21. Without waiting for a response, send a Get Connector Capability Command as defined in sections 3.2.1.1 and 4.2.6 of the USB Type-C Bridge specification to the Device Under Test. In this Command, set the Connector Number field to the number of the Connector Under Test.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.1#1)
- 22. Wait to receive a PPM Response with *Command Completed Indicator* set to 1 and *Error Indicator* set to 0.
  - a. Test fails if this doesn't happen within 5 seconds.
- 23. Compare the Response data to the Response data received in step 19.
  - a. Test fails if the data doesn't match.
- 24. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and exits the test if this fails for any reason.

## **TD 4.7 Get Connector Status Command - No Device Attached Test**

### **Device States for Test**

This test is run with the device in the configured state.

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

Run test for each **Accessible Connector**.

**[Initialization]**

- 1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
- 2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
- 3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
- 4. Save the *bmConnectorMask* field of the Bridge Capability Descriptor for later use.
- 5. If *bmConnectorMask* is set to 0, then test returns success and exits.
- 6. Perform **General PPM Initialization**, only set bit 0 and bit 14 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1 in step 2.



- a. If any steps fail for any reason, then test records an abort and goes to step 24.

**[Get user to remove device attached to Connector Under Test]**

**7. Ensure Connector Under Test does not have a Connector Partner.**

- a. Test records an abort and goes to step 24 if any step fails for any reason.
- b. Test records an abort and goes to step 24 if Connector still has a Connector Partner.

**[Get Connector Status – good command]**

**8. Perform *General PPM Initialization*.**

- a. If any steps fail for any reason, then test records an abort and goes to step 15.

**9. Send a Get Connector Status Command as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the number of the Connector Under Test.**

- a. Test records an abort and goes to step 15 if PPM fails to return a Response for any reason.

**10. Perform *General PPM Notification Validation*.**

**11. Parse the Get Connector Status Response in the PPM Notification payload field.**

- a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.10#8)
- b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.10#9)
- c. Test fails if the *Data Length* field is not set to 09h. (4.2.10#10)
- d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
- e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.10#12)
- f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.10#13)
- g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.10#14)
- h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.10#15)
- i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.10#16)
- j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.10#17)
- k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.10#18)

**12. If *Error Indicator* is set to 1 or *Data Length* is not set to 09h, then go to step 15.**

**13. Parse the Get Connector Status data structure.**

- a. Test fails if the *Reserved* fields are not set to 0. (4.2.10#19)
- b. Test fails if bits 0, 1, 3, 4, 8, 10 and 13 of the *Connector Status Change* field are not all set to 0. (4.2.10#20)
- c. Test fails if the *Request Data Object* field is not set to 0. (4.2.10#56).
- d. Test fails if bits 2-7 of the *Connector Partner Flags* field are not set to 0. (4.2.10#45)
- e. Test fails if the *Power Operation Mode* field is not set to 0. (4.2.10#30)
- f. Test fails if the *Connection Status Field* is not set to 0. (4.2.10#41)
- g. Test fails if the *Connector Partner Flags* field is not set to 0. (4.2.10#44)
- h. Test fails if the *Connector Partner Type* field is not set to 0. (4.2.10#48)
- i. Test fails if the *Provider Capabilities Limited Reason* is not set to 0. (4.2.10#58)

**14. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.**

- a. Test records an abort and goes to step 15 if this fails for any reason.

**[Get Connector Status – bad command]**

**15. Perform *General PPM Initialization*.**

- a. If any steps fail for any reason, then test records an abort and goes to step 18.

**16. Send a Get Connector Status Command as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the value corresponding**

with the lowest significant bit of the *bmConnectorMask* field that is set to 1 and the *Data Length* field set to 01h.

- a. Test records an abort and goes to step 18 if PPM fails to return a Response for any reason.
17. Parse the Get Connector Status Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.10#8)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.10#9)
  - c. Test fails if the *Data Length* field is not set to 0h. (4.2.10#11)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.10#12)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.10#13)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.10#14)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.10#15)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.10#16)
  - j. Test fails if the *Error Indicator* field is not set to 1. (4.1.2#17, 4.2.10#17)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.10#18)

**[STALL when prior PPM Command not complete.]**

18. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 24.
19. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 24 if this fails for any reason.
  - b. Save the Response data for later use.
20. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 24 if this fails for any reason.
21. Without waiting for a response, send a Get Connector Status Command as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test. In this Command, set the Connector Number field to the number of the Connector Under Test.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.1#1)
22. Wait to receive a PPM Response with *Command Completed Indicator* set to 1 and *Error Indicator* set to 0.
  - a. Test fails if this doesn't happen within 5 seconds.
23. Compare the Response data to the Response data received in step 19.
  - a. Test fails if the data doesn't match.
24. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and exits the test if this fails for any reason.

## **TD 4.8 Get PDOs Command Test**

### **Device States for Test**

This test is run with the device in the configured state.

### **Equipment**

This test requires a Known Good PD Source and a Known Good PD Sink. (These may be the same device, if available.)

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

Run test for each **Connector**.

### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
4. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 39.
5. **Determine whether Power Delivery is supported** on the current Connector.
  - a. If Power Delivery is not supported on the current Connector, then go to step 39.
6. Prompt user to enter to what extent the Connector Under Test supports PD. Provide the following options:
  - Does not support PD.
  - Can act as a PD Source (only if either bit 0 or bit 2 of the *Operation Mode Capability* field of the Get Connector Capability Response data is set to 1 AND *Provider* of the Get Connector Capability Response data is set to 1)
  - Can act as a PD Sink (only if either bit 1 or bit 2 of the *Operation Mode Capability* field of the Get Connector Capability Response data is set to 1 AND *Consumer* of the Get Connector Capability Response data is set to 1)
  - Can act as both a PD Source and a PD Sink (only if bit 2 of the *Operation Mode Capability* field of the Get Connector Capability Response data is set to 1 AND both *Provider* and *Consumer* of the Get Connector Capability Response data are both set to 1)
  - a. Save the result for future use.
7. If user selects "Does not support PD, then go to step 39.

