

MHL™ **(Mobile High-definition Link)**

Compliance Test Specification

Main Required Methods

Confidential information of:

Nokia Corporation,
Samsung Electronics Corporation,
Silicon Image, Inc.,
Sony Corporation,
Toshiba Corporation,
each, an “MHL Promoter” and collectively, the “MHL Promoters”.

Revision 2.1

Release Candidate fc02

MHL, LLC

April 2013

Notice

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, NO WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION, OR SAMPLE.

Nokia Corporation, Samsung Corporation, Silicon Image, Inc., Sony Corporation, Toshiba Corporation and **MHL, LLC** disclaim all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification.

Copyright © 2001-2013 by one or more of the MHL Promoters. All rights reserved. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein. Unauthorized use or duplication prohibited. "MHL" and all associated logos are trademarks of **MHL, LLC**. Third-party trademarks and servicemarks are property of their respective owners.

Document Revision History

| | |
|---------|---------------------------------------|
| 2010-11 | Distributed "December 2010" revision. |
| 2011-02 | Distributed "February 2011" revision. |
| 2011-06 | Distributed Revision 1.1. |
| 2012-01 | Distributed Revision 1.2. |
| 2012-08 | Distributed Revision 2.0. |
| 2013-04 | Distributed Revision 2.1. |

Please refer also to the Revision Highlights section, beginning on Section 2.6 on page 30.

1 Introduction

This document contains a version of the MHL Compliance Test Specification, as such term is defined in the Mobile High-Definition Link Specification Adopter Agreement. A product's satisfaction of the criteria and tests set forth herein do not guarantee that such product is Fully Compliant, as such term is defined in the Mobile High-Definition Link Specification Adopter Agreement. This version of the MHL Compliance Test Specification shall be effective until a superseding version is made available by the MHL Promoters to the MHL Adopters. In the event that an Authorized Testing Center has tested and certified a product by performing the criteria and tests set forth herein and the product is later reported to the Agent as not being Fully Compliant, then the respective Promoter or Adopter shall, in a timely manner following the date it is notified by the Agent of such report, resubmit the product to an Authorized Testing Center for re-testing and re-certification.

Abbreviated Table of Contents

| Section | Title | Page |
|---------|--|------|
| 1 | Introduction | 1 |
| 2 | Background | 12 |
| 3 | Source Test | 31 |
| 4 | Sink Test | 114 |
| 5 | Dongle Test | 191 |
| 6 | Tests Common to Sources, Sinks and Dongles | 269 |
| 7 | Cable Test | 366 |
| 8 | Appendices | 389 |

Table of Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 3 |
| 2 | Background | 12 |
| 2.1 | Formatting of this Document | 12 |
| 2.1.1 | Formatting of Tests | 12 |
| 2.1.2 | Symbolic Terms Used in Test Definitions | 12 |
| 2.1.3 | Test Descriptions | 13 |
| 2.1.4 | Capability Declaration Form and Test Results Form | 13 |
| 2.1.5 | Usages and Conventions | 13 |
| 2.2 | Test Results | 13 |
| 2.2.1 | Definitions of Test Result Terminology | 14 |
| 2.2.2 | Requirement for Successful Completion of Compliance Testing | 14 |
| 2.2.3 | Test Flow Usage of PASS, FAIL and PASS (SKIP) | 15 |
| 2.3 | Test Equipment | 15 |
| 2.3.1 | TMDS Test Equipment | 16 |
| 2.3.2 | MHL System-level Equipment | 18 |
| 2.3.3 | CBUS Equipment | 19 |
| 2.3.4 | MSC Test Tool | 22 |
| 2.3.5 | HDCP Equipment | 22 |
| 2.3.6 | Adapters | 22 |
| 2.3.7 | Miscellaneous | 25 |
| 2.4 | Required Video and Audio Patterns | 26 |
| 2.4.1 | Required Video Patterns | 26 |
| 2.4.2 | Required Audio Patterns | 28 |
| 2.5 | Measurement Methods | 29 |
| 2.5.1 | CBUS Electrical Measurements | 29 |
| 2.6 | Revision History | 30 |
| 3 | Source Test | 31 |
| 3.1 | Electrical Tests | 31 |
| 3.1.1 | TMDS Electrical Tests | 31 |
| 3.2 | System Tests | 50 |
| 3.2.2 | TMDS Coding | 50 |
| 3.2.3 | Video Modes | 57 |
| 3.2.4 | Audio Test | 62 |
| 3.2.5 | HDCP Test | 64 |
| 3.2.6 | EDID Test and Device Capability / Device Status Register Test | 65 |
| 3.2.7 | RCP Sub-Command Tests | 68 |
| 3.2.8 | RAP Sub-command Tests | 69 |
| 3.2.9 | 3D Video Test | 70 |
| 3.2.10 | UCP Sub-Command Tests | 73 |
| 3.3 | CBUS Tests | 74 |
| 3.3.1 | CBUS Source DUT Common Test Equipment Setups | 75 |
| 3.3.2 | CBUS Source DUT Common Methodologies | 75 |
| 3.3.3 | Link Layer Electrical – Source: Absolute Maximum Voltages | 79 |
| 3.3.4 | Link Layer Timing – Source DUT Output: Pre-Discovery | 80 |

| | | |
|----------|---|------------|
| 3.3.5 | Link Layer Electrical – Source DUT Output: Discovery | 80 |
| 3.3.6 | Link Layer Timing – Source DUT Output: Discovery | 85 |
| 3.3.7 | Link Layer Electrical – Source DUT Output: Arbitration/Sync/Data Signaling | 88 |
| 3.3.8 | Link Layer Timing – Source DUT Output: Arbitration/Sync/Data in Nanoseconds | 91 |
| 3.3.9 | Link Layer Timing – Source DUT Output: Arbitration/Sync/Data in Bit Times | 93 |
| 3.3.10 | Link Layer Timing – Source DUT Output: Link-Level NACK | 95 |
| 3.3.11 | Link Layer Timing – Source DUT Output: ACK | 96 |
| 3.3.12 | Link Layer Timing – Source DUT Output: Bus Re-Arbitration | 98 |
| 3.3.13 | Link Layer Behavior – Source DUT Output: Ill-formed packets | 100 |
| 3.3.14 | Link Layer Timing – Source DUT Input: Discovery | 101 |
| 3.3.15 | Link Layer Electrical – Source DUT Input: Arbitration/Sync/Data signaling | 104 |
| 3.3.16 | Link Layer Timing – Source DUT Input: Arbitration | 105 |
| 3.3.17 | Link Layer Timing – Source DUT Input: Data | 106 |
| 3.3.18 | Link Layer Timing – Source DUT Input: NACK | 108 |
| 3.3.19 | Link Layer Timing – Source DUT Input: ACK | 109 |
| 3.3.20 | Link Layer Timing – Source DUT Input: Bus Re-Arbitration | 110 |
| 3.3.21 | Link Layer Behavior – Source DUT Input: Ill-formed packets | 111 |
| 3.3.22 | Link Layer Timing – Source DUT Input: Disconnect | 112 |
| 3.3.23 | Link Layer Electrical – Source DUT VBUS Control | 114 |
| 4 | Sink Test | 118 |
| 4.1 | Electrical Tests | 118 |
| 4.1.1 | TMDS Electrical Tests | 118 |
| 4.2 | System Tests | 133 |
| 4.2.1 | TMDS Coding Tests | 133 |
| 4.2.2 | Video Test | 135 |
| 4.2.3 | Audio Test | 138 |
| 4.2.4 | HDCP Test | 139 |
| 4.2.5 | EDID Test and Device Capability / Device Status Register Test | 140 |
| 4.2.6 | RCP Sub-Commands Tests | 142 |
| 4.2.7 | RAP Sub-Commands Test | 144 |
| 4.2.8 | 3D Video Test | 145 |
| 4.2.9 | UCP Sub-Command Tests | 147 |
| 4.3 | CBUS Tests | 148 |
| 4.3.1 | CBUS Sink DUT Common Test Equipment Setups | 148 |
| 4.3.2 | CBUS Sink DUT Common Required Methodologies | 149 |
| 4.3.3 | Link Layer Electrical – Sink: Absolute Maximum Voltages | 150 |
| 4.3.4 | Link Layer Electrical – Sink DUT Output: Standby Discovery Impedance | 151 |
| 4.3.5 | Link Layer Timing – Sink DUT Output: Pre-Discovery | 152 |
| 4.3.6 | Link Layer Electrical – Sink DUT Output: Arbitration/Sync/Data Signaling | 153 |
| 4.3.7 | Link Layer Timing – Sink DUT Output: Arbitration/Sync/Data in Nanoseconds | 155 |
| 4.3.8 | Link Layer Timing – Sink DUT Output: Arbitration/Sync/Data in Bit Times | 158 |
| 4.3.9 | Link Layer Timing – Sink DUT Output: Link Level NACK | 160 |
| 4.3.10 | Link Layer Timing – Sink DUT Output: Link Level ACK | 161 |
| 4.3.11 | Link Layer Timing – Sink DUT Output: Bus Re-Arbitration | 163 |
| 4.3.12 | Link Layer Timing – Sink DUT Output: Ill-formed packets | 166 |
| 4.3.13 | Link Layer Electrical – Sink DUT Input: Discovery | 167 |
| 4.3.14 | Link Layer Timing – Sink DUT Input: Discovery OK | 167 |
| 4.3.15 | Link Layer Timing – Sink DUT Input: Discovery Reject | 174 |
| 4.3.16 | Link Layer Electrical – Sink DUT Input: Arbitration/Sync/Data Signaling | 179 |

| | | |
|----------|---|------------|
| 4.3.17 | Link Layer Timing – Sink DUT Input: Arbitration | 180 |
| 4.3.18 | Link Layer Timing – Sink DUT Input: Data | 182 |
| 4.3.19 | Link Layer Timing – Sink DUT Input: NACK..... | 183 |
| 4.3.20 | Link Layer Timing – Sink DUT Input: ACK | 184 |
| 4.3.21 | Link Layer Timing – Sink DUT Input: Bus Re-Arbitration | 185 |
| 4.3.22 | Link Layer Timing – Sink DUT Input: Ill-formed Packets..... | 186 |
| 4.3.23 | Link Layer Timing – Sink DUT Input: Disconnect..... | 187 |
| 4.3.24 | Link Layer Electrical – Sink DUT VBUS Output | 189 |
| 4.3.25 | Link Layer Timing – Sink DUT VBUS Turn On Transition | 190 |
| 5 | Dongle Test | 191 |
| 5.1 | Electrical Tests..... | 191 |
| 5.1.1 | TMDS Electrical Tests..... | 191 |
| 5.2 | System Tests..... | 207 |
| 5.2.1 | TMDS Coding | 207 |
| 5.2.2 | Video Tests | 210 |
| 5.2.3 | Audio Test..... | 213 |
| 5.2.4 | HDCP Test | 214 |
| 5.2.5 | EDID Test and Device Capability / Device Status Register Test..... | 214 |
| 5.2.6 | RCP Sub-Commands Tests | 217 |
| 5.2.7 | RAP Sub-Commands Test..... | 218 |
| 5.2.8 | 3D Video Test..... | 219 |
| 5.2.9 | UCP Sub-Command Tests | 221 |
| 5.3 | CBUS Tests..... | 223 |
| 5.3.1 | CBUS Dongle DUT Common Test Equipment Setups | 223 |
| 5.3.2 | CBUS Dongle DUT Common Required Methodologies | 223 |
| 5.3.3 | Link Layer Electrical – Dongle: Absolute Maximum Voltages | 225 |
| 5.3.4 | Link Layer Electrical – Dongle DUT Output: Discovery Impedance | 226 |
| 5.3.5 | Link Layer Timing – Dongle DUT Output: Pre-Discovery..... | 229 |
| 5.3.6 | Link Layer Electrical – Dongle DUT Output: Arbitration/Sync/Data Signaling..... | 230 |
| 5.3.7 | Link Layer Timing – Dongle DUT Output: Arbitration/Sync/Data in Nanoseconds | 232 |
| 5.3.8 | Link Layer Timing – Dongle DUT Output: Arbitration/Sync/Data in Bit Times | 234 |
| 5.3.9 | Link Layer Timing – Dongle DUT Output: Link Level NACK | 237 |
| 5.3.10 | Link Layer Timing – Dongle DUT Output: Link Level ACK | 238 |
| 5.3.11 | Link Layer Timing – Dongle DUT Output: Bus Re-Arbitration..... | 240 |
| 5.3.12 | Link Layer Timing – Dongle DUT Output: Ill-formed packets..... | 242 |
| 5.3.13 | Link Layer Electrical – Dongle DUT Input: Discovery..... | 243 |
| 5.3.14 | Link Layer Timing – Dongle DUT Input: Discovery OK..... | 244 |
| 5.3.15 | Link Layer Timing – Dongle DUT Input: Discovery Reject | 250 |
| 5.3.16 | Link Layer Electrical – Dongle DUT Input: Arbitration/Sync/Data Signaling | 256 |
| 5.3.17 | Link Layer Timing – Dongle DUT Input: Arbitration | 256 |
| 5.3.18 | Link Layer Timing – Dongle DUT Input: Data | 259 |
| 5.3.19 | Link Layer Timing – Dongle DUT Input: NACK..... | 261 |
| 5.3.20 | Link Layer Timing – Dongle DUT Input: ACK | 262 |
| 5.3.21 | Link Layer Timing – Dongle DUT Input: Bus Re-Arbitration | 263 |
| 5.3.22 | Link Layer Timing – Dongle DUT Input: Ill-formed Packets..... | 264 |
| 5.3.23 | Link Layer Timing – Dongle DUT Input: Disconnect | 265 |
| 5.3.24 | Link Layer Electrical – Dongle DUT VBUS Output..... | 267 |
| 5.3.25 | Link Layer Electrical – Dongle DUT VBUS Input | 268 |
| 5.3.26 | Link Layer Timing – Dongle DUT VBUS Transition..... | 269 |

| | | |
|----------|---|------------|
| 6 | Tests Common to Sources, Sinks and Dongles | 269 |
| 6.1 | Electrical Tests..... | 270 |
| 6.2 | System Tests..... | 270 |
| 6.3 | CBUS Tests..... | 270 |
| 6.3.1 | MSC – Source and Sink DUT Input: Device Register Space Contents; Reads | 270 |
| 6.3.2 | MSC – Source and Sink DUT Output: Vendor-specific and Reserved Header Values..... | 271 |
| 6.3.3 | MSC – Source and Sink DUT Output: Normal Commands | 272 |
| 6.3.4 | MSC – Source and Sink DUT Output: NACK Packet Response to MSC_MSG | 280 |
| 6.3.5 | MSC – Source and Sink DUT Output: Never Initiates Bad Commands | 281 |
| 6.3.6 | MSC – Source and Sink DUT Output: Errors and Exceptions..... | 286 |
| 6.3.7 | MSC – Source and Sink DUT Output; Disconnect..... | 291 |
| 6.3.8 | MSC – Source and Sink DUT Input: Device Register Space Contents; Writes | 292 |
| 6.3.9 | MSC – Source and Sink DUT Input: Vendor-specific and Reserved Header Values..... | 294 |
| 6.3.10 | MSC – Source and Sink DUT Input: Normal Commands | 295 |
| 6.3.11 | MSC – Source and Sink DUT Input: Errors and Exceptions | 303 |
| 6.3.12 | MSC – Source and Sink DUT Input: Argument Timeouts | 324 |
| 6.3.13 | MSC – Source DUT Output: Never Initiates Bad Commands | 332 |
| 6.3.14 | MSC – Source DUT Input: Normal Commands..... | 332 |
| 6.3.15 | MSC – Sink DUT Output: Normal Commands | 334 |
| 6.3.16 | MSC – Sink DUT Input: Errors and Exceptions | 335 |
| 6.3.17 | DDC – Source DUT Output; DUT Never Sends Illegal DDC Command..... | 337 |
| 6.3.18 | DDC – Source DUT Output; Normal Operation | 338 |
| 6.3.19 | DDC – Source DUT Output; Illegal Responses..... | 346 |
| 6.3.20 | DDC – Sink DUT Input; Continuous Monitors and Normal Operation | 353 |
| 6.3.21 | DDC – Sink DUT Input; Normal Operation | 355 |
| 6.3.22 | DDC – Sink DUT Input; Illegal Responses | 363 |
| 7 | Cable Test..... | 366 |
| 7.1 | Mechanical tests | 366 |
| 7.1.1 | Cable Assembly Mechanical Tests | 366 |
| 7.2 | Electrical Tests..... | 367 |
| 7.2.1 | Cable Assembly Electrical Tests | 367 |
| 8 | Appendices..... | 389 |
| 8.1 | Capability Declaration Form (CDF) | 389 |
| 8.2 | Test Results Form (TRF)..... | 391 |

List of Figures

| | |
|--|-----|
| Figure 1. Flowchart for CTS Test Flow | 15 |
| Figure 2. Recommended MHL Signal Generator and Adapter for Dongle Test. | 18 |
| Figure 3. Recommended MHL Signal Generator and Adapter for Sink Test. | 19 |
| Figure 4. MHL Test Cable | 19 |
| Figure 5. CBUS Source Board | 20 |
| Figure 6. CBUS Sink Board..... | 20 |
| Figure 7. MHL Direct Attach Source Extension Cable..... | 21 |
| Figure 8. MHL Direct Attach Sink Extension Cable | 21 |
| Figure 9. MSC Test Tool | 22 |
| Figure 10. MHL HDCP Analyzer | 22 |
| Figure 11. MHL HDCP Emulator | 22 |
| Figure 12. TPA Board with Termination..... | 23 |
| Figure 13. TPA Board without Termination | 23 |
| Figure 14. TPA Board for Source RxSense Test (Source TPA-RSEN) | 24 |
| Figure 15. TPA Board for Sink and Dongle RxSense Test (Sink/Dongle TPA-RSEN) | 25 |
| Figure 16. Gray Ramp Color Derivation | 26 |
| Figure 17. Representative Gray Ramp Video Patterns..... | 26 |
| Figure 18. Chess Board Video Pattern | 27 |
| Figure 19. CBUS Voltage Level Waveform | 29 |
| Figure 20. Source – V_{OFF} – Test Setup | 31 |
| Figure 21. Source – V_{SE_HIGH} Test Setup | 33 |
| Figure 22. Source – V_{SE_LOW} Test Setup | 34 |
| Figure 23. Source – $V_{DFSWING}$ Test Setup | 35 |
| Figure 24. Source – $V_{CMSWING}$ Test Setup | 36 |
| Figure 25. Source – T_{R_DF} and T_{F_DF} Test Setup | 38 |
| Figure 26. Source – T_{R_CM} and T_{F_CM} Test Setup | 39 |
| Figure 29. Source – TP2 Eye Diagram Test Setup..... | 47 |
| Figure 30. Source – CBUS Setup #2 | 78 |
| Figure 31. Single-Ended Signals Calibration Setup for Sink Tests..... | 118 |
| Figure 32. Differential & Common-Mode Signals Calibration Setup for Sink Tests..... | 118 |
| Figure 33. Sink – Input Signal DC Voltage Level Test Setup | 119 |
| Figure 34. Sink – Input Signal Min and Max Swing Voltage Tolerance Test Setup | 121 |
| Figure 35. Sink – Intra-Pair Skew Tolerance Test Setup | 122 |
| Figure 36. Sink – Jitter Tolerance Test Setup | 124 |
| Figure 37. Sink – Differential Impedance Test Setup | 125 |
| Figure 38. Set reference point for differential mode | 126 |
| Figure 39. Sink – Common-mode Impedance Test Setup | 128 |
| Figure 40. Set reference point for common-mode | 128 |
| Figure 41. Sink – RxSense Impedance Test Setup | 130 |
| Figure 42. Single-Ended Signals Calibration Setup for Dongle Tests | 191 |
| Figure 43. Differential & Common-Mode Signals Calibration Setup for Dongle Tests | 191 |
| Figure 44. Dongle – V_{SE_HIGH} and V_{SE_LOW} Tolerance Test Setup | 193 |
| Figure 45. Dongle – $V_{DFSWING}$ and $V_{CMSWING}$ Tolerance Test Setup..... | 194 |
| Figure 46. Dongle – T_{SKEW_DF} and T_{SKEW_CM} Tolerance Test Setup | 196 |
| Figure 47. Dongle – Jitter Tolerance Test Setup | 197 |
| Figure 48. Dongle – Differential Impedance Test Setup | 198 |
| Figure 49. Set reference point for differential mode | 199 |
| Figure 50. Dongle – Differential Impedance Test Symbolology | 199 |

| | |
|--|-----|
| Figure 51. Set reference point for common mode | 202 |
| Figure 52. Dongle – Common-mode Impedance Test Setup | 202 |
| Figure 53. Dongle – RxSense Impedance Test Setup | 204 |
| Figure 54. Dongle – RxSense Impedance Test Setup | 205 |
| Figure 55. Differential Intra-Pair Skew Test Setup | 368 |
| Figure 56. Common-mode Intra-Pair Skew Test Setup | 369 |
| Figure 57. Illustration of Cable Length for the Impedance Test | 370 |
| Figure 58. Differential Characteristic Impedance Test Setup | 370 |
| Figure 59. Set reference point for differential mode | 371 |
| Figure 60. Common-Mode Characteristic Impedance Test Setup | 373 |
| Figure 61. Set reference point for common mode | 373 |
| Figure 62. CBUS Line Capacitance Test Setup | 375 |
| Figure 63. CBUS Cable Delay Test Setup | 376 |
| Figure 64. Differential Insertion Loss Test Setup | 377 |
| Figure 65. Common-mode Insertion Loss Test Setup | 378 |
| Figure 66. Differential and Common-mode Conversion Test Setup | 379 |
| Figure 67. CBUS Insertion Loss Test Setup | 380 |
| Figure 68. Far-End Crosstalk Test Setup | 381 |
| Figure 69. Cable VBUS Drop Test Setup | 382 |
| Figure 70. Cable Ground Resistance and Ground Shift Test Setup | 383 |
| Figure 71. Cable Detect Circuit for MHL Cable with HDMI Type A Plug | 384 |
| Figure 72. Cable Detect for MHL Cable with HDMI Type A Plug Test Setup | 384 |
| Figure 73 Single-Ended Signals Calibration Setup for Cable Tests | 385 |
| Figure 74 Differential & Common-Mode Signals Calibration Setup for Cable Tests | 386 |
| Figure 75. Cable – Minimum Clock Swing Test Setup | 386 |

List of Tables

| | |
|--|-----|
| Table 2-1. MHL TPA Boards Description | 23 |
| Table 3-1. Tester Sink Device Capability Registers..... | 78 |
| Table 3-2. Response to Initial Plug-in to MHL Device | 81 |
| Table 3-3. Response to Sink Priming Pulse to MHL Device | 83 |
| Table 3-4. Pre-Discovery Success Pull-up HIGH Voltage | 84 |
| Table 3-5. Discovery Pulse Drive HIGH Voltage | 85 |
| Table 3-6. Discovery Pulse Float LOW Voltage..... | 85 |
| Table 3-7. Wake/Discovery Pulse Edge Rate..... | 88 |
| Table 3-8. Arbitration/Sync/Data Edge Rate..... | 93 |
| Table 3-9. ACK Output Timing in Nanoseconds | 97 |
| Table 3-10. ACK Drive HIGH Duration | 98 |
| Table 3-11. Discovery – Sink Responds Correctly – Time to Source Pull-up Change..... | 102 |
| Table 3-12. Discovery – Sink Responds Late | 103 |
| Table 3-13. Discovery – Sink Never Drives MHL+/- HIGH..... | 104 |
| Table 3-14. Sensitivity to VIH/VIL{min} | 104 |
| Table 3-15. Source Behaves Appropriately when Sink Wins Arbitration | 105 |
| Table 3-16. Bit Timing Sensitivity | 107 |
| Table 3-17. Check Response to Bad Parity Packet | 109 |
| Table 3-18. Response to ACKs with Different Timings | 110 |
| Table 3-19. Source Receives Back-to-Back Transfers | 111 |
| Table 3-20. Tester Sink Emulation | 117 |
| Table 4-1. Tester Source Device Capability Registers | 150 |
| Table 4-2. Standby $Z_{CBUS_SINK_DISCOVER}$ | 152 |
| Table 4-3. Arbitration/Sync/Data Edge Rate..... | 157 |
| Table 4-4. ACK Output Timing in Nanoseconds | 162 |
| Table 4-5. ACK Drive HIGH Duration. | 163 |
| Table 4-6. Sensitivity to VIH/VIL..... | 167 |
| Table 4-7. Wake Pulse Timing 1 | 169 |
| Table 4-8. Wake Pulse Timing 2 | 169 |
| Table 4-9. Discovery Pulse Timing 1..... | 171 |
| Table 4-10. Discovery Pulse Timing 2..... | 172 |
| Table 4-11. Discovery Pulse Timing 3..... | 172 |
| Table 4-12. Discovery Pulse Reject Timing 1..... | 175 |
| Table 4-13. Discovery Pulse Reject Timing 2..... | 176 |
| Table 4-14. Discovery Pulse Reject Timing 1..... | 178 |
| Table 4-15. Discovery Pulse Reject Timing 2..... | 179 |
| Table 4-16. Sensitivity to VIH/VIL..... | 180 |
| Table 4-17. End of Discovery to Early Source-side Arbitration | 181 |
| Table 4-18. Loses Arbitration Collision Correctly | 182 |
| Table 4-19. Bit Timing Sensitivity | 183 |
| Table 4-20. Check Response to Bad Parity Packet | 184 |
| Table 4-21. Response to ACKs with Different Timings | 185 |
| Table 4-22. Sink Receives Back-to-Back Transfers | 186 |
| Table 5-1. Tester Source Device Capability Registers | 225 |
| Table 5-2. Powered-Off $Z_{CBUS_DONGLE_DISCOVER}$ | 227 |
| Table 5-3. VBUS-Powered $Z_{CBUS_DONGLE_DISCOVER}$ | 228 |
| Table 5-4. Self-Powered $Z_{CBUS_DONGLE_DISCOVER}$ | 229 |
| Table 5-5. Arbitration/Sync/Data Edge Rate..... | 234 |

| | |
|---|-----|
| Table 5-6. ACK Output Timing in Nanoseconds | 239 |
| Table 5-7. ACK Drive HIGH Duration. | 240 |
| Table 5-8. Sensitivity to VIH/VIL..... | 244 |
| Table 5-9. Wake Pulse Timing 1 | 245 |
| Table 5-10. Wake Pulse Timing 2 | 246 |
| Table 5-11. Discovery Pulse Timing 1..... | 248 |
| Table 5-12. Discovery Pulse Timing 2..... | 248 |
| Table 5-13. Discovery Pulse Timing 3..... | 249 |
| Table 5-14. Discovery Pulse Reject Timing 1..... | 251 |
| Table 5-15. Discovery Pulse Reject Timing 2..... | 252 |
| Table 5-16. Discovery Pulse Reject Timing 1..... | 254 |
| Table 5-17. Discovery Pulse Reject Timing 2..... | 255 |
| Table 5-18. Sensitivity to VIH/VIL..... | 256 |
| Table 5-19. Dongle Behaves Appropriately when Dongle Loses Arbitration | 257 |
| Table 5-20. End of Discovery to Early Source-side Arbitration | 258 |
| Table 5-21. Loses Arbitration Collision Correctly | 259 |
| Table 5-22. Bit Timing Sensitivity | 260 |
| Table 5-23. Check Response to Bad Parity Packet | 261 |
| Table 5-24. Response to ACKs with Different Timings | 263 |
| Table 5-25. Dongle Receives Back-to-Back Transfers..... | 264 |
| Table 6-1. Read Device Capability Registers | 271 |
| Table 6-2. GET_STATE Command..... | 273 |
| Table 6-3. GET_VENDOR_ID Command | 274 |
| Table 6-4. GET_MSC_ERRORCODE Command | 275 |
| Table 6-5. SET_INT and WRITE_STAT Commands | 276 |
| Table 6-6. WRITE_BURST Command..... | 278 |
| Table 6-7. MSC_MSG Command..... | 279 |
| Table 6-8. GET_DDC_ERRORCODE Command | 280 |
| Table 6-9. DUT Receives NACK to MSC_MSG | 281 |
| Table 6-10. Legal MSC Commands..... | 282 |
| Table 6-11. DUT Receives Bad Reply – Control instead of Data..... | 286 |
| Table 6-12. DUT Receives Bad Reply – Data instead of Control..... | 287 |
| Table 6-13. DUT Receives Bad Reply – Control-then-Control instead of Control-then-Data | 288 |
| Table 6-14. DUT Receives Result Timeout | 289 |
| Table 6-15. Verify No Next Command until Hold-Off after ABORT Seen | 291 |
| Table 6-16. DUT Receives Disconnect During Various Commands | 292 |
| Table 6-17. TEST_OFFSET for Interrupt Registers..... | 293 |
| Table 6-18. TEST_OFFSET for Status Registers..... | 294 |
| Table 6-19. DUT Receives Vendor-specific and Reserved Header Values | 295 |
| Table 6-20. DUT Receives (0x61) READ_DEVCAP Command | 297 |
| Table 6-21. DUT Receives (0x60) WRITE_STAT Command..... | 300 |
| Table 6-22. DUT Receives (0x68) MSC_MSG Command – RCP_Support == 1 and RAP_Support == 1 | 302 |
| Table 6-23. DUT Receives Reserved Commands..... | 303 |
| Table 6-24. Valid Commands when Tester is Acting as Source | 304 |
| Table 6-25. Valid Commands when Tester is Acting as Sink | 304 |
| Table 6-26. DUT Issues DDC Short Read and Current Read | 340 |
| Table 6-27. DUT Issues Regular DDC Read | 341 |
| Table 6-28. DUT Issues DDC Segment Read | 344 |
| Table 6-29. DUT Issues DDC Write..... | 346 |
| Table 6-30. DUT Issues DDC Short Read and Current Read | 347 |

| | |
|---|-----|
| Table 6-31. DUT Issues Regular DDC Read | 349 |
| Table 6-32. DUT Issues DDC Segment Read | 351 |
| Table 6-33. DUT Issues DDC Write..... | 353 |
| Table 6-34. DUT Receives DDC Write..... | 356 |
| Table 6-35. DUT Receives DDC Current Read..... | 358 |
| Table 6-36. DUT Receives Regular DDC Read..... | 359 |
| Table 6-37. DUT Receives DDC Segment Read..... | 361 |
| Table 6-38. DUT Receives DDC Write..... | 365 |
| Table 6-39. DUT Receives Short Read and Current Read – Various Errors | 366 |

2 Background

2.1 Formatting of this Document

2.1.1 Formatting of Tests

Each test description uses a sequence of headings as shown in the following table:

| Level | Style | Designation |
|-------|-----------|--|
| 1 | Heading 1 | Device Type Section: Source, Sink, Dongle, Cable |
| 2 | Heading 2 | Test Category: Electrical, System, etc. |
| 3 | Heading 3 | Test Group: Video, Audio, HDCP, etc. |
| 4 | Heading 4 | Individual Major Test (common setup) |
| 5 | Heading 5 | Test Objective, References, Equipment, Method |
| 6 | Heading 6 | Minor Test (using previous major test setup) |

2.1.2 Symbolic Terms Used in Test Definitions

A number of symbolic terms – variable names – are used in the description of tests. These terms indicate aspects of the device under test, as described in the device’s Capability Declaration Form (CDF); and aspects of the test equipment, as described in the methodologies.

Variables containing information about the device under test use the prefix CDF_.

Variables controlling aspects of the test equipment use the prefix TEST_.

Measured parameters during tests use the prefix MEAS_.

A variable name which uses a Greek letter in its name in the MHL Specification may be represented here with the Greek name spelled out. This facilitates use of the variable name in the test equipment. For example: ΔT_{RF} is represented in this document as DELTA TRF.

2.1.3 Test Descriptions

2.1.3.1 Use of Subsections

Each test is described in a series of sections:

- XXX.1 Test Objective
- XXX.2 References
- XXX.3 Required Test Equipment
- XXX.4 Required Methodology

2.1.3.2 “Required” and “Recommended” Test Methodologies

Each test is described first by a required methodology, which explains the methodology for proper measurement of the parameters or performance to PASS, FAIL or PASS (SKIP) the test. This method refers to one or more of the required test equipment.

The test is then described by one or more recommended methodology, using specific fixtures and test equipment, to guide the test to a compliant test result in actual practice. These recommended methodologies are included in one or more separate documents, usually one per vendor of test equipment. Contact the MHL, LLC for details on the documents available.

2.1.4 Capability Declaration Form and Test Results Form

Each device under test (DUT) arrives at the test center with a Capabilities Declaration Form (CDF). This form lists the features of the DUT, as well as providing directions on how to setup the DUT for each feature and input or output mode. Through the use of the details in the CDF the test engineer is able to setup each test and properly exercise the DUT.

A test results form (TRF) is generated by the test center as the tests are performed. This TRF is delivered back to the Adopter to show the PASS, FAIL or PASS (SKIP) result of each test.

2.1.5 Usages and Conventions

In addition to the Usages and Conventions defined in the MHL Specification (q.v.), the following are used in this Compliance Test Specification:

- 0x#_##_###** Symbolizes a hexadecimal value, with the first field (#) representing exactly two bits, the second field (##) representing exactly one bit, and the third field (###) representing exactly 8 bits. This syntax is used throughout the tests covering CBUS. The 2 bits in the first field are the two header bits in a CBUS packet; the 1 bit in the second field is the control bit; the 8 bits in the third field are the eight payload bits. For example, 0x2_0_E3. Refer to Section 7.2.3 in the MHL Specification.
- Normal Mode** Normal Mode is equivalent to 24-bit Mode in the context of the format of video handling. Normal Mode is different from PackedPixel Mode, and separate tests are defined in this CTS for each.

2.2 Test Results

Each test in this Compliance Test Specification ends in a PASS, FAIL or PASS (SKIP) result.

2.2.1 Definitions of Test Result Terminology

| | |
|--------------------|--|
| PASS | Indicates that the device under test completed the recommended methodology with no FAIL or SKIP results. |
| FAIL | Indicates that the device under test failed to meet the requirements of one (or more) steps in the recommended methodology. Usually this means that the device's performance does fall within the bounds prescribed by the methodology, such as the minimum and maximum limits of an electrical test. It may also mean that the device did not implement a required feature, and the lack of such implementation was determined by one or more steps in the methodology. |
| PASS (SKIP) | Indicates that the device under test did not complete the recommended methodology because the conditions defined in the device's Capability Declaration Form, when checked in the first step of the methodology, mean that the test does not apply to this device under test. An example is the implementation of driving VBUS output, which is not required for Source devices. |

2.2.2 Requirement for Successful Completion of Compliance Testing

A device under test passes the compliance test suite if it completes the CTS with no FAIL results.

2.2.3 Test Flow Usage of PASS, FAIL and PASS (SKIP)

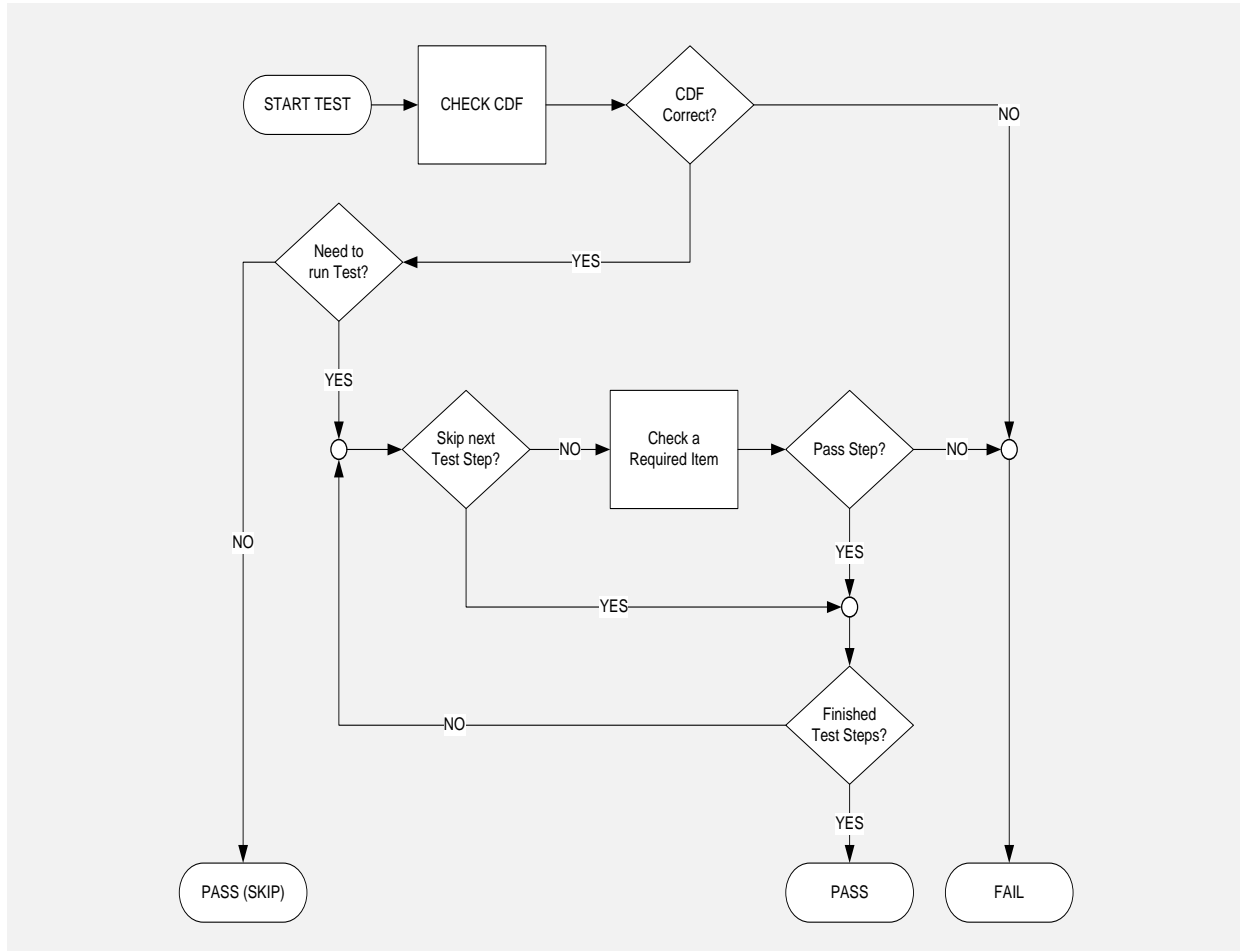


Figure 1. Flowchart for CTS Test Flow

2.3 Test Equipment

A variety of equipment is needed for testing MHL products completely. Each piece is authorized and included by name in the separate documents associated with the Main Required Methods of the Compliance Test Specification. This section describes the technical requirements of the major pieces of equipment used in testing.

NOTE: The proper configuration of each piece of test equipment is critical to the proper functioning of the test. Compliance testing shall follow closely the instructions in each Recommended Methodology to perform each test. Settings on equipment shall not be varied outside of the Recommended Methodology in the performance of compliance testing.

NOTE: Care shall be taken by the test engineer in eliminating from the proximity of the device under test any equipment or other device which may interfere with the accuracy of the tests. Some DUTs use wireless technologies which, when active, may affect the accuracy of MHL-related tests. Adopters shall indicate in the CDF scenarios which affect their DUT, as instructional to the test engineer.

2.3.1 TMDs Test Equipment

Refer also to equipment in Section 2.3.3 for electrical tests which require CBUS Test Equipment assistance.

2.3.1.1 High-bandwidth Digital Oscilloscope

TMDs measurements require a High-bandwidth Digital Oscilloscope:

- -3dB Bandwidth: DC to 8GHz or greater
- Sampling rate > 20Gsample/sec, when 2 or more channels are simultaneously sampling.
- Sample memory: more than 20M samples per channel.