### [Get Local PDOs]

8. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 39.
9. If the Connector Under Test cannot act as a PD Source, then go to step 13.
10. Send a Get PDOs Command as defined in sections 3.2.1.1 and 4.2.9 of the USB Type-C Bridge specification to the Device Under Test with the following arguments:
  - Data Length: 28
  - Connector Number: number of Connector Under Test.
  - Partner PDO: 0
  - Source or Sink PDOs: 1
  - Source Capabilities Type: 2
  - Number of PDOs: 6
  - a. Test records an abort and goes to step 17 if PPM fails to return a Response for any reason.
11. Perform **General PPM Notification Validation**.
12. Parse the Get Connector Status Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.9#23)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.9#24)
  - c. Test fails if the *Data Length* field is not a multiple of 4. (4.2.9#25)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.9#27)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.9#28)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.10#29)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.9#30)



- i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.9#31)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.9#32)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.9#33)
13. If the Connector Under Test cannot act as a PD Sink, then go to step 17.
14. Send a Get PDOs Command as defined in sections 3.2.1.1 and 4.2.9 of the USB Type-C Bridge specification to the Device Under Test with the following arguments:
- Data Length: 28
  - Connector Number: number of Connector Under Test.
  - Partner PDO: 0
  - Source or Sink PDOs: 0
  - Source Capabilities Type: 0
  - Number of PDOs: 6
- a. Test records an abort and goes to step 17 if PPM fails to return a Response for any reason.
15. Perform **General PPM Notification Validation**.
16. Parse the Get Connector Status Response in the PPM Notification payload field.
- a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.9#23)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.9#24)
  - c. Test fails if the *Data Length* field is not a multiple of 4. (4.2.9#25)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.9#27)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.9#28)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.10#29)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.9#30)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.9#31)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.9#32)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.9#33)

#### [Get PDOs From Sink Connector Partner]

17. If Connector Under Test cannot act as a Source (see step 6), then go to step 24.
18. Perform **General PPM Initialization**, only set bit 0 and bit 14 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1 in step 2.
- a. If any steps fail for any reason, then test records an abort and goes to step 24.
19. **Ensure Connector Under Test does not have a Connector Partner.**
- a. Test records an abort and goes to step 24 if any step fails for any reason.
  - b. Test records an abort and goes to step 24 if Connector is **Accessible** and still has a Connector Partner.
  - c. Test goes to step 24 if Connector is not **Accessible** and does not have a Connector partner.
20. **Ensure Connector Under Test has a Connector Partner** that supports PD as a Sink.
- a. Test records an abort and goes to step 24 if any step fails for any reason.
  - b. Test records an abort and goes to step 24 if Connector does not have a Connector Partner.
  - c. Test records an abort and goes to step 24 if Power Operation Mode is not PD (*Power Operation Mode* field of GET\_CONNECTOR\_STATUS is not set to 3).
21. Send a Get PDOs Command as defined in sections 3.2.1.1 and 4.2.9 of the USB Type-C Bridge specification to the Device Under Test with the following arguments:
- Data Length: 28
  - Connector Number: number of Connector Under Test.
  - Partner PDO: 1
  - Source or Sink PDOs: 1
  - Source Capabilities Type: 0

- Number of PDOs: 6
  - a. Test records an abort and goes to step 24 if PPM fails to return a Response for any reason.
22. Perform **General PPM Notification Validation**.
23. Parse the Get Connector Status Response in the PPM Notification payload field.
- a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.9#23)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.9#24)
  - c. Test fails if the *Data Length* field is not a multiple of 4. (4.2.9#25)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.9#27)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.9#28)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.10#29)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.9#30)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.9#31)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.9#32)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.9#33)

#### [Get PDOs From Source Connector Partner]

24. If Connector Under Test cannot act as a Sink (see step 6), then go to step 29.
25. Perform **General PPM Initialization**.
- a. If any steps fail for any reason, then test records an abort and goes to step 29.
26. Send a Get PDOs Command as defined in sections 3.2.1.1 and 4.2.9 of the USB Type-C Bridge specification to the Device Under Test with the following arguments:
- Data Length: 28
  - Connector Number: number of Connector Under Test.
  - Partner PDO: 1
  - Source or Sink PDOs: 0
  - Source Capabilities Type: 0
  - Number of PDOs: 6
- a. Test records an abort and goes to step 29 if PPM fails to return a Response for any reason.
27. Perform **General PPM Notification Validation**.
28. Parse the Get Connector Status Response in the PPM Notification payload field.
- a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.9#23)
  - b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.9#24)
  - c. Test fails if the *Data Length* field is not set to 0. (4.2.9#26)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.9#27)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.9#28)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.10#29)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.9#30)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.9#31)
  - j. Test fails if the *Error Indicator* field is not set to 1. (4.1.2#17, 4.2.9#32)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.9#33)

#### [Get PDOs—bad command]

29. Perform **General PPM Notification Validation**.
30. Send a Get PDOs Command as defined in sections 3.2.1.1 and 4.2.9 of the USB Type-C Bridge specification to the Device Under Test with the following arguments:
- Data Length: 28
  - Connector Number: 0.

- Partner PDO: 0
  - Source or Sink PDOs: 1 if Connector Under Test can act as Source, otherwise 0.
  - Source Capabilities Type: 0
  - Number of PDOs: 6
- a. Test records an abort and goes to step 39 if PPM fails to return a Response for any reason.

31. Perform **General PPM Notification Validation**.

32. Parse the Get Connector Status Response in the PPM Notification payload field.

- a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.9#23)
- b. Test fails if the *Connector Change Indicator* field is not set to 00h. (4.1.2#3, 4.2.9#24)
- c. Test fails if the *Data Length* field is not set to 0. (4.2.9#26)
- d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
- e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.9#27)
- f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.9#28)
- g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.10#29)
- h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.9#30)
- i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.9#31)
- j. Test fails if the *Error Indicator* field is not set to 1. (4.1.2#17, 4.2.9#32)
- k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.9#33)

*[STALL when prior PPM Command not complete.]*

33. Perform **General PPM Initialization**.

34. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.

- a. Test records an abort and goes to step 39 if this fails for any reason.
- b. Save the Response data for later use.

35. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.

- a. Test records an abort and goes to step 39 if this fails for any reason.

36. Without waiting for a response, send a valid Get PDOs Command as defined in sections 3.2.1.1 and 4.2.9 of the USB Type-C Bridge specification to the Device Under Test. In this Command, set the Connector Number field to the number of the Connector Under Test.

- a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.1#1)

37. Wait to receive a PPM Response with *Command Completed Indicator* set to 1 and *Error Indicator* set to 0.

- a. Test fails if this doesn't happen within 5 seconds.

38. Compare the Response data to the Response data received in step 34.

- a. Test fails if the data doesn't match.

39. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.

- a. Test records an abort and exits the test if this fails for any reason.

## **TD 4.9 Get Error Status Command Test**

### **Device States for Test**

This test is run with the device in the configured state.

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

Run test for each **Connector**.

#### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
4. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 51.

#### [Unknown Command]

5. Send a PPM command with *Command* field set to FFh, the *Data Length* field set to 0, and the *Reserved* field set to 0.
  - a. Test records an abort and goes to step 14 if PPM fails to return a Response for any reason.
6. Perform **General PPM Notification Validation**.
7. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 14 if this fails for any reason.
8. Send a Get Error Status Command as defined in sections 3.2.1.1 and 4.2.11 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 14 if PPM fails to return a Response.
9. Perform **General PPM Notification Validation**.
10. Parse the Get Error Status Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.11#9)
  - b. Test fails if the *Connector Change Indicator* field is not set to 0. (4.1.2#3, 4.2.11#10)
  - c. Test fails if the *Data Length* field is not set to 10h (GET\_ERROR\_STATUS\_DATA\_LENGTH). (4.2.11#11)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#11)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.11#12)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.11#13)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.11#14)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.11#15)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.11#16)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.11#17)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.11#18)
11. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 14 if this fails for any reason.
12. If *Error Indicator* is set to 1 or *Data Length* is not set to 10h, then go to step 14.
13. Parse the Get Error Status data structure.
  - a. Test fails if bits 13-15 of the *Error Information* field are not all set to 0. (4.2.11#20)
  - b. Test fails if bit 0 of the *Error Information* field is not set to 1. (4.2.11#21)

#### [Bad connector number]

14. Send a Get Connector Capability Command as defined in sections 3.2.1.1 and 4.2.8 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to 10h
  - a. Test records an abort and goes to step 22 if PPM fails to return a Response for any reason.

15. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 22 if this fails for any reason.
16. Send a Get Error Status Command as defined in sections 3.2.1.1 and 4.2.11 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 22 if this fails to return a PPM Response for any reason.
17. Perform **General PPM Notification Validation**.
18. Parse the Get Error Status Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.11#9)
  - b. Test fails if the *Connector Change Indicator* field is not set to 0. (4.1.2#3, 4.2.11#10)
  - c. Test fails if the *Data Length* field is not set to 10h (GET\_ERROR\_STATUS\_DATA\_LENGTH). (4.2.11#11)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#11)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.11#12)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.11#13)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.11#14)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.11#15)
  - i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.11#16)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.11#17)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.11#18)
19. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 22 if this fails for any reason.
20. If *Error Indicator* is set to 1 or *Data Length* is not set to 10h, then go to step 22.
21. Parse the Get Error Status data structure.
  - a. Test fails if bits 13-15 of the *Error Information* field are not all set to 0. (4.2.11#20)
  - b. Test fails if bit 1 of the *Error Information* field is not set to 1. (4.2.11#22)

#### [Bad data length]

22. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test with the *Data Length* field set to 1h.
  - a. Test records an abort and goes to step 29 if PPM fails to return a Response for any reason.
23. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 29 if this fails for any reason.
24. Send a Get Error Status Command as defined in sections 3.2.1.1 and 4.2.11 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 29 if this fails to return a PPM Response for any reason.
25. Perform **General PPM Notification Validation**.
26. Parse the Get Error Status Response in the PPM Notification payload field.
  - a. Test fails if either one of the *Reserved* fields is not set to 0h. (4.1.2#2, 4.2.11#9)
  - b. Test fails if the *Connector Change Indicator* field is not set to 0. (4.1.2#3, 4.2.11#10)
  - c. Test fails if the *Data Length* field is not set to 10h. (4.2.11#11)
  - d. Test fails if the *Data Length* field is greater than 38h (MAX\_DATA\_LENGTH). (4.1.2#6)
  - e. Test fails if the *Not Supported Indicator* field is not set to 0. (4.1.2#8, 4.2.11#12)
  - f. Test fails if the *Cancel Completed Indicator* field is not set to 0. (4.1.2#9, 4.2.11#13)
  - g. Test fails if the *Reset Completed Indicator* field is not set to 0. (4.1.2#10, 4.2.11#14)
  - h. Test fails if the *Busy Indicator* field is not set to 0. (4.1.2#13, 4.2.11#15)