2.3.1.2 High-Bandwidth Probe

The High-Bandwidth Probe shall have the following performance characteristics:

- -3dB Bandwidth:
 - Differential Probe for differential signal and single-ended signal: 8GHz or higher
 - Active Probe for common-mode signal: 2GHz or higher

2.3.1.3 Low-bandwidth Digital Oscilloscope

RxSense measurements require a Low-bandwidth Digital Oscilloscope:

- -3dB Bandwidth: DC to 100MHz or greater
- Sampling rate > 500Msample/sec, when 3 or more channels are simultaneously sampling.
- Sample memory: more than 1M samples per channel.

2.3.1.4 Passive Probe

The Passive Probe shall have the following performance characteristics to measure RSEN:

- -3dB Bandwidth: DC to 100MHz or greater
- Input resistance > 1M ohms
- Input capacitance < 200pF.
- More than 1 mega samples per second per channel.

2.3.1.5 Jitter and Eye Analyzer

The Jitter and Eye Analyzer shall function and process the MHL clock and data signals following the MHL Clock Recovery Unit and the Reference Cable Equalizer specified in MHL specification 13.3.1.8 and 13.3.1.7. These are usually implemented in Digital Oscilloscope software.

2.3.1.6 MHL Signal Generator

The MHL Signal Generator creates content streams at all modes supported by MHL.

- Generate MHL clock and data for all MHL defined format
- Maximum output data bit rate: > 3Gbps
- Internal clock and data jitter generation (optional)
 - Two independent jitter signals
 - Jitter frequency: 100 KHz to 20MHz
 - Jitter amplitude: maximum 1UI for 750Mbps – 3Gbps with 0.05UI granules

2.3.1.7 Clock Jitter Generator

The Clock Jitter Generator shall generate clock (750MHz – 3GHz) with the following jitter characteristics:

- Two independent jitter signals
- Jitter frequency: 100 KHz to 20MHz
- Jitter amplitude: maximum 1UI for 750Mbps – 3Gbps with 0.05UI granules

2.3.1.8 MHL Cable Emulator

The MHL Cable Emulator shall represent the differential and common-mode insertion losses specified in MHL specification 13.8.2.1 and 13.8.2.2 for MHL+ and MHL- channels.

2.3.1.9 Transition Time Converter (TTC)

The transition time converter shall slow down the rise and fall times of output signals of MHL Signal Generator to 200ps (differential 20-80%) and 600ps (common-mode 20-80%).

2.3.1.10 Combiner

- DC to 18 GHz
- Rise time 17 picoseconds typical
- DC insertion loss 6 dB \pm 0.05 dB maximum
- AC insertion loss 6 dB \pm 0.5 dB maximum

2.3.1.11 Bias Tee

- Wideband, 0.1 – 6 GHz
- Low insertion loss, 0.6 dB typical
- Good isolation, 40 dB typical

2.3.1.12 Network Analyzer

- 4 ports
- 300 KHz to 8GHz
- Dynamic accuracy over the frequency range 300KHz – 8GHz
 - Magnitude: \leq (+/-)0.5dB from 0 to - 50dBm
 - Phase: \leq (+/-)4 degrees from 0 to - 50dBm

2.3.1.13 TDR/TDT Oscilloscope

For TDR measurement:

- Bandwidth: \geq 18GHz
- Pulse rise time: \leq 75ps (10-90%)
- 2 port
- Rise time adjustment capability to 200ps (20-80% differential) and 600ps (20-80% common-mode)

For TDT measurement:

- Bandwidth: \geq 18GHz
- Pulse rise time: \leq 75ps (10-90%)
- 4 port

2.3.2 MHL System-level Equipment

2.3.2.1 MHL Protocol Analyzer

The MHL Protocol Analyzer is able to take in a complete stream of content with discovery, authentication, etc.

- MHL Pseudo Sink: Receive MHL stream and display on HDMI capable TV.
 - Receive MHL A/V stream, decode and re-transmit in HDMI format.
 - Detect A/V format, pixel encoding, pixel repetition.
 - Measure A/V timing.
 - Analyze A/V data and AVI InfoFrame.
 - Monitor EDID access.
 - Verify RCP commands.

2.3.2.2 MHL Signal Generator

The MHL Signal Generator creates content streams at all modes supported by MHL.

- Generate MHL clock and data for all MHL defined format
- Maximum output data bit rate: > 3Gbps
- Internal clock and data jitter generation (optional)
 - Two independent jitters
 - Jitter frequency: 100 KHz to 20MHz
 - Jitter amplitude: maximum 1UI for 750Mbps – 3Gbps with 0.05UI granules
 - Generate A/V format.
 - Generate A/V test pattern.
 - EDID access
 - Verify RCP commands.

Figure 2 and Figure 3 show the connection from MHL Signal Generator to the Sink DUT and to the Dongle DUT. The Dongle DUT requires a downstream Sink device to show the signal. A Sink DUT which has no display requires a downstream display device to show the signal.

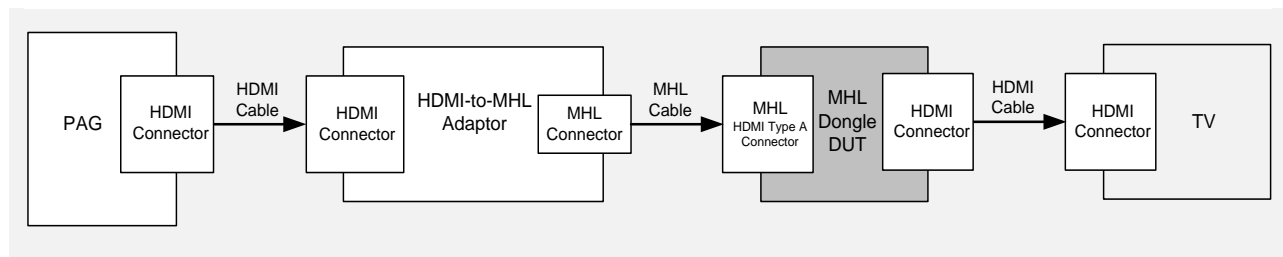


Figure 2. Recommended MHL Signal Generator and Adapter for Dongle Test.

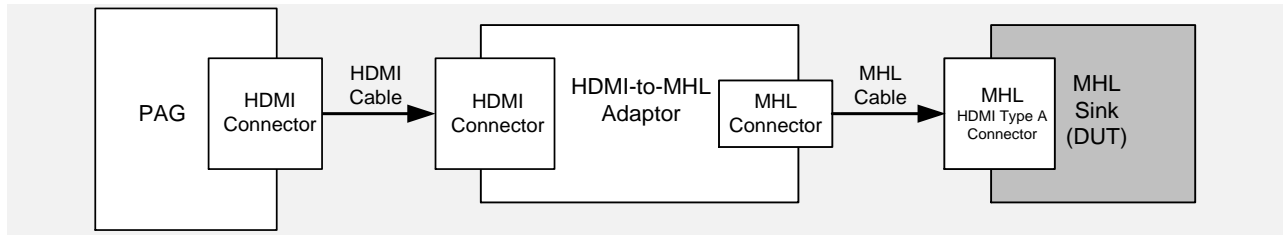


Figure 3. Recommended MHL Signal Generator and Adapter for Sink Test.

2.3.3 CBUS Equipment

2.3.3.1 MHL-CBUS Protocol Generator / Analyzer

The MHL CBUS Protocol Generator / Analyzer is a test tool for analyzing and exercising Mobile High-Definition Link (MHL) CBUS protocol. The tool comprising custom hardware and software is capable of generating and running CBUS sequences for limits testing of the MHL Certification Test Specification, and generating validation reports for in spec and out of spec conditions.

The tool:

1. Generates CBUS sequences for emulating either Source or Sink MHL devices. The tool is capable of varying the electrical and timing parameters of the CBUS signal for conformance testing of MHL devices.
2. Monitors electrical and timing parameters of the CBUS, VBUS and MHL pins and indicates out of spec conditions.
3. Captures CBUS, VBUS and MHL activity to a file, decodes the CBUS stream and flags errors.

2.3.3.2 MHL Test Cable (20cm)

The MHL Test Cable (20cm) is a minimum capacitance test cable for use with the CBUS tester. The cable has a male uUSB connector for connection to an MHL Source port, and an HDMI connector for connection to an MHL Sink port.



Figure 4. MHL Test Cable

2.3.3.3 Digital Oscilloscope

The Digital Oscilloscope is a digital waveform viewing instrument that allows observation of constantly varying signal voltages.

1. -3dB Bandwidth: DC to 500MHz or greater
2. Sampling rate > 1Gsample/sec, when 2 or more channels are simultaneously sampling.
3. Minimum 2 channel, 4 channel preferred

2.3.3.4 CBUS Source Board

A CBUS Source board provides CBUS connection and discovery to an MHL Sink or MHL Dongle DUT. The CBUS Source Board can be an MHL source device or an MHL CBUS Protocol Generator / Analyzer or an MHL RX/TX Analyzer.

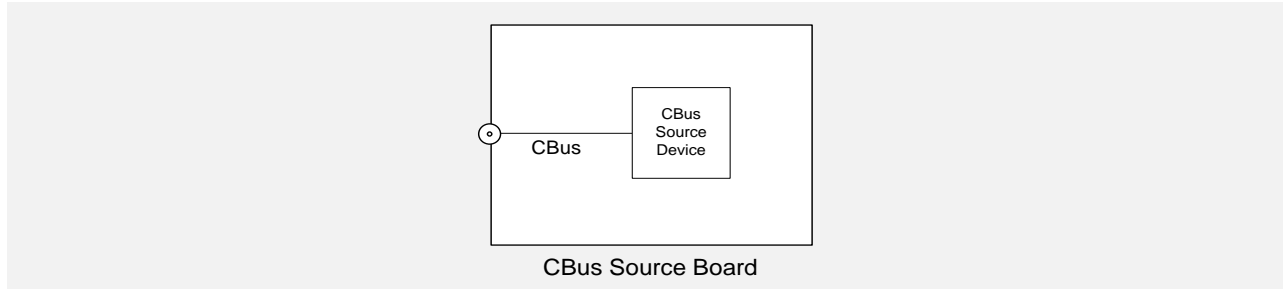


Figure 5. CBUS Source Board

2.3.3.5 CBUS Sink Board

A CBUS Sink board provides CBUS connection and discovery to MHL Source DUT. The CBUS Sink board can be an MHL sink device or an MHL CBUS Protocol Generator / Analyzer or an MHL RX/TX Analyzer.

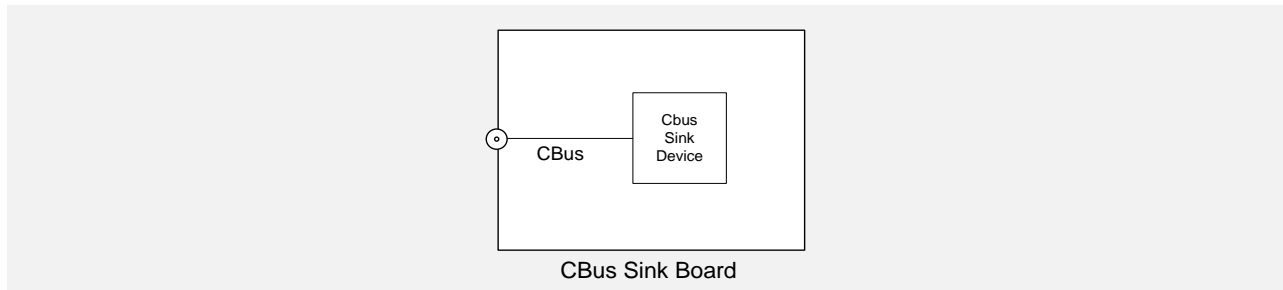


Figure 6. CBUS Sink Board

2.3.3.6 CBUS RxSense Board

A CBUS RxSense Board (Source Emulation) provides CBUS control for RxSense test of a Sink DUT or Dongle DUT. A CBUS RxSense Board (Sink Emulation) provides CBUS control for RxSense of a Source DUT.

2.3.3.7 MHL Direct Attach Source Extension Cable

The MHL Direct Attach Source Extension Cable is a minimum capacitance test cable for use with the CBUS tester in cases where the DA Source DUT cannot successfully mate with the tester's Sink (HDMI) input for mechanical reasons. The cable has a female HDMI receptacle for connection to the DA MHL Source plug and an HDMI plug for connection to an MHL Tester's Sink port receptacle.



Figure 7. MHL Direct Attach Source Extension Cable

2.3.3.8 MHL Direct Attach Sink Extension Cable

The MHL Direct Attach Sink Extension Cable is a minimum capacitance test cable for use with the CBUS tester in cases where the DA Sink DUT cannot successfully mate with the tester's Source (uUSB) input for mechanical reasons. The cable has a female uUSB receptacle for connection to the DA MHL Sink plug and an uUSB plug for connection to an MHL Tester's Source port receptacle.



Figure 8. MHL Direct Attach Sink Extension Cable

2.3.4 MSC Test Tool

The MSC Test Tool allows bi-directional communication with MSC packets between the test tool and the DUT.

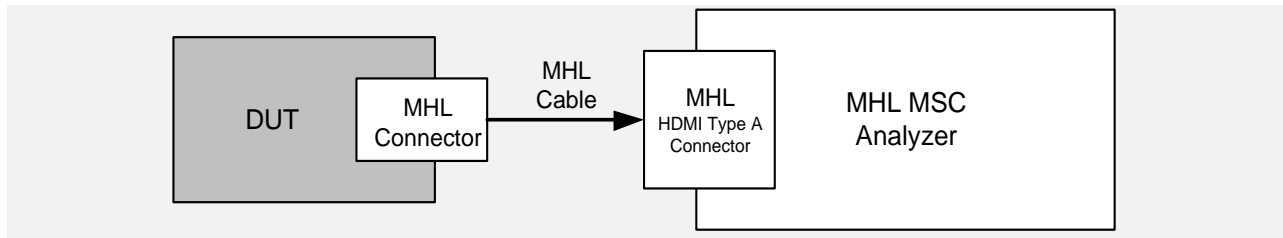


Figure 9. MSC Test Tool

2.3.5 HDCP Equipment

2.3.5.1 HDCP Analyzer

The HDCP Analyzer is used for testing HDCP Transmitters by checking the stream of TMDS and the DDC traffic for proper authentication and encryption.

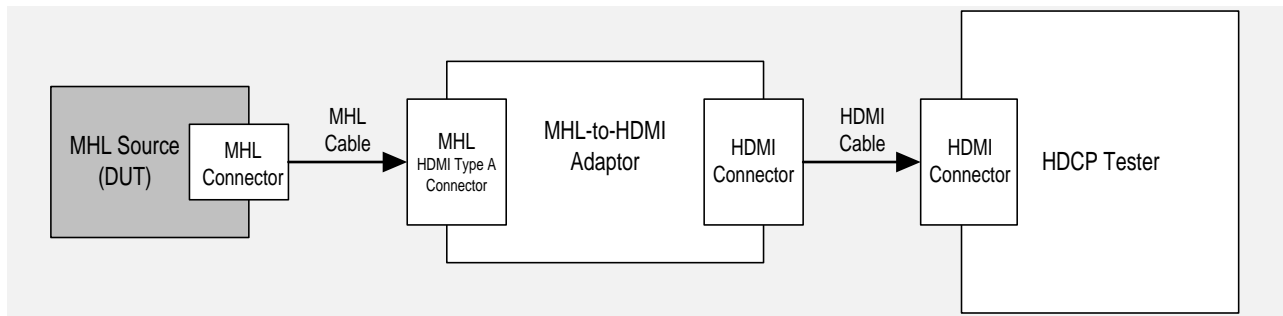


Figure 10. MHL HDCP Analyzer

2.3.5.2 HDCP Emulator

The HDCP Emulator is used for testing HDCP Receivers, by creating a TMDS stream and managing the DDC transactions in parallel for HDCP authentication. Responses from the Receiver are checked for compliance. The output of the Receiver-under-test is checked to show proper decryption.

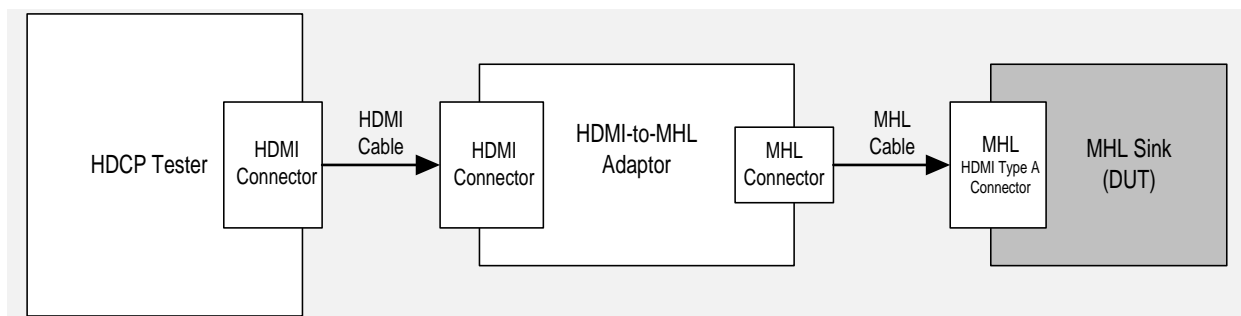


Figure 11. MHL HDCP Emulator

2.3.6 Adapters

2.3.6.1 Test Point Access (TPA) Boards

A TPA board provides connection between a DUT and test equipment. It includes a mating connector (Source side Plug, Source side Receptacle, Sink side Plug, or Sink side Receptacle), connection ports (MHL+, MHL-, VBus, CBUS,

and GND), probe ports (V_{TERM} , V_{D+} , V_{D-} , V_{CM}). It may or may not include a termination-resistor-network for the differential and common-mode signals.

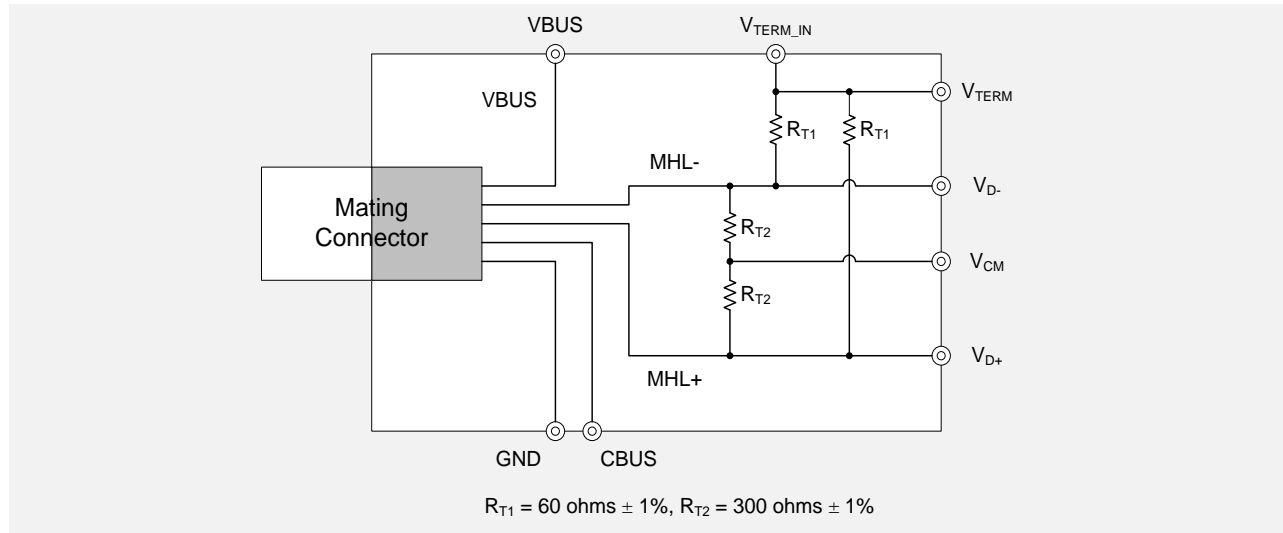


Figure 12. TPA Board with Termination

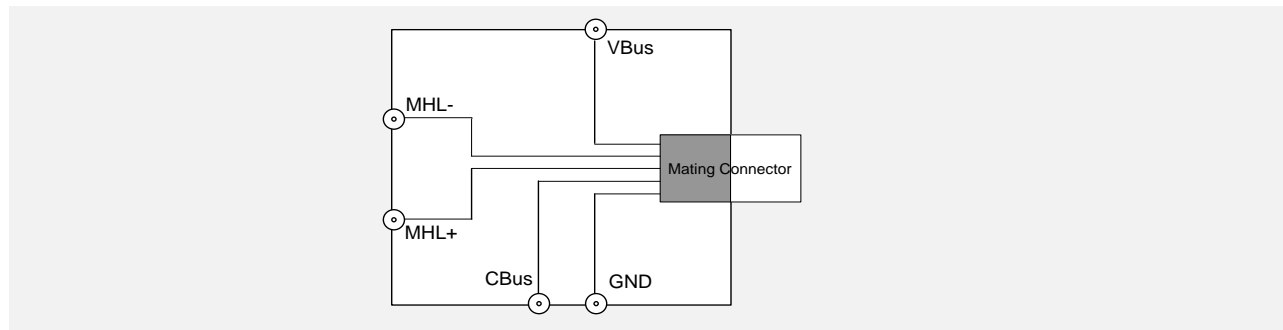


Figure 13. TPA Board without Termination

The following types of MHL TPA boards are used in the subsequent CTS testing procedures. They are differentiated by the mating connector type and existence of termination-resistor-network:

Table 2-1. MHL TPA Boards Description

| MHL TPA | Description | MHL Testing |
|--------------------|---|---|
| Source TPA-P-WT | Source side Plug connector and with termination-resistor-network | Source DUT Testing and Dongle Test Signal Calibration |
| DirSource TPA-R-WT | Source side Receptacle connector and with termination-resistor-network | Direct Attach Source DUT Testing |
| Source TPA-R-WO | Source side Receptacle connector and without termination-resistor-network | Dongle DUT Testing and Cable Testing (Source Side) |
| Sink TPA-R-WT | Sink side Receptacle connector and with termination-resistor-network | Sink Test Signal Calibration |
| Sink TPA-P-WO | Sink side Plug connector and without termination-resistor-network | Sink DUT Testing |
| Sink TPA-R-WO | Sink side Receptacle connector and without termination-resistor-network | Cable Testing (Sink Side) |

| | | |
|-------------------------------|---|--------------------------------|
| DirSink TPA-R-WO ¹ | Sink side Receptacle connector and without termination-resistor-network | Direct Attach Sink DUT Testing |
|-------------------------------|---|--------------------------------|

Notes on Table 2-1:

1. If direct attach sink DUT has the same plug type connector as the Dongle device, then DirSink TPA-R-WO is the same as Source TPA-R-WO.

2.3.6.2 RxSense (RSEN) Test Point Access (TPA) Boards

The following two types of TPA boards will be used for RxSense Impedance testing on Source, Sink, and Dongle DUTs:

1. Source TPA-RSEN: Used for Source DUT RxSense Impedance detection test
2. Sink/Dongle TPA-RSEN: Used for Sink and Dongle DUTs RxSense Impedance test

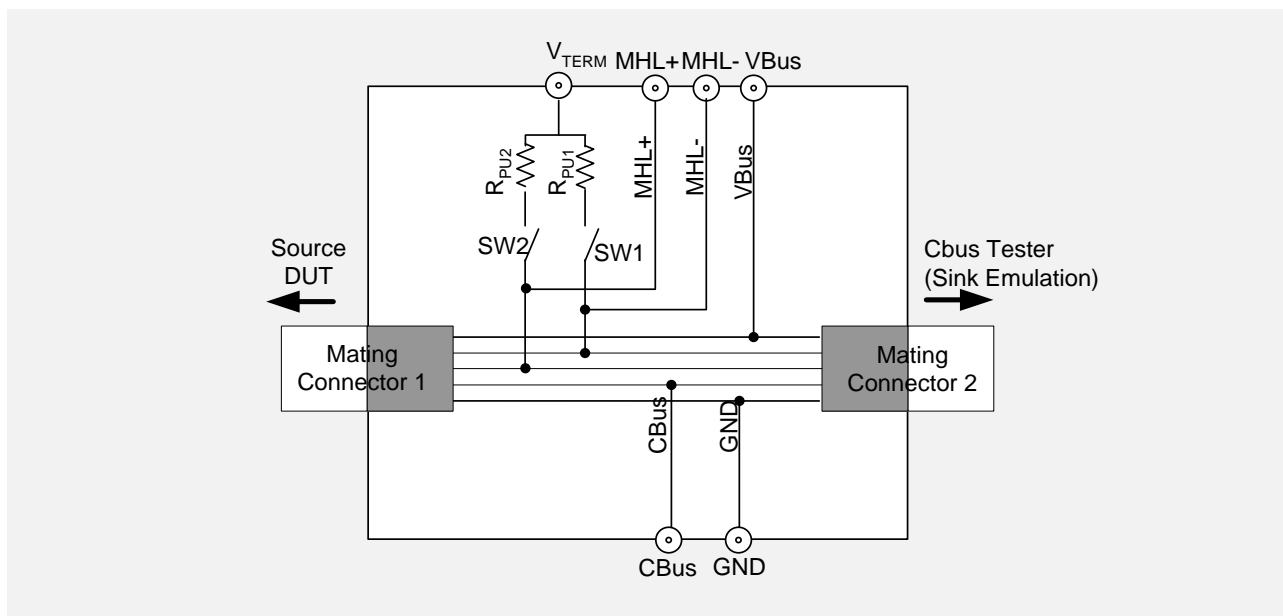


Figure 14. TPA Board for Source RxSense Test (Source TPA-RSEN)

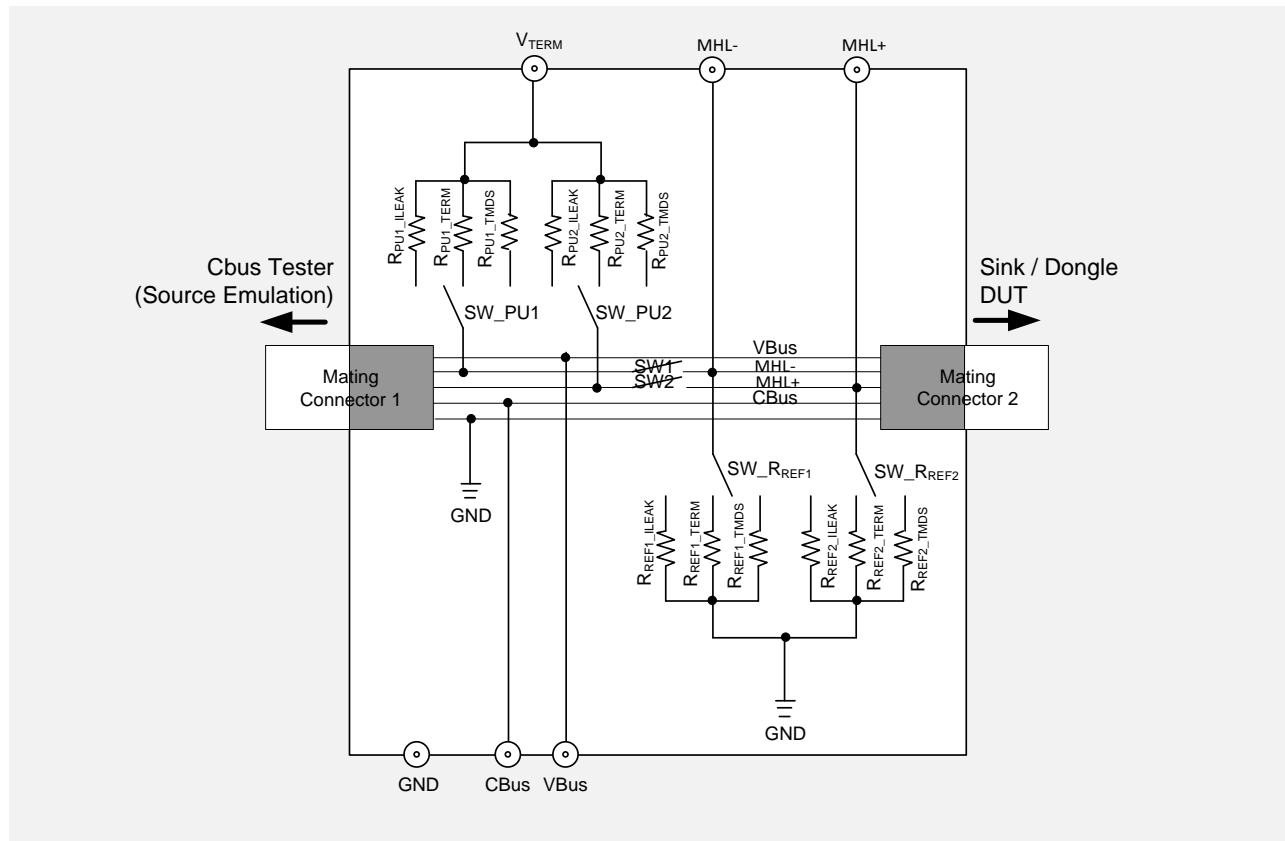


Figure 15. TPA Board for Sink and Dongle RxSense Test (Sink/Dongle TPA-RSEN)

2.3.7 Miscellaneous

2.3.7.1 SMA Coaxial Cable

- -3dB Bandwidth: 10GHz or greater
- 50 ohm impedance

2.3.7.2 DC Source-Meter

- ISVM: Can measure the voltage (0 – 10V, 0.1mV resolution and accuracy or better) with controlling the drain current (0 – 1A, 0.1uA resolution and accuracy or better)

2.3.7.3 Digital Multi-meter

A DC Multi-meter is needed to measure DC voltages at various test points.

- DC Voltage
 - DC voltage resolution $\leq 1\mu\text{V}$ when range is 0 – 1mV
 - DC voltage accuracy $\leq \pm 10\mu\text{V}$ when range is 0 – 1mV
- DC Resistance
 - DC resistance resolution is more than 3 digits
 - DC resistance accuracy $\leq 1\%$
 - Maximum measurable resistance $\geq 1\text{M ohms}$

2.3.7.4 DC Power Supply

A DC Power Supply is needed to drive the TPA boards. A dual mode power supply would be preferred.

- Voltage range: 0 – 10V
- Voltage accuracy: $\leq \pm 1\%$
- Current: maximum 1A

2.3.7.5 DC Block

- -3dB BW: 18GHz or greater

2.4 Required Video and Audio Patterns

Some tests in this specification require the generation or measurement of specific video patterns or audio content streams.

2.4.1 Required Video Patterns

2.4.1.1 Gray Ramp Pattern

The Gray Ramp pattern creates a full video field or frame with a changing color, ranging from black on the left to white on the right, repeating across the width of the screen. The pixel data values for the range from 'black' to 'white' depend on the color space used on the link. A formula for 'gray' in RGB and YCbCr color spaces is shown in Figure 16.

$$\{[Red = Green = Blue = Mod(i + Black, White)] \text{ for } i = 0 \text{ to } (LineWidth - 1) \text{ step } 1\} \text{ for each line in frame}$$

$$\{[Luma = Mod(i + Black, White), Chroma = 128] \text{ for } i = 0 \text{ to } (LineWidth - 1) \text{ step } 1\}$$

Figure 16. Gray Ramp Color Derivation

The limited range and full range values for R, G, B, Y, Cb and Cr for 'black' and 'white' are defined in the CEA-861E Specification, Section 5.4.

The Gray Ramp pattern for PackedPixel mode is the corresponding pattern varying from the minimum luma on the left to the maximum luma on the right.

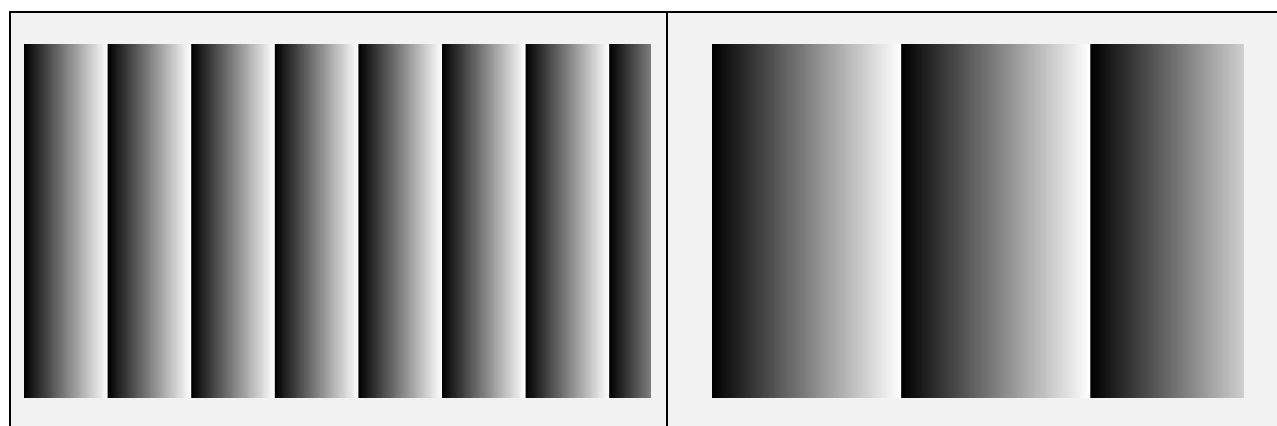


Figure 17. Representative Gray Ramp Video Patterns

Representative Gray Ramp patterns are shown in Figure 17, with a 1920x1080 pattern on the left, and a 720x480 pattern on the right.

MHL Source DUTs are required to output this Gray Ramp pattern in some tests. MHL Sink and Dongle DUTs are required to render, or pass downstream this Gray Ramp pattern in some tests. Refer to each test's required methodology for specific video timings, normal versus PackedPixel mode, etc.

Some test equipment is also required to output this Gray Ramp pattern in some tests. Refer to each test's required methodology for specific video timings, normal versus PackedPixel mode, etc.

2.4.1.2 Chess Board Pattern

The fully saturated black and white Chess Board pattern is required in the Source pixel encoding test and the Source full/limited range test. MHL Source DUTs are required to output this Chess Board pattern in some tests.

For the Chess Board pattern, the Source device creates forty-eight blocks of color, either black or white, as shown in Figure 18. Each rectangular region thus created alternates from black in one region to white in the neighboring regions, beginning with a white rectangle in the upper-left portion of the screen, and ending with a white rectangle in the lower-right portion of the screen.

Note that exact dimensions in pixels and rows of each rectangular region will vary according to the video format, but the color will change seven times moving left-to-right across each row, and change five times moving top-to-bottom on each column.

The limited range and full range values for R, G, B, Y, Cb and Cr are defined in the CEA-861E Specification, Section 5.4.

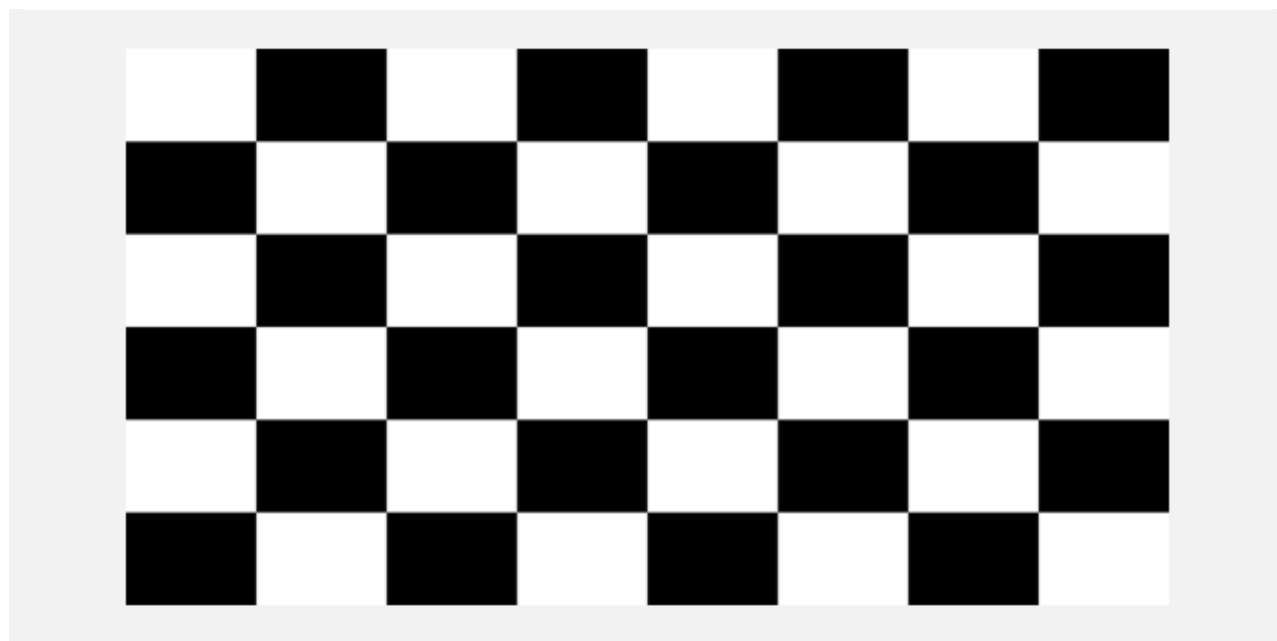


Figure 18. Chess Board Video Pattern

2.4.1.3 3D Video Patterns

No specific 3D video pattern is required, but when testing 3D video mode support the pattern output from the Source DUT should distinctly show the desired 3D effect.

2.4.2 Required Audio Patterns

2.4.2.1 1 Kiloherertz Audio Sinusoidal Tone

An MHL Source DUT, while being tested, shall be capable to generate a 1 kHz sinusoidal tone at a 32kHz, 44.1kHz and/or 48kHz sampling rate that is supported on its MHL output.

An MHL Signal Generator shall be capable to generate a 1kHz sinusoidal tone at 32kHz, 44.1kHz and 48kHz sampling rate.

2.5 Measurement Methods

2.5.1 CBUS Electrical Measurements

2.5.1.1 CBUS Voltage Level Measurement Technique

Various measurement points for determining the HIGH and LOW logic levels on CBUS are shown in Figure 19.

2.5.1.1.1 Measuring Falling Edge Rate

The measurement is the time from when CBUS voltage falls below $VOH_CBUS\{min\}$ [Point A in the figure] until the voltage falls below $VOL_CBUS\{min\}$ [Point B in the figure].

2.5.1.1.2 Measuring Rising Edge Rate

The measurement is the time from when CBUS voltage rises above $VOL_CBUS\{min\}$ [Point D in the figure] until the voltage rises above $VOH_CBUS\{min\}$ [Point E in the figure].

2.5.1.1.3 Measuring Drive High Time

The measurement is the time from when the CBUS voltage starts rising above its stable LOW value [Point C in the figure] until the time the voltage starts dropping after the drive is removed [Point F in the figure].