- i. Test fails if the *Acknowledge Command Indicator* field is not set to 0. (4.1.2#14, 4.2.11#16)
  - j. Test fails if the *Error Indicator* field is not set to 0. (4.1.2#17, 4.2.11#17)
  - k. Test fails if the *Command Completed Indicator* field is not set to 1. (4.1.2#18, 4.2.11#18)
27. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
- a. Test records an abort and goes to step 29 if this fails for any reason.
28. Parse the Get Error Status data structure.
- a. Test fails if bits 13-15 of the *Error Information* field are not all set to 0. (4.2.11#20)
  - b. Test fails if bit 2 of the *Error Information* field is not set to 1. (4.2.11#23)

**[Error bits are sticky, and are cleared by Acknowledgement of Get Error Status]**

29. Perform **General PPM Initialization**.
- a. If any steps fail for any reason, then test records an abort and goes to step 51.
30. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test with the *Data Length* field set to 1h.
- a. Test records an abort and goes to step 39 if PPM fails to return a Response for any reason.
31. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
- a. Test records an abort and goes to step 39 if this fails for any reason.
32. Send a Get Connector Capability Command as defined in sections 3.2.1.1 and 4.2.8 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to 10h
- a. Test records an abort and goes to step 39 if PPM fails to return a Response for any reason.
33. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
- a. Test records an abort and goes to step 39 if this fails for any reason.
34. Send a Get Error Status Command as defined in sections 3.2.1.1 and 4.2.11 of the USB Type-C Bridge specification to the Device Under.
- a. Test records an abort and goes to step 39 if this fails to return a PPM Response for any reason.
35. Parse the Get Error Status data structure.
- a. Test fails if bits 13-15 of the *Error Information* field are not all set to 0. (4.2.11#20)
  - b. Test fails if bits 1 and 2 of the *Error Information* field are not both set to 1. (4.2.11#4)
36. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
- a. Test records an abort and goes to step 39 if this fails for any reason.
37. Send a Get Error Status Command as defined in sections 3.2.1.1 and 4.2.11 of the USB Type-C Bridge specification to the Device Under.
- a. Test records an abort and goes to step 39 if this fails to return a PPM Response for any reason.
38. Parse the Get Error Status data structure.
- a. Test fails if any bits of the *Error Information* field are not set to 0. (4.2.11#2)
  - b. Test fails if bits 1 and 2 of the *Error Information* field are not both set to 1. (4.2.11#22)

**[Error bits cleared by PPM Reset]**

39. Perform **General PPM Initialization**.
- a. If any steps fail for any reason, then test records an abort and goes to step 51.

40. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test with the *Data Length* field set to 1h.
  - a. Test records an abort and goes to step 46 if PPM fails to return a Response for any reason.
41. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 46 if this fails for any reason.
42. Send a Get Error Status Command as defined in sections 3.2.1.1 and 4.2.11 of the USB Type-C Bridge specification to the Device Under.
  - a. Test records an abort and goes to step 46 if this fails to return a PPM Response for any reason.
43. Perform **General PPM Initialization**.
  - a. Test records an abort and goes to step 46 if this fails for any reason.
44. Send a Get Error Status Command as defined in sections 3.2.1.1 and 4.2.11 of the USB Type-C Bridge specification to the Device Under.
  - a. Test records an abort and goes to step 46 if this fails to return a PPM Response for any reason.
  - b. Test fails if any bits of the *Error Information* field are not set to 0. (4.2.11#3)

**[STALL when prior PPM Command not complete.]**

45. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 51.
46. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 51 if this fails for any reason.
  - b. Save the Response data for later use.
47. Send a Get Capability Command as defined in sections 3.2.1.1 and 4.2.7 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and goes to step 51 if this fails for any reason.
48. Without waiting for a response, send a Get Error Status Command as defined in sections 3.2.1.1 and 4.2.9 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test fails if the USB SEND\_PPM\_COMMAND doesn't STALL. (5.2.1#1)
49. Wait to receive a PPM Response with *Command Completed Indicator* set to 1 and *Error Indicator* set to 0.
  - a. Test fails if this doesn't happen within 5 seconds.
50. Compare the Response data to the Response data received in step 46.
  - a. Test fails if the data doesn't match.
51. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and exits the test if this fails for any reason.

## **TD 4.10 Connect/Disconnect Notification Test**

### **Device States for Test**

This test is run with the device in the configured state.

### **Equipment**

This test requires a device with a USB Type-C Upstream Facing Port (UFP).

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

Run test for each **Accessible Connector**.

### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
4. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
5. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
6. If the Device Under Test does not have an **Accessible Connector**, then test records success and exits.
7. Perform **General PPM Initialization**, but set bit 0 and bit 14 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1 in step 2.
  - a. If any steps fail for any reason, then test records an abort and goes to step 21.
8. **Ensure Connector Under Test does not have a Connector Partner**.
  - a. Test records an abort and goes to step 21 if any step fails for any reason.
  - b. Test records an abort and goes to step 21 if Connector still has a Connector Partner.

### [Connection change notifications ON – prompt connect]

9. Prompt user to attach device with UFP to the Connector Under Test.
  - a. Test fails if Device Under Test does not return an asynchronous notification. (4.2.6#14)
  - b. Test fails if the *Connector Change Indicator* field of the Asynchronous Notification structure is not set to the connector number of the Connector Under Test. (4.1.2#3, 4.1.2#4)
  - c. Test fails if the *Data Length* field of the Asynchronous Notification structure is not set to 00h. (4.1.2#7)
10. Send a Get Connector Status Command as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the number of the Connector Under Test.
  - a. Test records an abort and goes to step 21 if the request fails for any reason.
  - b. Parse the Get Connector Status Response structure.
  - c. Test fails if bit 14 of the *Connector Status Change* field is not set to 1. (4.2.10#30)
  - d. Test fails if the *Connection Status* field is not set to 1. (4.2.10#41)
11. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 21 if the request fails for any reason.
12. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 1 and the *Command Completed Acknowledge* field set to 0.
  - a. Test records an abort and goes to step 21 if the request fails for any reason.

### [Connection change notifications ON – prompt disconnect]

13. Prompt user to detach device with UFP to the Connector Under Test.
  - a. Test fails if Device Under Test does not return an asynchronous notification. (4.2.6#14)
  - b. Test fails if the *Connector Change Indicator* field of the Asynchronous Notification structure is not set to the connector number of the Connector Under Test. (4.1.2#3, 4.1.2#4)
  - c. Test fails if the *Data Length* field of the Asynchronous Notification structure is not set to 00h. (4.1.2#7)



14. Send a Get Connector Status Command as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the number of the Connector Under Test.
  - a. Test records an abort and goes to step 21 if the request fails for any reason.
15. Parse the Get Connector Status Response structure.
  - a. Test fails if bit 14 of the *Connector Status Change* field is not set to 1. (4.2.10#30)
  - b. Test fails if the *Connection Status* field is not set to 0. (4.2.10#41)
16. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
  - a. Test records an abort and goes to step 21 if the request fails for any reason.
17. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 1 and the *Command Completed Acknowledge* field set to 0.
  - a. Test records an abort and goes to step 21 if the request fails for any reason.

#### [Connection change notifications OFF]

18. Perform **General PPM Initialization**.
19. Prompt user to attach device with UFP to Connector Under Test.
  - a. Test fails if Device Under Test returns an asynchronous notification. (4.2.6#14, 5.2#4)
20. Prompt user to detach device with UFP from Connector Under Test
  - a. Test fails if Device Under Test returns an asynchronous notification. (4.2.6#14, 5.2#4)
21. Send a PPM RESET Command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort and exits if this fails for any reason.

## TD 4.11 Default Notifications Test

### Device States for Test

This test is run with the device in the configured state.

### Equipment

This test requires a device with a USB Type-C Upstream Facing Port (UFP).

### Repetitions

Run this test for all Configurations the Device Under Test supports.

#### [Initialization]

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
4. Perform **General PPM Initialization**.
  - a. If any steps fail for any reason, then test records an abort and goes to step 15.

#### [Notifications turned off after PPM\_RESET.]

5. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification.
  - a. Record an abort and go to step 10 this fails for any reason.

6. Send a GET\_CAPABILITY Command as defined in sections 3.2.1.1 and 4.2.7 and do nothing for 5 seconds.
  - a. Test fails if Device Under Test returns a PPM Response in that time. (5.2#2, 5.2.3#1)
7. If the Device Under Test does not have an **Accessible Connector**, then go to step 10.
8. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification.
  - a. Record an abort and exit test if this fails for any reason.
9. Prompt the user to change the attachment status (that is, detach the attached USB Type-C Device if there is one, attach one if there isn't) of an Accessible Connector on the Device Under Test, then do nothing for 5 seconds.
  - a. Test fails if Device Under Test sends an Asynchronous Notification to the Host. (5.2#2, 5.2#4)

**[Notifications turned off after Enumeration.]**

10. Prompt user to detach Device Under Test and then reattach it to the same port.
  - a. Record an abort and exit test if Device Under Test fails to re-enumerate.
11. Send a GET\_CAPABILITY Command as defined in sections 3.2.1.1 and 4.2.7 and do nothing for 5 seconds.
  - a. Test fails if Device Under Test returns a PPM Response in that time. (5.2#2, 5.2.3#1)
12. If the Device Under Test does not have an **Accessible Connector**, then go to step 15.
13. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification.
  - a. Record an abort and exit test if this fails for any reason.
14. Prompt the user to change the attachment status (that is, detach the attached USB Type-C Device if there is one, attach one if there isn't) of an Accessible Connector on the Device Under Test, then do nothing for 5 seconds.
  - a. Test fails if Device Under Test sends an Asynchronous Notification to the Host. (5.2#2, 5.2#4)
15. Send a PPM RESET Command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - b. Test records an abort and exits if this fails for any reason.

## ***TD 4.12 PD Hard Reset Notification Test***

### **Device States for Test**

This test is run with the device in the configured state.

### **Equipment**

This test requires a QuadraMAX device. If the CV test machine has multiple USB hosts, the QuadraMAX can be run from it. Otherwise, an additional computer is required on which to run the QuadraMAX.

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

Run test for each **Accessible Connector**.

**[Initialization]**

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
3. Determine whether the Device Under Test **supports PPM functionality**.

- a. If it doesn't, then test records success and exits.
- 4. Perform **General PPM Initialization**, only set bit 0 and bit 7 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1 in step 2.
  - a. If any steps fail for any reason, then test records an abort and goes to step 19.
- 5. **Determine whether Power Delivery is supported** on the Connector Under Test.
  - a. If Power Delivery is not supported, then go to step 19.