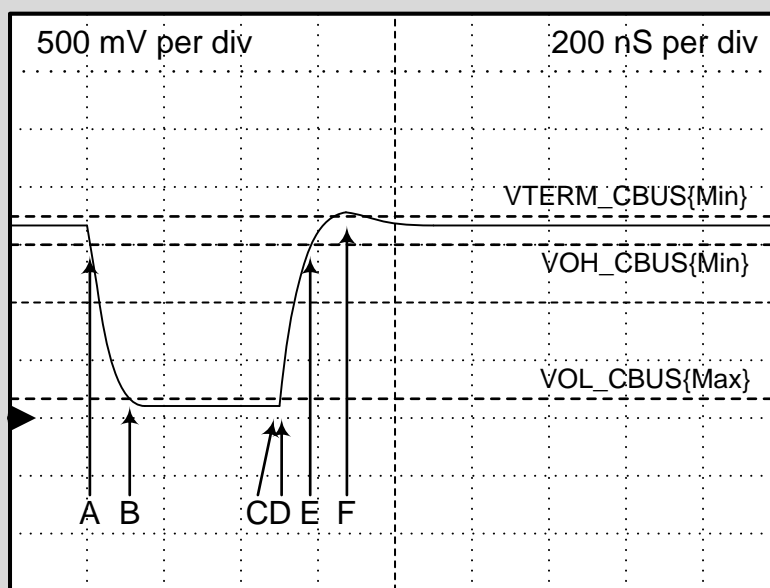


Figure 19. CBUS Voltage Level Waveform

2.6 Revision History

Each released version of this specification shall record the highlights of its changes in this section. Cross-references to the pertinent pages may be included as appropriate.

Revision Highlights

December 2010 MHL Compliance Test Specification

Contains no test methodologies for PackedPixel mode.

February 2011 MHL Compliance Test Specification

Add recommended methodologies for Dongle and Cable device testing.

MHL Compliance Test Specification Revision 1.1

Add automated methods for specific test equipment.

MHL Compliance Test Specification Revision 1.2

MHL Compliance Test Specification Revision 2.0

Add tests to cover new requirements in MHL 2.0 Specification.
Remove Recommended Methods to separate documents.
Divide tests with both Normal and PackedPixel mode sub-tests into separate test indices.
Add CDF field definition listing to Appendix.

MHL Compliance Test Specification Revision 2.1

Remove CDF_XXX_CBUS_THRESHOLD_V parameter and use MHL 2.1 measurement method.

3 Source Test

3.1 Electrical Tests

3.1.1 TMDs Electrical Tests

3.1.1.1 Standby (Off) Output Voltage: V_{OFF}

Application: Source

3.1.1.1.1 Test Objective

This test measures that the MHL source output voltage is within the specified level limits when the source device is in Standby State or power off mode as specified in the CDF.

3.1.1.1.2 References

[MHL] Table 13-5 Source Device DC Characteristics at TP1: $V_{TERM} - 10mV \leq V_{OFF} \leq V_{TERM} + 10mV$

3.1.1.1.3 Required Test Equipment

Digital Multi-meter

Source TPA-P-WT or DirSource TPA-R-WT

DC Power Supply

CBUS Sink Board

3.1.1.1.4 Required Methodology

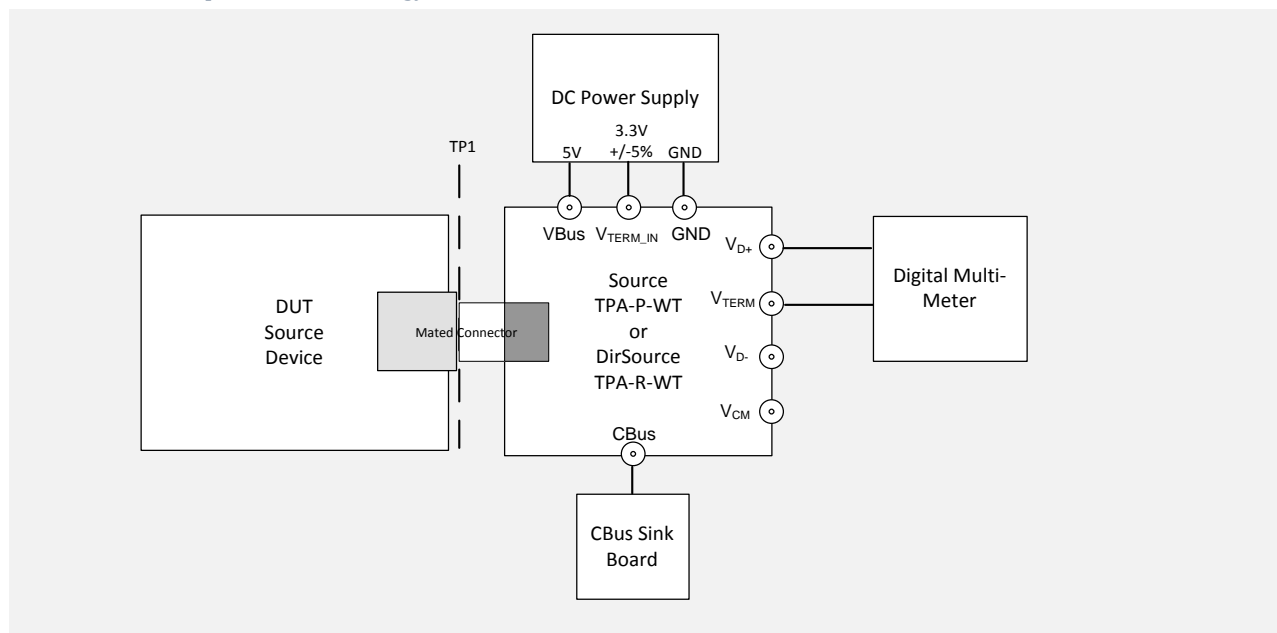


Figure 20. Source – V_{OFF} – Test Setup

Perform the following test for all stand-by and power off modes specified in the CDF. (If neither stand-by nor power off mode is specified in the CDF, this test is not performed. Record a PASS (SKIP) in the TRF.)

1. If **CDF_PROC_SOURCE_POWER_STANDBY** is defined, use the procedure to put the DUT into standby mode; else end the test with PASS (SKIP).
2. Connect the equipment as shown in Figure 20 and unplug the DUT from the Source TPA-P-WT or DirSource TPA-R-WT board.
3. Turn on the DC Power Supply and provide power to $V_{\text{TERM_IN}}$ and VBus.
4. Measure baseline voltages (V_{BASELINE}) between $V_{\text{D+}}$ and V_{TERM} and between $V_{\text{D-}}$ and V_{TERM} for $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\}$ 3.135V and $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\}$ 3.465V without DUT attached.
5. Plug the DUT to the Source TPA-P-WT or DirSource TPA-R-WT board.
6. Put the DUT in the power mode following the instruction in CDF. VBus provision should follow the instruction in CDF.
7. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\}$ 3.135V.
8. Measure the voltage ($V_{\text{OFF_BEFORE_COMPENSATION}}$) between $V_{\text{D+}}$ and V_{TERM} .
9. Measure the voltage ($V_{\text{OFF_BEFORE_COMPENSATION}}$) between $V_{\text{D-}}$ and V_{TERM} .
10. Record V_{OFF} values for $V_{\text{D+}}$ and $V_{\text{D-}}$; $V_{\text{OFF}} = V_{\text{OFF_BEFORE_COMPENSATION}} - V_{\text{BASELINE}}$.
11. Repeat steps 7 – 9 for $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\}$ 3.465V..
12. If all recorded V_{OFF} values are within +/- 10mV (inclusive), PASS. Otherwise FAIL.

3.1.1.2 Single-ended High Level Voltage: $V_{\text{SE_HIGH}}$

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.2.1 Test Objective

This test confirms that the single-ended high output voltage level is within the specified limits when the source device is in normal mode.

3.1.1.2.2 References

[MHL] Table 13-5 Source Device DC Characteristics at TP1: $V_{\text{TERM}} - 540\text{mV} \leq V_{\text{SE_HIGH}} \leq V_{\text{TERM}} + 10\text{mV}$

3.1.1.2.3 Required Test Equipment

High-bandwidth Digital Oscilloscope
High-Bandwidth Probe
CBUS Sink Board
Source TPA-P-WT or DirSource TPA-R-WT
DC Power Supply

3.1.1.2.4 Required Methodology

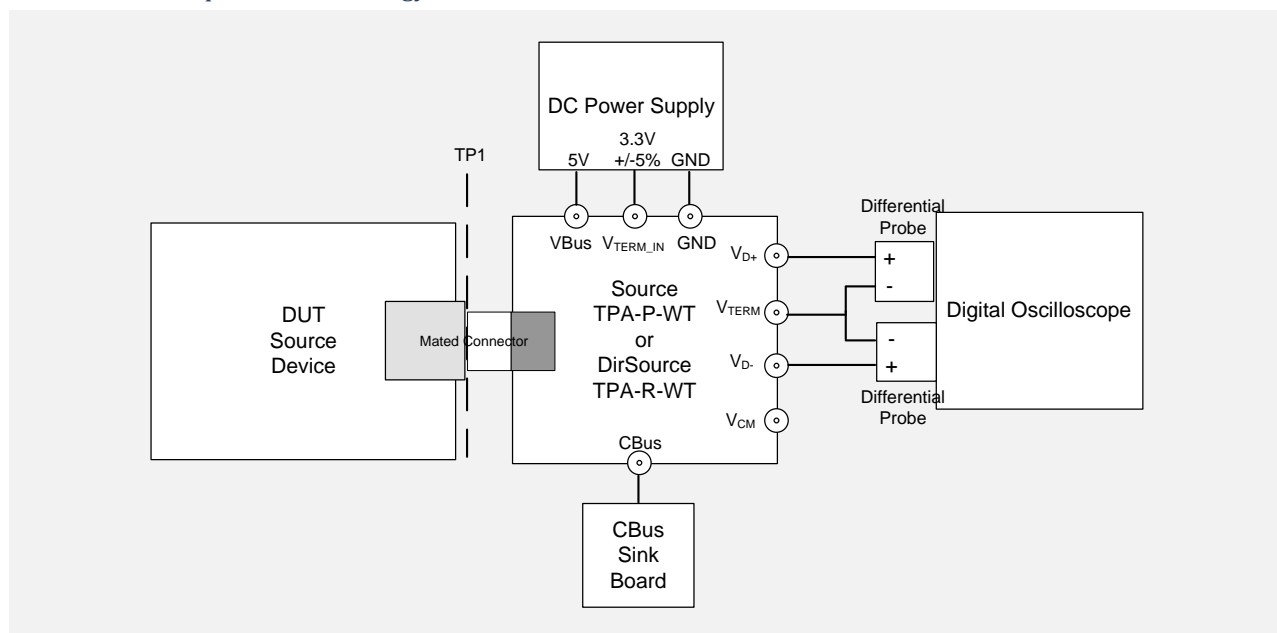


Figure 21. Source – V_{SE_HIGH} Test Setup

1. Connect the equipment as shown in Figure 21.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the lowest MHL data bit rate.
5. Set the Digital Oscilloscope horizontal setting; sampling rate $\geq 20\text{Gsa/sec}$ and approximately 10 bits displayed on the screen.
6. Probe the measurement points with two differential probes as shown in Figure 21.
7. Set $V_{TERM} = V_{TERM}\{\text{min}\} 3.135\text{V}$.
8. Measure V_{SE_HIGH} .
 - a. Pattern trigger for L-H-H-H at the mid-point of the transition edge to the V_{SE_HIGH} Level.
 - b. Display voltage histogram for the third 'H' bit.
 - c. Capture 1,000 repetitions.
 - d. Record histogram mode value for the V_{SE_HIGH} value for the given V_{TERM} .
9. Set $V_{TERM} = V_{TERM}\{\text{max}\} 3.465\text{V}$.
10. Repeat step 8.
11. If $V_{TERM} - 540\text{mV} \leq V_{SE_HIGH} \leq V_{TERM} + 10\text{mV}$ for all recorded V_{SE_HIGH} values, PASS. Otherwise FAIL.

3.1.1.3 Single-ended Low Level Voltages: V_{SE_LOW}

Application: Source

3.1.1.3.1 Test Objective

This test confirms that the single-ended low output voltage level is within the specified limits when the source device is in normal mode.

3.1.1.3.2 References

[MHL] Table 13-5 Source Device DC Characteristics at TP1: $V_{\text{TERM}} - 1760\text{mV} \leq V_{\text{SE_LOW}} \leq V_{\text{TERM}} - 700\text{mV}$

3.1.1.3.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

CBUS Sink Board

Source TPA-P-WT or DirSource TPA-R-WT

DC Power Supply

3.1.1.3.4 Required Methodology

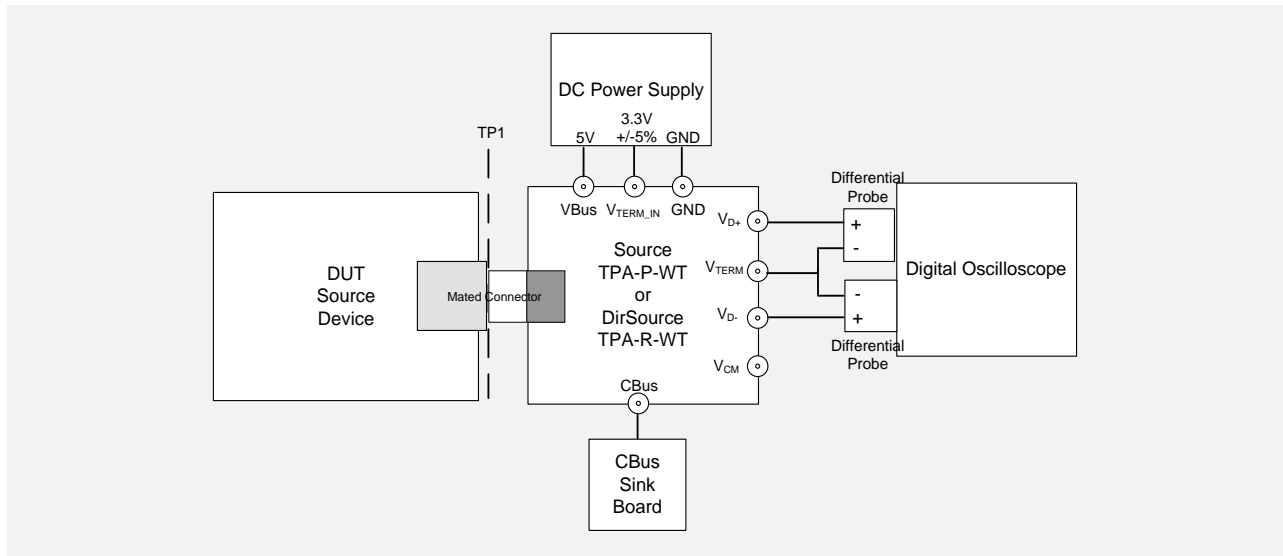


Figure 22. Source – $V_{\text{SE_LOW}}$ Test Setup

1. Connect the equipment as shown in Figure 22.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the lowest MHL data bit rate.
5. Set the Digital Oscilloscope horizontal setting: sampling rate $\geq 20\text{Gsa/sec}$ and approximately 10 bits displayed on the screen.
6. Probe the measurement points with two differential probes as shown in Figure 22.
7. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\} 3.135\text{V}$.
8. Measure $V_{\text{SE_LOW}}$.
 - a. Pattern trigger for H-L-L-L-L at the mid-point of the transition edge to the $V_{\text{SE_LOW}}$ Level.
 - b. Display voltage histogram for the third 'L' bit.
 - c. Capture 1,000 repetitions.
 - d. Record histogram mode value for the $V_{\text{SE_LOW}}$ value for the given V_{TERM} .
9. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\} 3.465\text{V}$.
10. Repeat step 8.
11. If $V_{\text{TERM}} - 1760\text{mV} \leq V_{\text{SE_LOW}} \leq V_{\text{TERM}} - 700\text{mV}$ for all recorded $V_{\text{SE_LOW}}$ values, PASS. Otherwise FAIL.

3.1.1.4 Differential Output Swing Voltage: $V_{DFSWING}$

Application: Source

3.1.1.4.1 Test Objective

This test confirms that the differential output voltage swing amplitude is within the specified limits when the source device is in normal mode.

3.1.1.4.2 References

[MHL] Table 13-5 Source Device DC Characteristics at TP1: $600\text{mV} \leq V_{DFSWING} \leq 1000\text{mV}$

3.1.1.4.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

CBUS Sink Board

Source TPA-P-WT or DirSource TPA-R-WT

DC Power Supply

3.1.1.4.4 Required Methodology

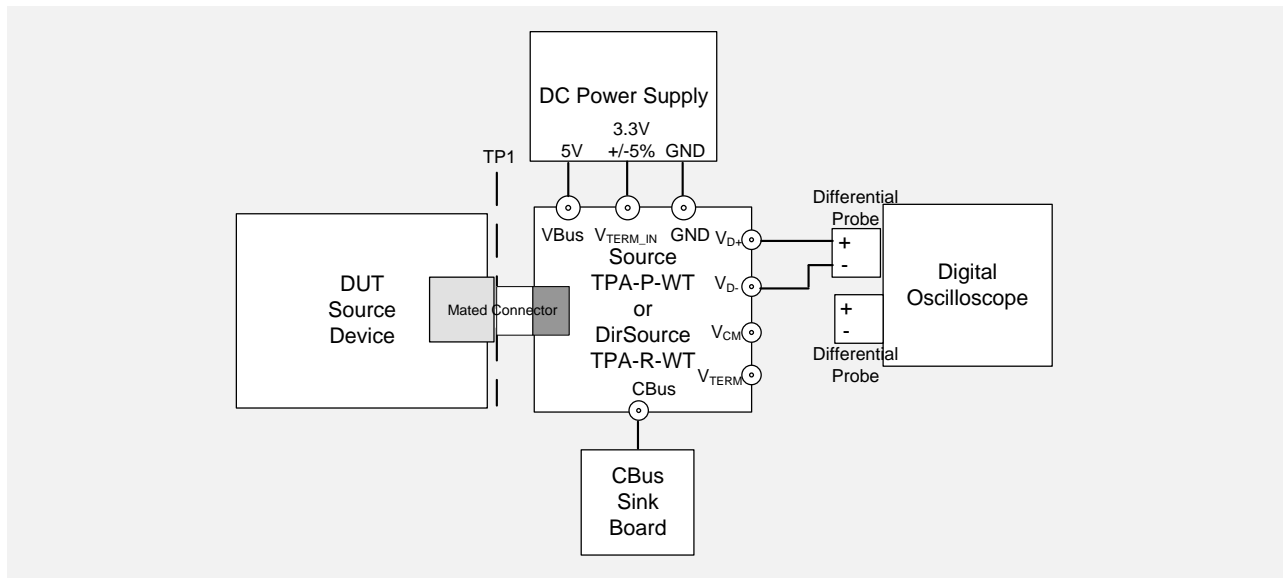


Figure 23. Source – $V_{DFSWING}$ Test Setup

1. Connect the equipment as shown in Figure 23.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the lowest MHL data bit rate.
5. Set the Digital Oscilloscope horizontal setting: sampling rate $\geq 20\text{Gsa/sec}$ and approximately 10 bits displayed on the screen.
6. Probe the measurement points with the differential probe as shown in Figure 23.
7. Set $V_{TERM} = V_{TERM}\{\text{min}\} 3.135\text{V}$.
8. Measure $V_{DFSWING}$.

- a. Display voltage histogram for differential high and low levels of the middle five bits.
 - b. Capture 10,000 repetitions.
 - c. Record the histogram mode values of the differential high and low levels and calculate V_{DFSWING} for the given V_{TERM} .
9. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\} 3.465\text{V}$.
 10. Repeat step 8.
 11. If $600\text{mV} \leq V_{\text{DFSWING}} \leq 1000\text{mV}$ for all recorded V_{DFSWING} values, PASS. Otherwise FAIL.

3.1.1.5 Common-mode Output Swing Voltage: V_{CMSWING}

Application: Source

3.1.1.5.1 Test Objective

This test confirms that the common-mode output voltage swing amplitude is within the specified limits when the source device operates in normal mode.

3.1.1.5.2 References

[MHL] Table 13-5 Source Device DC Characteristics at TP1: $360\text{mV} \leq V_{\text{CMSWING}} \leq \text{Min}(720\text{mV}, 0.85 V_{\text{DFSWING}})$

3.1.1.5.3 Required Test Equipment

High-bandwidth Digital Oscilloscope
High-Bandwidth Probe
CBUS Sink Board
Source TPA-P-WT or DirSource TPA-R-WT
DC Power Supply

3.1.1.5.4 Required Methodology

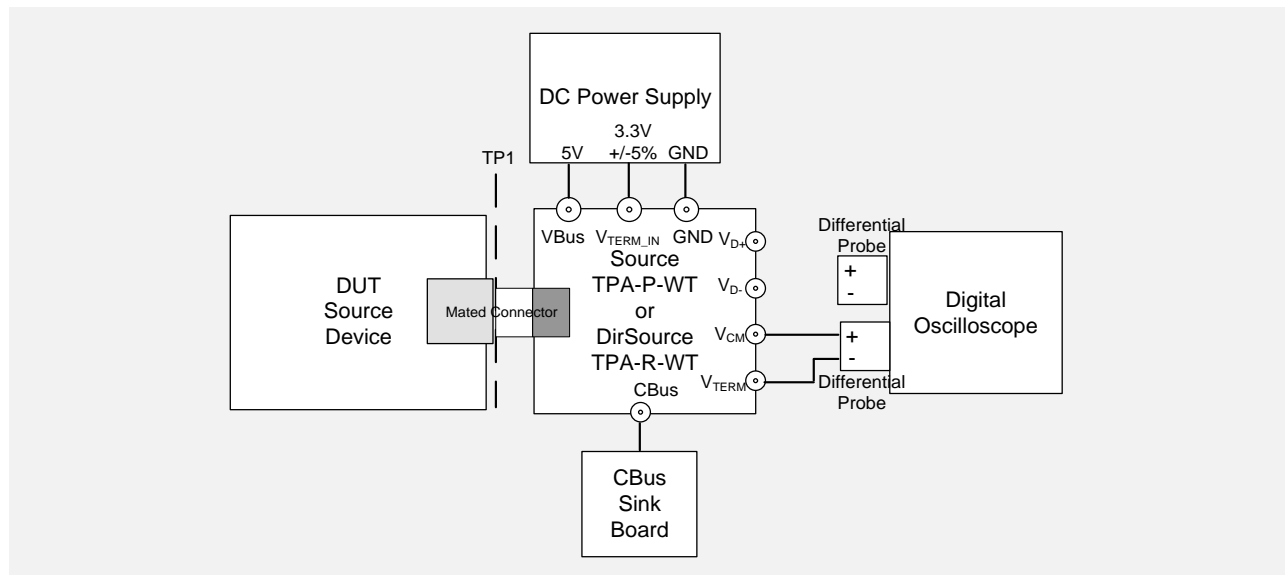


Figure 24. Source – V_{CMSWING} Test Setup

1. Connect the equipment as shown in Figure 24.
2. Turn on the DC Power Supply.

3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the lowest MHL data bit rate.
5. Set the Digital Oscilloscope horizontal setting: sampling rate $\geq 20\text{Gsa/sec}$ and approximately 3 MHL clock cycles displayed on the screen.
6. Probe the measurement points with the differential probe as shown in Figure 24.
7. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\} 3.135\text{V}$.
8. Measure V_{CMSWING} .
 - a. Display voltage histogram on the middle 20 % of MHL Clock high period for the common-mode high level.
 - b. Capture 10,000 repetitions.
 - c. Record the histogram mode value for the common-mode high level.
 - d. Display voltage histogram on the middle 20 % of MHL Clock low period for the common-mode low level.
 - e. Capture 10,000 repetitions.
 - f. Record the histogram mode value for the common-mode low level.
 - g. Calculate V_{CMSWING} for the given V_{TERM} .
9. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\} 3.465\text{V}$.
10. Repeat step 8.
11. If $360\text{mV} \leq V_{\text{CMSWING}} \leq \text{Min}(720\text{mV}, 0.85 V_{\text{DFSWING}})$ for all recorded V_{CMSWING} values, then PASS. Otherwise FAIL. [V_{DFSWING} shall be obtained prior to this test following the procedures in Section 3.1.1.4.4.]

3.1.1.6 Differential Rise and Fall Times: $T_{\text{R_DF}}$, $T_{\text{F_DF}}$

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.6.1 Test Objective

This test confirms that the rise and fall times of the differential output signal are equal to or larger than the minimum limit.

3.1.1.6.2 References

[MHL] Table 13-6 Source Device AC Characteristics at TP1: $T_{\text{R_DF}} \geq 75\text{ps}$ and $T_{\text{F_DF}} \geq 75\text{ps}$

3.1.1.6.3 Required Test Equipment

High-bandwidth Digital Oscilloscope
High-Bandwidth Probe
CBUS Sink Board
Source TPA-P-WT or DirSource TPA-R-WT
DC Power Supply

3.1.1.6.4 Required Methodology

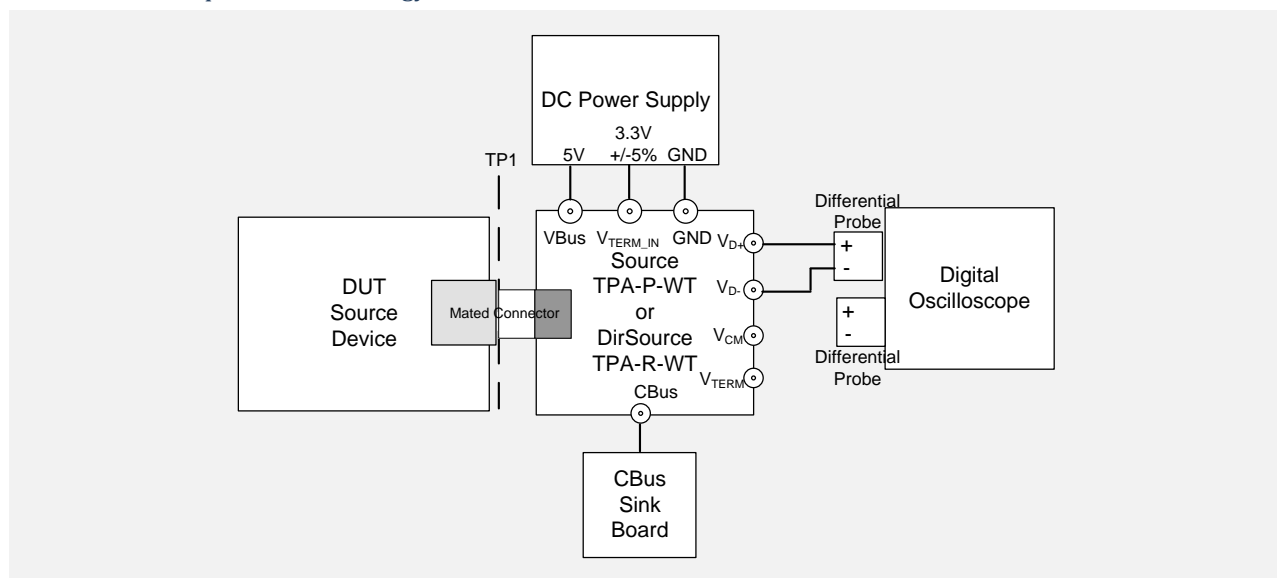


Figure 25. Source – T_{R_DF} and T_{F_DF} Test Setup

1. Connect the equipment as shown in Figure 25.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the highest supported MHL data bit rate.
5. Set $V_{TERM} = V_{TERM}\{min\}$ 3.135V.
6. Measure the differential high and low levels following the procedures in 3.1.1.4 for the given V_{TERM} and MHL data bit rate.
7. Set the Digital Oscilloscope horizontal setting: sampling rate ≥ 20 Gsa/sec and approximately 2 bits displayed on the screen.
8. Probe the measurement points with the differential probe as shown in Figure 25.
9. Measure T_{R_DF} and T_{F_DF} .
 - a. Edge-trigger at the mid-level of rising transition for T_{R_DF} and falling transition for T_{F_DF} .
 - b. Capture 10,000 repetitions for each case.
 - c. Record the mean values of the 20% - 80% rising (T_{R_DF}) and falling (T_{F_DF}) transition times for the given V_{TERM} and MHL data bit rate.
10. Set $V_{TERM} = V_{TERM}\{max\}$ 3.465V.
11. Repeat steps 6 -9.
12. If $T_{R_DF} \geq 75$ ps and $T_{F_DF} \geq 75$ ps for all recorded T_{R_DF} and T_{F_DF} values, then PASS. Otherwise FAIL.

3.1.1.7 Common-mode Rise and Fall Times: T_{R_CM} , T_{F_CM}

Application: Source

3.1.1.7.1 Test Objective

This test confirms that the rise and fall times of the common-mode output signal are within the specified limits.

3.1.1.7.2 References

[MHL] Table 13-6 Source Device AC Characteristics at TP1: $600\text{ps} \leq T_{R_CM} \leq 2500\text{ps}$ and $600\text{ps} \leq T_{F_CM} \leq 2500\text{ps}$

3.1.1.7.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

CBUS Sink Board

Source TPA-P-WT or DirSource TPA-R-WT

DC Power Supply

3.1.1.7.4 Required Methodology

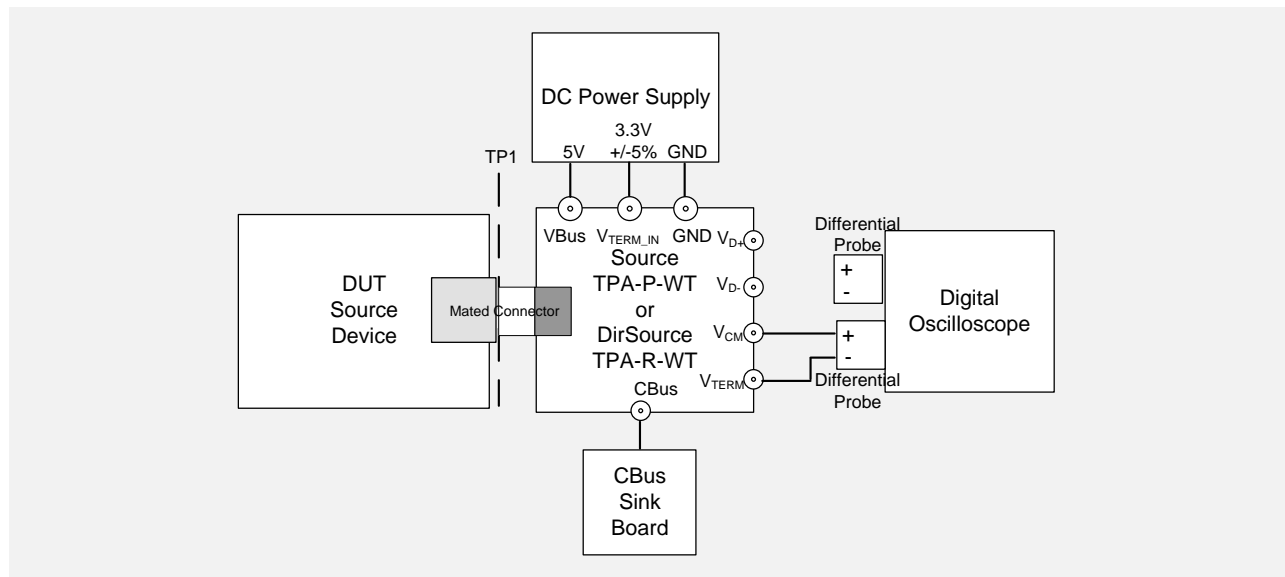


Figure 26. Source – T_{R_CM} and T_{F_CM} Test Setup

1. Connect the equipment as shown in Figure 26.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL data streams in the highest supported MHL data bit rate.
5. Set $V_{TERM} = V_{TERM}\{\min\} 3.135V$.
6. Measure the common-mode high and low levels following the procedures in 3.1.1.5 for the given V_{TERM} and MHL data bit rate.
7. Set the Digital Oscilloscope horizontal setting: sampling rate $\geq 20\text{Gsa/sec}$ and approximately 1 MHL clock cycle displayed on the screen.
8. Probe the measurement points with the differential probe as shown in Figure 26.
9. Measure T_{R_CM} and T_{F_CM} .
 - a. Edge-trigger at the mid-level of rising transition for T_{R_CM} and falling transition for T_{F_CM} .
 - b. Capture 10,000 repetitions for each case.
 - c. Record the mean values of the 20% - 80% rising (T_{R_CM}) and falling (T_{F_CM}) transition times for the given V_{TERM} and MHL data bit rate.
10. Set $V_{TERM} = V_{TERM}\{\max\} 3.465V$.

11. Repeat steps 6 -9.
12. If $600\text{ps} \leq T_{R_CM} \leq 2500\text{ps}$ and $600\text{ps} \leq T_{F_CM} \leq 2500\text{ps}$ for all recorded T_{R_CM} and T_{F_CM} values, then PASS. Otherwise FAIL.

3.1.1.8 Differential Intra-Pair Skew: T_{SKEW_DF}

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.8.1 Test Objective

This test is replaced with tests 3.1.1.17, 3.1.1.18, 3.1.1.19, and 3.1.1.20 for DUTs which support PackedPixel mode; and with tests 3.1.1.17, 3.1.1.18 for DUTs which do not support PackedPixel mode.

3.1.1.8.2 References

[MHL] Table 13-6 Source Device AC Characteristics at TP1: $T_{SKEW_DF} \leq \text{Min}(0.12T_{BIT}, 50\text{ps})$

3.1.1.8.3 Required Test Equipment

3.1.1.8.4 Required Methodology

This test is replaced with tests for DUTs which support PackedPixel mode.
This test is replaced with tests for DUTs which do not support PackedPixel mode.

3.1.1.9 Common-Mode Intra-Pair Skew: T_{SKEW_CM}

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.9.1 Test Objective

This test is replaced with tests 3.1.1.17, 3.1.1.18, 3.1.1.19, and 3.1.1.20 for DUTs which support PackedPixel mode; and with tests 3.1.1.17, 3.1.1.18 for DUTs which do not support PackedPixel mode..

3.1.1.9.2 References

[MHL] Table 13-6 Source Device AC Characteristics at TP1: $T_{SKEW_CM} \leq \text{Min}(0.12T_{BIT}, 50\text{ps})$

3.1.1.9.3 Required Test Equipment

3.1.1.9.4 Required Methodology

This test is replaced with tests for DUTs which support PackedPixel mode.
This test is replaced with tests for DUTs which do not support PackedPixel mode.

3.1.1.10 MHL Clock Duty Cycle in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.10.1 Test Objective

This test confirms that the MHL clock duty cycle in Normal Mode does not exceed the limits allowed by the specification.

3.1.1.10.2 References

[MHL] Table 13-6 Source Device AC Characteristics at TP1: $35\%T_{MHL} \leq \text{Clock Duty Cycle} \leq 65\%T_{MHL}$

3.1.1.10.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

CBUS Sink Board

Source TPA-P-WT or DirSource TPA-R-WT

DC Power Supply

3.1.1.10.4 Required Methodology

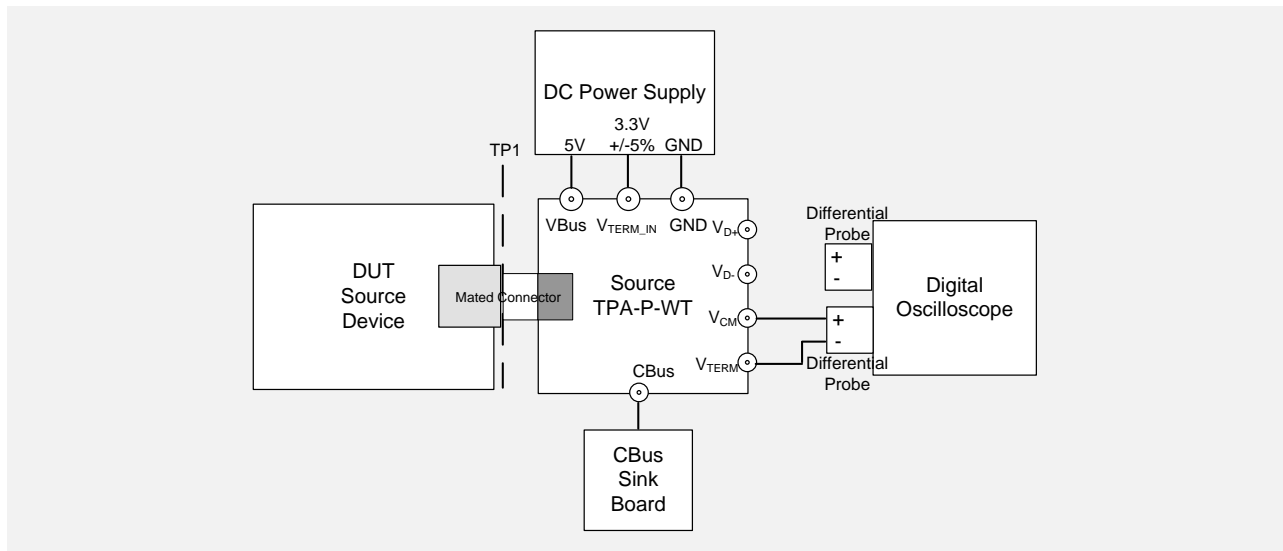


Figure 27. Source – MHL Clock Duty Cycle Test Setup

1. Connect the equipment as shown in Figure 27.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the highest supported MHL data bit rate in Normal Mode.
5. Set $V_{TERM} = V_{TERM}\{min\}$ 3.135V.
6. Set the Digital Oscilloscope horizontal setting: sampling rate $\geq 20\text{Gsa/sec}$ and approximately 10 MHL clock cycle displayed on the screen.
7. Probe the measurement points with the differential probe as shown in Figure 27.
8. Measure the duty cycle.
 - a. Edge-trigger at the mid-level of rising transition edge of the common-mode signal.
 - b. Capture 10,000 repetitions.
 - c. Record the mean values of the duty cycle for the given V_{TERM} and MHL data bit rate.
9. Set $V_{TERM} = V_{TERM}\{max\}$ 3.465V.
10. Repeat step 8.
11. If $35\%T_{MHL} \leq \text{Clock Duty Cycle} \leq 65\%T_{MHL}$ for all recorded Clock Duty Cycle values, then PASS. Otherwise FAIL.

3.1.1.11 MHL Clock Jitter in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.11.1 Test Objective

This test confirms that the MHL Clock output does not contain excessive jitter larger than the limit allowed by the specification in Normal Mode.

3.1.1.11.2 References

[MHL] Figure 13-13 MHL Clock Jitter at TP1:

$$T_{CLK_JITTER_TP1} \leq 0.25T_{BIT} + 200ps, \text{ up to } 2.25Gbps$$

$$T_{CLK_JITTER_TP1} \leq 0.7T_{BIT}, \text{ above } 2.25Gbps$$

3.1.1.11.3 Required Test Equipment

3.1.1.11.4 Required Methodology

This test is replaced by test 3.1.1.17.

3.1.1.12 MHL Data Eye Diagram in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.12.1 Test Objective

This test confirms that the MHL Data output has signal quality that meets the eye opening required by the specification in Normal Mode.

3.1.1.12.2 References

[MHL] Figure 13-14 Differential Eye Diagram Mask at TP1

[MHL] Figure 13-15 MHL Data Jitter at TP1

3.1.1.12.3 Required Test Equipment

3.1.1.12.4 Required Methodology

This test is replaced by test 3.1.1.18.

3.1.1.13 RxSense Impedance

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.13.1 Test Objective

This test confirms that the Source recognizes, and responds correctly to, the allowed ranges of Sink RxSense impedance as defined in the specification.

3.1.1.13.2 References

[MHL] Table 13-9 Sink Device Impedance Characteristics at TP2:

RxSense Active: $Z_{RXSENSE_TERM} \leq 100K \text{ ohms}$

TMDs Termination: $Z_{RXSENSE_TMDs} \leq 70 \text{ ohms}$

RxSense Open (leakage current): $I_{RXSENSE_LEAK} \leq 2 \text{ uA}$

[MHL] Table 13-31 and 13-32

$T_{SRC:RXSENSE_CHK}$
 $T_{SRC:RXSENSE_DEGLITCH}$
 $T_{SRC:CBUS_TMDS_DIS}$

3.1.1.13.3 Required Test Equipment

Low-Bandwidth Digital Oscilloscope
Passive Probe
CBUS Tester (Sink Emulation)
Source TPA-RSEN
DC Power Supply

3.1.1.13.4 Required Methodology

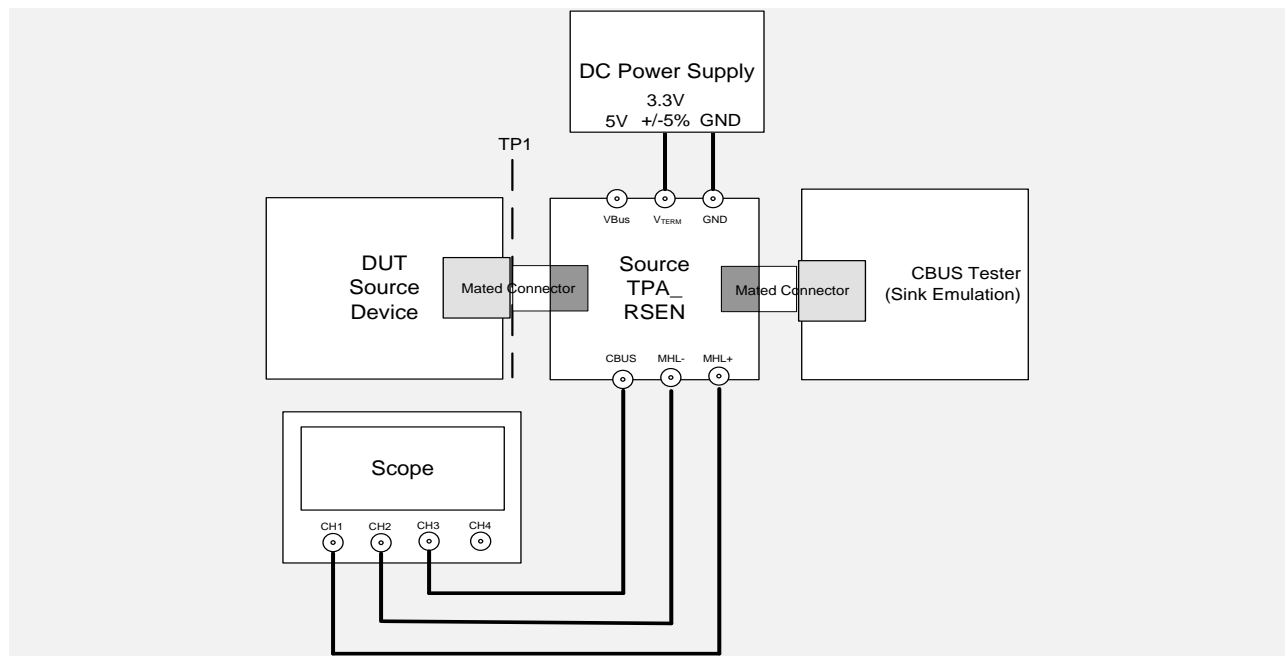


Figure 28. Source – RxSense Test Setup

1. Connect the equipment as shown in Figure 28.
2. Turn on Source DUT and CBUS Tester (Sink Emulation).
3. Perform MHL discovery using CBUS Tester (Sink Emulation).
 - a. Information: At the end of this step, Source DUT should be in MHL discovered state.
4. Verify that the Source DUT is in a discovered state. FAIL if DUT does not respond to CBUS command.
5. Wait $T_{SRC:RXSENSE_CHK}\{min\}$ then set CBUS Tester(Sink Emulation) RxSense to $Z_{RXSENSE_TERM}\{max\}$.
6. Wait 1 sec, verify that Source DUT stays connected. FAIL if DUT does not respond to CBUS command.
7. Set CBUS Tester (Sink Emulation) RxSense to $Z_{RXSENSE_TMDS}\{max\}$.
8. Use the CBUS Tester (Sink Emulation) to send PATH_EN=1 packet to Source DUT.
9. Wait for Source DUT to send PATH_EN=1 packet to CBUS Tester (Sink Emulation).
 - a. Source device may send video to the Sink after completion of this step.
10. Wait 1 sec, verify that Source DUT stays connected. FAIL if DUT does not respond to CBUS command.
11. Use the CBUS Tester (Sink Emulation) to send PATH_EN=0 packet to Source DUT.
12. Wait for Source DUT to send PATH_EN=0 packet to CBUS Tester (Sink Emulation).

13. Set CBUS Tester (Sink Emulation) RxSense to $Z_{RXSENSE_TERM} \{max\}$.
14. Wait 1 sec, verify that Source DUT stays connected. FAIL if DUT does not respond to CBUS command.
15. Set CBUS Tester (Sink Emulation) to drive $I_{RXSENSE_LEAK} \{max\}$ for $T_{SRC_RXSENSE_DEGLITCH} \{min\}$ into Source DUT.
 - a. Info: This time is shorter than the amount the Source DUT must wait to disconnect upon loss of RxSense.
 - b. After $T_{SRC_RXSENSE_DEGLITCH} \{min\}$, set CBUS Tester to revert to $Z_{RXSENSE_TERM} \{max\}$ impedance.
16. Verify that Source DUT stays connected. FAIL if DUT does not respond to CBUS command.
17. Set CBUS Tester (Sink Emulation) to drive $I_{RXSENSE_LEAK} \{max\}$ for $T_{SRC_RXSENSE_DEGLITCH} \{max\} + T_{SRC_CBUS_TMDS_DIS} - T_{SRC_RXSENSE_DEGLITCH} \{min\}$ into Source DUT.
 - a. Info: This time is longer than the time within which the Source must disconnect upon loss of RxSense.
 - b. After $(T_{SRC_RXSENSE_DEGLITCH} \{max\} + T_{SRC_CBUS_TMDS_DIS} - T_{SRC_RXSENSE_DEGLITCH} \{min\})$, set CBUS Tester to revert to $Z_{RXSENSE_TERM} \{max\}$ impedance.
18. Verify that Source DUT disconnects. FAIL if DUT responds to CBUS command.
19. Verify time between falling edges of MHL+/- and CBUS signal. FAIL if time is $> (T_{SRC_RXSENSE_DEGLITCH} + T_{SRC_CBUS_TMDS_DIS})$ after MHL disconnect, else PASS.

3.1.1.14 MHL Clock Duty Cycle in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.14.1 Test Objective

This test confirms that the MHL clock duty cycle in PackedPixel Mode does not exceed the limits allowed by the specification.

3.1.1.14.2 References

[MHL] Table 13-6 Source Device AC Characteristics at TP1: $35\%T_{MHL_PP} \leq \text{Clock Duty Cycle} \leq 65\%T_{MHL_PP}$

3.1.1.14.3 Required Test Equipment

High-bandwidth Digital Oscilloscope
High-Bandwidth Probe
CBUS Sink Board
Source TPA-P-WT or DirSource TPA-R-WT
DC Power Supply

3.1.1.14.4 Required Methodology

1. Connect the equipment as shown in Figure 27.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the highest supported MHL data bit rate in PackedPixel Mode.
5. Set $V_{TERM} = V_{TERM} \{min\} 3.135V$.
6. Set the Digital Oscilloscope horizontal setting: sampling rate $\geq 20Gsa/sec$ and approximately 10 MHL clock cycle displayed on the screen.
7. Probe the measurement points with the differential probe as shown in Figure 27.

8. Measure the duty cycle.
 - d. Edge-trigger at the mid-level of rising transition edge of the common-mode signal.
 - e. Capture 10,000 repetitions.
 - f. Record the mean values of the duty cycle for the given V_{TERM} and MHL data bit rate.
9. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\}$ 3.465V.
10. Repeat step 8.
11. If $35\%T_{\text{MHL_PP}} \leq \text{Clock Duty Cycle} \leq 65\%T_{\text{MHL_PP}}$ for all recorded Clock Duty Cycle values, then PASS.
Otherwise FAIL.

3.1.1.15 MHL Clock Jitter in PackedPixel Mode

| Application: | Source |
|--------------|--------|
|--------------|--------|

3.1.1.15.1 Test Objective

This test confirms that the MHL Clock output does not contain excessive jitter larger than the limit allowed by the specification in PackedPixel Mode.

3.1.1.15.2 References

[MHL] Figure 13-13 MHL Clock Jitter at TP1:

$$T_{\text{CLK_JITTER_TP1}} \leq 0.25T_{\text{BIT}} + 200\text{ps, up to 2.25Gbps}$$

$$T_{\text{CLK_JITTER_TP1}} \leq 0.7T_{\text{BIT}}, \text{ above 2.25Gbps}$$

3.1.1.15.3 Required Test Equipment

3.1.1.15.4 Required Methodology

This test is replaced by test 3.1.1.19.

3.1.1.16 MHL Data Eye Diagram in PackedPixel Mode

| Application: | Source |
|--------------|--------|
|--------------|--------|

3.1.1.16.1 Test Objective

This test confirms that the MHL Data output has signal quality that meets the eye opening required by the specification in PackedPixel Mode.

3.1.1.16.2 References

[MHL] Figure 13-14 Differential Eye Diagram Mask at TP1

[MHL] Figure 13-15 MHL Data Jitter at TP1

3.1.1.16.3 Required Test Equipment

3.1.1.16.4 Required Methodology

This test is replaced by test 3.1.1.20.

3.1.1.17 TP2 Clock Jitter in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.17.1 Test Objective

This test confirms that the MHL Clock output after a compliant cable does not contain excessive jitter larger than the limit allowed by the specification in Normal Mode.

3.1.1.17.2 References

[MHL] Figure 13-16 MHL Clock Jitter at TP2:

$$T_{CLK_JITTER_TP2} \leq 0.3T_{BIT} + 266.7\text{ps, up to 2.25Gbps}$$

$$T_{CLK_JITTER_TP2} \leq 0.9T_{BIT}, \text{ above 2.25Gbps}$$

3.1.1.17.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

Jitter and Eye Analyzer

CBUS Sink Board

Source TPA-P-WT or DirSource TPA-R-WT

DC Power Supply

3.1.1.17.4 Required Methodology

1. Connect the equipment as shown in Figure 29.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.
4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the lowest supported MHL data bit rate.
5. Set $V_{TERM} = V_{TERM}\{min\}$ 3.135V.
6. Probe the measurement points as shown in Figure 29.
7. Set the oscilloscope to the following conditions.
 - a. Sampling rate $\geq 20\text{Gs/sec}$
 - b. Sampling length $\geq 0.5\text{msec}$
 - c. Low pass filtering for the MHL Clock signal: $f_{3dB} = 500\text{MHz}$
 - d. Software CRU for the MHL Clock signal: 1st order, 3dB Bandwidth = 4MHz
 - e. Reference equalizer for the MHL Clock signal
 - f. Histogram Window Vertical Width = 5mV
8. Set the oscilloscope to convolve s-parameter of worst cable model whose intra-pair skew is +43ps.
9. Measure the MHL clock jitter on the rising edge and record the average value over ten measurements for the given V_{TERM} and MHL data bit rate.
10. Set $V_{TERM} = V_{TERM}\{max\}$ 3.465V.
11. Repeat step 9.
12. Set the oscilloscope to convolve s-parameter of worst cable model whose intra-pair skew is -43ps and set $V_{TERM} = V_{TERM}\{min\}$ 3.135V.
13. Repeat steps 9 – 11.
14. Repeat steps 5 – 13 for the highest supported MHL data bit rate in Normal mode.

15. If all recorded jitter values meet the MHL Clock Jitter spec, then PASS. Otherwise FAIL.

3.1.1.18 TP2 Eye Diagram in Normal Mode

Application: Source

3.1.1.18.1 Test Objective

This test confirms that the MHL output after a compliant cable has signal quality that meets the eye opening required by the specification in Normal Mode.

3.1.1.18.2 References

[MHL] Figure 13-17 MHL Data Eye Mask at TP2

[MHL] Figure 13-18 MHL Data Jitter at TP2

3.1.1.18.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

Jitter and Eye Analyzer

CBUS Sink Board

Source TPA-P-WT or DirSource TPA-R-WT

DC Power Supply

3.1.1.18.4 Required Methodology

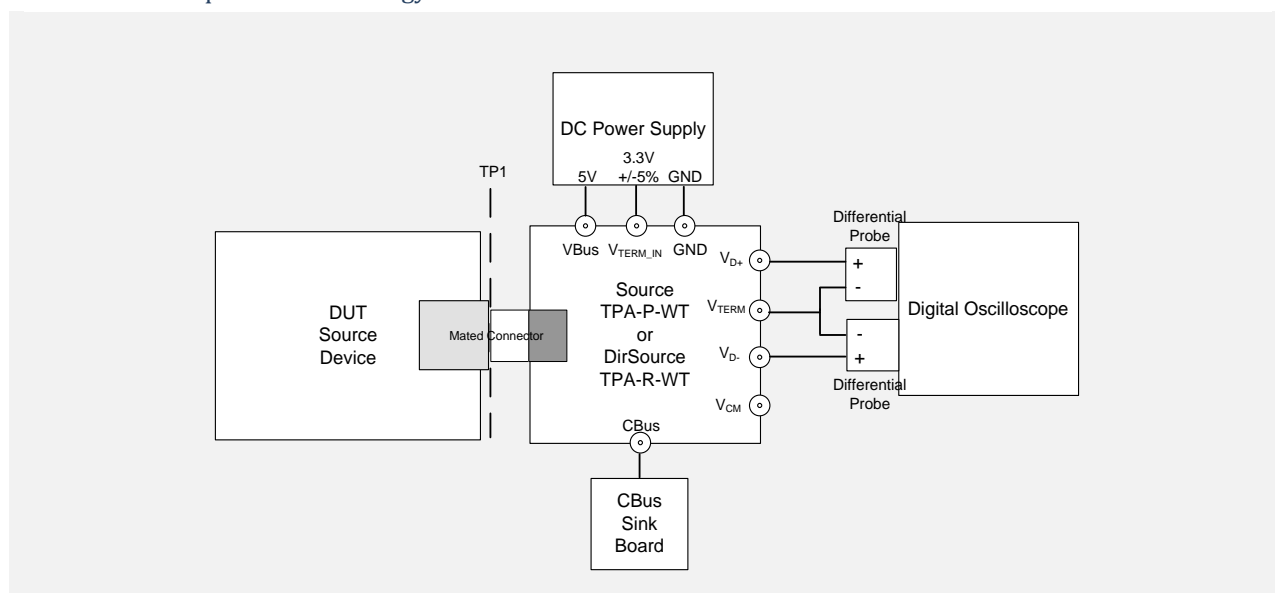


Figure 29. Source – TP2 Eye Diagram Test Setup

1. Connect the equipment as shown in Figure 29.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.

4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams in the lowest supported MHL data bit rate in Normal Mode.
5. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\} 3.135\text{V}$.
6. Probe the measurement points as shown in Figure 29.
7. Set the oscilloscope to the following conditions.
 - a. Sampling rate $\geq 20\text{Gs/sec}$
 - b. Sampling length $\geq 0.5\text{msec}$
 - c. Low pass filtering for the MHL Clock signal: $f_{3\text{dB}} = 500\text{MHz}$
 - d. Software CRU for the MHL Clock signal: 1st order, 3dB Bandwidth = 4MHz
 - e. Histogram Window Vertical Width = 5mV
8. Set the oscilloscope to convolve s-parameter of worst cable model whose intra-pair skew is +43ps.
9. Measure and record the eye diagram with reference equalizer.
10. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\} 3.465\text{V}$.
11. Repeat step 9.
12. Set the oscilloscope to convolve s-parameter of worst cable model whose intra-pair skew is -43ps and set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\} 3.135\text{V}$.
13. Repeat steps 9 – 11.
14. Repeat steps 5 – 13 for the highest supported MHL data bit rate in Normal Mode.
15. If all recorded Data Eye diagrams meet the MHL Data Eye diagram spec, then PASS. Otherwise FAIL.

3.1.1.19 TP2 Clock Jitter in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.19.1 Test Objective

This test confirms that the MHL Clock output after a compliant cable does not contain excessive jitter larger than the limit allowed by the specification in PackedPixel Mode.

3.1.1.19.2 References

[MHL] Figure 13-16 MHL Clock Jitter at TP2:

$$T_{\text{CLK_JITTER_TP2}} \leq 0.3T_{\text{BIT}} + 266.7\text{ps, up to } 2.25\text{Gbps}$$

$$T_{\text{CLK_JITTER_TP2}} \leq 0.9T_{\text{BIT}}, \text{ above } 2.25\text{Gbps}$$

3.1.1.19.3 Required Test Equipment

High-bandwidth Digital Oscilloscope
High-Bandwidth Probe
Jitter and Eye Analyzer
CBUS Sink Board
Source TPA-P-WT or DirSource TPA-R-WT
DC Power Supply

3.1.1.19.4 Required Methodology

1. Connect the equipment as shown in Figure 29.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.

4. Set the DUT so that the DUT outputs MHL Gray Ramp data streams data in the highest supported MHL data bit rate in PackedPixel Mode.
5. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\} 3.135\text{V}$.
6. Probe the measurement points as shown in Figure 29.
7. Set the oscilloscope to the following conditions.
 - a. Sampling rate $\geq 20\text{Gs/sec}$
 - b. Sampling length $\geq 0.5\text{msec}$
 - c. Low pass filtering for the MHL Clock signal: $f_{3\text{dB}} = 500\text{MHz}$
 - d. Software CRU for the MHL Clock signal: 1st order, 3dB Bandwidth = 4MHz
 - e. Reference equalizer for the MHL Clock signal
 - f. Histogram Window Vertical Width = 5mV
8. Set the oscilloscope to convolve s-parameter of worst cable model whose intra-pair skew is +43ps.
9. Measure the MHL clock jitter on the rising edge and record the average value over ten measurements for the given V_{TERM} and MHL data bit rate.
10. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\} 3.465\text{V}$.
11. Repeat step 9.
12. Set the oscilloscope to convolve s-parameter of worst cable model whose intra-pair skew is -43ps and set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\} 3.135\text{V}$.
13. Repeat steps 9 – 11.
14. If all recorded jitter values meet the MHL Clock Jitter spec, then PASS. Otherwise FAIL.

3.1.1.20 TP2 Eye Diagram in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.1.1.20.1 Test Objective

This test confirms that the MHL output after a compliant cable has signal quality that meets the eye opening required by the specification in PackedPixel Mode.

3.1.1.20.2 References

[MHL] Figure 13-17 MHL Data Eye Mask at TP2

[MHL] Figure 13-18 MHL Data Jitter at TP2

3.1.1.20.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

Jitter and Eye Analyzer

CBUS Sink Board

Source TPA-P-WT or DirSource TPA-R-WT

DC Power Supply

3.1.1.20.4 Required Methodology

1. Connect the equipment as shown in Figure 29.
2. Turn on the DC Power Supply.
3. Turn on the DUT and enable MHL output signal through Device Discovery and CBUS Information Exchange with the CBUS Sink Board.

4. Set the DUT so that the DUT outputs MHL Gray Ramp data in the highest supported MHL data bit rate in PackedPixel Mode.
5. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\} 3.135\text{V}$.
6. Probe the measurement points as shown in Figure 29.
7. Set the oscilloscope to the following conditions.
 - a. Sampling rate $\geq 20\text{Gs/sec}$
 - b. Sampling length $\geq 0.5\text{msec}$
 - c. Low pass filtering for the MHL Clock signal: $f_{3\text{dB}} = 500\text{MHz}$
 - d. Software CRU for the MHL Clock signal: 1st order, 3dB Bandwidth = 4MHz
 - e. Histogram Window Vertical Width = 5mV
8. Set the oscilloscope to convolve s-parameter of worst cable model whose intra-pair skew is +43ps.
9. Measure and record the eye diagram with reference equalizer.
10. Set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{max}\} 3.465\text{V}$.
11. Repeat step 9.
12. Set the oscilloscope to convolve s-parameter of worst cable model whose intra-pair skew is -43ps and set $V_{\text{TERM}} = V_{\text{TERM}}\{\text{min}\} 3.135\text{V}$.
13. Repeat steps 9- 11.
14. If all recorded Data Eye diagrams meet the MHL Data Eye diagram spec, then PASS. Otherwise FAIL.

3.2 System Tests

3.2.1 General Test Setups

The equipment setups shown in Section 2.3.2 are used throughout Section 3.2.

3.2.2 TMDS Coding

3.2.2.1 Legal Codes in Normal Mode

| Application: | Source |
|--------------|--------|
|--------------|--------|

3.2.2.1.1 Test Objective

Confirm that the source DUT never transmits illegal 10-bit codes in Normal Mode.

3.2.2.1.2 References

[MHL] 4.1.4 Character Synchronization

[MHL] 4.2.1.1.1 Guard Bands

[MHL] 4.4.3 Control Period Coding

[MHL] 4.4.4 TERC4 Coding

[MHL] 4.4.5 Video Data Coding

[MHL] 4.3 Guard Band Values

3.2.2.1.3 Required Test Equipment

MHL Protocol Analyzer

A/V Display

3.2.2.1.4 Required Methodology

1. If **CDF_VIDEO_480p_60** field in the CDF is “NO” and if **CDF_VIDEO_576p_50** field in the CDF is “NO”, then FAIL, else continue to test.
2. Configure DUT to send out one of the video formats indicated in the CDF which use Normal Mode, while also transmitting 2-channel PCM audio if supported.
3. Verify that all transmitted 10-bit codes, within an analysis period of 2 seconds, belong to one of the following groups:
 - Guard Band codes: as listed in Table 4-3 of the MHL Specification.
 - 4 Control Period codes
 - 16 TERC4 codes
 - legal Video Data codes (there are 460 valid video data codes)
 If any of the transmitted 10-bit codes does not belong to one of these groups, then FAIL, else continue to test.
4. For Video Data codes, if any of the codes has an incorrect “data stream disparity” value, then FAIL, else continue to test.
5. For each pixel, if any of the multiplexed (i.e., logical) channels belong to different groups, then FAIL, else continue to test.
6. Repeat the test from step 2 for all video formats indicated in the CDF which use Normal Mode.
7. If all video formats specified in the CDF which use Normal Mode are verified, then PASS.

3.2.2.2 Basic Protocol in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.2.2.1 Test Objective

Confirm that the source DUT always transmits code sequences for Control Periods, Data Island Periods, and Video Data Periods conforming to the basic protocol requirements when in Normal Mode.

3.2.2.2.2 References

[MHL] 4.2 Operating Modes

3.2.2.2.3 Required Test Equipment

MHL Protocol Analyzer

A/V Display

3.2.2.2.4 Required Methodology

1. Configure DUT to send out one of the Normal Mode Pixel Video formats indicated in the CDF, while also transmitting 2-channel PCM audio if supported.
2. For each transition from a character in Control Period encoding to a character in non-Control Period Encoding.
 - a. If the 12 pixels prior to the transition contain any pixels not encoded with Control Period Coding, then FAIL (Control Period too short).
 - b. Examine the CTL3:CTL2:CTL1:CTL0 values for the 8 (Control-encoded) pixels immediately prior to the transition and compare to the values 0b0001 (Video Data Period Preamble) and 0b0101 (Data Island Preamble).
 - c. [Check for Invalid Data Island Preamble control code usage]
Examine whole control period prior to the Preamble. If the period includes Data Island Preamble control code (CTL0:3=1010), then FAIL.

- d. [Inconsistent Preamble]
If any of the 8 pixels does not match the CTLx value for any of the other 7 pixels, then FAIL.
- e. [Illegal Preamble]
If the Preamble value is neither Data Island Preamble nor Video Data Preamble, then FAIL.
- f. If the Preamble value is Data Island Preamble:
 - i. Examine the first two pixels following the Preamble (Leading Guard Band).
If Logical Channel 0 for either of these pixels does not equal one of the 4 permitted Data Island Guard Band characters (0xC, 0xD, 0xE, 0xF), then FAIL.
 - ii. If Logical Channel 1 or 2 for either of these pixels does not equal the specified Data Island Guard Band character, then FAIL.
 - iii. Scan through following pixels, while counting pixels, until finding a transition to Control Period Coding.
If any character is not encoded with Data Island Coding, then FAIL.
 - iv. Examine the last two pixels preceding this transition (Trailing Guard Band).
If Logical Channel 0 for either of these pixels does not equal one of the 4 permitted Data Island Guard Band characters (0xC, 0xD, 0xE, 0xF), then FAIL.
 - v. If Logical Channel 1 or 2 for either of these pixels does not equal the specified Data Island Guard Band character, then FAIL.
 - vi. If any character following the Leading Guard Band but preceding the Trailing Guard Band is not a legal TERC4 code, then FAIL.
 - vii. If first character following the Leading Guard Band has TERC4 ch. 0, bit 3 == 1, then FAIL.
 - viii. If any other character prior to Trailing Guard Band has TERC4 ch. 0, bit 3 != 1, then FAIL.
 - ix. Length of Data Island is equal to number of pixels following Leading Guard Band and prior to Trailing Guard Band. Number of packets = Length of Data Island / 32.
If number of packets is not an integer, or number of packets == 0, or number of packets >18, then FAIL.
- g. For every packet within the Data Island:
 - i. For each of the 5 ECC blocks within the packet, if BCH parity bits are incorrect, then FAIL.
- h. If the Preamble value is Video Data Preamble:
 - i. Examine the first two pixels following the Preamble (Leading Guard Band).
If either of these pixels does not equal Video Data Guard Band character, then FAIL.
 - ii. Scan through following pixels until finding a transition to Control Period Coding.
If any character is not encoded with Video Data Coding, then FAIL.
3. Repeat the test from step 1 for all Normal Mode video formats indicated in the CDF.
4. If all video formats specified in the CDF are verified, then PASS.

3.2.2.3 Packet Types in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.2.3.1 Test Objective

Confirm that the packets transmitted by the source DUT only are in the specified format when in Normal Mode.

3.2.2.3.2 References

[MHL] 4.3 Data Island Packet Definitions
[MHL] 10.2 EIA / CEA-861E Info Frames

3.2.2.3.3 Required Test Equipment

MHL Protocol Analyzer
A/V Display

3.2.2.3.4 Required Methodology

1. Configure DUT to send out one of the Normal Mode video formats indicated in the CDF, while also transmitting 2-channel PCM audio if supported.
2. If no Data Island is detected at least once per two video fields, then FAIL.
3. For each transmitted packet:
 - a. If Packet Type is 0x00 (Null Packet), then
 - if any of the bytes HB1, HB2, or any of the bytes in the Packet Body is not equal to 0x00, then FAIL.
 - b. If Packet Type is 0x01 (Audio Clock Regeneration Packet), then
 - if any of the bytes HB1, HB2, PB0 is not equal to 0x00, then FAIL.
 - if any of the bits [7:4] in bytes PB1, PB4 is not equal 0b0, then FAIL.
 - if the Subpacket in bytes PB7 to PB13 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - if the Subpacket in bytes PB14 to PB20 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - if the Subpacket in bytes PB21 to PB27 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - c. If Packet Type is 0x02 (Audio Sample Packet), then
 - if any of the bits [7:5] in byte HB1 is not equal 0b0, then FAIL.
 - d. If Packet Type is 0x03 (Content Mute Packet), then
 - if any of the bytes HB1, HB2, PB1, PB2, PB3, PB4, PB5, PB6 is not equal to 0x00, then FAIL.
 - if any of the bits [7:5] in byte PB0 is not equal 0b0, then FAIL.
 - if any of the bits [3:1] in byte PB0 is not equal 0b0, then FAIL.
 - if bit 4 in byte PB0 is equal to 0b1 and bit 0 in byte PB0 is equal to 0b1, then FAIL.
 - if the Subpacket in bytes PB7 to PB13 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - if the Subpacket in bytes PB14 to PB20 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - if the Subpacket in bytes PB21 to PB27 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - e. If Packet Type is 0x82 (AVI InfoFrame), then
 - if byte HB1 is not equal 0x02, then FAIL.
 - if byte HB2 is not equal 0x0D, then FAIL.
 - if bit 7 of bytes PB1, PB4, or any of the bits [7:2] of byte PB3, or any of the bits [7:4] of byte PB5 is not equal 0b0, then FAIL.
 - if any of the bytes PB14 to PB27 is not equal 0x00, then FAIL.
 - f. If Packet Type is 0x84 (Audio InfoFrame), then:
 - if byte HB1 is not equal 0x01, then FAIL.
 - if byte HB2 is not equal 0x0A, then FAIL.
 - if bit 3 of PB1 is not equal to 0b0, then FAIL.
 - if byte PB2 is not equal to 0x00, then FAIL.
 - if bits [7:5] of PB3 are not 0b000, then FAIL.
 - if bit 7 of PB5 is not equal to 0b0, then FAIL.
 - if bit 2 of PB5 is not equal to 0b0, then FAIL.
 - if the two bit field in bits [1:0] of PB5 (LFEPBL in CEA-861E) is equal to 0b11, then FAIL.
 - if any of the bytes PB6 to PB27 is not equal 0x00, then FAIL.
 - g. If Packet Type is not [0x00 to 0x03 or 0x81 to 0x87] (i.e. unknown Packet Type), then FAIL.
4. Repeat the test from step 1 for all Normal Mode video formats indicated in the CDF.
5. If all video formats specified in the CDF which use Normal Mode are verified, then PASS.

3.2.2.4 Legal Codes in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.2.4.1 Test Objective

Confirm that the source DUT never transmits illegal 10-bit codes in PackedPixel Mode.

3.2.2.4.2 References

[MHL] 4.1.4 Character Synchronization

[MHL] 4.2.2.1.1 Guard Bands (PackedPixel Mode)

[MHL] 4.4.3 Control Period Coding

[MHL] 4.4.4 TERC4 Coding

[MHL] 4.4.5 Video Data Coding

[MHL] 4.3 Guard Band Values

3.2.2.4.3 Required Test Equipment

MHL Protocol Analyzer

A/V Display

3.2.2.4.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is “Yes”, then continue test; else end with PASS (SKIP).
2. Configure DUT to send out one of the video formats indicated in the CDF which use PackedPixel Mode, while also transmitting 2-channel PCM audio if supported.
3. Verify that all transmitted 10-bit codes, within an analysis period of 2 seconds, belong to one of the following groups:
 - Guard Band codes: as listed in Table 4-3 of the MHL Specification.
 - 4 Control Period codes
 - 16 TERC4 codes
 - legal Video Data codes (there are 460 valid video data codes)If any of the transmitted 10-bit codes does not belong to one of these groups, then FAIL, else continue to test.
4. For Video Data codes, if any of the codes has an incorrect “data stream disparity” value, then FAIL, else continue to test.
5. For each pixel, if any of the multiplexed (i.e., logical) channels belong to different groups, then FAIL, else continue to test.
6. Repeat the test from step 3 for all video formats indicated in the CDF which use PackedPixel Mode.
7. If all video formats specified in the CDF which use PackedPixel Mode are verified, then PASS.

3.2.2.5 Basic Protocol in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.2.5.1 Test Objective

Confirm that the source DUT always transmits code sequences for Control Periods, Data Island Periods, and Video Data Periods conforming to the basic protocol requirements when in PackedPixel Mode.

3.2.2.5.2 References

[MHL] 4.2 Operating Modes

3.2.2.5.3 Required Test Equipment

MHL Protocol Analyzer

A/V Display

3.2.2.5.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is “Yes”, then continue test; else end with PASS (SKIP).
2. Configure DUT to send out one of the PackedPixel Mode video formats indicated in the CDF, while also transmitting 2-channel PCM audio if supported.
3. For each transition from a character in Control Period encoding to a character in non-Control Period Encoding.
 - a. If the 12 pixels prior to the transition contain any pixels not encoded with Control Period Coding, then FAIL (Control Period too short).
 - b. Examine the CTL3:CTL2 values for the 8 (Control-encoded) pixels immediately prior to the transition and compare to the values 0b00 (Video Data Period Preamble) and 0b01 (Data Island Preamble).
 - c. [Check for Invalid Data Island Preamble control code usage]
Examine whole control period prior to the Preamble. If the period includes Data Island Preamble control code (CTL2:3=10), then FAIL.
 - d. [Inconsistent Preamble]
If any of the 8 pixels does not match the CTLx value for any of the other 7 pixels, then FAIL.
 - e. [Illegal Preamble]
If the Preamble value is neither Data Island Preamble nor Video Data Preamble, then FAIL.
 - f. If the Preamble value is Data Island Preamble:
 - i. Examine the first two pixels following the Preamble (Leading Guard Band).
If Logical Channel 0 for either of these pixels does not equal one of the 4 permitted Data Island Guard Band characters (0xC, 0xD, 0xE, 0xF), then FAIL.
 - ii. If Logical Channel 1 for either of these pixels does not equal the specified Data Island Guard Band character, then FAIL.
 - iii. Scan through following pixels, while counting pixels, until finding a transition to Control Period Coding.
If any character is not encoded with Data Island Coding, then FAIL.
 - iv. Examine the last two pixels preceding this transition (Trailing Guard Band).
If Logical Channel 0 for either of these pixels does not equal one of the 4 permitted Data Island Guard Band characters (0xC, 0xD, 0xE, 0xF), then FAIL.
 - v. If Logical Channel 1 for either of these pixels does not equal the specified Data Island Guard Band character, then FAIL.
 - vi. If any character following the Leading Guard Band but preceding the Trailing Guard Band is not a legal TERC4 code, then FAIL.
 - vii. If first character following the Leading Guard Band has TERC4 ch. 0, bit 3 == 1, then FAIL.
 - viii. If any other character prior to Trailing Guard Band has TERC4 ch. 0, bit 3 != 1, then FAIL.
 - ix. Length of Data Island is equal to number of pixels following Leading Guard Band and prior to Trailing Guard Band. Number of packets = Length of Data Island / 32.
If number of packets is not an integer, or number of packets == 0, or number of packets >18, then FAIL.
 - g. For every packet within the Data Island:
 - i. For each of the 5 ECC blocks within the packet, If BCH parity bits are incorrect, then FAIL.
 - h. If the Preamble value is Video Data Preamble:
 - i. Examine the first two pixels following the Preamble (Leading Guard Band).
If either of these pixels does not equal Video Data Guard Band character, then FAIL.

- ii. Scan through following pixels until finding a transition to Control Period Coding.
If any character is not encoded with Video Data Coding, then FAIL.
4. Repeat the test from step 2 for all PackedPixel Mode video formats indicated in the CDF.
5. If all video formats specified in the CDF which use PackedPixel Mode are verified, then PASS.

3.2.2.6 Packet Types in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.2.6.1 Test Objective

Confirm that the packets transmitted by the source DUT only are in the specified format when in PackedPixel Mode.

3.2.2.6.2 References

[MHL] 4.3 Data Island Packet Definitions
[MHL] 10.2 EIA / CEA-861E Info Frames

3.2.2.6.3 Required Test Equipment

MHL Protocol Analyzer
A/V Display

3.2.2.6.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is "Yes", then continue test; else end with PASS (SKIP).
2. Configure DUT to send out one of the PackedPixel Mode video formats indicated in the CDF, while also transmitting 2-channel PCM audio if supported.
3. If no Data Island is detected at least once per two video fields, then FAIL.
4. For each transmitted packet:
 - a. If Packet Type is 0x00 (Null Packet) then
 - if any of the bytes HB1, HB2, or any of the bytes in the Packet Body is not equal to 0x00, then FAIL.
 - b. If Packet Type is 0x01 (Audio Clock Regeneration Packet), then
 - if any of the bytes HB1, HB2, PB0 is not equal to 0x00, then FAIL.
 - if any of the bits [7:4] in bytes PB1, PB4 is not equal 0b0, then FAIL.
 - if the Subpacket in bytes PB7 to PB13 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - if the Subpacket in bytes PB14 to PB20 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - if the Subpacket in bytes PB21 to PB27 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - c. If Packet Type is 0x02 (Audio Sample Packet), then
 - if any of the bits [7:5] in byte HB1 is not equal 0b0, then FAIL.
 - d. If Packet Type is 0x03 (Content Mute Packet), then
 - if any of the bytes HB1, HB2, PB1, PB2, PB3, PB4, PB5, PB6 is not equal to 0x00, then FAIL.
 - if any of the bits [7:5] in byte PB0 is not equal 0b0, then FAIL.
 - if any of the bits [3:1] in byte PB0 is not equal 0b0, then FAIL.
 - if bit 4 in byte PB0 is equal to 0b1 and bit 0 in byte PB0 is equal to 0b1, then FAIL.
 - if the Subpacket in bytes PB7 to PB13 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
 - if the Subpacket in bytes PB14 to PB20 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.

- if the Subpacket in bytes PB21 to PB27 is not equal to the Subpacket in bytes PB0 to PB6, then FAIL.
- e. If Packet Type is 0x82 (AVI InfoFrame), then
 - if byte HB1 is not equal 0x02, then FAIL.
 - if byte HB2 is not equal 0x0D, then FAIL.
 - if bit 7 of bytes PB1, PB4, or any of the bits [7:2] of byte PB3, or any of the bits [7:4] of byte PB5 is not equal 0b0, then FAIL.
 - if any of the bytes PB14 to PB27 is not equal 0x00, then FAIL.
- f. If Packet Type is 0x84 (Audio InfoFrame), then:
 - if byte HB1 is not equal 0x01, then FAIL.
 - if byte HB2 is not equal 0x0A, then FAIL.
 - if bit 3 of PB1 is not equal to 0b0, then FAIL.
 - if byte PB2 is not equal to 0x00, then FAIL.
 - if bits [7:5] of PB3 are not 0b000, then FAIL.
 - if bit 7 of PB5 is not equal to 0b0, then FAIL.
 - if bit 2 of PB5 is not equal to 0b0, then FAIL.
 - if the two bit field in bits [1:0] of PB5 (LFEPBL in CEA-861E) is equal to 0b11, then FAIL.
 - if any of the bytes PB6 to PB27 is not equal 0x00, then FAIL.
- g. If Packet Type is not [0x00 to 0x03 or 0x81 to 0x87] (i.e. unknown Packet Type), then FAIL.
- 5. Repeat the test from step 2 for all PackedPixel Mode video formats indicated in the CDF.
- 6. If all video formats specified in the CDF which use PackedPixel Mode are verified, then PASS.

3.2.3 Video Modes

NOTE: Individual tests in section 3.2.3 confirm aspects of proper video mode handling. Users should perform the tests in this group as a suite of tests to assure coverage of all aspects of video mode handling.

3.2.3.1 Video Formats in Normal Mode

Application: **Source**

3.2.3.1.1 Test Objective

Verify that Source supports the required video formats and the optional video formats which use the Normal Mode, compliant with the EIA/CEA-861E specification.

3.2.3.1.2 References

[MHL] Section 5.2.1.1 MHL Source Video Format Support Requirements

3.2.3.1.3 Required Test Equipment

MHL Protocol Analyzer

3.2.3.1.4 Required Methodology

1. If **CDF_VIDEO_480p_60** field in the CDF is “NO” and if **CDF_VIDEO_576p_50** field in the CDF is “NO”, then FAIL, else continue to test.
2. Configure DUT to send out one of the video formats indicated in the CDF which uses Normal Mode.
3. Measure:
 - a. H-front porch, H-sync pulse width, H-back porch, H-active, H-sync polarity.
 - b. V-front porch, V-sync pulse width, V-back porch, V-active, V-sync polarity.
 - c. Clock frequency, offset of the active edge of V-sync from the active edge of H-sync.

4. Compare the measurement values with the video format timings defined in the EIA/CEA-861E specification. If any value does not equal the proper values for the select video format, then FAIL.
5. Examine if the measured clock frequency is within +0.5%/-0.6% range of the pixel clock frequency defined in the EIA/CEA-861E if vertical frequency is either an integer multiple of 6 Hz or an integer multiple of 6 Hz adjusted by factor of 1000/1001, or if the measured clock frequency is within $\pm 0.5\%$ range if the vertical frequency does not fall in those two cases. If the pixel clock frequency is out of the allowable range, then FAIL.
6. Repeat from step 2 until all video formats indicated in the CDF which use Normal Mode are tested.
7. If all video formats specified in the CDF which use Normal Mode are verified, then PASS.