**[Prompt user to attach Hard Reset Generator]**

- 6. Send a Get Connector Status as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the number of the Connector Under Test.
  - a. If any steps fail for any reason, then test records an abort and goes to step 19.
  - b. Test fails if Device Under Test does not return an asynchronous notification. (4.2.6#14)
  - c. Test fails if the *Connector Change Indicator* field of the Asynchronous Notification structure is not set to the connector number of the Connector Under Test. (4.1.2#3)
  - d. Test fails if the *Data Length* field of the Asynchronous Notification structure is not set to 00h. (4.1.2#7)
- 7. Determine whether a device is attached to the Connector Under Test, by checking the *Connection Status* field of the Get Connector Status Data structure.
- 8. Prompt the user to remove the device attached to the Connector Under Test.
- 9. Wait for the Device Under Test to send an Asynchronous Notification.
  - a. If no Asynchronous Notification arrives, then test records an abort and goes to step 19.
  - b. If the Asynchronous Notification is not a PPM notification, then test records an abort and goes to step 19.
  - c. If the Asynchronous Notification is for a different Connector than the Connector Under Test, then test records an abort and goes to step 19.
- 10. Prompt user to attach QuadraMAX device to Connector Under Test.
  - a. Test fails if Device Under Test does not return an asynchronous notification. (4.2.6#14)
  - b. Test fails if the *Connector Change Indicator* field of the Asynchronous Notification structure is not set to the connector number of the Connector Under Test. (4.1.2#3)
  - c. Test fails if the *Data Length* field of the Asynchronous Notification structure is not set to 00h. (4.1.2#7)

**[Hard Reset – notification on]**

- 11. Perform **General PPM Initialization**, only set bit 0 and bit 7 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1 in step 2.
  - a. If any steps fail for any reason, then test records an abort and goes to step 19.
- 12. Generate a PD Hard Reset from the QuadraMAX.
  - a. Test fails if Device Under Test does not return an asynchronous notification within 5 seconds. (4.2.6#11)
  - b. Test fails if the *Connector Change Indicator* field of the Asynchronous Notification structure is not set to the connector number of the Connector Under Test. (4.1.2#3)
  - c. Test fails if the *Data Length* field of the Asynchronous Notification structure is not set to 00h. (4.1.2#7)
- 13. Send a Get Connector Status Command as defined in sections 3.2.1.1 and 4.2.10 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Number* set to the number of the Connector Under Test.

- a. Test records an abort and goes to step 19 if the request fails for any reason.
- 14. Parse the Get Connector Status Response structure.
  - a. Test fails if bit 7 of the *Connector Status Change* field is not set to 1. (4.2.10#26)
- 15. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 0 and the *Command Completed Acknowledge* field set to 1.
- 16. Send an ACK\_PCR\_PAN Command as defined in sections 3.2.1.1 and 4.2.5 of the USB Type-C Bridge specification to the Device Under Test with the *Connector Change Acknowledge* field set to 1 and the *Command Completed Acknowledge* field set to 0.

#### **[Hard Reset – notification off]**

- 17. Perform **General PPM Initialization**.
- 18. Generate a PD Hard Reset from the QuadraMAX and wait 5 seconds.
  - a. Test fails if Device Under Test returns an asynchronous notification in that period. (4.2.6#11, 5.2#4)
- 19. Send a PPM RESET command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort if this fails for any reason.

## **TD 5.1 Type-C Authentication Test**

### **Device States for Test**

This test is run with the device in the configured state.

### **Equipment**

A Known Good USB PD Device that supports Type-C Authentication (**Authentication Device**).

A Known Good USB PD Device that does not support Type-C Authentication (**Not Supported Device**).

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

Run test for each **Accessible Connector**.

#### **[Initialization]**

- 1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.
- 2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
- 3. Determine whether the Device Under Test **supports PAM functionality**.
  - a. If it doesn't, then test records success and exits.
- 4. Prompt user to attach **Authentication Device** to Connector Under Test.

#### **[For All Authentication Requests]**

- 5. Attempt to read 5 bytes from the Notification Endpoint for all Auth Requests or Responses.
  - a. Test fails if it receives more than 4 bytes. (3.2.2.3#8)

#### **[GET\_DIGESTS/DIGEST]**

- 6. Use SEND\_PD\_DATA to send a GET\_DIGESTS request as defined in the Type-C Authentication Specification to the Authentication Device.
  - a. Test fails if this fails for any reason. In this case, proceed to step 9.

7. Read the GET\_DIGESTS notification from the DUT's Notification Endpoint.
  - a. Test fails if this fails for any reason. In this case, proceed to step 9. (3.2.2.3#1)
  - b. Test fails if *bNotificationType* is not equal to 0x02 (PD\_NOTIFICATION). (3.2.2.3#2)
  - c. Test fails if *bParam1* is not equal to the number of the Connector Under Test. (3.2.2.3#3)
  - d. Test fails if *bParam2* is not equal to 0x01 (AUTH). (3.2.2.3#4)
  - e. Test fails if *bParam3* is not set to 0x00 (Success). (3.2.2.3#5)
8. Use GET\_PD\_DATA to read the DIGESTS response.
  - a. Test fails if this fails for any reason. In this case, proceed to step 9. (3.2.2.2#1)