3.2.3.2 Pixel Encoding in Normal Mode

| Application: | Source |
|--------------|--------|
|--------------|--------|

3.2.3.2.1 Test Objective

Verify that Source is capable of supporting RGB pixel encoding, and YCbCr 4:4:4 Encoding.

3.2.3.2.2 References

[MHL] Section 5.2.3 24-Bit Pixel Encoding Requirement.

[MHL] Section 5.5 Pixel Encodings.

3.2.3.2.3 Required Test Equipment

MHL Protocol Analyzer

3.2.3.2.4 Required Methodology

1. If **CDF_VIDEO_RGB** field in the CDF is "YES", then continue to test, else FAIL.
2. Configure DUT's output mode as 720x480p 59.94/60Hz or 720x576p 50Hz.
3. Configure DUT's output mode as RGB pixel encoding.
4. Configure DUT to send out fully saturated black and white chess board pattern.
5. If video data of the fully saturated black and white chess board pattern is detected as RGB pixel encoding, then continue to test, else FAIL.
6. If **CDF_VIDEO_YCBCR_444** and **CDF_VIDEO_YCBCR_422** field in the CDF is "NO" and if source is capable of sending out color-difference color space across any other analog or digital video interface, then FAIL, else continue to test.
7. If **CDF_VIDEO_YCBCR_422** field in the CDF is "YES", then configure DUT's output mode as YCbCr 4:2:2 pixel encoding.
8. Configure DUT to send out fully saturated black and white chess board pattern.
9. If video data of the fully saturated black and white chess board pattern is detected as YCbCr 4:2:2 encoding, then continue to test, else FAIL.
10. If **CDF_VIDEO_YCBCR_444** field in the CDF is "YES", then repeat the test from step 8 with **CDF_VIDEO_YCBCR_444**.
11. If all Pixel Encodings indicated in the CDF is verified, then PASS.

3.2.3.3 AVI InfoFrame in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.3.3.1 Test Objective

Verify that source transmit an accurate AVI InfoFrame at least once every two video fields for video modes using Normal Mode.

3.2.3.3.2 References

[MHL] Section 10.2.1. AVI InfoFrame.

3.2.3.3.3 Required Test Equipment

MHL Protocol Analyzer

3.2.3.3.4 Required Methodology

1. If **CDF_AVI_SUPPORT** field in the CDF is "NO" and if any of **CDF_VIDEO_480i_60_2X**, **CDF_VIDEO_480i_60_4X**, **CDF_VIDEO_480p_60**, **CDF_VIDEO_480p_60_2X**, **CDF_VIDEO_576i_50_2X**, **CDF_VIDEO_576i_50_4X**, **CDF_VIDEO_576p_50**, **CDF_VIDEO_576p_50_2X**, **CDF_VIDEO_YCBCR_444**, **CDF_VIDEO_YCBCR_422** field in the CDF is "YES" then FAIL, else continue to test.
2. If **CDF_AVI_SUPPORT** field in the CDF is "Yes", then continue to test, else end the test with PASS (SKIP).
3. Connect DUT to a Pseudo MHL sink device.
4. Configure DUT as one of the video formats indicated in the CDF which use Normal Mode.
5. Configure DUT to send out fully saturated black and white chess board pattern.
6. If DUT transmits AVI InfoFrame at least once every two video fields, then continue to test, else FAIL.
7. If HB1 is 0x02, then continue to test, else FAIL.
8. If Y1, Y0 bits match the RGB or YCbCr pixel encoding, then continue to test, else FAIL.
9. If S1, S0 bits match the overscan or underscan or no {S1,S0} data, then continue to test, else FAIL.
10. If PR3, PR2, PR1, PR0 bits match the Pixel Repetition Factor, then continue to test, else FAIL.
11. If VIC6-VIC0 bits match the video format, then continue to test, else FAIL.
12. If all reserved fields are 0, then continue to test, else FAIL.
13. Repeat the test from step 4 for all video formats indicated in the CDF which use Normal Mode.
14. If any of video formats indicated in the CDF which use Normal Mode fails then FAIL, else PASS.

3.2.3.4 Video Quantization Ranges

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.3.4.1 Test Objective

Verify that Source complies with quantization ranges and QY and QS bits in the Video Capability Data Block defined in EIA/CEA-861E.

3.2.3.4.2 References

[MHL] Section 5.6 Video Quantization Ranges.

3.2.3.4.3 Required Test Equipment

MHL Protocol Analyzer

3.2.3.4.4 Required Methodology

1. If **CDF_VIDEO_YCBCR_444** or **CDF_VIDEO_YCBCR_422** field in the CDF is "YES", then continue to test, else move to step 2.
 - a. If **CDF_AVI_SUPPORT** is "YES", then continue to test, else FAIL.
 - b. Configure pseudo MHL sink device QY bit to '0'.
 - c. Configure DUT with YCbCr 4:2:2 or YCbCr 4:4:4 pixel encoding in 720x480p 59.94/60Hz or 720x576p 50Hz format.
 - d. Configure DUT to send out fully saturated white and black chess board pattern.
 - e. Examine the YQ1 and YQ0 fields in the AVI InfoFrame. If {YQ1, YQ0} are {0,0}, then continue to test, else FAIL.
 - f. Examine the Y component of the video that the DUT sends out. If the MSB 8 bits of the Y component in any area of the pattern is 0 or 255, then FAIL, else continue to test.
 - g. If **CDF_SOURCE_YQ_FULL** is "YES", then continue to test, else move to step 2.
 - h. Configure pseudo MHL sink device QY bit to '1'.
 - i. Configure DUT to send out fully saturated white and black chess board pattern.
 - j. Examine the YQ1 and YQ0 fields in the AVI InfoFrame. If {YQ1, YQ0} are {0,1} then continue to test, else FAIL.
2. Configure DUT with RGB pixel encoding in 720x480p 59.94/60Hz or 720x576p 50Hz format.
 - a. Configure DUT to send out fully saturated white and black chess board pattern.
 - b. Examine the R, G and B components of the video that DUT sends out. If the MSB 8 bits of any R, G, or B component in any area of the pattern is 0 or 255, then FAIL. Else PASS.

3.2.3.5 Video Formats in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.3.5.1 Test Objective

Verify that Source supports the optional video formats which use PackedPixel mode, compliant with the EIA/CEA-861E specification.

3.2.3.5.2 References

[MHL] Section 5.2.1.1 MHL Source Video Format Support Requirements

3.2.3.5.3 Required Test Equipment

MHL Protocol Analyzer

3.2.3.5.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is "YES", then continue to test, else PASS (SKIP).
2. Configure DUT to send out one of the video formats indicated in the CDF which uses PackedPixel Mode.
3. Measure:
 - a. H-front porch, H-sync pulse width, H-back porch, H-active, H-sync polarity.
 - b. V-front porch, V-sync pulse width, V-back porch, V-active, V-sync polarity.
 - c. Clock frequency, offset of the active edge of V-sync from the active edge of H-sync.
4. Compare the measurement values with the video format timings defined in the EIA/CEA-861E specification. If any value does not equal the proper values for the select video format, then FAIL.

5. Examine if the measured clock frequency is within +0.5%/-0.6% range of the pixel clock frequency defined in the EIA/CEA-861E if vertical frequency is either an integer multiple of 6 Hz or an integer multiple of 6 Hz adjusted by factor of 1000/1001, or if the measured clock frequency is within $\pm 0.5\%$ range if the vertical frequency does not fall in those two cases. If the pixel clock frequency is out of the allowable range, then FAIL.
6. Repeat from step 2 until all video formats indicated in the CDF which use PackedPixel Mode are tested.
7. If all video formats specified in the CDF which use PackedPixel Mode are verified, then PASS; else FAIL.

3.2.3.6 Pixel Encoding in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.3.6.1 Test Objective

Verify that Source is capable of supporting PackedPixel Encoding.

3.2.3.6.2 References

[MHL] Section 5.2.4 PackedPixel Encoding Requirement.

[MHL] Section 5.5 Pixel Encodings.

3.2.3.6.3 Required Test Equipment

MHL Protocol Analyzer

3.2.3.6.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is "YES", then continue to test, else PASS (SKIP).
2. Configure DUT's output mode as 720x480p 59.94/60Hz or 720x576p 50Hz.
3. Configure DUT's output mode as PackedPixel encoding.
4. Configure DUT to send out fully saturated black and white chess board pattern.
5. If video data of the fully saturated black and white chess board pattern is detected as PackedPixel encoding, then PASS, else FAIL.

3.2.3.7 AVI InfoFrame in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.3.7.1 Test Objective

Verify that source transmit an accurate AVI InfoFrame at least once every two video fields for video modes using PackedPixel Mode.

3.2.3.7.2 References

[MHL] Section 10.2.1. AVI InfoFrame.

3.2.3.7.3 Required Test Equipment

MHL Protocol Analyzer

3.2.3.7.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is "YES", then continue to test, else PASS (SKIP).
2. If **CDF_AVI_SUPPORT** field in the CDF is "NO" and if **CDF_VIDEO_PACKEDPIXEL** field in the CDF is "YES" then FAIL, else continue to test.

3. If **CDF_AVI_SUPPORT** field in the CDF is "Yes", then continue to test, else end the test with PASS (SKIP).
4. Connect DUT to a Pseudo MHL sink device.
5. Configure DUT as one of the video formats indicated in the CDF which use PackedPixel Mode.
6. Configure DUT to send out fully saturated black and white chess board pattern.
7. If DUT transmits AVI InfoFrame at least once every two video fields, then continue to test, else FAIL.
8. If HB1 is 0x02, then continue to test, else FAIL.
9. If Y1, Y0 bits match the RGB or YCbCr pixel encoding, then continue to test, else FAIL.
10. If S1, S0 bits match the overscan or underscan or no {S1,S0} data, then continue to test, else FAIL.
11. If PR3, PR2, PR1, PR0 bits match the Pixel Repetition Factor, then continue to test, else FAIL.
12. If VIC6-VIC0 bits match the video format, then continue to test, else FAIL.
13. If all reserved fields are 0, then continue to test, else FAIL.
14. Repeat the test from step 4 for all video formats indicated in the CDF which use PackedPixel Mode.
15. If any of video formats indicated in the CDF which use PackedPixel Mode fails then FAIL, else PASS.

3.2.4 Audio Test

3.2.4.1 IEC 60958 / IEC61937

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.4.1.1 Test Objective

Verify that Source is capable of audio transmission that is complied with either IEC60958 format or IEC61937 format.

3.2.4.1.2 References

[MHL] Section 6.3 Audio Sample Rates and Support Requirements.

3.2.4.1.3 Required Test Equipment

MHL Protocol Analyzer

3.2.4.1.4 Required Methodology

1. If **CDF_AUDIO_2CH_32kHz** and **CDF_AUDIO_2CH_44kHz** and **CDF_AUDIO_2CH_48kHz** fields in the CDF are all three set to "NO", and if the DUT has no other audio output, then end test with PASS (SKIP).
2. If **CDF_AUDIO_2CH_32kHz** and **CDF_AUDIO_2CH_44kHz** and **CDF_AUDIO_2CH_48kHz** fields in the CDF are all three set to "NO", and if the DUT has any other audio output, then FAIL.
3. Configure DUT's output as one of PCM_2CH_32kHz, PCM_2CH_44.1kHz or PCM_2CH_48kHz indicated in the CDF with an appropriate video format that can carry the Audio-mode.
4. Count audio samples by using Layout and sample_present bits in the header, and get Channel Status Bits.
 - a. If Layout Value is 1, then FAIL, else continue to test.
 - b. If repetition period of B bit is 192 Frames, then continue to test, else FAIL.
 - c. If Channel Status Bit 24, 25, 26, 27 matches sample frequency in the table below, then continue to test, else FAIL.
 - d. If Sample Size matches the **CDF_AUDIO_SAMPLE_SIZE** field in the CDF, then continue to test, else FAIL.
5. Repeat the test from step 3 for PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz if it is indicated in the CDF.
6. If any of Audio formats fails, then FAIL, else PASS.

| Channel Status Bit Number | | | | Sample Frequency |
|---------------------------|----|----|----|------------------|
| 24 | 25 | 26 | 27 | |
| 1 | 1 | 0 | 0 | 32 kHz |
| 0 | 0 | 0 | 0 | 44.1 kHz |
| 0 | 1 | 0 | 0 | 48 kHz |

3.2.4.2 Audio Clock Regeneration

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.4.2.1 Test Objective

Verify that N, CTS and audio sample frequency have correct correlation.

3.2.4.2.2 References

[MHL] Section 6.2 Audio Sample Clock Capture and Regeneration.

3.2.4.2.3 Required Test Equipment

MHL Protocol Analyzer

3.2.4.2.4 Required Methodology

1. If **CDF_AUDIO_2CH_32kHz**, **CDF_AUDIO_2CH_44kHz** or **CDF_AUDIO_2CH_48kHz** field in the CDF are "YES", then continue to test, else end test with PASS (SKIP).
2. Configure DUT's output mode as one of PCM_2CH_32kHz, PCM_2CH_44.1kHz or PCM_2CH_48kHz indicated in the CDF with an appropriate video format that can carry the Audio-mode.
3. Examine the Audio Clock Regeneration Packet.
 - a. If N value is greater than or equal to $\{128 \times F_s\}/1500$, and is less than or equal to $\{128 \times F_s\}/300$, then continue to test, else FAIL.
 - b. If Bits 28 and 29 of the Channel Status Bits in the audio stream are set to '1', and '0', respectively, then the expected accuracy is 50ppm:
 - i. If the average CTS value over more than 2 seconds of measurement meets the limitations of $\{\text{PixelClock} \times N\}/\{128 \times F_s\}$ within an accuracy of plus or minus 50 ppm, then PASS, else FAIL.
NOTE: PixelClock must be multiplied by the Pixel Repetition Factor.
 - c. Otherwise the expected accuracy is 100ppm:
 - i. If the average CTS value over more than 2 seconds of measurement meets the limitations of $\{\text{PixelClock} \times N\}/\{128 \times \text{actual audio sample rate}\}$ within an accuracy of plus or minus 100 ppm, then PASS, else FAIL.
NOTE: PixelClock must be multiplied by the Pixel Repetition Factor.
3. Repeat the test from step 2 for PCM_2CH_32kHz, PCM_2CH_44.1kHz or PCM_2CH_48kHz indicated in the CDF.
4. If any of Audio mode fails the test, then FAIL, else PASS.

3.2.4.3 Audio InfoFrame

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.4.3.1 Test Objective

Verify that whenever an active audio stream is being transmitted an accurate Audio InfoFrame is transmitted at least as often as T_{AIF} .

3.2.4.3.2 References

[MHL] Section 10.2.2 Audio InfoFrame.

3.2.4.3.3 Required Test Equipment

MHL Protocol Analyzer

3.2.4.3.4 Required Methodology

1. If **CDF_AUDIO_2CH_32kHz**, **CDF_AUDIO_2CH_44kHz** or **CDF_AUDIO_2CH_48kHz** field in the CDF are "YES", then continue to test, else end the test with PASS (SKIP).
2. Configure DUT's output mode as one of PCM_2CH_32kHz, PCM_2CH_44.1kHz or PCM_2CH_48kHz indicated in the CDF with an appropriate video format that can carry the Audio-mode.
3. If Audio InfoFrame of which HB0 is 0x84 is detected at least as often as T_{AIF} , then continue to test, else FAIL.
4. If HB1 – HB2 of the Audio InfoFrame Packet are 0x01 and 0xA (decimal 10) respectively, then continue test, else FAIL.
5. If CC2, CC1 and CC0 match the number of Audio Channel, then continue to test, else FAIL.
6. Examine CC2-CC0 and CA7-CA0. If they have valid combination, then continue to test, else FAIL.
7. If SF2, SF1 and SF0 are 0,0,0, then continue to test, else FAIL.
8. If LFEPBL0 is 0x01 and LFEPBL1 is 0x01 then FAIL, else continue to test.
9. If sum of all values is 0x00, then continue to test, else FAIL.
10. Repeat the test from step 2 for PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz indicated in the CDF.
11. If all Audio modes indicated in the CDF are verified, then PASS.

3.2.5 HDCP Test

3.2.5.1 HDCP

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.5.1.1 Test Objective

Verify that source complies with HDCP Specification.

3.2.5.1.2 References

[MHL] Section 11.3 HDCP Compliance.

3.2.5.1.3 Required Test Equipment

HDCP Analyzer

3.2.5.1.4 Required Methodology

1. If the **CDF_HDCP_SUPPORT** field in the CDF is "YES" then continue to test, else PASS (SKIP).

2. Follow the HDCP CTS procedure. If the **CDF_HDCP_REPEATER** field in the CDF is “YES”, then include all HDCP Repeater tests.
3. If all tests in the HDCP CTS procedure result in pass, then PASS; else FAIL.

3.2.6 EDID Test and Device Capability / Device Status Register Test

3.2.6.1 EDID Reading Test

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.6.1.1 Test Objective

Verify that DUT reads EDID while MHL link is being established and when EDID is updated.

3.2.6.1.2 References

[MHL] Section 10.3 EDID Access.

3.2.6.1.3 Required Test Equipment

MHL Protocol Analyzer

3.2.6.1.4 Required Methodology

1. Configure a Pseudo MHL sink device to have a valid 2 block EDID with CEA timing extension ver3.
2. Connect DUT to the Pseudo MHL sink device.
 - a. If DUT reads block 0 and block 1 while MHL link is being established, then continue to test, else FAIL.
3. Update EDID in the Pseudo MHL sink device, and indicate to the DUT by EDID_CHG interrupt bit in DCHANGE_INT register that EDID has changed.
 - a. If DUT reads block 0 and block 1 again, then continue to test, else FAIL.
4. Send CLR_HPD command to the DUT, wait $T_{HPD_WIDTH}\{min\}$, then send SET_HPD to the DUT.
 - a. If DUT reads block 0 and block 1 again, then continue to test, else FAIL.
5. Disconnect Pseudo MHL sink device from the DUT.
6. Configure a Pseudo MHL sink device to have a valid 4 block EDID with extension map and CEA timing extension ver3.
7. Connect DUT to the Pseudo MHL sink device.
8. If DUT reads block 0,1,2 and block 3 while MHL link is being established, then continue to test, else FAIL.
9. Update EDID in the Pseudo MHL sink device.
10. If DUT reads block 0,1,2 and block 3 again, then PASS, else FAIL.

3.2.6.2 Device Capability Registers Test

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.6.2.1 Test Description

Verify that Device Capability Registers have accurate value.

3.2.6.2.2 Reference

Device Capability Registers

3.2.6.2.3 Required Test Equipment

MHL Protocol Analyzer

3.2.6.2.4 Required Methodology

Perform the test in the following steps:

1. Connect DUT to MHL Protocol Analyzer.
2. Configure MHL Protocol Analyzer read DUT's MHL Capability Register.
3. Examine the registers and compare them with CDF.
 - a. If the values in MHL_VERSION(offset:0x01) register do not match the **CDF_CR_MHL_VER_MAJOR** and **CDF_CR_MHL_VER_MINOR** field in the CDF, then FAIL.
 - b. If DEV_TYPE in the DEV_CAT(offset:0x02) register is not 0b0010:Source, then FAIL.
 - c. If POW in the DEV_CAT(offset:0x02) register does not match the **CDF_CR_POW** field in CDF, then FAIL.
 - d. If ADOPTER_ID_H(offset:0x03) or ADOPTER_ID_L(offset:0x04) register does not match the corresponding **CDF_CR_ADOPTER_ID_H** and **CDF_CR_ADOPTER_ID_L** fields in the CDF, then FAIL.
 - e. If any of the SUPP_RGB444, SUPP_YCBCR444, SUPP_YCBCR422, SUPP_PPIXEL, SUPP_ISLANDS and SUPP_VGA bits in the VID_LINK_MODE(offset:0x05) register do not match the corresponding **CDF_CR_SUPP_RGB444**, **CDF_CR_SUPP_YCBCR444**, **CDF_CR_SUPP_YCBCR422**, **CDF_CR_SUPP_PPIXEL**, **CDF_CR_SUPP_ISLANDS** and **CDF_CR_SUPP_VGA** field in the CDF, then FAIL.
 - f. If AUD_2CH or AUD_8CH bits in the AUD_LINK_MODE(offset:0x06) do not match the corresponding **CDF_CR_AUD_2CH** and **CDF_CR_AUD_8CH** fields in the CDF, then FAIL.
 - g. If any of the LD_DISPLAY, LD_VIDEO, LD_AUDIO, LD_MEDIA, LD_TUNER, LD_RECORD, LD_SPEAKER, LD_GUI bits in the LOG_DEV_MAP(offset:0x06) register do not match the corresponding **CDF_CR_LD_DISPLAY**, **CDF_CR_LD_VIDEO**, **CDF_CR_LD_AUDIO**, **CDF_CR_LD_MEDIA**, **CDF_CR_LD_TUNER**, **CDF_CR_LD_RECORD**, **CDF_CR_LD_SPEAKER**, **CDF_CR_LD_GUI** fields in the CDF, then FAIL.
 - h. Check the following support bits in the FEATURE_FLAG register against the CDF:
 - If RCP_SUPPORT does not match **CDF_CR_RCP_SUPPORT**, then FAIL.
 - If RAP_SUPPORT does not match **CDF_CR_RAP_SUPPORT**, then FAIL.
 - If SP_SUPPORT does not match **CDF_CR_SP_SUPPORT**, then FAIL.
 - If UCP_SEND_SUPPORT does not match **CDF_CR_UCP_SEND_SUPPORT**, then FAIL.
 - If UCP_RECV_SUPPORT does not match **CDF_CR_UCP_RECV_SUPPORT**, then FAIL.
 - i. If DEVICE_ID_H(offset:0x0b) or DEVICE_ID_L(offset:0x0c) does not match the corresponding **CDF_CR_DEVICE_ID_H** and **CDF_CR_DEVICE_ID_L** field in the CDF, then FAIL.
 - j. If SP_SUPPORT bit is 0b1 and if SCRATCHPAD_SIZE(offset:0x0d) register does not match the **CDF_CR_SCRATCHPAD_SIZE** field in the CDF, then FAIL.
 - k. If INT_SIZE or STAT_SIZE fields in INT_STAT_SIZE register(offset:0x0e) do not match the corresponding **CDF_CR_INT_SIZE** and **CDF_CR_STAT_SIZE** fields in the CDF, then FAIL.
4. If all previous steps pass, then PASS; else FAIL.

3.2.6.3 Device Status Registers in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.6.3.1 Test Description

Verify that Device Status Registers have proper value.

3.2.6.3.2 Reference

[MHL] Section 7.9.3 Device Status Registers

[MHL] Table 7-42 Mandatory Device Status Registers

3.2.6.3.3 Required Test Equipment

MHL Protocol Analyzer

3.2.6.3.4 Required Methodology

1. Configure DUT and MHL Protocol Analyzer to Normal Mode.
2. Connect DUT to MHL Protocol Analyzer while let the MHL Protocol Analyzer monitoring LINK_MODE status register.
3. During initial discovery if CLK_MODE bits are 0b011 : “Normal(24-Bit) clock mode”, then continue to test, else FAIL.
4. Write PATH_EN = 1 into DUT.
5. If TMDS transmission from DUT is enabled and if DUT writes PATH_EN = 1 to MHL Protocol Analyzer, then continue to test, else FAIL.
6. Write PATH_EN = 0 into DUT.
7. If TMDS transmission from DUT is disabled and if DUT writes PATH_EN = 0 to MHL Protocol Analyzer, then continue to test, else FAIL.
8. If DUT writes PATH_EN into MHL Protocol Analyzer at other times, then FAIL else PASS.

3.2.6.4 Device Status Registers in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.6.4.1 Test Description

Verify that Device Status Registers have proper value in PackedPixel Mode.

3.2.6.4.2 Reference

[MHL] Section 7.9.3 Device Status Registers

[MHL] Table 7-42 Mandatory Device Status Registers

3.2.6.4.3 Required Test Equipment

MHL Protocol Analyzer

3.2.6.4.4 Required Methodology

1. If CDF_VIDEO_PACKEDPIXEL field in the CDF is “YES”, then continue to test, else PASS (SKIP).
2. Configure DUT and MHL Protocol Analyzer to enable PackedPixel mode.
3. Connect DUT to MHL Protocol Analyzer while let the MHL Protocol Analyzer monitoring LINK_MODE status register.

4. During initial discovery if CLK_MODE bits are 0b011 : “Normal(24-Bit) clock mode”, then continue to test, else FAIL.
5. Keep monitoring LINK_MODE status register until the end of the test.
6. Write PATH_EN = 1 into DUT.
7. If clock mode changes on content stream, and if CLK_MODE is updated accordingly within T_{MODE_CHG_DLY}:100mS Max, continue to test, else FAIL.
8. If TMDS transmission from DUT is enabled and if DUT writes PATH_EN = 1 to MHL Protocol Analyzer, then continue to test, else FAIL.
9. Write PATH_EN = 0 into DUT.
10. If TMDS transmission from DUT is disabled and if DUT writes PATH_EN = 0 to MHL Protocol Analyzer, then continue to test, else FAIL.
11. If DUT writes PATH_EN into MHL Protocol Analyzer at other times, then FAIL else PASS.

3.2.7 RCP Sub-Command Tests

3.2.7.1 RCP Sub-Commands Receiving Test

| Application: | Source |
|--------------|--------|
|--------------|--------|

3.2.7.1.1 Test Objective

Verify that Source DUT responds to RCP sub-commands with the expected behavior based on the definitions in the MHL Specification, for each Logical Device claimed to be supported by the Source DUT.

3.2.7.1.2 References

[MHL] Section 7.7.8.1 Support when Receiving RCP Command

[MHL] Table 7-17 Key Code Receiving Support Map

[MHL] Table 7-16 Key Code Behaviors

3.2.7.1.3 Required Test Equipment

MHL Protocol Analyzer

3.2.7.1.4 Required Methodology

1. If **CDF_RCP_RECEIVE** field in the CDF is “YES”, then continue to test, else FAIL.
2. Examine the Logical Device Type field in the CDF.
 - a. If no Logical Device Type is defined, then FAIL, else continue to test.
3. For each logical device type marked in the CDF:
 - a. Send each RCP command marked in for the Logical Device in MHL Specification.
 - b. If Source DUT does not demonstrate proper behavior as described in MHL Specification.
4. if **CDF_LOG_DEV_MAP_CHANGE** in the CDF is “YES”, then run a proper application and repeat the test from the step 3.
5. If any additional RCP sub-commands are defined in the CDF, repeat the test in step 3 for the additional RCP sub-commands.
6. If all required RCP sub-commands are verified, then PASS.

3.2.7.2 RCP Sub-Commands Transmitting Test

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.7.2.1 Test Objective

Verify that the Source DUT outputs each RCP sub-command supported as identified in the CDF, demonstrating the proper opcode and sub-command.

3.2.7.2.2 References

[MHL] Table 7-19 Key Code Sending Support Map for Source

3.2.7.2.3 Required Test Equipment

MHL Protocol Analyzer

3.2.7.2.4 Required Methodology

1. If **CDF_RCP_SEND** field in the CDF is “YES”, then continue to test, else PASS (SKIP).
2. Set DUT send each RCP sub-command marked in the CDF Table 12, and verify the opcode.
 - a. Refer to the procedure in the CDF Table 6 to have the DUT send out each RCP command.
3. If opcode does not match, then FAIL.
4. Repeat for all RCP commands marked in the CDF Table 12.
5. If all RCP sub-commands marked in the CDF Table 12 are verified, then PASS.

3.2.8 RAP Sub-command Tests

3.2.8.1 RAP and RAPK Sub-Commands Test

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.8.1.1 Test Objective

Verify that Source DUT supports all RAP sub-commands and responds with the proper behavior.

3.2.8.1.2 References

[MHL] Section 7.6 Request Action Protocol Sub-commands

[MHL] Table 7-11 MSC Request Action Protocol Sub-Command Action Codes

[MHL] Table 7-12 MSC Request Action Protocol Status Codes

3.2.8.1.3 Required Test Equipment

MHL Protocol Analyzer

3.2.8.1.4 Required Methodology

1. If **CDF_RAP_SUPPORT** field in the CDF is YES, then continue to test, else FAIL.
2. Connect Source DUT and initiate discovery sequence, then test equipment sends HPD and PATH_EN{sink} = 1 to initiate Source AV link before proceeding with test.
3. Send “Poll”: OPcode(0x00) to the DUT,
 - a. If the DUT responds with RAPK and Status Code “No Error”: 0x00, then continue to test, else FAIL.
 - b. If the DUT responds with RAPK and Status Code “Unrecognized Action Code”: 0x01 or “Unsupported Action Code”: 0x02, then FAIL, else continue to test.

- c. If the DUT responds with RAPK and Status Code "Responder Busy": 0x03, then wait 1 second and repeat from the beginning of step 2.
 4.
 - a.
 - b.
 - c.
 5. Send "CONTENT_OFF": OPcode(0x11) to the DUT,
 - a. If the DUT responds with RAPK and Status Code "No Error": 0x00 and if the DUT disables TMDS output, then continue to test, else FAIL.
 - b. If the DUT responds with RAPK and Status Code "Unrecognized Action Code": 0x01 or "Unsupported Action Code": 0x02, then FAIL, else continue to test.
 - c. If the DUT responds with RAPK and Status Code "Responder Busy": 0x03, then wait 1 second and repeat from the beginning of step 4.
 6. Send "CONTENT_ON": OPcode(0x10) to the DUT,
 - a. If the DUT responds with RAPK and Status Code "No Error": 0x00 and if the DUT enables TMDS output, then continue to test, else FAIL.
 - b. If the DUT responds with RAPK and Status Code "Unrecognized Action Code": 0x01 or "Unsupported Action Code": 0x02, then FAIL, else continue to test.
 - c. If the DUT responds with RAPK and Status Code "Responder Busy": 0x03, then wait 1 second and repeat from the beginning of step 3.
 7. If DUT passes all step 2-4, then PASS.

3.2.9 3D Video Test

3.2.9.1 3D Video Mode Support (3D REQ)

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.9.1.1 Test Objective

Verify that Source DUT requests 3D video mode support data from the connected Sink.

3.2.9.1.2 References

[MHL] Section 5.9.1 3D Video Support Detection

3.2.9.1.3 Required Test Equipment

MHL Protocol Analyzer

3.2.9.1.4 Required Methodology

1. If **CDF_VIDEO_3D** field in the CDF is "Yes", then continue to test, else FAIL.
2. Connect DUT to MHL Protocol Analyzer while letting the MHL Protocol Analyzer monitor 3D_REQ bit on RCHANGE_INT register and keeping DCAP_RDY bit 0b0.
3. If the DUT writes 3D_REQ bit to 0b1, then FAIL, else continue to test.
4. Change DCAP_RDY bit to 0b1.
5. If the DUT writes 3D_REQ bit to 0b1, then continue to test, else FAIL.
6. Disconnect and connect DUT to MHL Protocol Analyzer while let the MHL Protocol Analyzer monitoring 3D_REQ bit on RCHANGE_INT register and set MHL_VERSION 0x10 .
7. Configure DCAP_RDY bit to 0b1.

8. If the DUT writes 3D_REQ bit to 0b1, then FAIL, else PASS.

3.2.9.2 3D Video Format Timings in Normal Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.9.2.1 Test Objective

Verify that Source DUT supports 3D video formats in Normal Mode if DUT indicates 3D support, and verify the timing.

3.2.9.2.2 References

[MHL] Section 5.9.4 3D Video Support Requirements

[MHL] Section 5.9.4.1 MHL Source 3D Video Format Support Requirements

3.2.9.2.3 Required Test Equipment

MHL Protocol Analyzer

3.2.9.2.4 Required Methodology

1. If CDF_VIDEO_3D field in the CDF is “Yes”, then continue to test, else FAIL.
2. If none of CDF_VIDEO_1280x720P_60_3D_Top_Bottom, CDF_VIDEO_1280x720P_50_3D_Top_Bottom, CDF_VIDEO_1920x1080i_60_3D_Left_Right, CDF_VIDEO_1920x1080i_50_3D_Left_Right, CDF_VIDEO_1920x1080p_24_Top_Bottom in the CDF is “Yes”, then FAIL, else continue to test.
3. Configure DUT to send out one of the 3D video formats indicated in the CDF which use Normal Mode.
4. Measure:
 - a. H-front porch, H-sync pulse width, H-back porch, H-active, H-sync polarity.
 - b. V-front porch, V-sync pulse width, V-back porch, V-active, V-sync polarity.
 - c. Clock frequency, offset of the active edge of V-sync from the active edge of H-sync.
5. Compare the measurement values with the video format timings defined in the EIA/CEA-861E specification. If any value does not equal the proper values for the select video format, then FAIL.
6. Examine if the measured clock frequency is within +0.5%/-0.6% range of the pixel clock frequency defined in the EIA/CEA-861E if vertical frequency is either an integer multiple of 6 Hz or an integer multiple of 6 Hz adjusted by factor of 1000/1001, or if the measured clock frequency is within $\pm 0.5\%$ range if the vertical frequency does not fall in those two cases. If the pixel clock frequency is out of the allowable range, then FAIL.
7. Repeat from step 4 until all 3D video formats indicated in the CDF which use Normal Mode are tested.
8. If all 3D Video formats specified in the CDF which use Normal Mode are verified, then PASS.

3.2.9.3 3D Video Mode Indicator

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.9.3.1 Test Objective

Verify that Source DUT indicate 3D video mode using MHL Vendor-Specific InforFrame (VSIF).

3.2.9.3.2 References

[MHL] Section 5.9.2 Active 3D Video Mode Indicator.

3.2.9.3.3 Required Test Equipment

MHL Protocol Analyzer

3.2.9.3.4 Required Methodology

1. Configure the DUT to send out one of the 3D Video formats ,displaying a 3D Video pattern, indicated in the CDF which use PackedPixel mode..
2. If the DUT transmits an MHL VSIF at least once every two video fields during the vertical blanking period, then continue to test, else FAIL.
3. Examine MHL VSIF:
 - a. If HB1 is 0x01 then, continue to test, else FAIL.
 - b. If HB2 is 0x04, then continue to test, else FAIL.
 - c. If byte-wide sum of all three bytes of the Packet Header and all the InfoFrame Packet contents determined by InfoFrame_length, including PB0 equals zero, then continue to test, else FAIL.
 - d. If PB3, PB2 and PB1 is 0x7CA61D, then continue to test, else FAIL.
 - e. If MHL_VID_FMT field in PB4 is 0b01, then continue to test, else FAIL.
 - f. If MHL_3D_FMT_TYPE field in PB4 matches the 3D video mode in the video stream , then PASS, else FAIL.
4. Perform the steps above for all the 3D modes supported in the DUT's CDF.

3.2.9.4 3D Video Format Timings in PackedPixel Mode

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.9.4.1 Test Objective

Verify that Source DUT support 3D video format which use PackedPixel Mode if DUT indicates 3D support, and verify the timing.

3.2.9.4.2 References

[MHL] Section 5.9.4 3D Video Support Requirements

[MHL] Section 5.9.4.1 MHL Source 3D Video Format Support Requirements

3.2.9.4.3 Required Test Equipment

MHL Protocol Analyzer

3.2.9.4.4 Required Methodology

1. If **CDF_VIDEO_3D** field in the CDF is “Yes”, then continue to test, else FAIL.
2. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is “Yes” and if none of **CDF_VIDEO_1280x720P_60_3D_Frame**, **CDF_VIDEO_1280x720P_50_3D_Frame**, **CDF_VIDEO_1920x1080p_24_3D_Frame** in the CDF is “Yes”, then FAIL, else continue to test.
3. Configure DUT to send out one of the 3D video formats indicated in the CDF which use PackedPixel Mode.
4. Measure:
 - d. H-front porch, H-sync pulse width, H-back porch, H-active, H-sync polarity.
 - e. V-front porch, V-sync pulse width, V-back porch, V-active, V-sync polarity.
 - f. Clock frequency, offset of the active edge of V-sync from the active edge of H-sync.
5. Compare the measurement values with the video format timings defined in the EIA/CEA-861E specification. If any value does not equal the proper values for the select video format, then FAIL.

6. Examine if the measured clock frequency is within +0.5%/-0.6% range of the pixel clock frequency defined in the EIA/CEA-861E if vertical frequency is either an integer multiple of 6 Hz or an integer multiple of 6 Hz adjusted by factor of 1000/1001, or if the measured clock frequency is within $\pm 0.5\%$ range if the vertical frequency does not fall in those two cases. If the pixel clock frequency is out of the allowable range, then FAIL.
7. Repeat from step 4 until all 3D video formats indicated in the CDF which use PackedPixel Mode are tested.
8. If all 3D Video formats specified in the CDF which use PackedPixel Mode are verified, then PASS.

3.2.10 UCP Sub-Command Tests

The test implementation shall include UTF character sequences for all lengths of characters (1 to 4 8-bit bytes or octets), with both invalid and valid values in each of those lengths.

3.2.10.1 UCP Sub-Commands Receiving Test

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.10.1.1 Test Objective

Verify that DUT responds to valid UCP sub-commands by displaying the character or characters sent in the UCP command, or response to invalid UCP sub-commands by displaying an error message.

3.2.10.1.2 References

[MHL] Section 7.8.4 MSC UTF-8 Character Protocol Support Requirements

3.2.10.1.3 Required Test Equipment

MHL Protocol Analyzer

The test implementation shall include UTF character sequences for all lengths of characters, with both invalid and valid values in those lengths.

3.2.10.1.4 Required Methodology

1. If CDF indicates **CDF_CR_UCP_RECV_SUPPORT** is 1 then continue test, else PASS (SKIP).
2. Connect the Test Equipment to the DUT.
3. Perform discover sequence between the Test Equipment and the DUT.
 - a. If DUT fails to discover properly, then FAIL.
4. For each UCP command described in the CDF, perform the following steps:
 - a. Enter into the test equipment the expected character or character sequence to send to the DUT, according to the information in the CDF entry.
 - b. Configure the DUT to receive the selected character or character sequence. Use the Application or other software on the DUT as described in the CDF with field **CDF_UCP_RECV_APPLICATION**.
 - c. Trigger the sending of the character or character sequence by Test Equipment action to the DUT.
 - d. Test Equipment displays all bits and interpretation of the UCP payload(s) it sends.
 - e. DUT displays all bits and interpretation of the UCP payload(s) it receives.
 - f. Test Engineer compares the DUT displayed data with the Test Equipment data.
 - g. If the payload received by the DUT does not match that sent by the Test Equipment, then FAIL.
5. If DUT passes all iterations of the test, then PASS. Else FAIL.

3.2.10.2 UCP Sub-Commands Transmitting Test

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.2.10.2.1 Test Objective

Verify that DUT sends valid UCP sub-commands by initiating the sending of UTF-8 characters in various formats through the user interface on the DUT.

3.2.10.2.2 References

[MHL] Section 7.8.4 MSC UTF-8 Character Protocol Support Requirements

3.2.10.2.3 Required Test Equipment

TBD

3.2.10.2.4 Required Methodology

1. If CDF indicates **CDF_CR_UCP_SEND_SUPPORT** is 1 then continue test, else PASS (SKIP).
2. Connect the Test Equipment to the DUT.
3. Perform discover sequence between the Test Equipment and the DUT.
 - a. If DUT fails to discover properly, then FAIL.
4. For each UCP command described in the CDF, perform the following steps:
 - a. Enter into the test equipment the expected character or character sequence from the DUT, according to the information in the CDF entry.
 - b. Configure the DUT to send the selected character or character sequence. Use the Application or other software on the DUT as described in the CDF with field **CDF_UCP_SEND_APPLICATION**.
 - c. Trigger the sending of the character or character sequence by user action on the DUT.
 - d. Test Equipment displays all bits and interpretation of the UCP payload(s) it receives.
 - e. If the Test Equipment displays "Mismatch in received UCP information", then FAIL.
5. If DUT passes all iterations of the test, then PASS. Else FAIL.

3.3 CBUS Tests

Test the CBUS timing and functionality of the Source DUT. Many of the tests in this section use common test equipment setups and common methodologies (listed in 3.3.1). These common elements are delineated before the individual tests.

Note 1: Preparation: Many Sources may be in a Sleep, Standby or Unpowered state. All testing assumes that a suitable method, as specified in the CDF, is used to put the Source into an MHL-sensitive state before testing begins.

Note 2: Typical resistances, capacitances, and timings: Where called out, use $Z_{CBUS_SINK_DISCOVER}\{typ\}$, $Z_{CBUS_SINK_ON}\{typ\}$ and $C_{CBUS}=180pF$ ($C_{SINK_CBUS}+150pF$ Cable) and $T_{BIT_CBUS}\{typ\}$.

Note 3: Continuous monitoring: Continuous monitoring assumes that a parameter measurement is checked continuously for the duration of the testing. Continuous monitoring is usually applied to detect out of range voltage levels and timings.

Note 4: Direct Attach Source DUT testing: If possible, Direct Attach Source DUTs should be connected directly to the MHL Tester's Sink input port. In cases where the mechanical design of the DUT or Tester does not allow a direct connection, the use of a "Direct Attach Source Extension Cable" is allowed.

Note 5: Slow responding Source DUT testing: There is no MHL specification for the maximum amount of time a Source DUT may wait before starting a discovery sequence. Because of this, Source DUT CTS testing may fail in a timeout if it does not wait long enough for a discovery sequence to begin. This is especially true for DA Sources that are powered directly from the Sink Tester and which may require extra time to initialize upon powerup. Rather than failing devices due to early timeout expiration, it is preferred that TE vendors accommodate this by:

- a. Allowing at least 10 seconds for the Source DUT to boot after the application of VBUS.
- b. Allowing the Source DUT adequate time to respond to the application of power and the 1K ohm pulldown before signaling a timeout.

3.3.1 CBUS Source DUT Common Test Equipment Setups

3.3.2 CBUS Source DUT Common Methodologies

Many of the tests in Section 3.3 use a common methodology, followed by test steps specific to the individual test definition. Refer back to this section for the initial test steps using the hyperlink if one is shown in the individual test definition.

Most of the tests start by specifying that the tester should be set to "disconnected state". "Tester disconnected state" implies the following:

1. The tester does not drive the VBUS.
2. The tester does not apply a load to the VBUS.
3. The tester does not apply a pullup to the MHL+/- wires which would result in a current greater than $I_{\text{RSEN_LEAK}}$.
4. The tester does not apply a pulldown to the MHL+/- wires which would result in a current greater than $I_{\text{RSEN_LEAK}}$.
5. The tester does not apply a pullup to the CBUS wire which would result in a current greater than $I_{\text{cbus_LEAK_SRC}}$ or $I_{\text{cbus_LEAK_SINK}}$.
6. The tester does not apply a pulldown to the CBUS wire which would result in a current greater than $I_{\text{cbus_LEAK_SRC}}$ or $I_{\text{cbus_LEAK_SINK}}$.
7. The tester does not actively drive the CBUS wire.
8. The tester applies the minimum possible capacitor loading to the CBUS.

3.3.2.1 Tester_Unpowered_Sink_Incomplete_Discovery_Procedure

1. Set Tester Sink to not react to Discovery pulses.
2. Set Tester Sink to not drive the VBUS.
3. Direct Tester Sink to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
4. Direct Tester Sink to initiate Discovery by resistively pulling down with $Z_{\text{CBUS_SINK_DISCOVER}}$.

3.3.2.2 Tester_Powered_Sink_Incomplete_Discovery_Procedure

1. Set Tester Sink to not react to Discovery pulses.
2. Set Tester Sink to drive the VBUS.
3. Direct Tester Sink to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
4. Direct Tester Sink to initiate Discovery by resistively pulling down with $Z_{\text{CBUS_SINK_DISCOVER}}$.

3.3.2.3 *Tester_Powered_Sink_Incomplete_R_Discovery Procedure*

1. Set Tester Sink to not react to Discovery pulses.
2. Set Tester Sink to drive the VBUS.
3. Direct Tester Sink to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
4. Direct Tester Sink to initiate Discovery by resistively pulling down with TEST_RESISTOR.

3.3.2.4 *Tester_Sink_Incomplete_VR_Discovery Procedure*

1. Set Tester Sink to not react to Discovery pulses.
2. Set Tester Sink port to drive TEST_VBUS_DRIVE to the VBUS.
3. Direct Tester Sink to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
4. Direct Tester Sink to initiate Discovery by resistively pulling down with TEST_RESISTOR.

3.3.2.5 *Tester_Unpowered_Sink_Discovery Procedure*

1. Start Continuous Monitor procedures.
2. Set Tester Sink to use its Sink Discovery engine to enable normal MHL discovery.
3. Set Tester Sink to not drive the VBUS.
4. Direct Tester Sink to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
5. Direct Tester Sink to initiate Discovery by resistively pulling down with $Z_{\text{CBUS_SINK_DISCOVER}}\{\text{typ}\}$.
6. After Discovery completes, Tester Sink will change its CBUS pulldown to $Z_{\text{CBUS_SINK_ON}}\{\text{typ}\}$, and will pull up on the MHL+/- wires with $Z_{\text{RXSENSE_TERM}}\{\text{max}\}$.

3.3.2.6 *Tester_Sink_VR_Discovery Procedure*

1. Start Continuous Monitor procedures.
2. Set Tester Sink to use its Sink Discovery engine to enable normal MHL discovery.
3. Set Tester Sink port to drive TEST_VBUS_DRIVE to the VBUS.
4. Direct Tester Sink to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
5. Direct Tester Sink to initiate Discovery by resistively pulling down with TEST_RESISTOR.
6. After Discovery completes, Tester Sink will change its CBUS pulldown to $Z_{\text{CBUS_SINK_ON}}\{\text{typ}\}$.
7. Pull up on the MHL+/- wires with $Z_{\text{RXSENSE_TERM}}\{\text{max}\}$.

3.3.2.7 *Tester_Powered_Sink_R_Discovery Procedure*

1. Start Continuous Monitor procedures.
2. Set Tester Sink to use its Sink Discovery engine to enable normal MHL discovery.
3. Set Tester Sink to drive the VBUS.
4. Direct Tester Sink to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
5. Direct Tester Sink to initiate Discovery by resistively pulling down with $Z_{\text{CBUS_SINK_DISCOVER}}\{\text{typ}\}$.
6. After Discovery completes, Tester Sink will change its CBUS pulldown to TEST_RESISTOR.
7. Pull up on the MHL+/- wires with $Z_{\text{RXSENSE_TERM}}\{\text{max}\}$.

3.3.2.8 *Tester_Powered_Sink_Discovery Procedure*

1. Start Continuous Monitor procedures.
2. Set Tester Sink to use its Sink Discovery engine to enable normal MHL discovery.
3. Set Tester Sink to drive the VBUS.
4. Direct Tester Sink to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
5. Direct Tester Sink to initiate Discovery by resistively pulling down with $Z_{\text{CBUS_SINK_DISCOVER}}\{\text{typ}\}$.
6. After Discovery completes, Tester Sink will change its CBUS pulldown to $Z_{\text{CBUS_SINK_ON}}\{\text{typ}\}$.
7. Pull up on the MHL+/- wires with $Z_{\text{RXSENSE_TERM}}\{\text{max}\}$.

3.3.2.9 *Tester_Powered_Sink_Modified_Discovery Procedure*

Note that Discovery state machine can detect illegal discovery pulses and start over.

The procedure requires 2 arguments: TEST_SINK_PULSE_COUNT and TEST_MHL_PU_TIME.

1. Set Tester Sink to use its Sink Discovery engine to enable normal MHL discovery.
2. Set Tester Sink to drive the VBUS.
3. Set Tester Sink to respond to Discovery Pulses after TEST_SINK_PULSE_COUNT.
4. Set Tester Sink to enable MHL+/- Pull-ups after TEST_MHL_PU_TIME (0 default).
5. Direct Tester to HIGH-Z the CBUS for $T_{\text{SINK_CBUS_FLOAT}}\{\text{min}\}$.
6. Direct tester to initiate Discovery by resistively pulling down with $Z_{\text{CBUS_SINK_DISCOVER}}\{\text{typ}\}$.
7. After the Tester Sink counts TEST_SINK_PULSE_COUNT rising edges it will wait for the next valid rising edge of the CBUS.
8. Once the Tester sees the final rising edge, it will change from $Z_{\text{CBUS_SINK_DISCOVER}}\{\text{typ}\}$ to $Z_{\text{CBUS_SINK_ON}}\{\text{typ}\}$.
9. After changing to $Z_{\text{CBUS_SINK_ON}}$, wait TEST_MHL_PU_TIME and then enable pull-up of $Z_{\text{RXSENSE_TERM}}\{\text{max}\}$ on MHL+/- wires.

3.3.2.10 Tester Powered Sink Immediate Discovery Procedure

1. Start Continuous Monitor procedures.
2. Set Tester Sink to use its Sink Discovery engine to enable normal MHL discovery.
3. Set Tester Sink to drive the VBUS.
4. Direct Tester Sink to initiate Discovery by resistively pulling down with $Z_{\text{CBUS_SINK_DISCOVER}}\{\text{typ}\}$.
5. After Discovery completes, Tester Sink will change its CBUS pulldown to $Z_{\text{CBUS_SINK_ON}}\{\text{typ}\}$.
6. Pull up on the MHL+/- wires with $Z_{\text{RXSENSE_TERM}}\{\text{max}\}$.

3.3.2.11 Tester Wait Capability Registers Valid Procedure

1. Wait until the first of the following 3 events:
 - a. DUT indicates DCAP_RDY by issuing a WRITE_STAT command with an offset of 0x30 (CONNECTED_RDY) and a data value of 0x01 (DCAP_RDY).
 - b. DUT indicates that a DCAP_CHG interrupt has occurred by issuing a SET_INT with an offset of 0x20 (RCHANGE_INT) and a data byte containing the 0x01 bit (DCAP_CHG).
 - c. Tester selected timeout (typically 60 Sec) suggesting that something went wrong and the Capability registers are not going to become readable.
2. Return a Success indication if the Capability Registers are readable; otherwise indicate TIMEOUT.

3.3.2.12 Packets to Induce Link-Level NACK Behavior

Several of the tests require the Tester to send a specially-constructed packet which causes the DUT to respond with a link-level NACK. There are many edge sequences which should be interpreted as bad packets by the DUT. This test specification defines 2 bad packets to represent the larger set of all bad patterns. Both error packets are created by starting with a good packet and moving one edge ½ bit time earlier. This might happen on the CBUS when impulse noise is added to the driven signal.

Error-Packet-1 is a packet containing Header bits: 0x0, Control Bit: 0x1, Data bits: 0x00.

The packet has no falling edge at the beginning of the first Data bit D7. That edge is moved ½ bit time earlier, and is interpreted as a middle-bit-time edge by the DUT.

When Error-Packet-1 is sent using Source Arbitration, the packet timing appears as shown in Figure 30.

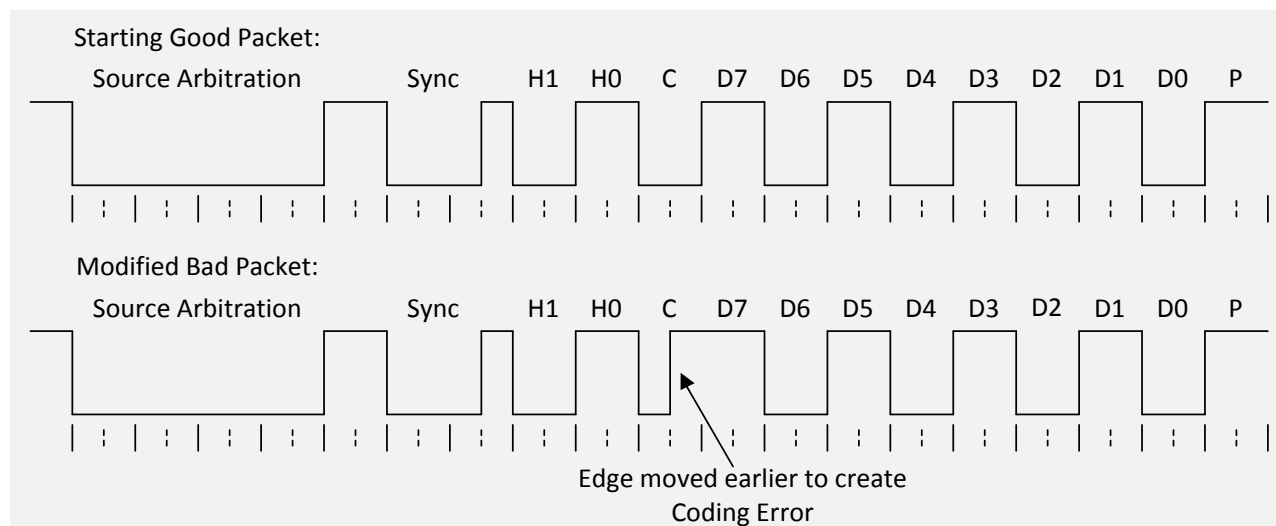


Figure 30. Source – CBUS Setup #2

Error-Packet-2 is a packet containing Header bits: 0x0, Control Bit: 0x1, Data bits: 0x01.
The D7 bit edge is moved to create the Coding Error the same as in Error-Packet-1.

3.3.2.13 Tester Sink Device Capability Registers

Unless changed in specific tests, the Tester will exhibit the following Device Capability Register contents:

Table 3-1. Tester Sink Device Capability Registers

| | Register Name | Register Value | Description |
|----|-----------------|-------------------------------|--|
| 0 | DEV_STATE | 0x00 | Reserved |
| 1 | MHL_VERSION | 0x20 | Level of MHL Spec supported |
| 2 | DEV_CAT | 0x11 | Identify the type of MHL system (Sink) + POW bit |
| 3 | ADOPTER_ID_H | Set by Equipment Manufacturer | High-order byte of Adopter ID |
| 4 | ADOPTER_ID_L | Set by Equipment Manufacturer | Low-order byte of Adopter ID |
| 5 | VID_LINK_MODE | 0x37 | List of link modes supported for Video |
| 6 | AUD_LINK_MODE | 0x03 | List of link modes supported for Audio |
| 7 | VIDEO_TYPE | 0x85 | Video Type Support |
| 8 | LOG_DEV_MAP | 0x41 | Logical Device Map |
| 9 | BANDWIDTH | 0x0F | Link Bandwidth limit |
| 10 | FEATURE_FLAG | 0x07 | Flags for MHL Optional feature |
| 11 | DEVICE_ID_H | Set by Equipment Manufacturer | High-order byte of system identifier |
| 12 | DEVICE_ID_L | Set by Equipment Manufacturer | Low-order byte of system identifier |
| 13 | SCRATCHPAD_SIZE | 0x10 | Count of Scratchpad registers |
| 14 | INT_STAT_SIZE | 0x33 | Count of interrupt and status registers |
| 15 | No Name | 0x00 | Reserved |

3.3.3 Link Layer Electrical – Source: Absolute Maximum Voltages

3.3.3.1 Common Test Environment

3.3.3.1.1 Test Objective

Monitor the voltages on VBUS and CBUS wires continuously to make sure they never exceed their absolute maximum values.

3.3.3.1.2 References

See individual tests in this section.

3.3.3.1.3 Required Test Equipment

Use the Standard Test Suite.

3.3.3.1.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. Set Tester Sink VBUS Limit comparator to appropriate voltage.
2. Set Tester Sink CBUS Limit comparator to appropriate voltage.
3. Set Tester to report VBUS Limit errors. FAIL if VBUS limits are violated.
4. Set Tester to report CBUS Limit errors. FAIL if CBUS limits are violated.
5. If no limits are violated, then record PASS.

3.3.3.2 CBE-Source: VBUS Absolute Maximum Positive Voltage

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.3.2.1 Test Objective

Continuous monitor the VBUS wire to make sure it never exceeds the absolute maximum positive voltage.

3.3.3.2.2 References

[MHL] Section 13.9.1.1; Table 13-21: $V_{VBUS_TP2_SINK_DRV}$; $V_{VBUS_TP1_SRC_DRV}$

3.3.3.2.3 Required Test Equipment

Use the Standard Test Suite.

3.3.3.2.4 Required Methodology

Continuous monitor runs in the background, using the setup in Section 3.3.3.1.

FAIL if VBUS positive limit is exceeded.

3.3.3.3 CBE-Source: CBUS Absolute Maximum Positive Voltage

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.3.3.1 Test Objective

Continuous monitor the CBUS wire to make sure it never exceeds the absolute maximum positive voltage.

3.3.3.3.2 References

[MHL] Section 13.10.1; Table 13-23: CBUS Absolute Max Voltage

3.3.3.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.3.3.4 Required Methodology

Continuous monitor runs in the background, using the setup in Section 3.3.3.1.

FAIL if CBUS positive limit is exceeded.

3.3.4 Link Layer Timing – Source DUT Output: Pre-Discovery

Measure the timing of the Source DUT before it starts emitting Discovery pulses.

3.3.4.1 CBT-Source: Time from Source VBUS Application to Discovery Pulses

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.4.1.1 Test Objective

Verify that the Source DUT waits $T_{\text{SRC_VBUS_OUT_TO_STABLE}}$ after applying VBUS before starting the Discovery procedure.

3.3.4.1.2 References

[MHL] Section 8.2.1.1 (SRC8->SRC8A)

[MHL] Section 8.3.2

[MHL] Section 13.10.1.1; Table 13-29: $T_{\text{SRC_VBUS_OUT_TO_STABLE}}$

[MHL] Section 13.10.1.1; Table 13-30: $T_{\text{SINK_CBUS_FLOAT}}$

3.3.4.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.4.1.4 Required Methodology

1. If the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source then continue, else end test with PASS (SKIP).
2. If necessary, connect Source DUT to Tester Sink port.
3. Set Tester Sink port to disconnected state.
4. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
5. Execute the Tester_Unpowered_Sink_Discovery Procedure.
6. FAIL if DUT does not enable VBUS.
7. FAIL if DUT starts driving CBUS HIGH Discovery pulses before the minimum or after the maximum $T_{\text{SRC_VBUS_OUT_TO_STABLE}}$ from enable of VBUS.
8. If all preceding steps pass, then PASS; else FAIL.

3.3.5 Link Layer Electrical – Source DUT Output: Discovery

3.3.5.1 CBE-Source: Response to Initial Plug-in to MHL Device

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.5.1.1 Test Objective

Verify correct Source DUT behavior when going from unplugged to $Z_{\text{CBUS_SINK_DISCOVER}}$ within the MHL range.

3.3.5.1.2 References

[MHL] Section 8.2.1.1 (SRC1)

[MHL] Section 8.3.1

[MHL] Section 13.10.1; Table 13-26, Table 13-27: $Z_{CBUS_SINK_DISCOVER}$, $Z_{CBUS_DONGLE_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SRC_START_IMP_MEAS}$

3.3.5.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.5.1.4 Required Methodology

1. Execute the following test procedure once for each row in Table 3-2 substituting TEST_VBUS_DRIVE and TEST_RESISTOR.
2. If necessary, connect Source DUT to Tester Sink port.
3. If the CDF contains a definition of the procedure **CDF_SRC_PROC_STANDBY_DISCOVERY**, execute it.
4. Set Tester Sink port to disconnected state.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
6. Execute the Tester_Sink_VR_Discovery Procedure.
7. When Tester Sink not driving VBUS and the CDF value **CDF_SRC_POWERED** indicates that the Source DUT is a Powered Source, FAIL if DUT does not wait $T_{SRC_VBUS_CBUS_STABLE\{min\}}$ from the assertion of $Z_{CBUS_SINK_DISCOVER}$ before driving VBUS.
8. When Tester Sink is not driving VBUS and the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source, FAIL if DUT does not make wake pulses followed by discovery pulses starting within (minimum to maximum) $T_{SRC_VBUS_OUT_TO_STABLE}$ after it enables VBUS.
9. When Tester Sink driving the VBUS, FAIL if DUT does not make wake pulses followed by discovery pulses after the assertion of $Z_{CBUS_SINK_DISCOVER}$.
10. If all preceding steps pass, then PASS; else FAIL.

Table 3-2. Response to Initial Plug-in to MHL Device

| | TEST_VBUS_DRIVE | TEST_RESISTOR | Purpose |
|---|--------------------------|-----------------------------------|-----------------------------------|
| 1 | Sink does not drive VBUS | $Z_{CBUS_SINK_DISCOVER\{min\}}$ | Verify that Source does Discovery |
| 2 | Sink does not drive VBUS | $Z_{CBUS_SINK_DISCOVER\{typ\}}$ | Verify that Source does Discovery |
| 3 | Sink does not drive VBUS | $Z_{CBUS_SINK_DISCOVER\{max\}}$ | Verify that Source does Discovery |
| 4 | Sink drives VBUS | $Z_{CBUS_SINK_DISCOVER\{min\}}$ | Verify that Source does Discovery |
| 5 | Sink drives VBUS | $Z_{CBUS_SINK_DISCOVER\{typ\}}$ | Verify that Source does Discovery |
| 6 | Sink drives VBUS | $Z_{CBUS_SINK_DISCOVER\{max\}}$ | Verify that Source does Discovery |

3.3.5.2 CBE-Source: Response to Sink Priming Pulse to MHL device

Application: Source

3.3.5.2.1 Test Objective

Verify correct Source DUT behavior when Sink HIGH-Z's the CBUS to invoke a new (second or subsequent) CBUS Discovery, then attaches with $Z_{CBUS_SINK_DISCOVER}$ within the MHL range.

There are several cases when a Sink or Dongle posts an event to a Source to indicate that the Source should re-run its MHL Discovery Procedure. Examples are when a Sink or Dongle has its power applied, and when one of these devices leaves Standby State.

The Sink or Dongle signals the Source to retry its Discovery procedure by removing and then re-applying the Sink or Dongle Discovery resistor. The Source must notice the change from “no discovery pulldown resistor” to “valid discovery pulldown resistor”, and should retry MHL Discovery based on the change.

3.3.5.2.2 References

[MHL] Section 8.2.1; Table 8-2; SRC1-SRC2
[MHL] Section 8.2.2 (SINK5)
[MHL] Section 8.2.3 (DONGLE4, DONGLE8)
[MHL] Section 8.3.3
[MHL] Section 13.10.1; Table 13-26, Table 13-27: $Z_{CBUS_SINK_DISCOVER}$, $Z_{CBUS_DONGLE_DISCOVER}$
[MHL] Section 13.10.1.1; Table 13-29: $T_{SRC_START_IMP_MEAS}$
[MHL] Section 13.10.1.1; Table 13-30: $T_{SINK_CBUS_FLOAT}$

3.3.5.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.5.2.4 Required Methodology

1. Execute steps 2 to 15 once for each row in Table 3-3.
2. If necessary, connect Source DUT to Tester Sink port.
3. Set Tester Sink port to disconnected state.
4. If the CDF contains a definition of the procedure **CDF_SRC_PROC_STANDBY_DISCOVERY**, execute it.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
6. Execute the Tester Sink Incomplete VR Discovery procedure, setting TEST_RESISTOR to $Z_{CBUS_SINK_DISCOVER\{typ\}}$ and TEST_VBUS_DRIVE values from the column TEST_VBUS_DRIVE_FIRST.
7. If Tester Sink is not driving VBUS and the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source, then FAIL if DUT does not wait at least $T_{SRC_VBUS_CBUS_STABLE\{min\}}$ from the assertion of $Z_{CBUS_SINK_DISCOVER}$ before driving VBUS.
8. If Tester Sink is not driving VBUS and the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source, then FAIL if DUT does not make discovery pulses within (minimum to maximum) $T_{SRC_VBUS_OUT_TO_STABLE}$ after it enables VBUS.
9. If Tester Sink is driving VBUS and the CDF value **CDF_SRC_POWERED** indicates that Source DUT is NOT a Powered Source, then FAIL if DUT does not make discovery pulses after the assertion of $Z_{CBUS_SINK_DISCOVER}$.
10. Set Tester Sink to HIGH-Z its Discovery resistor, and to drive VBUS as specified in column TEST_VBUS_DRIVE_SINK5.
11. Wait for at least the period specified in HIGH-Z Duration.
12. Execute the Tester Powered Sink Immediate Discovery procedure.
13. If Tester Sink is not driving VBUS and the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source, then FAIL if DUT does not wait at least $T_{SRC_VBUS_CBUS_STABLE\{min\}}$ from the assertion of $Z_{CBUS_SINK_DISCOVER}$ before driving VBUS.
14. If Tester Sink is not driving VBUS and the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source, then FAIL if DUT does not make wake pulses followed by discovery pulses starting within (minimum to maximum) $T_{SRC_VBUS_OUT_TO_STABLE}$ after it enables VBUS.
15. If Tester Sink is driving the VBUS, then FAIL if DUT does not make wake pulses followed by discovery pulses starting after the assertion of $Z_{CBUS_SINK_DISCOVER}$.
16. If the test does not otherwise FAIL, then PASS.

Table 3-3. Response to Sink Priming Pulse to MHL Device

| | TEST_VBUS_DRIVE_FIRST | TEST_VBUS_DRIVE_SINK5 | HIGH-Z Duration | Purpose |
|----|--|-----------------------|------------------------------------|--|
| 1 | VBUS, No Z _{CBUS_SINK_DISCOVER} | No VBUS | T _{SINK_CBUS_FLOAT} {min} | SINK0 -> SINK5 -> SINK1, no SINK5 VBUS |
| 2 | No VBUS, No Z _{CBUS_SINK_DISCOVER} | VBUS | T _{SINK_CBUS_FLOAT} {min} | SINK0 -> SINK5 -> SINK1, SINK5 VBUS |
| 3 | VBUS, Z _{CBUS_SINK_DISCOVER} {typ} | No VBUS | T _{SINK_CBUS_FLOAT} {min} | SINK1/SINK4 -> SINK5 -> SINK1, no SINK5 VBUS |
| 4 | VBUS, Z _{CBUS_SINK_DISCOVER} {typ} | VBUS | T _{SINK_CBUS_FLOAT} {min} | SINK1/SINK4 -> SINK5 -> SINK1, SINK5 VBUS |
| 5 | No VBUS, Z _{CBUS_SINK_DISCOVER} {typ} | No VBUS | T _{SINK_CBUS_FLOAT} {min} | SINK4 -> SINK5 -> SINK1, no SINK5 VBUS |
| 6 | No VBUS, Z _{CBUS_SINK_DISCOVER} {typ} | VBUS | T _{SINK_CBUS_FLOAT} {min} | SINK4 -> SINK5 -> SINK1, SINK5 VBUS |
| 7 | VBUS, No Z _{CBUS_SINK_DISCOVER} | No VBUS | 5 seconds | SINK0 -> SINK5 -> SINK1, no SINK5 VBUS |
| 8 | No VBUS, No Z _{CBUS_SINK_DISCOVER} | VBUS | 5 seconds | SINK0 -> SINK5 -> SINK1, SINK5 VBUS |
| 9 | VBUS, Z _{CBUS_SINK_DISCOVER} {typ} | No VBUS | 5 seconds | SINK1/SINK4 -> SINK5 -> SINK1, no SINK5 VBUS |
| 10 | VBUS, Z _{CBUS_SINK_DISCOVER} {typ} | VBUS | 5 seconds | SINK1/SINK4 -> SINK5 -> SINK1, SINK5 VBUS |
| 11 | No VBUS, Z _{CBUS_SINK_DISCOVER} {typ} | No VBUS | 5 seconds | SINK4 -> SINK5 -> SINK1, no SINK5 VBUS |
| 12 | No VBUS, Z _{CBUS_SINK_DISCOVER} {typ} | VBUS | 5 seconds | SINK4 -> SINK5 -> SINK1, SINK5 VBUS |

3.3.5.3 CBE-Source: Pre-Discovery Success Pull-up HIGH Voltage

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.5.3.1 Test Objective

Verify that Source DUT Pull-up Voltage has correct value when connecting Z_{CBUS_SRC_DISCOVER}.

3.3.5.3.2 References

[MHL] Section 13.10.1; Table 13-25: V_{TERM_CBUS}

3.3.5.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.5.3.4 Required Methodology

Execute the following test procedure once for each row in Table 3-4 substituting TEST_VBUS_DRIVE and TEST_RESISTOR.

1. If necessary, connect Source DUT to Tester Sink port.
2. If the CDF contains a definition of the procedure CDF_SRC_PROC_STANDBY_DISCOVERY, execute it.
3. Set Tester Sink port to disconnected state.
4. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.

5. If the CDF value CDF_SRC_REQUIRES_SINK_VBUS indicates that the Source requires Sink VBUS to operate, then SKIP steps 7-14 for table entry 1 (TEST_VBUS_DRIVE = Sink does not drive VBUS), else execute the Tester_Sink_VR_Discovery Procedure, setting TEST_RESISTOR to $Z_{CBUS_SINK_DISCOVER}\{min\}$ and TEST_VBUS_DRIVE values from the column TEST_VBUS_DRIVE.
6. Watch for Source to drive Wake pulses.
7. Measure the CBUS Voltage before, during, and after the Discovery pulses when the CBUS is not actively driven HIGH by the Source DUT.
8. Calculate the pullup voltage and the pullup resistor.
9. FAIL if VTERM_CBUS is not within the allowable range.
10. FAIL if ZCBUS_SRC_DISCOVER is not within the allowable range.
11. Repeat steps 1 through 10 for each row in Table 3-4.
12. If all preceding steps pass, then PASS; else FAIL.

Table 3-4. Pre-Discovery Success Pull-up HIGH Voltage

| | TEST_VBUS_DRIVE | TEST_RESISTOR | Purpose |
|---|--------------------------|-----------------------------------|--------------------------------|
| 1 | Sink does not drive VBUS | $Z_{CBUS_SINK_DISCOVER}\{min\}$ | Measure Source Pull-up Voltage |
| 2 | Sink drives VBUS | $Z_{CBUS_SINK_DISCOVER}\{min\}$ | Measure Source Pull-up Voltage |

3.3.5.4 CBE-Source: Discovery Pulse Drive HIGH Voltage

Application: Source

3.3.5.4.1 Test Objective

Verify that Source DUT drives Discovery Pulses with HIGH voltage for Sink to observe.

3.3.5.4.2 References

[MHL] Section 13.10.1; Table 13-24: V_{IH_CBUS}

[MHL] Section 13.10.1; Table 13-25: V_{TERM_CBUS}

[MHL] Section 8.4;

3.3.5.4.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.5.4.4 Required Methodology

Execute the following test procedure once for each row in Table 3-5, substituting for TEST_VBUS_DRIVE and TEST_RESISTOR.

1. If necessary, connect Source DUT to Tester Sink port.
2. If the CDF contains a definition of the procedure **CDF_SRC_PROC_STANDBY_DISCOVERY**, execute it.
3. Set Tester Sink port to disconnected state.
4. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
5. If the CDF value CDF_SRC_REQUIRES_SINK_VBUS indicates that the Source DUT requires Sink VBUS to operate, then SKIP steps 7-9 for table entry 1 (TEST_VBUS_DRIVE = Sink does not drive VBUS), else execute the Tester_Sink_VR_Discovery Procedure setting TEST_RESISTOR to $Z_{CBUS_SINK_DISCOVER}\{min\}$ and TEST_VBUS_DRIVE values from the column TEST_VBUS_DRIVE.
6. Observe the Source DUT making Discovery pulses.
7. Measure the HIGH voltage of the pulses.
8. FAIL if HIGH voltage is not greater than $V_{IH_CBUS}\{min\}$.
9. If all preceding steps pass, then PASS; else FAIL.

Table 3-5. Discovery Pulse Drive HIGH Voltage

| | TEST_VBUS_DRIVE | TEST_RESISTOR | Purpose |
|---|--------------------------|-----------------------------------|--------------------------------|
| 1 | Sink does not drive VBUS | $Z_{CBUS_SINK_DISCOVER}\{min\}$ | Measure Source Pull-up Voltage |
| 2 | Sink drives VBUS | $Z_{CBUS_SINK_DISCOVER}\{min\}$ | Measure Source Pull-up Voltage |

3.3.5.5 CBE-Source: Discovery Pulse float LOW Voltage

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.5.5.1 Test Objective

Verify that Source DUT floats the CBUS so that the Sink $Z_{CBUS_SINK_DISCOVER}$ pulls down to make Discovery Pulses with LOW voltage for Sink to observe.

3.3.5.5.2 References

[MHL] Section 13.10.1; Table 13-24: V_{IL_CBUS}

[MHL] Section 8.4;

3.3.5.5.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.5.5.4 Required Methodology

Execute the following test procedure once for each row in Table 3-5, substituting for TEST_VBUS_DRIVE and TEST_RESISTOR.

1. If necessary, connect Source DUT to Tester Sink port.
2. If the CDF contains a definition of the procedure **CDF_SRC_PROC_STANDBY_DISCOVERY**, execute it.
3. Set Tester Sink port to disconnected state.
4. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
5. If the CDF value CDF_SRC_REQUIRES_SINK_VBUS indicates that the Source DUT requires Sink VBUS to operate, then SKIP steps 7-9 for table entry 1 (TEST_VBUS_DRIVE = Sink does not drive VBUS), else executethe Tester_Sink_VR_Discovery Procedure setting TEST_RESISTOR to $Z_{CBUS_SINK_DISCOVER}\{min\}$ and TEST_VBUS_DRIVE values from the column TEST_VBUS_DRIVE.
6. Observe the Source DUT making Discovery pulses.
7. Measure the LOW voltage of the pulses.
8. FAIL if LOW voltage is not less than $V_{IL_CBUS}\{max\}$.
9. If all preceding steps pass, then PASS; else FAIL.

Table 3-6. Discovery Pulse Float LOW Voltage

| | TEST_VBUS_DRIVE | TEST_RESISTOR | Purpose |
|---|--------------------------|-----------------------------------|--------------------------------|
| 1 | Sink does not drive VBUS | $Z_{CBUS_SINK_DISCOVER}\{max\}$ | Measure Source Pull-up Voltage |
| 2 | Sink drives VBUS | $Z_{CBUS_SINK_DISCOVER}\{max\}$ | Measure Source Pull-up Voltage |

3.3.6 Link Layer Timing – Source DUT Output: Discovery

Measure the timing of the Source DUT as it drives Discovery pulses.

3.3.6.1 CBT-Source: Wake Pulse HIGH, LOW and GAP Times

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.6.1.1 Test Objective

Verify that Source DUT drives Wake Pulses which meet HIGH timing requirements.

Verify that Source DUT drives Wake Pulses which meet LOW timing requirements.

Verify that Source DUT drives 2 pairs of Wake Pulse bursts which meet timing requirements.

3.3.6.1.2 References

[MHL] Section 8.3.1

[MHL] Section 8.4.2

[MHL] Section 13.10.1.1; Table 13-28: $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$

[MHL] Section 13.10.1.1; Table 13-28: $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}$

3.3.6.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.6.1.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Connect Oscilloscope to Tester to allow it to monitor the Source DUT CBUS wire.
3. Set Oscilloscope to trigger on a rising edge at about 1V.
4. Set Tester Sink port to disconnected state.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
7. Execute the Tester_Powered_Sink_Discovery_Procedure.
8. FAIL if DUT does not drive 2 Wake Pulses, followed by 2 more Wake Pulses.
9. FAIL if each Pulse does not meet the HIGH time requirement $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$.
10. FAIL if each Pulse does not meet the LOW time requirement $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$.
11. FAIL if the gap between pulse pairs does not meet the LOW time requirement $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}$.
12. If all preceding steps pass, then PASS; else FAIL.

3.3.6.2 CBT-Source: Wake Pulse Delay Until Discovery Pulse

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.6.2.1 Test Objective

Verify that Source DUT waits after Wake Pulse bursts before starting Discovery pulses.

3.3.6.2.2 References

[MHL] Section 8.3.1

[MHL] Section 8.4.2

[MHL] Section 13.10.1.1; Table 13-28: $T_{\text{SRC_WAKE_TO_DISCOVER}}$

3.3.6.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.6.2.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Connect Oscilloscope to Tester to allow it to monitor the Source DUT CBUS wire.
3. Set Oscilloscope to trigger on a rising edge at about 1V.
4. Set Tester Sink port to disconnected state.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.

6. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
7. Execute the Tester_Powered_Sink_Discovery_Procedure.
8. FAIL if the gap between the last wake pulse and the start of discovery pulses does not meet the LOW time requirement $T_{SRC_WAKE_TO_DISCOVER}$.
9. If all preceding steps pass, then PASS; else FAIL.

3.3.6.3 CBT-Source: Discovery Pulse HIGH and LOW Time

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.6.3.1 Test Objective

Verify that Source DUT drives Discovery Pulses which meet HIGH timing requirements.

Verify that Source DUT drives Discovery Pulses which meet LOW timing requirements.

Note that Discovery LOW pulses get LOW due to the Sink-side $Z_{CBUS_SINK_DISCOVER}$ pulldown resistor. The fall-time might reduce the LOW time (below V_{IL_CBUS}) of the pulses.

3.3.6.3.2 References

[MHL] Section 8.4.2

[MHL] Section 13.10.1.1; Table 13-30: $T_{SRC_PULSE_WIDTH}$

3.3.6.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.6.3.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Connect Oscilloscope to Tester to allow it to monitor the Source DUT CBUS wire.
3. Set Oscilloscope to trigger on a rising edge at about 1V.
4. Set Tester Sink port to disconnected state.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
7. Execute the Tester_Powered_Sink_Discovery_Procedure.
8. FAIL if DUT does not drive Discovery pulses which meet the HIGH time requirement $T_{SRC_PULSE_WIDTH}$.
9. FAIL if DUT does not drive Discovery pulses which meet the LOW time requirement $T_{SRC_PULSE_WIDTH}$.
10. If all preceding steps pass, then PASS; else FAIL.

3.3.6.4 CBT-Source: Wake/Discovery Pulse Edge Rate

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.6.4.1 Test Objective

Verify that Source DUT drives Discovery Pulses which have legal edge rates.

Note: Edge rates during discovery are not specified in the MHL 1.0 Specification.

3.3.6.4.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OH_CBUS}

[MHL] Section 13.10.1; Table 13-23: V_{OL_CBUS}

[MHL] Section 13.10.1; Table 13-23: T_{R_CBUS}

[MHL] Section 13.10.1; Table 13-23: T_{F_CBUS}

3.3.6.4.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.6.4.4 Required Methodology

Execute the following test procedure once for each row in Table 3-7, substituting for TEST_CAPACITOR and TEST_RESISTOR.

1. If necessary, connect Source DUT to Tester Sink port.
2. Connect Oscilloscope to Tester to allow it to monitor the Source DUT CBUS wire.
3. Set Oscilloscope to trigger on a rising edge at about 1V.
4. Set Tester Sink port to disconnected state.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Tester Sink capacitance to TEST_CAPACITOR.
7. Set tester thresholds to V_{IH_CBUS} and V_{IL_CBUS} .
8. Execute the Tester_Sink_VR_Discovery_Procedure, setting TEST_RESISTOR from the column TEST_RESISTOR with TEST_VBUS_DRIVE set to have the Tester drive VBUS.
9. FAIL if discovery pulses have a maximum rise or fall time of greater than 1 microsecond.
10. If all preceding steps pass, then PASS; else FAIL.

NOTE: A Source DUT which fails this test shall not be considered non-compliant, as there is presently no minimum or maximum limit on Wake Pulse and Discovery Pulse rise or fall times in the MHL Specification.