#### [GET\_CERTIFICATE/CERTIFICATE]

9. Use SEND\_PD\_DATA to send a GET\_CERTIFICATE request as defined in the Type-C Authentication Specification to the Authentication Device with slot = 0, offset = 0, length = 36.
  - a. Test fails if request fails for any reason. In this case, proceed to step 12.
10. Read the GET\_CERTIFICATE notification from the DUT's Notification Endpoint.
  - a. Test fails if this fails for any reason. In this case, proceed to step 12. (3.2.2.3#1)
  - b. Test fails if *bNotificationType* is not equal to 0x02 (PD\_NOTIFICATION). (3.2.2.3#2)
  - c. Test fails if *bParam1* is not equal to the number of the Connector Under Test. (3.2.2.3#3)
  - d. Test fails if *bParam2* is not equal to 0x01 (AUTH). (3.2.2.3#4)
  - e. Test fails if *bParam3* is not set to 0x00 (Success). (3.2.2.3#5)
11. Use GET\_PD\_DATA to read first 40 bytes of the CERTIFICATE response.
  - a. Test fails if this fails for any reason. In this case, proceed to step 12. (3.2.2.2#1)

#### [CHALLENGE/CHALLENGE\_AUTH]

12. Use SEND\_PD\_DATA to send a CHALLENGE request as defined in the Type-C Authentication Specification to the Authentication Device with slot = 0, and a randomly generated Nonce.
  - a. Test fails if request fails for any reason. In this case, proceed to step 15.
13. Read the CHALLENGE notification from the DUT's Notification Endpoint.
  - a. Test fails if this fails for any reason. In this case, proceed to step 15. (3.2.2.3#1)
  - b. Test fails if *bNotificationType* is not equal to 0x02 (PD\_NOTIFICATION). (3.2.2.3#2)
  - c. Test fails if *bParam1* is not equal to the number of the Connector Under Test. (3.2.2.3#3)
  - d. Test fails if *bParam2* is not equal to 0x01 (AUTH). (3.2.2.3#4)
  - e. Test fails if *bParam3* is not set to 0x00 (Success). (3.2.2.3#5)
14. Use GET\_PD\_DATA to read the CHALLENGE\_AUTH response.
  - a. Test fails if this fails for any reason. In this case, proceed to step 15. (3.2.2.2#1)

#### [Invalid request/ERROR]

15. Use SEND\_PD\_DATA to send an Authentication request with message type of 0x84 as defined in the Type-C Authentication Specification to the Authentication Device.
  - a. Test fails if request fails for any reason. In this case, proceed to step 18.
16. Read the notification from the DUT's Notification Endpoint.
  - a. Test fails if this fails for any reason. In this case, proceed to step 18. (3.2.2.3#1)
  - b. Test fails if *bNotificationType* is not equal to 0x02 (PD\_NOTIFICATION). (3.2.2.3#2)
  - c. Test fails if *bParam1* is not equal to the number of the Connector Under Test. (3.2.2.3#3)
  - d. Test fails if *bParam2* is not equal to 0x01 (AUTH). (3.2.2.3#4)
  - e. Test fails if *bParam3* is not set to 0x00 (Success). (3.2.2.3#5)
17. Use GET\_PD\_DATA to read the response.
  - a. Test fails if this fails for any reason. In this case, proceed to step 18. (3.2.2.2#1)

**[Auth Not Supported]**

18. Prompt user to detach **Authentication Device** and attach the **Not Supported Device** to the same connector.
19. Use SEND\_PD\_DATA to send a GET\_DIGESTS request as defined in the Type-C Authentication Specification to the Authentication Device.
  - a. Test fails if this fails for any reason. In this case, proceed to step 21.
20. Read the GET\_DIGESTS notification from the DUT's Notification Endpoint.
  - a. Test fails if this fails for any reason. In this case, proceed to step 21.
  - b. Test fails if *bNotificationType* is not equal to 0x02 (PD\_NOTIFICATION). (3.2.2.3#2)
  - c. Test fails if *bParam1* is not equal to the number of the Connector Under Test. (3.2.2.3#3)
  - d. Test fails if *bParam2* is not equal to 0x01 (AUTH). (3.2.2.3#4)
  - e. Test fails if *bParam3* is not set to 0x01 (Not Supported). (3.2.2.3#7)
21. Use SEND\_PD\_DATA to send a GET\_CERTIFICATE request as defined in the Type-C Authentication Specification to the Authentication Device.
  - a. Test fails if this fails for any reason. In this case, proceed to step 23.
22. Read the GET\_CERTIFICATE notification from the DUT's Notification Endpoint.
  - a. Test fails if this fails for any reason. In this case, proceed to step 23. (3.2.2.3#1)
  - b. Test fails if *bNotificationType* is not equal to 0x02 (PD\_NOTIFICATION). (3.2.2.3#2)
  - c. Test fails if *bParam1* is not equal to the number of the Connector Under Test. (3.2.2.3#3)
  - d. Test fails if *bParam2* is not equal to 0x01 (AUTH). (3.2.2.3#4)
  - e. Test fails if *bParam3* is not set to 0x01 (Not Supported). (3.2.2.3#7)
23. Use SEND\_PD\_DATA to send a CHALLENGE request as defined in the Type-C Authentication Specification to the Authentication Device.
  - a. Test fails if this fails for any reason. In this case, proceed to step 25.
24. Read the CHALLENGE notification from the DUT's Notification Endpoint.
  - a. Test fails if this fails for any reason. In this case, proceed to step 25.
  - b. Test fails if *bNotificationType* is not equal to 0x02 (PD\_NOTIFICATION). (3.2.2.3#2)
  - c. Test fails if *bParam1* is not equal to the number of the Connector Under Test. (3.2.2.3#3)
  - d. Test fails if *bParam2* is not equal to 0x01 (AUTH). (3.2.2.3#4)
  - e. Test fails if *bParam3* is not set to 0x01 (Not Supported). (3.2.2.3#7)

## **TD 5.2 Remote Wake Test**

### **Device States for Test**

This test is run with the device in the configured state.