Table 3-7. Wake/Discovery Pulse Edge Rate

| | TEST_CAPACITOR | TEST_RESISTOR | Purpose |
|---|----------------------------------|-----------------------------------|------------------------------------|
| 1 | minimum cable + Sink capacitance | $Z_{CBUS_SINK_DISCOVER}\{max\}$ | Wake/Discovery Pulse Rise time MIN |
| 2 | maximum cable + Sink capacitance | $Z_{CBUS_SINK_DISCOVER}\{min\}$ | Wake/Discovery Pulse Rise time MAX |
| 3 | maximum cable + Sink capacitance | $Z_{CBUS_SINK_DISCOVER}\{max\}$ | Wake/Discovery Pulse Fall time MAX |

3.3.7 Link Layer Electrical – Source DUT Output: Arbitration/Sync/Data Signaling

Measure the electrical behavior of the Source DUT as it drives Arbitration, Sync, and Data pulses.

3.3.7.1 CBE-Source: Post-Discovery Pull-up HIGH Voltage

Application: Source

3.3.7.1.1 Test Objective

Verify that Source DUT pull-up Voltage has correct value when Source is connected with $Z_{CBUS_SRC_ON}$.

3.3.7.1.2 References

[MHL] Section 13.10.1; Table 13-23: V_{TERM_CBUS}

[MHL] Section 13.10.1.1; Table 13-30: T_{SRC_CONN}

3.3.7.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.7.1.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything .
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. After Discovery completes, wait greater than $2 * T_{SRC_PULSE_WIDTH}\{max\} + T_{SRC_CONN}\{max\}$ but less than $T_{SRC_PULSE_WIDTH}\{min\} + T_{SRC_ARBITRATE}\{min\}$ from the rising edge of the last discovery pulse Remove $Z_{CBUS_SINK_ON}$.
7. Measure CBUS voltage as V_{CBUS_MEAS} .
8. FAIL if V_{CBUS_MEAS} not within the range of V_{TERM_CBUS} .
9. If all preceding steps pass, then PASS; else FAIL.

3.3.7.2 CBE-Source: Post-Discovery Passive Pull-up HIGH $Z_{CBUS_SRC_ON}$ Resistance

| |
|----------------------------|
| Application: Source |
|----------------------------|

3.3.7.2.1 Test Objective

Verify that Source DUT pull-up $Z_{CBUS_SRC_ON}$ has correct value.

3.3.7.2.2 References

[MHL] Section 13.10.1; Table 13-25: $Z_{CBUS_SRC_ON}$

3.3.7.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.7.2.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. After Discovery completes, wait $2 * T_{SRC_PULSE_WIDTH}\{max\} + T_{SRC_CONN}\{max\}$ from the rising edge of the last discovery pulse.
7. Measure CBUS source impedance as Z_{CBUS_MEAS} .
8. FAIL if Z_{CBUS_MEAS} is not in the valid range of $Z_{CBUS_SRC_ON}$.
9. If all preceding steps pass, then PASS; else FAIL.

3.3.7.3 CBE-Source: CBUS Capacitance

| |
|----------------------------|
| Application: Source |
|----------------------------|

3.3.7.3.1 Test Objective

Verify that the Source DUT has a low-enough input capacitance

3.3.7.3.2 References

[MHL] Section 13.10.1; Table 13-25: $C_{SRC_ON_CBUS}$

3.3.7.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.7.3.4 Required Methodology

Characterize the capacitance of the Test Equipment, so that it can be subtracted from any measurement of DUT + Test Equipment.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. Measure the CBUS Capacitance (C_{CBUS_MEAS}) after Discovery completes.
7. FAIL if the calculated Source capacitance is greater than $C_{SRC_ON_CBUS}$.
8. If all preceding steps pass, then PASS; else FAIL.

NOTE: The capacitance measurement may depend on compliant termination of the CBUS line in the DUT. The test may result in a FAIL if the resistance $Z_{CBUS_SRC_ON}$ is out of range.

3.3.7.4 CBE-Source: Arbitrate/Sync/Data Drive LOW Voltage

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.7.4.1 Test Objective

Verify that Source DUT drives Arbitration, Sync, and Data Pulses with the correct DRIVE LOW voltage.

3.3.7.4.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OL_CBUS}

3.3.7.4.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.7.4.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. Set Tester Sink CBUS impedance to $Z_{CBUS_SINK_ON}\{max\}$.
7. Set Tester Sink to send CLR_HPD to Source DUT.
8. Observe LOW levels in the waveform while the Source DUT arbitrates for the bus.
9. FAIL if DUT does not do link-level ACK to CLR_HPD.
10. FAIL if DUT does not arbitrate for the use of the CBUS to send an ACK Packet.
11. FAIL if DUT does not drive CBUS LOW during the Arbitration pulse to a voltage below $V_{OL_CBUS}\{max\}$.
12. FAIL if DUT does not drive CBUS LOW during the Sync pulse to a voltage below $V_{OL_CBUS}\{max\}$.
13. FAIL if DUT does not drive CBUS LOW during the Header, Control, Data, and Parity pulse to a voltage below $V_{OL_CBUS}\{max\}$.
14. If all preceding steps pass, then PASS; else FAIL.

3.3.7.5 CBE-Source: Arbitrate/Sync/Data Drive HIGH Voltage

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.7.5.1 Test Objective

Verify that Source DUT drives Arbitration, Sync, and Data Pulses with correct DRIVE HIGH voltage.

3.3.7.5.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OH_CBUS}

3.3.7.5.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.7.5.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. Set Tester Sink CBUS impedance to $Z_{CBUS_SINK_ON}\{min\}$.
7. Set Tester Sink to send CLR_HPDP to Source DUT.
8. Observe HIGH levels in the waveform while the Source DUT arbitrates for the bus.
9. FAIL if DUT does not do link-level ACK to CLR_HPDP.
10. FAIL if DUT does not arbitrate for the use of the CBUS to send an ACK Packet.
11. FAIL if DUT does not drive CBUS HIGH after the Arbitration pulse to a voltage above $V_{OH_CBUS}\{min\}$ and below $V_{OH_CBUS}\{max\}$.
12. FAIL if DUT does not drive CBUS HIGH during the Sync pulse to a voltage above $V_{OH_CBUS}\{min\}$ and below $V_{OH_CBUS}\{max\}$.
13. FAIL if DUT does not drive CBUS HIGH during the Header, Control, Data, and Parity pulse to a voltage above $V_{OH_CBUS}\{min\}$ and below $V_{OH_CBUS}\{max\}$. If all preceding steps pass, then PASS; else FAIL.

Note: $V_{OH_CBUS}\{max\}$ should be measured at the end of the CBUS drive high time $T_{DRV_HI_CBUS}\{max\}$ to avoid failing devices due to impedance/driver signal overshoot.

3.3.8 Link Layer Timing – Source DUT Output: Arbitration/Sync/Data in Nanoseconds

Measure the timing of the Source DUT as it drives Arbitration and data pulses.

The Source DUT output Bit Time must be measured.

3.3.8.1 CBT-Source: Arbitration/Sync/Data Active Drive HIGH Duration

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.8.1.1 Test Objective

Verify that Source DUT drives CBUS HIGH, then releases bus, when changing the CBUS from LOW to HIGH. (The HIGH voltage will be sustained by a Resistor pull-up at the Source end.)

3.3.8.1.2 References

[MHL] Section 7.2.4

[MHL] Section 13.10.1; Table 13-23: $T_{DRV_HI_CBUS}$

3.3.8.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.8.1.4 Required Methodology

Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Source DUT to Tester Sink port.
2. Connect Oscilloscope to Tester to allow it to monitor the Source DUT CBUS wire.
3. Set Oscilloscope to trigger on a rising edge at about 1V.
4. Set Tester Sink port to disconnected state.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Tester Sink to use its Sink Packet engine to ACK everything.
7. Adjust the Tester resistances and voltages so that the DUT Drive High time is observable.
8. Set Tester Sink to measure the CBUS transition times during the test.
9. Execute the Tester_Powered_Sink_Discovery_Procedure.
10. After Discovery completes, wait $T_{SRC_CONN}\{max\}$.
11. Set Tester Sink to send CLR_HPDP to Source DUT.
12. Observe post-discovery Source activity on CBUS Tester Sink port.
13. FAIL if DUT has problems doing Discovery.
14. FAIL if DUT does not arbitrate for access to CBUS to return ACK to CLR_HPDP.
15. FAIL if the CBUS Drive HIGH time (the period starting when the CBUS goes from LOW to HIGH and lasting until only the Source resistor pull-up drives the CBUS) is outside the limits set for $T_{DRV_HI_CBUS}$.
16. If all preceding steps pass, then PASS; else FAIL.

3.3.8.2 CBT-Source: Arbitration/Sync/Data Edge Rate

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.8.2.1 Test Objective

Verify that Source DUT drives Arbitration/Sync/Data and ACK Pulses which have valid Rise or Fall times.

Verify that Source DUT Rise and Fall times for CBUS Arbitration/Sync/Data pulses vary at most by the allowable amount.

3.3.8.2.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OH_CBUS}

[MHL] Section 13.10.1; Table 13-23: V_{OL_CBUS}

[MHL] Section 13.10.1; Table 13-23: T_{F_CBUS}

[MHL] Section 13.10.1; Table 13-23: T_{R_CBUS}

[MHL] Section 13.10.1; Table 13-23: ΔT_{RF}

3.3.8.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.8.2.4 Required Methodology

Execute the following test procedure once for each row in Table 3-8, substituting for TEST_CAPACITOR and TEST_RESISTOR. Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Source DUT to Tester Sink port using a short cable with minimum capacitance.
2. Connect Oscilloscope to Tester to allow it to monitor the Source DUT CBUS wire.

3. Set Oscilloscope to trigger on a rising edge at about 1V.
4. Set Tester Sink port to disconnected state.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Tester Sink capacitance to TEST_CAPACITANCE.
7. Set Tester Sink to its Sink Packet engine to ACK everything.
8. Execute the Tester_Powered_Sink_Discovery_Procedure.
9. After Discovery completes, wait $T_{SRC_CONN}\{max\}$.
10. Set Tester Sink to send CLR_HPD to Source DUT.
11. FAIL if DUT does not arbitrate for access to CBUS to return ACK to CLR_HPD.
12. For Test 1 FAIL if Arbitration/Sync/Data pulses have a rise time less than $T_{R_CBUS}\{min\}$.
13. For Test 2 FAIL if Arbitration/Sync/Data pulses have a fall time less than $T_{F_CBUS}\{min\}$.
14. For Test 3 FAIL if Arbitration/Sync/Data pulses have a rise time greater than $T_{R_CBUS}\{max\}$.
15. For Test 4 FAIL if Arbitration/Sync/Data pulses have a fall time greater than $T_{F_CBUS}\{max\}$.
16. FAIL if rise times and fall times measured do not meet the ΔT_{RF} specification.
17. If all preceding steps pass, then PASS; else FAIL.

Table 3-8. Arbitration/Sync/Data Edge Rate

| | TEST_CAPACITOR | TEST_RESISTOR | Purpose |
|---|----------------------------------|-----------------------------|-------------------------------------|
| 1 | minimum cable + Sink capacitance | $Z_{CBUS_SINK_ON}\{max\}$ | Arbitration/Sync/Data Rise time MIN |
| 2 | minimum cable + Sink capacitance | $Z_{CBUS_SINK_ON}\{min\}$ | Arbitration/Sync/Data Fall time MIN |
| 3 | maximum cable + Sink capacitance | $Z_{CBUS_SINK_ON}\{min\}$ | Arbitration/Sync/Data Rise time MAX |
| 4 | maximum cable + Sink capacitance | $Z_{CBUS_SINK_ON}\{max\}$ | Arbitration/Sync/Data Fall time MAX |

3.3.9 Link Layer Timing – Source DUT Output: Arbitration/Sync/Data in Bit Times

Measure the timing of the Source DUT as it drives Data pulses.

The Source DUT output Bit Time must be measured based on observing CBUS edges.

3.3.9.1 CBT-Source: Arb, Sync, Data HIGH and LOW Times

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.9.1.1 Test Objective

Verify that the Source DUT does not arbitrate to send a first Packet over the CBUS until $T_{SRC_ARBITRATE}$ after discovery completes.

Verify that the Source DUT drives a CBUS LOW arbitration pulse for T_{ARB_SRC} . Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Source DUT drives CBUS HIGH before Sync pulse after winning arbitration. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Source DUT drives CBUS LOW to make a Sync pulse after winning arbitration. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Source DUT drives CBUS HIGH after the Sync pulse until it starts driving data. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Check the timing for Arbitration pulses in terms of “Bit Times”, and verify that it meets the required spec.

Check the Idle period between Arbitration and the Sync pulse in terms of “Bit Times”, and verify that it meets the required spec.

Check the Sync LOW pulse in terms of “Bit Times”, and verify that it meets the required spec.

Check the delay between the Sync LOW end and the first Data transition in terms of “Bit Times”, and verify that it meets the required spec.

Check the timing of 0 and 1 bits, taking care to observe duty-cycle and edge rate differences.

Measure the timings using the method described in Section 2.5.1.1.

3.3.9.1.2 References

[MHL] Section 7.2.2

[MHL] Section 8.5.1

[MHL] Section 13.10.1.1; Table 13-30: T_{SRC_CONN}

[MHL] Section 13.10.1.1; Table 13-31: $T_{SRC_ARBITRATE}$

[MHL] Section 13.10.1; Table 13-23: T_{BIT_CBUS}

[MHL] Section 13.10.2; Table 13-33: T_{ARB_SRC}

[MHL] Section 13.10.2; Table 13-33: T_{REQ_OPP}

[MHL] Section 13.10.1; Table 13-23; T_{CBUS_DUTY}

[MHL] Section 13.10.1; Table 13-23; T_{SYNC_TOTAL}

[MHL] Section 13.10.1; Table 13-23; T_{SYNC_DUTY}

[MHL] Section 7.2.3.1.

3.3.9.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.9.1.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
5. Set Tester Sink to use its Sink Packet engine to ACK everything.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.
7. After Discovery completes, wait $T_{SRC_CONN}\{max\}$.
8. Tester Sink does CLR_HPDP to Source DUT.
9. Measure the timing of the ACK packet from the Source DUT in nanoseconds.
10. Measure timing information in nanoseconds and assign values to a set of named timing variables.
 - a. ARB_LOW time
 - b. ARB_HIGH time
 - c. Idle_Time
 - d. Sync_LOW time
 - e. Sync_HIGH time
 - f. HIGH_0 time
 - g. LOW_0 time
 - h. Starting_LOW_1 timings
 - i. Starting_HIGH_1 timings
11. FAIL if DUT does CBUS arbitration which is both before its response to the CLR_HPDP and sooner than $T_{SRC_ARBITRATE}$ from the rising edge of the last Discovery pulse.
12. FAIL if CBUS ARB LOW time is not between (minimum or maximum T_{BIT_CBUS}) * T_{ARB_SRC} .
13. FAIL if CBUS HIGH time after arbitration but before Sync is not between (minimum or maximum T_{BIT_CBUS}) * T_{REQ_OPP} .
14. FAIL if CBUS LOW time during Sync is not between (minimum or maximum T_{BIT_CBUS}) * (minimum or maximum T_{SYNC_DUTY}).

15. FAIL if CBUS HIGH time between Sync LOW time and falling edge of first Data bit is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * ($T_{\text{SYNC_TOTAL}} - (\text{minimum or maximum } T_{\text{SYNC_DUTY}})$).
16. FAIL if time between Sync going LOW time and falling edge of first Data bit is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * ($T_{\text{SYNC_TOTAL}}$).
17. FAIL if DUT does not drive Data 0 bits which are asserted entirely LOW on the CBUS for the required period (in nanoseconds).
18. FAIL if DUT does not drive Data 0 bits which are asserted entirely HIGH on the CBUS for the required period (in nanoseconds).
19. FAIL if DUT does not drive Data 1 bits which start HIGH for the first half of a bit time for the required HIGH period (in nanoseconds).
20. FAIL if DUT does not drive Data 1 bits which start HIGH for the first half of a bit time for the following required LOW period (in nanoseconds).
21. FAIL if DUT does not drive Data 1 bits which start LOW for the first half of a bit time for the required LOW period (in nanoseconds).
22. FAIL if DUT does not drive Data 1 bits which start LOW for the first half of a bit time for the following required HIGH period (in nanoseconds).
23. FAIL if DUT does not drive Data 1 bits with required symmetry.
24. Calculate Bit Time range based on Arbitration pulse LOW.
25. Calculate Bit Time range based on HIGH period after the arbitration pulse.
26. Calculate Bit Time range based on Sync pulse LOW.
27. Calculate Bit Time range based on "1" bit timing.
28. Calculate Bit Time range based on pairs of "0" bit timings.
29. FAIL if the bit time ranges based on Arbitration, Sync, and Data bits differ.
30. If all preceding steps pass, then PASS; else FAIL.

3.3.9.2 CBT-Source: Continuous Monitor: Bit Timing Variation within a Packet

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.9.2.1 Test Objective

Continuously monitor Source DUT individual received packets to verify that Bit Timing does not change too quickly.

3.3.9.2.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_VARY_PACKET}}$

3.3.9.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.9.2.4 Required Methodology

1. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
2. Continuously monitor the CBUS to look for failures of this spec.
3. FAIL if the bit timing of any two consecutive bits within a packet differ by more than $T_{\text{BIT_VARY_PACKET}}$.
4. If all preceding steps pass, then PASS; else FAIL.

3.3.10 Link Layer Timing – Source DUT Output: Link-Level NACK

Measure the behavior of the Source DUT as it receives a NACK indication.

3.3.10.1 CBT-Source: Response to Link-Level NACK

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.10.1.1 Test Objective

Verify that Source DUT is willing to retransmit data $N_{\text{RETRY}}\{\text{min}\}$ times when it receives NACKs.

3.3.10.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.2; Table 13-33; N_{RETRY}

3.3.10.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.10.1.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to operate using a typical legal $T_{\text{BIT_CBUS}}$.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
6. Set Tester Sink to use Its Sink Packet engine to NACK the first $N_{\text{RETRY}}\{\text{min}\}-1$ packets from the Source DUT, then to ACK all subsequent packets.
7. Execute the Tester_Powered_Sink_Discovery_Procedure.
8. After Discovery completes, wait $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$.
9. Tester sends CLR_HPD.
10. Observe that the Source DUT responds with an ACK packet.
11. NACK this packet as it is retransmitted $N_{\text{RETRY}}\{\text{min}\}-1$ times.
12. Finally link-level ACK the ACK Packet from the Source DUT .
13. FAIL if DUT does not try to send an ACK packet.
14. FAIL if DUT does not release the bus based on the NACK.
15. FAIL if DUT does not retry the ACK Packet the required minimum number of times.
16. FAIL if DUT does not stop retrying the ACK Packet based on the final link-level ACK. This indicates that it is sensitive to the ACK where placed.
17. If all preceding steps pass, then PASS; else FAIL.

3.3.11 Link Layer Timing – Source DUT Output: ACK

Measure the timing of the Source DUT as it drives ACK pulses.

The Source DUT output Bit Time must be measured.

3.3.11.1 CBT-Source: ACK Output Timing in Nanoseconds

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.11.1.1 Test Objective

Measure when Source DUT drives ACK when Tester uses various Sink Bit timing and various Parity.

3.3.11.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 7.2.4

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$ Max-Min

[MHL] Section 7.2.4.1; Table 13-33; $T_{\text{REQ_OPP}}$

[MHL] Section 7.2.4.1; Table 13-33; $T_{\text{REQ_CONT}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{CBUS_ACK_FALL}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{CBUS_ACK_0}}$

3.3.11.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.11.1.4 Required Methodology

Execute the test procedure once for every row in Table 3-9, substituting TEST_IDLE_PERIOD and TEST_DATA from the table for each execution. Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. After Discovery completes, wait $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$.
7. Set Tester Sink to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
8. Tester Sink does SET_INT to Source DUT, releases the bus.
9. Tester Sink sends TEST_DATA for an offset, releases the bus.
10. Tester Sink sends Data of all 0's.
11. Observe post-discovery Source DUT link-level ACK activity as it replies to the Tester Sink command.
12. Observe that the Source DUT replies with an ACK packet.
13. FAIL if DUT has problems doing Discovery.
14. FAIL if DUT does not drive the Link Level ACK LOW for all 3 packets sent from the Tester.
15. FAIL if DUT does not drive the Link Level ACK LOW starting within the window defined by (0 to max) of $T_{\text{CBUS_ACK_FALL}}$ from the end of the Tester-driven packet parity bit.
16. FAIL if DUT drives an ACK LOW pulse which not between $T_{\text{CBUS_ACK_0}}\{\text{min}\}$ and $T_{\text{CBUS_ACK_0}}\{\text{max}\}$ wide.
17. If all preceding steps pass, then PASS; else FAIL.

Table 3-9. ACK Output Timing in Nanoseconds

| | TEST_IDLE_PERIOD | TEST_DATA | Purpose |
|---|------------------|-----------|--------------------|
| 1 | REQ_OPP{min} | 0x20 | Measure ACK timing |
| 2 | REQ_OPP{min} | 0x21 | Measure ACK timing |

3.3.11.2 CBT-Source: ACK Drive HIGH Duration

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.11.2.1 Test Objective

Measure the time the Source DUT drives ACK HIGH.

3.3.11.2.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.1; Table 13-23; $T_{\text{DRV_HI_CBUS}}$

[MHL] Section 7.2.4.1; Table 13-33; $T_{\text{REQ_CONT}}$

3.3.11.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.11.2.4 Required Methodology

Connect the Oscilloscope to observe the Tester Sink CBUS.

Either use the Oscilloscope to examine the CBUS HIGH period, or adjust the CBUS HIGH and CBUS LOW Limit comparators, to separate out the period at the beginning of a HIGH period where the Source DUT is actively pulling HIGH to signal ACK.

This HIGH voltage should be higher than the voltage throughout the rest of the CBUS HIGH period, where only a Source-side resistor is pulling the bus up.

Execute procedure once for every row in Table 3-10, substituting TEST_IDLE_PERIOD and TEST_DATA from the table for each execution.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Adjust the Tester resistances and voltages so that the DUT Drive High time is observable.
5. Set Tester Sink to use Its Sink Packet engine to ACK everything.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.
7. After Discovery completes, wait $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$.
8. Set Tester Sink to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
9. Tester Sink does SET_INT to Source DUT, releases the bus.
10. Tester Sink sends TEST_DATA for an offset, releases the bus.
11. Tester Sink sends Data of all 0's.
12. Observe post-discovery Source DUT link-level ACK activity as it replies to the Tester Sink command.
13. Observe that the Source DUT replies with an ACK packet.
14. FAIL if the CBUS Drive HIGH time during the ACK period (the period starting when the CBUS goes from LOW to HIGH and lasting until only the Source resistor pull-up drives the CBUS) is outside the limits set for $T_{\text{DRV_HI_CBUS}}$.
15. If all preceding steps pass, then PASS; else FAIL.

Table 3-10. ACK Drive HIGH Duration

| | TEST_IDLE_PERIOD | TEST_DATA | Purpose |
|---|------------------|-----------|--------------------|
| 1 | REQ_OPP{min} | 0x20 | Measure ACK timing |

3.3.12 Link Layer Timing – Source DUT Output: Bus Re-Arbitration

Measure the behavior and timing of the Source DUT as it re-arbitrates for the bus.

3.3.12.1 CBT-Source: Source Uses Case 2 Regular Arbitration after NACK

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.12.1.1 Test Objective

Verify that Source DUT backs off after a link-level NACK, and uses Case 2 regular arbitration timing to reacquire the bus.

3.3.12.1.2 References

[MHL] Section 7.2.4; Case 2

[MHL] Section 13.10.2; Table 13-33; T_{REQ_HOLD}

3.3.12.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.12.1.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use Its Sink Packet engine to NACK each received packet $N_{RETRY}\{min\}$ -1 times, then ACK it .
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.
7. After Discovery completes, wait $T_{SINK_ARBITRATE}\{min\}$.
8. Set Tester Sink to send CLR_HPDP to Source DUT.
9. After CLR_HPDP is sent, Source DUT sends an ACK packet.
10. Observe that the Source DUT does send an ACK packet.
11. Observe that the ACK Packet is retransmitted $N_{RETRY}\{min\}$ -1 times.
12. Carefully measure the DUT T_{BIT_CBUS} during the retransmissions of the ACK packet.
13. Finally link-level ACK the ACK Packet from the Source DUT.
14. Use the timing of the observed waveforms to calculate the DUT T_{BIT_CBUS} .
15. FAIL if DUT does not try to send an ACK packet.
16. FAIL if DUT does not release the bus based on the NACK.
17. FAIL if DUT does not retransmit the ACK packet as expected.
18. FAIL if DUT does arbitration (after a NACK) in less than $T_{REQ_HOLD}\{min\}$ (expressed in DUT Bit Times of the following packet).
19. FAIL if DUT does not eventually complete its command after retransmission.
20. If all preceding steps pass, then PASS; else FAIL.

3.3.12.2 CBT-Source: Source Uses Case 3 Long Re-arbitration

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.12.2.1 Test Objective

Verify that Source DUT uses Case 3 Long Arbitration whenever it gives up the bus and later re-acquires it.

3.3.12.2.2 References

[MHL] Section 7.2.4; Case 3

[MHL] Section 13.10.2; Table 13-33; T_{WAIT}

3.3.12.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.12.2.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.

4. Set Tester Sink to use its Sink Packet engine to ACK all packets.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.
7. After Discovery completes, wait $T_{SINK_ARBITRATE}\{min\}$.
8. Set Tester Sink to send READ_DEVCAP to Source DUT with offset of zero.
9. After READ_DEVCAP is sent, Source DUT sends an ACK packet followed by a data packet.
10. Use the timing of the observed waveforms to calculate the Source DUT T_{BIT_CBUS} .
11. FAIL if DUT does not respond with an ACK packet followed by a data packet.
12. FAIL if DUT does re-arbitration (after an ACK) sooner than T_{WAIT} (expressed in Source Bit Times of the subsequent packets).
13. If all preceding steps pass, then PASS; else FAIL.

3.3.12.3 CBT Source: Source Never Sends Too Many Back-to-Back Packets

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.12.3.1 Test Objective

Continuously monitor the CBUS to verify that the Source DUT does not send too many packets back-to-back.

3.3.12.3.2 References

[MHL] Section 7.2.4; Case 4

[MHL] Section 13.10.2; Table 13-33; N_{MAX}

3.3.12.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.12.3.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. Continuous monitor the CBUS wire after Discovery to detect Source DUT arbitration and packet activity.
2. FAIL if DUT ever sends more than N_{MAX} back-to-back packets without giving up the bus.
3. If all preceding steps pass, then PASS; else FAIL.

3.3.13 Link Layer Behavior – Source DUT Output: Ill-formed packets

Verify that the Source DUT does not send ill-formed packets (for instance, packets shorter than 16 bit-times long).

3.3.13.1 CBT-Source: Source Never Sends Impulse Noise

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.13.1.1 Test Objective

Continuously monitor Source DUT to check that it does not drive CBUS LOW for short periods.

3.3.13.1.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; T_{BIT_CBUS}

3.3.13.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.13.1.4 Required Methodology

Continuous monitor runs in the background.

FAIL if DUT seems to be responsible for LOW periods on the CBUS of less than 25% of T_{BIT_CBUS} .

If all preceding steps pass, then PASS; else FAIL.

3.3.13.2 CBT-Source: Source Never Sends Partial Packets

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.13.2.1 Test Objective

Continuously monitor Source DUT to check that it does not arbitrate for the CBUS, then send a short packet.

3.3.13.2.2 References

3.3.13.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.13.2.4 Required Methodology

Continuous monitor runs in the background.

FAIL if DUT drives arbitration pulse, then Sync pulse, then less than a full packet consisting of Control, Type, Data, and Parity bits.

If all preceding steps pass, then PASS; else FAIL.

3.3.14 Link Layer Timing – Source DUT Input: Discovery

Check that the Source DUT detects the Sink based on its response to Discovery pulses.

3.3.14.1 CBT-Source: Discovery; Sink Responds Correctly; Time to Source Pull-up Change

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.14.1.1 Test Objective

Verify that Source DUT discovers a Sink which responds after the correct number of Discovery pulses expected of an MHL device.

3.3.14.1.2 References

[MHL] Section 8.4.2

[MHL] Section 13.10.1; Table 13-25: $Z_{CBUS_SRC_DISCOVER}$, $Z_{CBUS_SRC_ON}$

[MHL] Section 13.10.1.1; Table 13-30: T_{SRC_CONN}

[MHL] Section 13.10.1.1; Table 13-31: $N_{SRC_PULSE_COUNT}$

3.3.14.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.14.1.4 Required Methodology

Execute the following test procedure once for each row in Table 3-11, substituting for TEST_SINK_PULSE_COUNT and TEST_MHL_PU_TIME.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Modified_Discovery_Procedure.
6. Carefully watch the HIGH voltage after the Sink allows discovery.
7. During Source Drive HIGH time of discovery pulses, the CBUS will go to a specific HIGH voltage.
8. The Source will free the CBUS one more time to detect a Discovered Sink by observing that CBUS stays HIGH.
9. Once the Sink is discovered, the Source DUT will change the value of its Pull-up from $Z_{CBUS_SRC_DISCOVER}$ to $Z_{CBUS_SRC_ON}$.
10. This will result in a modest voltage increase on the CBUS. (10K Pull-up/100K Pulldown to 5K Pull-up/100K Pulldown). This will result in a 5% increase in the CBUS voltage.
11. FAIL if Source does not complete Discovery.
12. FAIL if DUT does not switch its pull-up from $Z_{CBUS_SRC_DISCOVER}$ to $Z_{CBUS_SRC_ON}$ within T_{SRC_CONN} of when it HIGH-Z's the CBUS after the 6th discovery pulse rising edge.
13. If all preceding steps pass, then PASS; else FAIL.

Table 3-11. Discovery – Sink Responds Correctly – Time to Source Pull-up Change

| | TEST_SINK_PULSE_COUNT | TEST_MHL_PU_TIME | Purpose |
|---|--------------------------|------------------|--|
| 1 | $N_{SINK_PULSE_COUNT}$ | 0 | Response to Sink which does normal discovery |

3.3.14.2 CBT-Source: Discovery; Sink Responds Late

Application: Source

3.3.14.2.1 Test Objective

Verify that a Source DUT will detect a Sink which asserts $Z_{CBUS_SINK_ON}$ too late in Discovery as an invalid MHL device.

The Source DUT must not proceed beyond the Discovery phase. It must HIGH-Z the CBUS for a minimum of $T_{SRC_CBUS_FLOAT}$ after discovery fails.

3.3.14.2.2 References

[MHL] Section 8.2.1.1; Transition SRC9-SRC7

[MHL] Section 8.4.2

[MHL] Section 13.10.1.1; Table 13-31: $N_{SRC_PULSE_COUNT}$

3.3.14.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.14.2.4 Required Methodology

Execute the following test procedure once for each row in Table 3-12, substituting for TEST_SINK_PULSE_COUNT and TEST_MHL_PU_TIME.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Modified_Discovery_Procedure.

6. Carefully watch the HIGH voltage after the Sink allows discovery.
7. During Source Drive HIGH time of discovery pulses, the CBUS will go to a specific HIGH voltage.
8. FAIL if Source sends ($N_{\text{SRC_PULSE_COUNT}}\{\text{max}\} + 1$) discovery pulses, causing the Sink Tester to complete discovery and change from $Z_{\text{BUS_SINK_DISCOVER}}$ to $Z_{\text{CBUS_SINK_ON}}$.
9. FAIL if the Source DUT does not HIGH-Z the CBUS for a minimum of $T_{\text{SRC_CBUS_FLOAT}}$ after discovery fails.
10. If all preceding steps pass, then PASS; else FAIL.

Table 3-12. Discovery – Sink Responds Late

| | TEST_SINK_PULSE_COUNT | TEST_MHL_PU_TIME | Purpose |
|---|--|------------------|-------------------------------------|
| 1 | $N_{\text{SRC_PULSE_COUNT}}\{\text{max}\} + 1$ | 0 | Response to Sink discovers too late |

3.3.14.3 CBT-Source: Discovery; Sink Never Drives MHL+/- HIGH

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.14.3.1 Test Objective

Verify that Source DUT detects that a Sink which never pulls up on MHL+/- wires is an invalid MHL device. When the Source detects this, it must HIGH-Z the CBUS for a minimum of $T_{\text{SRC_CBUS_FLOAT}}$ after discovery fails.

3.3.14.3.2 References

[MHL] Section 8.2.1.1; Transition SRC4-SRC7

[MHL] Section 8.4.2

[MHL] Section 13.2; Table 13-1

[MHL] Section 13.6.3; Table 13-9; $Z_{\text{RXSENSE_TERM}}$

[MHL] Section 13.10.1.1; Table 13-31: $T_{\text{SRC_RXSENSE_CHK}}$

3.3.14.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.14.3.4 Required Methodology

Execute the following test procedure once for each row in Table 3-13, substituting for TEST_SINK_PULSE_COUNT and TEST_MHL_PU_TIME.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Modified_Discovery_Procedure.
6. Carefully watch the CBUS HIGH voltage after the Sink allows discovery.
7. During Source Drive HIGH time of discovery pulses, the CBUS will go to a specific HIGH voltage.
8. Note that the Source will free the CBUS one more time to detect a Discovered Sink by observing that CBUS stays HIGH.
9. Note that once the Sink is discovered, the Source DUT will change the value of its Pull-up from $Z_{\text{CBUS_SRC_DISCOVER}}$ to $Z_{\text{CBUS_SRC_ON}}$.
10. Note that this will result in a modest voltage increase on the CBUS. (10K Pull-up/100K Pulldown to 5K Pull-up/100K Pulldown). This will result in a 5% increase in the CBUS voltage.
11. FAIL if Source does not undiscover after TEST_MHL_PU_TIME because neither $Z_{\text{RXSENSE_TERM}}$ nor $Z_{\text{RXSENSE_TMDS}}$ are attached.

12. FAIL if Source does not HIGH-Z the CBUS for $T_{SRC_CBUS_FLOAT}$ to indicate that the attached Sink device should reset.
13. If all preceding steps pass, then PASS; else FAIL.

Table 3-13. Discovery – Sink Never Drives MHL+/- HIGH

| | TEST_SINK_PULSE_COUNT | TEST_MHL_PU_TIME | Purpose |
|---|-------------------------------|---------------------------------------|--|
| 1 | N _{SINK_PULSE_COUNT} | $1.01 * T_{SRC_RXSENSE_CHK\{max\}}$ | Response to Sink driving MHL+/- too late |

3.3.15 Link Layer Electrical – Source DUT Input: Arbitration/Sync/Data signaling

Measure the electrical sensitivity of the Source DUT as it receives Arbitration, Sync, and Data pulses.

3.3.15.1 CBE-Source: Sensitivity to VIH/VIL

Application: Source

3.3.15.1.1 Test Objective

Verify that the Source DUT correctly receives data pulses above $V_{IH_CBUS\{min\}}$ and below $V_{IL_CBUS\{min\}}$.

3.3.15.1.2 References

[MHL] Section 8-4

[MHL] Section 13.10.1; Table 13-23; V_{OH_CBUS} , V_{OL_CBUS}

[MHL] Section 13.10.1; Table 13-24: V_{IH_CBUS} , V_{IL_CBUS}

3.3.15.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.15.1.4 Required Methodology

Execute the following test procedure once for each row in Table 3-14, substituting for TEST_VOH_CBUS and TEST_VOL_CBUS.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Prepare system to normally respond to device register reads.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.
7. Set Tester Sink HIGH output voltage to TEST_VOH_CBUS.
8. Set Tester Sink LOW output voltage to TEST_VOL_CBUS.
9. Set Tester Sink to do CLR_HPD to Source DUT.
10. FAIL if DUT does not ACK the CLR_HPD in less than 32 retries.
11. If all preceding steps pass, then PASS; else FAIL.

The Source DUT will probably NACK the first several attempts to do CLR_HPD. Retransmit.

Table 3-14. Sensitivity to VIH/VIL{min}

| | TEST_VOH_CBUS | TEST_VOL_CBUS | Purpose |
|---|-----------------------|------------------------------|--------------------|
| 1 | $V_{IH_CBUS\{min\}}$ | $V_{OL_CBUS\{min\}}$ (0.0V) | Sensitivity to VIH |
| 2 | $V_{OH_CBUS\{max\}}$ | $V_{IL_CBUS\{max\}}$ | Sensitivity to VIL |

3.3.16 Link Layer Timing – Source DUT Input: Arbitration

Measure the behavior of the Source DUT as it receives Arbitration pulses with different timings.

3.3.16.1 CBT-Source: Source Behaves Appropriately when Sink Wins Arbitration

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.16.1.1 Test Objective

Verify that Source DUT loses arbitration to the Tester acting as Sink using various Bit Timing.

3.3.16.1.2 References

[MHL] Section 8.5.1

[MHL] Section 13.10.1.1; Table 13-33: T_{REQ_OPP}

[MHL] Section 13.10.1.1; Table 13-31: $T_{SINK_ARBITRATE}$

3.3.16.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.16.1.4 Required Methodology

Execute the following test procedure once for each row in Table 3-15, substituting for TEST_BIT_TIME, TEST_IDLE_PERIOD and TEST_FIRST_SINK_COMMAND_DELAY.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
5. Set Tester Sink to use its Sink Packet engine to ACK everything.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.
7. Set Tester Sink to operate using TEST_BIT_TIME for T_{BIT_CBUS} .
8. Set Tester Sink to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
9. Wait TEST_FIRST_SINK_COMMAND_DELAY.
10. Tester Sink does CLR_HPD to Source DUT.
11. FAIL if DUT does not respond with an ACK packet to the CLR_HPD command from the Tester.
12. If all preceding steps pass, then PASS; else FAIL.

Note that the Source DUT might NACK the CLR_HPD command. Tester needs to re-issue this command up to N_{RETRY} times to give the Source DUT a chance to ACK it.

Table 3-15. Source Behaves Appropriately when Sink Wins Arbitration

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_FIRST_SINK_COMMAND_DELAY | Purpose |
|---|------------------------|-----------------------|-------------------------------|---|
| 1 | $T_{BIT_CBUS}\{typ\}$ | $T_{REQ_OPP}\{min\}$ | $T_{SINK_ARBITRATE}\{min\}$ | nominal Sink Bit Timing, shortest ARB to SYNC delay |
| 2 | $T_{BIT_CBUS}\{typ\}$ | $T_{REQ_OPP}\{max\}$ | $T_{SINK_ARBITRATE}\{min\}$ | nominal Sink Bit Timing, longest ARB to SYNC delay |
| 3 | $T_{BIT_CBUS}\{min\}$ | $T_{REQ_OPP}\{min\}$ | $T_{SINK_ARBITRATE}\{min\}$ | fast Sink Bit Timing, shortest ARB to SYNC delay |
| 4 | $T_{BIT_CBUS}\{min\}$ | $T_{REQ_OPP}\{max\}$ | $T_{SINK_ARBITRATE}\{min\}$ | fast Sink Bit Timing, longest ARB |

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_FIRST_SINK_COMMAND_DELAY | Purpose |
|---|--------------------------------------|-------------------------------------|--|--|
| | | | | to SYNC delay |
| 5 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$ | slow Sink Bit Timing, shortest ARB to SYNC delay |
| 6 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$ | slow Sink Bit Timing, longest ARB to SYNC delay |

3.3.16.2 CBT-Source: End of Discovery to Early Sink-side Arbitration

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.16.2.1 Test Objective

Verify that the Source DUT correctly responds to a packet if a Sink sends one before the minimum Discovery-to-Arbitration holdoff.