### **Equipment**

A Known Good Device with a Type-C Upstream Facing Port.

### **Repetitions**

Run this test for all Configurations the Device Under Test supports.

Run test for each **Accessible Connector**.

Run this test for the following Events:

Connect/Disconnect Change (SET\_NOTIFICATION\_ENABLE bit 14)

PD Hard Reset (SET\_NOTIFICATION\_ENABLE bit 7).

For each Event, run this test with the Event enabled and disabled.

**[Initialization]**

1. Perform **General Test Initialization**.
  - a. Test records an abort and exits if there are any failures.



- b. If bit 5 (Remote Wakeup) of the *bmAttributes* field of the Configuration Descriptor is not set to 1, then test records a failure and exits. (5.8.1#1)
- 2. Perform **USB Type-C Interface Validation**.
  - a. Test records an abort and exits if validation fails for any reason.
- 3. Determine whether the Device Under Test **supports PPM functionality**.
  - a. If it doesn't, then test records success and exits.
- 4. If the Event Type is Hard Reset, then **Determine whether Power Delivery is supported** on the Connector Under Test.
  - a. If it is not supported, then test records success and goes to step 14.

#### [Initialize Remote Wake]

- 5. Perform **General PPM Initialization**:
  - a. If this is an Event enabled test, set bit 0 and the appropriate bit for the Event type of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1 in step 2.
    - i. If any steps fail for any reason, then test records an abort and goes to step 14.
  - b. If this is an Event Disabled test, set bit 0 of the *Notification Enable* field of the SET\_NOTIFICATION\_ENABLE Command to 1 in step 2.
    - i. If any steps fail for any reason, then test records an abort and goes to step 14.
- 6. Enable Remote Wake for the DUT as a USB Device.
  - a. If Device Under Test is operating as a USB 2.0 device, refer to TD 9.12 in the **Universal Serial Bus Revision 2.0 USB Command Verifier Compliance Test Specification**.
    - i. If any steps fail for any reason, then test records an abort and goes to step 14.
  - b. If Device Under Test is operating as a USB 3.2 device enable Remote Wake for the Bridge Function. Refer to TD 9.15 in the **Universal Serial Bus Revision 3.2 USB Command Verifier Compliance Test Specification**.
    - i. If any steps fail for any reason, then test records an abort and goes to step 14.
- 7. Suspend the DUT as a USB Device
  - a. If Device Under Test is operating as a USB 2.0 device, refer to TD 9.12 in the **Universal Serial Bus Revision 2.0 USB Command Verifier Compliance Test Specification**.
    - i. If any steps fail for any reason, then test records an abort and goes to step 14.
  - b. If Device Under Test is operating as a USB 3.2 device, then suspend the Bridge Function. Refer to TD 9.15 in the **Universal Serial Bus Revision 3.2 USB Command Verifier Compliance Test Specification**.
    - i. If any steps fail for any reason, then test records an abort and goes to step 14.

#### [Perform Remote Wake]

- 8. Prompt user to generate a Remote Wake event by generating a Hard Reset.
- 9. Wait for Remote Wake to complete:
  - a. If Device Under Test is operating as a USB 2.0 device, refer to TD 9.12 in the **Universal Serial Bus Revision 2.0 USB Command Verifier Compliance Test Specification**.
    - i. If any steps fail for any reason, then test records an abort and goes to step 14.
  - b. If Device Under Test is operating as a USB 3.2 device, refer to TD 9.15 in the **Universal Serial Bus Revision 3.2 USB Command Verifier Compliance Test Specification**.
    - i. If any steps fail for any reason, then test records an abort and goes to step 14.
- 10. If the Event is enabled, then test fails if Remote Wake does not take place. (5.8.1#2)
- 11. If the Event is disabled, then test fails if Remote Wake takes place. (5.8.1#2)
- 12. Wait for the Device Under Test to send an Asynchronous Notification for the Event.
  - a. If the Event is enabled, then test fails if an Asynchronous Notification is not received. (5.8.1#2)



- b. If the Event is disabled, then test fails if an Asynchronous Notification is received. (5.8.1#2)
- 13. If Event is enabled and an Asynchronous Notification is received, examine it:
  - a. Test fails if the *Connector Number* field is not set to the number of the Connector Under Test. (4.1.2#3)
- 14. Send a PPM RESET Command as defined in sections 3.2.1.1 and 4.2.2 of the USB Type-C Bridge specification to the Device Under Test.
  - a. Test records an abort if this fails for any reason.

## **Appendix**

### ***Changes in Revision 0.90***

1. Initial Release.

### ***Changes in Revision 1.00***

1. Added TD 4.8 Get PDOs Command test. Renumbered following tests appropriately.
2. Added TD 5.2 Remote Wake test.
3. Removed TD 4.13.
4. Editorial fixes.

### ***Changes in Revision 1.01***

1. Added two new assertions: 5.7#8 and 5.7#9.

### ***Changes in Revision 1.02***

1. From several tests, removed consistency checks (that is, places where we query the user as to the value of a field in a descriptor or Command response and fail the test if they get it wrong). We determined that these checks were not useful, as the only thing they confirmed was whether the tester knew ahead of time what the values were supposed to be.
2. Consolidate steps for determining if a device supports PD into a single description. Update the tests that need to know this accordingly.