3.3.16.2.2 References

[MHL] Section 8.5.1

[MHL] Section 13.10.1.1; Table 13-31: $T_{\text{SRC_CONN}}$

[MHL] Section 13.10.1.1; Table 13-31: $T_{\text{SINK_ARBITRATE}}$

3.3.16.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.16.2.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
5. Set Tester Sink to use its Sink Packet engine to ACK everything.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.
7. Set Tester Sink to operate using $T_{\text{BIT_CBUS}}\{\text{typ}\}$ for $T_{\text{BIT_CBUS}}$.
8. Set Tester Sink to use $T_{\text{REQ_OPP}}\{\text{min}\}$ for delay from Arbitration pulse to Sync pulse.
9. Wait $T_{\text{SRC_CONN}}$.
10. Tester Sink does CLR_HPD to Source DUT.
11. FAIL if DUT does not do a link-layer ACK in response to the CLR_HPD command from the Tester.
12. If all preceding steps pass, then PASS; else FAIL.

Note that the Source DUT might NACK the CLR_HPD command. It needs to be re-issued up to N_{RETRY} times to give the Source DUT a chance to ACK it.

3.3.17 Link Layer Timing – Source DUT Input: Data

Measure the behavior of the Source DUT as it receives Data pulses with different timings.

3.3.17.1 CBT-Source: Bit Timing Sensitivity

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.17.1.1 Test Objective

Verify that Source DUT can receive data from a Sink with the fastest/slowest/most asymmetric legal bit timings.

3.3.17.1.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}} \pm T_{\text{CBUS_DUTY}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}} \pm \Delta T_{\text{RF}}$

3.3.17.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.17.1.4 Required Methodology

Execute the following test procedure once for each row in Table 3-16, substituting TEST_BIT_TIME, TEST_IDLE_PERIOD, TEST_CENTER_OFFSET, TEST_RISE_OFFSET.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
5. Set Tester Sink to use its Sink Packet engine to ACK everything.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.
7. After Discovery completes, wait $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$.
8. Set Tester Sink to operate using TEST_BIT_TIME for $T_{\text{BIT_CBUS}}$.
9. Set Tester Sink to TEST_IDLE_PERIOD delay from Arbitration pulse to Sync pulse.
10. Set Tester Sink to offset "1" bit center-bit transitions by TEST_CENTER_OFFSET.
11. Set Tester Sink to offset rising edges TEST_RISE_OFFSET compared to falling edges.
12. Tester Sink does CLR_HPD to Source DUT.
13. FAIL if DUT does not ACK the CLR_HPD command from the Tester.
14. If all preceding steps pass, then PASS; else FAIL.

Tester Sink switches to responding with ACK to everything.

Table 3-16. Bit Timing Sensitivity

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_CENTER_OFFSET | TEST_RISE_OFFSET | Purpose |
|---|--------------------------------------|-------------------------------------|------------------------------|------------------|---------------------------|
| 1 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | 0 | 0 | Fast; Bits symmetrical |
| 2 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{typ}\}$ | 0 | 0 | Typical; Bits symmetrical |
| 3 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | 0 | 0 | Slow; Bits symmetrical |
| 4 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | $-\text{TEST_BIT_TIME}/10$ | 0 | Fast; midpoint early |
| 5 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{typ}\}$ | $-\text{TEST_BIT_TIME}/10$ | 0 | Typical; midpoint early |
| 6 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | $-\text{TEST_BIT_TIME}/10$ | 0 | Slow; midpoint early |

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_CENTER_OFFSET | TEST_RISE_OFFSET | Purpose |
|----|-----------------------------|----------------------------|--------------------|-----------------------|--------------------------------|
| 7 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | +TEST_BIT_TIME/10 | 0 | Fast; midpoint late |
| 8 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {typ} | +TEST_BIT_TIME/10 | 0 | Typical; midpoint late |
| 9 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | +TEST_BIT_TIME/10 | 0 | Slow; midpoint late |
| 10 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | 0 | - ΔT _{RF} /2 | Fast; rise faster than fall |
| 11 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {typ} | 0 | - ΔT _{RF} /2 | Typical; rise faster than fall |
| 12 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | 0 | - ΔT _{RF} /2 | Slow; rise faster than fall |
| 13 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | 0 | + ΔT _{RF} /2 | Fast; fall faster than rise |
| 14 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {typ} | 0 | + ΔT _{RF} /2 | Typical; fall faster than rise |
| 15 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | 0 | + ΔT _{RF} /2 | Slow; fall faster than rise |

NOTE: A tester which cannot independently adjust the rise and fall times of its output can achieve the same result by displacing either its falling edge or its rising edge from their ideal timings.

The tester should place the mid-point transition (at the DUT) at the time when a slow rising or a falling edge would present by moving a fast edge one-half the rise and fall time difference.

3.3.18 Link Layer Timing – Source DUT Input: NACK

Bad parity is defined as a Packet containing mostly “0” bits, with a coding error at the beginning of D7. Refer to Section 3.3.2.12.

3.3.18.1 CBT-Source: Response to Packet with Bad Parity

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.18.1.1 Test Objective

Verify that Source DUT can detect bad parity.

3.3.18.1.2 References

[MHL] Section 7.2.3.5

3.3.18.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.18.1.4 Required Methodology

Execute the following test procedure once for each row in Table 3-17, substituting TEST_BIT_RATE, TEST_IDLE_PERIOD, and TEST_BAD_PACKET.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to operate using the TEST_BIT_RATE for $T_{\text{BIT_CBUS}}$.
5. Set Tester Sink to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
6. Set Tester Sink to use its Sink Packet engine to ACK everything.
7. Execute the Tester_Powered_Sink_Discovery_Procedure.
8. Tester Sink sends TEST_BAD_PACKET $N_{\text{RETRY}}\{\text{min}\}-1$ times.
9. FAIL if DUT does not NACK the bad Packet from the Tester $N_{\text{RETRY}}\{\text{min}\}-1$ times.
10. If all preceding steps pass, then PASS; else FAIL.

Table 3-17. Check Response to Bad Parity Packet

| | TEST_BIT_RATE | TEST_IDLE_PERIOD | TEST_BAD_PACKET | Purpose |
|---|--------------------------------------|-------------------------------------|-----------------|---------------------------|
| 1 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | Error-Packet-1 | bad parity (Nominal Sink) |
| 2 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | Error-Packet-1 | bad parity (Slow Sink) |
| 3 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | Error-Packet-1 | bad parity (Fast Sink) |
| 4 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | Error-Packet-2 | bad parity (Nominal Sink) |
| 5 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | Error-Packet-2 | bad parity (Slow Sink) |
| 6 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | Error-Packet-2 | bad parity (Fast Sink) |

3.3.19 Link Layer Timing – Source DUT Input: ACK

Measure the behavior of the Source DUT as it receives ACK pulses with different timings.

3.3.19.1 CBT-Source: Response to ACKs with Different Timings

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.19.1.1 Test Objective

Source DUT receives Tester ACK at various times. Verify that Source DUT recognizes ACK.

3.3.19.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.1; Table 13-23, $T_{\text{CBUS_ACK_0}}$

3.3.19.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.19.1.4 Required Methodology

Execute the following test procedure once for each row in Table 3-18, substituting for TEST_BIT_TIME, TEST_ACK_TIME, and TEST_ACK_DURATION.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
5. Set Tester Sink to use Its Sink Packet engine to ACK everything.
6. Execute the Tester_Powered_Sink_Discovery_Procedure.

7. Set Tester Sink to operate using TEST_BIT_TIME for $T_{\text{BIT_CBUS}}$.
8. Set Tester Sink to place ACK pulses at the TEST_ACK_TIME after expected ACK period starts.
9. Set Tester Sink to drive the ACK pulse LOW for TEST_ACK_DURATION.
10. After Discovery completes, wait $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$.
11. Set Tester Sink to send CLR_HPD to Source DUT.
12. Observe that Source DUT sends an ACK packet in response to the CLR_HPD.
13. ACK the ACK Packet from the DUT using the calculated bit timing and the previously set ACK pulse placement.
14. FAIL if DUT does not ACK the CLR_HPD.
15. FAIL if DUT retries sending the ACK Packet after the Tester link-level ACK of the DUT ACK packet. This indicates that the DUT is sensitive to where the link-level ACK is placed.
16. If all preceding steps pass, then PASS; else FAIL.

NOTE: the Source DUT might NACK the CLR_HPD Packet. Tester needs to re-issue this command up to $N_{\text{RETRY}}\{\text{min}\}$ times to give the Source DUT a chance to ACK it.

Table 3-18. Response to ACKs with Different Timings

| | TEST_BIT_TIME | TEST_ACK_TIME | TEST_ACK_DURATION | Purpose |
|----|--------------------------------------|--|---|-----------------------------|
| 1 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{min}\}$ | Earliest possible thin ACK |
| 2 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{min}\}$ | Latest possible thin ACK |
| 3 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{max}\}$ | Earliest possible thick ACK |
| 4 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{max}\}$ | Latest possible thick ACK |
| 5 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{min}\}$ | Earliest possible thin ACK |
| 6 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{min}\}$ | Latest possible thin ACK |
| 7 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{max}\}$ | Earliest possible thick ACK |
| 8 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{max}\}$ | Latest possible thick ACK |
| 9 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{min}\}$ | Earliest possible thin ACK |
| 10 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{min}\}$ | Latest possible thin ACK |
| 11 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{max}\}$ | Earliest possible thick ACK |
| 12 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{CBUS_ACK_FALL}}\{\text{max}\}$ | $T_{\text{CBUS_ACK_0}}\{\text{min}\}$ | Latest possible thick ACK |

3.3.20 Link Layer Timing – Source DUT Input: Bus Re-Arbitration

Measure the behavior of the Source DUT as it responds to Sink re-arbitration with different timings.

3.3.20.1 CBT-Source: Source receives Case 1 Back-to-Back Transfers

Application: Source

3.3.20.1.1 Test Objective

Verify that Source DUT can receive Case 1 Back-to-Back packets when connected to various speed Sinks using variable ARB-to-Sync Idle timing.

3.3.20.1.2 References

[MHL] Section 7.2.4; Case 1

3.3.20.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.20.1.4 Required Methodology

Execute the following test procedure once for each row in Table 3-19, substituting for TEST_BIT_TIME and TEST_IDLE_PERIOD.

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to operate using TEST_BIT_TIME as $T_{\text{BIT_CBUS}}$.
5. Set Tester Sink to use TEST_IDLE_PERIOD as delay from Arbitration pulse to Sync pulse.
6. Set Tester Sink to use its Sink Packet engine to ACK everything.
7. Execute the Tester_Powered_Sink_Discovery_Procedure.
8. After Discovery completes, wait $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$.
9. Tester Sink does SET_INT to Source DUT, followed by Offset 0x20, followed by Data 0x00. The Offset packet has Parity == 0, the Data packet has Parity == 1. The last 2 Packets are sent back-to-back.
10. Observe post-discovery Source DUT link-level ACK activity as it replies to the Tester Sink command.
11. Observe that the Source DUT replies with an ACK packet.
12. FAIL if DUT does not link level ACK all 3 packets sent from the Tester.
13. FAIL if DUT does not respond with an ACK packet.
14. If all preceding steps pass, then PASS; else FAIL.

NOTE: the Source DUT might NACK the packets it receives. Tester needs to re-issue this command up to N_{RETRY} times to give the Source DUT a chance to ACK it.

Table 3-19. Source Receives Back-to-Back Transfers

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | Purpose |
|---|--------------------------------------|--------------------------------------|--|
| 1 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | nominal Sink Bit Timing, shortest ARB to SYNC |
| 2 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | nominal Sink Bit Timing, longest ARB to SYNC delay |
| 3 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | minimum Sink Bit Timing, shortest ARB to SYNC |
| 4 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | minimum Sink Bit Timing, longest ARB to SYNC delay |
| 5 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | maximum Sink Bit Timing, shortest ARB to SYNC |
| 6 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | maximum Sink Bit Timing, longest ARB to SYNC delay |

3.3.21 Link Layer Behavior – Source DUT Input: Ill-formed packets

Verify that the Source DUT does not have problems when the Sink sends ill-formed packets.

3.3.21.1 CBT-Source: Sink wins Arbitration, then No Packet, when Source is Idle

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.21.1.1 Test Objective

Verify that Source DUT is not confused if the Sink arbitrates for the CBUS but then does not use it.

3.3.21.1.2 References

[MHL] Section 7.2.4

3.3.21.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.21.1.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. Set Tester Sink to occasionally drive a total of 100 CBUS LOW pulses for periods of 1000 nanoseconds (1 nominal bit time), followed by drive HIGH as usual. These are valid Sink-side arbitration pulses, but are not followed by Sync pulses.
7. Set Tester Sink to wait until the bogus arbitration test period ends, then do a CLR_HPDP.
8. FAIL if DUT responds in any way to the bad arbitration activity.
9. FAIL if DUT cannot respond to the valid CLR_HPDP command from the Tester with an ACK packet.
10. If all preceding steps pass, then PASS; else FAIL.

3.3.22 Link Layer Timing – Source DUT Input: Disconnect

Verify that the Source responds appropriately when the MHL bus is disconnected (or when the Sink pretends that the bus is disconnected).

3.3.22.1 CBT-Source: Remove MHL+/- Pull-ups for Less than Glitch Reject Time

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.22.1.1 Test Objective

Verify that Source DUT does not over-react to short glitch on $Z_{RXSENSE_TERM}$ on MHL+ and MHL-.

3.3.22.1.2 References

[MHL] Section 13.10.1.1; Table 13-31; $T_{SRC:RXSENSE_DEGLITCH}$

3.3.22.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.22.1.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. After Discovery completes, wait $T_{SRC:RXSENSE_CHK}\{max\}$.
7. Tester Sink HIGH-Z's MHL+/- for $T_{SRC:RXSENSE_DEGLITCH}\{min\} - 1\text{ msec}$.
8. Tester Sink returns MHL+/- Pull-ups to $Z_{RXSENSE_TERM}$.
9. FAIL if DUT disconnects the CBUS (as evidenced by the CBUS going LOW because of the Sink Tester pulldown), because it thinks the MHL+/- wires are LOW.
10. If all preceding steps pass, then PASS; else FAIL.

3.3.22.2 CBT-Source: Remove MHL+/- Pull-up for More than Glitch Reject Time

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.22.2.1 Test Objective

Verify that Source detects disconnect on $Z_{RXSENSE_TERM}$ on MHL+/- wires.

3.3.22.2.2 References

[MHL] Section 13.10.1.1; Table 13-31; $T_{SRC:RXSENSE_DEGLITCH}$

[MHL] Section 13.10.1.1; Table 13-32; $T_{SRC:CBUS_TMDS_DIS}$

[MHL] Section 8.2.1.1 (SRC7)

[MHL] Section 13.10.1.1; Table 13-30; $T_{SRC_CBUS_FLOAT}$

3.3.22.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.22.2.4 Required Methodology

1. If necessary, connect Source DUT to Tester Sink port.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
4. Set Tester Sink to use its Sink Packet engine to ACK everything.
5. Execute the Tester_Powered_Sink_Discovery_Procedure.
6. After Discovery completes, wait $T_{SRC:RXSENSE_CHK}\{max\}$.
7. Tester Sink HIGH-Z's MHL+/- for $T_{SRC:RXSENSE_DEGLITCH}\{max\} + 1$ msec.
8. Tester Sink returns MHL+/- Pull-ups to $Z_{RXSENSE_TERM}$.
9. FAIL if DUT does not detect the Sink Disconnect and float CBUS within $T_{SRC:RXSENSE_DEGLITCH}\{max\} + T_{SRC:CBUS_TMDS_DIS}\{max\}$ after the end of the MHL+/- float period. CBUS should drift low if the tester sink maintains a pulldown resistance that will overcome $I_{CBUS_LEAK_SRC}\{max\}$.
10. FAIL if DUT does not change impedance on CBUS to meet $I_{CBUS_LEAK_SRC}$ for at least $T_{SRC_CBUS_FLOAT}\{min\}$.
11. If all preceding steps pass, then PASS; else FAIL.

3.3.22.3 CBT-Source: Time from Disconnect until VOUT Falls

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.22.3.1 Test Objective

Verify that when MHL+/- are detected as disconnected by the Source DUT, it will stop driving VOUT to reduce power.

3.3.22.3.2 References

[MHL] Section 13.10.1.1; Table 13-32; $T_{SRC:CBUS_TMDS_DIS}$

[MHL] Section 8.6.2; Figure 8-15;

3.3.22.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.22.3.4 Required Methodology

1. If the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source then continue, else end test with PASS (SKIP).
2. If necessary, connect Source DUT to Tester Sink port.
3. Set Tester Sink port to disconnected state.
4. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Sink to use its Sink Packet engine to ACK everything.
6. Execute the Tester_Unpowered_Sink_Discovery_Procedure.
7. After Discovery completes, wait $T_{SINK_ARBITRATE}\{min\}$.

8. Tester Sink does CLR_HPDP to Source DUT.
9. After Discovery completes, wait $T_{SRC_RXSENSE_CHK}\{max\}$.
10. Tester Sink HIGH-Z's MHL+/- for $T_{SRC_RXSENSE_DEGLITCH}\{max\} + 1msec$.
11. FAIL if DUT does not enable VBUS as part of Discovery.
12. FAIL if DUT does not react to the long MHL glitch by floating CBUS. CBUS should drift low if the tester sink maintains a pulldown resistance that will overcome $I_{CBUS_LEAK_SRC}\{max\}$.
13. FAIL if DUT does not stop driving VBUS within $T_{SRC_CBUS_TMD_DIS}\{max\}$ from the end of the MHL+/- HIGH-Z period.
14. If all preceding steps pass, then PASS; else FAIL.

3.3.23 Link Layer Electrical – Source DUT VBUS Control

3.3.23.1 CBE-Source: Source DUT Prediscovery and Postdiscovery VBUS Input

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

This test is replaced with tests 3.3.23.4 and 3.3.23.5.

3.3.23.2 CBE-Source: Source DUT Prediscovery VBUS Output, Postdiscovery VBUS Input

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.23.2.1 Test Objective

Verify that a Powered Source supplies power over the VBUS until Sink provides VBUS.

3.3.23.2.2 References

[MHL] Section 8.2.1

[MHL] Section 8.2.2

[MHL] Section 13.9.1.1; Table 13-21: $V_{VBUS_TP1_SRC_DRV}$

[MHL] Section 13.9.1.2; VBUS Current Control, $I_{VBUS_SINK_TO_SOURCE}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SINK_VBUS_EN}$, $T_{SINK_READY_TO_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SRC_VBUS_EN}$, $T_{SRC_VBUS_CBUS_STABLE}$

3.3.23.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.23.2.4 Required Methodology

1. If the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source then continue, else end test with PASS (SKIP).
2. Connect Source DUT to Tester Sink port using 20cm MHL Test Cable.
3. Set Tester Sink port to disconnected state.
4. Set Tester Sink to not drive VBUS, and set its POW bit to '0'.
5. Set Tester Sink to apply an $I_{VBUS_PREDISCOVERY}\{max\}$ load (100mA) to the VBUS.
6. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
7. Set Source DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
8. Execute the Tester_Unpowered_Sink_Discovery procedure.
9. FAIL if Source DUT does not wait at least $T_{SRC_VBUS_CBUS_STABLE}\{min\}$ from the assertion of the Tester Discovery resistor, and after the wait completes the DUT drives VBUS.
10. After discovery completes, Tester Sink measures the VBUS driven by the Source DUT and loaded by the Tester Sink.

11. FAIL if VBUS Voltage is not within the range $V_{VBUS_TP1_SRC_DRV}\{min\}$ to $V_{VBUS_TP1_SRC_DRV}\{max\}$.
12. Do not set the Tester Sink to drive the VBUS. Since Tester is not driving the VBUS, it can observe the VBUS to see when the DUT stops driving the VBUS.
13. Set Tester Sink to write its own POW bit to '1'.
14. Set Tester Sink to send DCAP_CHG interrupt.
15. FAIL if Source fails to read at least Tester Sink Device Capabilities Register 2 to fetch the POW bit, starting after the DCAP_CHG interrupt is set.
16. FAIL if Source fails to disable its VBUS drive in $< (T_{SINK:DCAP_RDY} + T_{SRC:READ_DCAP} + T_{SRC:VBUS_DIS})$.
17. If all preceding steps pass, then PASS; else FAIL.

3.3.23.3 CBE-Source: Source DUT VBUS Output to Dongle

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.23.3.1 Test Objective

Verify that a Powered Source supplies power over the VBUS until Dongle provides VBUS.

3.3.23.3.2 References

[MHL] Section 8.2.1

[MHL] Section 8.2.2

[MHL] Section 13.9.1.1; Table 13-21: $V_{VBUS_TP1_SRC_DRV}$

[MHL] Section 13.9.1.2; VBUS Current Control, $I_{VBUS_SINK_TO_SOURCE}$

[[MHL] Section 13.10.1.1; Table 13-29: $T_{SINK:VBUS_EN}$, $T_{SINK:READY_TO_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SRC:VBUS_EN}$, $T_{SRC:VBUS_OUT_TO_STABLE}$

3.3.23.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.23.3.4 Required Methodology

1. If the CDF value **CDF_SRC_POWERED** indicates that Source DUT is a Powered Source then continue, else end test with PASS (SKIP).
2. Connect Source DUT to Tester Dongle (Sink) port using 20cm MHL Test Cable.
3. Set Tester Dongle (Sink) port to disconnected state.
4. Set Tester Dongle (Sink) Device Capability Registers to emulate Dongle, including the DEV_TYPE bits.
5. Set Tester Dongle (Sink) to not drive VBUS, and set its POW bit to '0'.
6. Set Tester Dongle (Sink) to apply an $I_{VBUS_SOURCE_TO_DONGLE}\{min\}$ load (200mA) to the VBUS.
7. Set Tester Dongle (Sink) port to use typical timings and to exhibit typical resistances and capacitances.
8. Set Source DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
9. Execute the Tester_Unpowered_Sink_Discovery procedure.
10. FAIL if Source DUT does not wait at least $T_{SRC_VBUS_CBUS_STABLE}\{min\}$ from the assertion of the Tester Discovery resistor, and after the wait completes the DUT drives VBUS.
11. After discovery completes, Tester Dongle (Sink) measures the VBUS driven by the Source DUT and loaded by the Tester Dongle (Sink).
12. FAIL if VBUS Voltage is not within the range $V_{VBUS_TP1_SRC_DRV}\{min\}$ to $V_{VBUS_TP1_SRC_DRV}\{max\}$.
13. Do not set the Tester Sink to drive the VBUS. Since Tester is not driving the VBUS, it can observe the VBUS to see when the DUT stops driving the VBUS.
14. Set Tester Dongle (Sink) to write its own POW bit to '1'.
15. Set Tester Dongle (Sink) to send DCAP_CHG interrupt.

16. FAIL if Source fails to read at least Tester Sink Device Capabilities Register 2 to fetch the POW bit, starting after the DCAP_CHG interrupt is set.
17. FAIL if Source fails to disable its VBUS drive in $< (T_{\text{DONGLE:DCAP_CHG}} + T_{\text{SRC:READ_DCAP}} + T_{\text{SRC:VBUS_DIS}})$.
18. If all preceding steps pass, then PASS; else FAIL.

3.3.23.4 CBE-Source: Source DUT Prediscovery VBUS Input Current Draw

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.23.4.1 Test Objective

Verify that a Source DUT does not violate Prediscovery current limits when connected to a Powered Sink.

3.3.23.4.2 References

[MHL] Section 8.2.1

[MHL] Section 8.2.2

[MHL] Section 13.9.1.1; Table 13-21: $V_{\text{VBUS_TP1_SINK_DRV}}$

MHL Section 13.9.1.2; VBUS Current Control, $I_{\text{VBUS_Prediscovery}}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{\text{SINK:VBUS_EN}}$, $T_{\text{SINK:READY_TO_DISCOVER}}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{\text{SRC:VBUS_EN}}$, $T_{\text{SRC:VBUS_CBUS_STABLE}}$

3.3.23.4.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.23.4.4 Required Methodology

1. Connect Source DUT to Tester Sink port using 20cm MHL Test Cable.
2. Set Tester Sink port to disconnected state.
3. Set Tester Sink to drive current limited $I_{\text{VBUS_PREDISCOVERY}}$ (100mA) VBUS.
4. Allow time for Source DUT to increase its current load on the VBUS.
5. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
6. Execute the Tester_Powered_Sink_Incomplete_Discovery procedure.
7. Measure $I_{\text{VBUS_PREDISCOVERY}}$ and VBUS during discovery attempt.
8. Fail if DUT Draws more than $I_{\text{VBUS_PREDISCOVERY}}$ (100mA) and $\text{VBUS} < V_{\text{VBUS_TP1_SINK_DRV}}\{\text{min}\}$.
9. Set Tester Sink port to disconnected state.
10. Set Tester Sink to drive current limited $I_{\text{VBUS_PREDISCOVERY}}$ (500mA) VBUS
11. Allow time for Source DUT to increase its current load on the VBUS.
12. Set Tester Sink port to use typical timings and to exhibit typical resistances and capacitances.
13. Execute the Tester_Sink_Incomplete_Discovery procedure.
14. Measure $I_{\text{VBUS_PREDISCOVERY}}$ and VBUS during discovery attempt.
15. Fail if DUT Draws more than $I_{\text{VBUS_PREDISCOVERY}}$ (500mA) and $\text{VBUS} < V_{\text{VBUS_TP1_SINK_DRV}}\{\text{min}\}$.
16. If all preceding steps pass, then PASS; else FAIL.

3.3.23.5 CBE-Source: Source DUT Postdiscovery VBUS Input Current Draw

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

3.3.23.5.1 Test Objective

Verify that a Source DUT does not violate Postdiscovery current limits when connected to a Powered Sink.

3.3.23.5.2 References

[MHL] Section 8.2.1

[MHL] Section 8.2.2

[MHL] Section 13.9.1.1; Table 13-21: $V_{VBUS_TP1_SINK_DRV}$

MHL] Section 13.9.1.2; VBUS Current Control, $I_{VBUS_Postdiscovery}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SINK:VBUS_EN}$, $T_{SINK:READY_TO_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SRC:VBUS_EN}$, $T_{SRC:VBUS_CBUS_STABLE}$

3.3.23.5.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

3.3.23.5.4 Required Methodology

Execute the following test procedure once for each row in Table 3-19, substituting for MHL version, device, PLIM bits, I_{VBUS} delivery.

1. Connect Source DUT to Tester Sink port using 20cm MHL Test Cable.
2. Set Tester port to disconnected state.
3. Set Tester port to drive current limited $I_{VBUS_Prediscovery}$ (100mA) onto VBUS.
4. Set Tester port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester port to display MHL version, device, and PLIM bits where applicable.
6. Set Tester port to drive current limited $I_{VBUS_Delivery}$ onto VBUS.
7. Set Tester port POW bit = 1.
8. Execute the Tester_Sink_Discovery procedure.
9. Wait until Discovery completes.
10. Tester port sends DCAP_RDY using WRITE_STAT and then DCAP_CHG using SET_INT to DUT.
11. Allow time for Source DUT to increase its current load on the VBUS.
12. FAIL if DUT draws more than $I_{VBUS_Delivery}$.
13. Check to see if DUT is simultaneously driving VBUS.
14. FAIL if DUT is simultaneously driving VBUS.
15. If all preceding steps pass, then PASS; else FAIL.

Table 3-20. Tester Sink Emulation

| | MHL Version | Device | PLIM | VBUS delivery | Purpose |
|---|-------------|--------|-------|---------------|-----------------|
| 1 | 1.x | Sink | N.A. | 500mA | 500mA draw test |
| 2 | 2.x | Sink | {0,0} | 500mA | 500mA draw test |
| 3 | 2.x | Sink | {0,1} | 900mA | 900mA draw test |
| 4 | 2.x | Sink | {1,0} | 1.5A | 1.5A draw test |
| 5 | 2.x | Dongle | {1,1} | 100mA | 100mA draw test |

4 Sink Test

4.1 Electrical Tests

4.1.1 TMD5 Electrical Tests

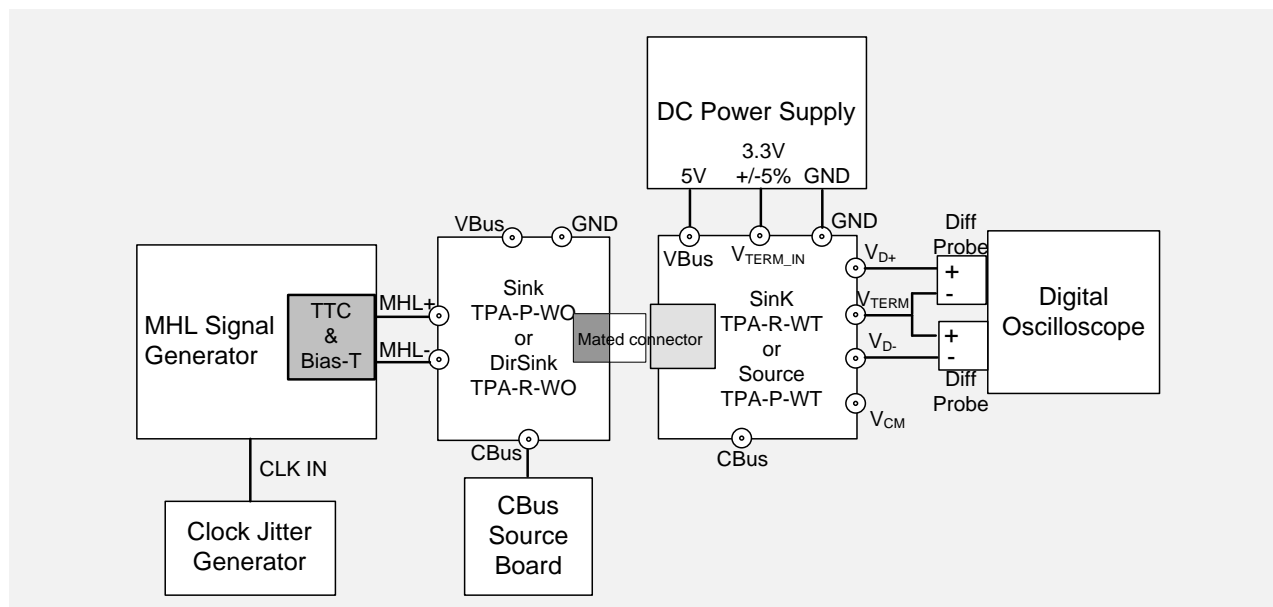


Figure 31. Single-Ended Signals Calibration Setup for Sink Tests

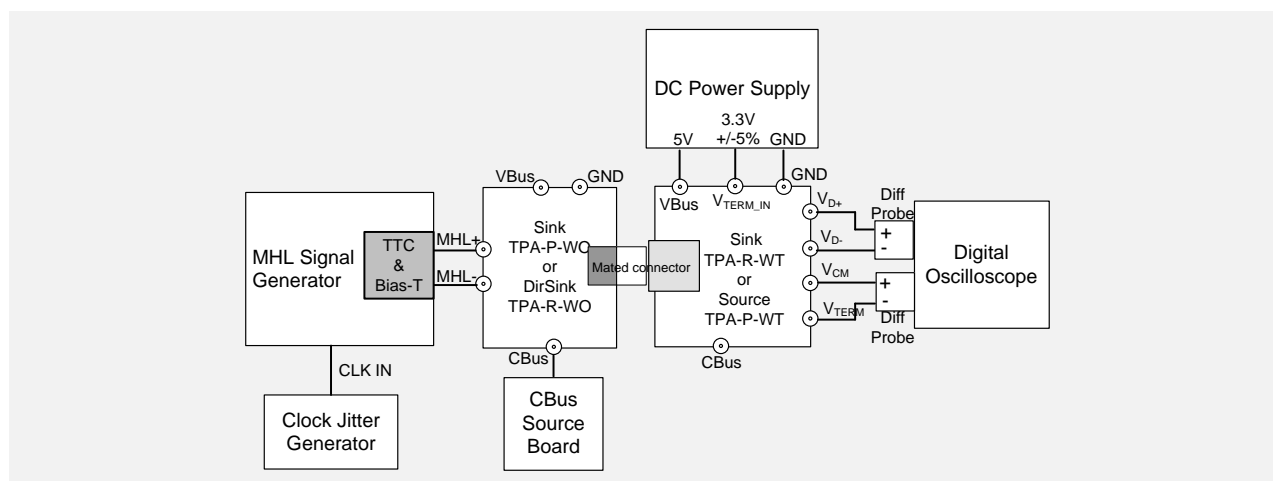


Figure 32. Differential & Common-Mode Signals Calibration Setup for Sink Tests

4.1.1.1 Input Signal DC Voltage Level Tolerance

Application: Sink

4.1.1.1.1 Test Objective

This test confirms that the sink device supports input signal DC voltage level allowed by the specification.

4.1.1.1.2 References

[MHL] Table 13-7 Receiver DC Characteristics at TP2: $V_{TERM} - 1200mV \leq V_{IDC} \leq V_{TERM} - 300mV$

4.1.1.1.3 Required Test Equipment

High-bandwidth Digital Oscilloscope
High-Bandwidth Probe
Transition Time Converter (TTC)
Bias Tee
MHL Signal Generator
Sink TPA-R-WT
Sink TPA-P-WO or DirSink TPA-R-WO
CBUS Source Board

4.1.1.1.4 Required Methodology

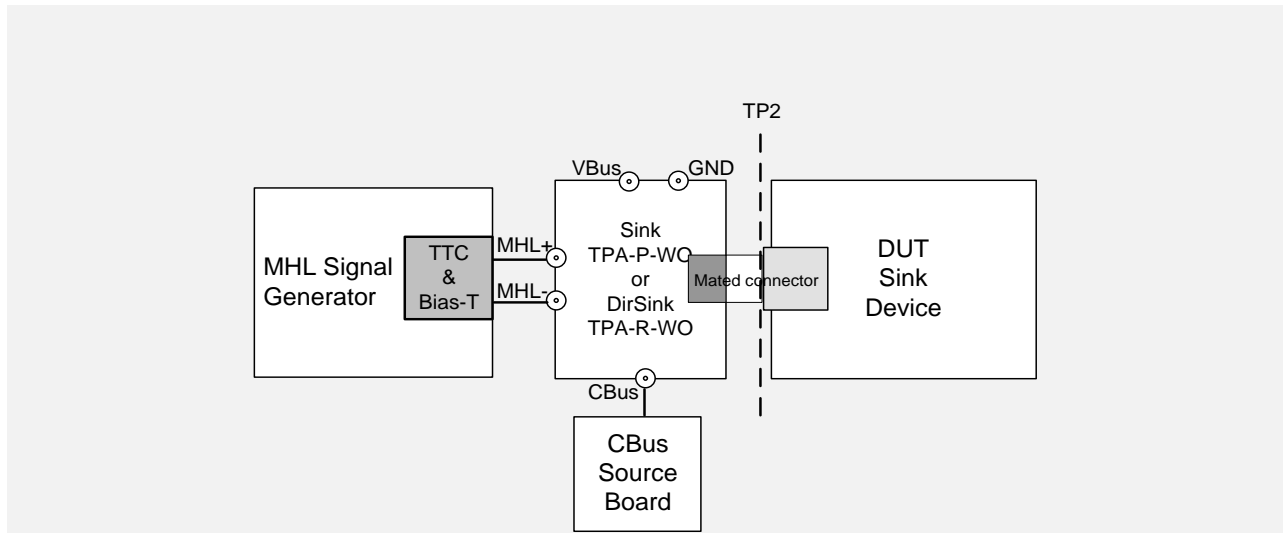


Figure 33. Sink – Input Signal DC Voltage Level Test Setup

1. Connect the equipment as shown in Figure 32. Set $V_{\text{TERM}} = 3.3\text{V}$.
2. Calibrate the MHL Signal Generator output for the following settings.
 - a. Frequency: Highest MHL data bit rate supported by the Sink DUT
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times:
 - i. Less than or equal to 2.25Gbps:
 1. Differential signal 200ps (20%-80%)
 2. Common-mode signal 600ps (20%-80%)
 - ii. Greater than 2.25Gbps:
 1. Differential signal 100ps (20%-80%)
 2. Common-mode signal 600ps (20%-80%)
 - d. Swing voltages: Differential signal 800mV, Common-mode signal 540mV
3. Connect the equipment as shown in Figure 31. Adjust the DC level of the output single-ended signal to $V_{\text{TERM}} - 300\text{mV}$.
4. Connect the equipment as shown in Figure 33.
5. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
6. Examine the output of the Sink through the display image.

7. Record errors in the display if any for the given V_{TERM} , MHL data bit rate and DC level.
8. Connect the equipment as shown in Figure 31. Adjust the DC level of the output single-ended signal to $V_{TERM} - 1200\text{mV}$.
9. Connect the equipment as shown in Figure 33.
10. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
11. Examine the output of the Sink through the display image and sound.
12. Record errors in the display if any for the given V_{TERM} , MHL data bit rate and DC level.
13. If there are no errors in the display in all recorded results, then PASS. Otherwise FAIL.

4.1.1.2 Input Signal Minimum and Maximum Swing Voltages Tolerance

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.1.1.2.1 Test Objective

This test confirms that the sink device supports input signal DC voltage level and swing voltage allowed by the specification.

4.1.1.2.2 References

[MHL] Table 13-7 Receiver DC Characteristics at TP2:

$$V_{TERM} - 1200\text{mV} \leq V_{IDC} \leq V_{TERM} - 300\text{mV}$$

$$200\text{mV} \leq V_{IDF} \leq 1000\text{mV}$$

$$170\text{mV} \leq V_{ICM} \leq \text{Min}(720\text{mV}, 0.85V_{IDF})$$

4.1.1.2.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

Transition Time Converter (TTC)

Bias Tee

MHL Signal Generator

Sink TPA-R-WT

Sink TPA-P-WO or DirSink TPA-R-WO

CBUS Source Board

4.1.1.2.4 Required Methodology

1. Connect the equipment as shown in Figure 31 for single-ended signals calibration and Figure 32. Set $V_{TERM} = 3.3\text{V}$.
2. Calibrate the Signal Generator output for the following settings.
 - a. Frequency: Highest MHL data bit rate supported by the Sink DUT
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps
Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - c. Swing voltages: Differential signal 800mV, Common-mode signal 540mV

- d. DC level of the output single-ended signal: $V_{\text{TERM}} - 750\text{mV}$
3. Connect the equipment as shown in Figure 34.
4. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
5. Examine the output of the Sink through the display image.
6. Record errors on the display if any for the given V_{TERM} , MHL data bit rate and differential and common-mode swing voltages.
7. Connect the equipment as shown in Figure 31 for single-ended signals calibration and Figure 32 for differential and common-mode signals adjustment.
8. Adjust the differential swing voltage to 1000mV, the common-mode swing voltage to 720mV and the DC level of the output single-ended signal to $V_{\text{TERM}} - 1200\text{mV}$.
9. Connect the equipment as shown in Figure 34.
10. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
11. Examine the output of the Sink through the display image.
12. Record errors on the display if any for the given MHL data bit rate and differential and common-mode swing voltages.
13. Connect the equipment as shown in Figure 31 for single-ended signals calibration and Figure 32 for differential and common-mode signals adjustment.
14. Adjust the differential swing voltage to 200mV, the common-mode swing voltage to 170mV and the DC level of the output single-ended signal to $V_{\text{TERM}} - 300\text{mV}$.
15. Connect the equipment as shown in Figure 34.
16. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
17. Examine the output of the Sink through the display image.
18. Record errors on the display if any for the given MHL data bit rate and differential and common-mode swing voltages.
19. If there is no error on the display in all recorded results, then PASS. Otherwise FAIL.

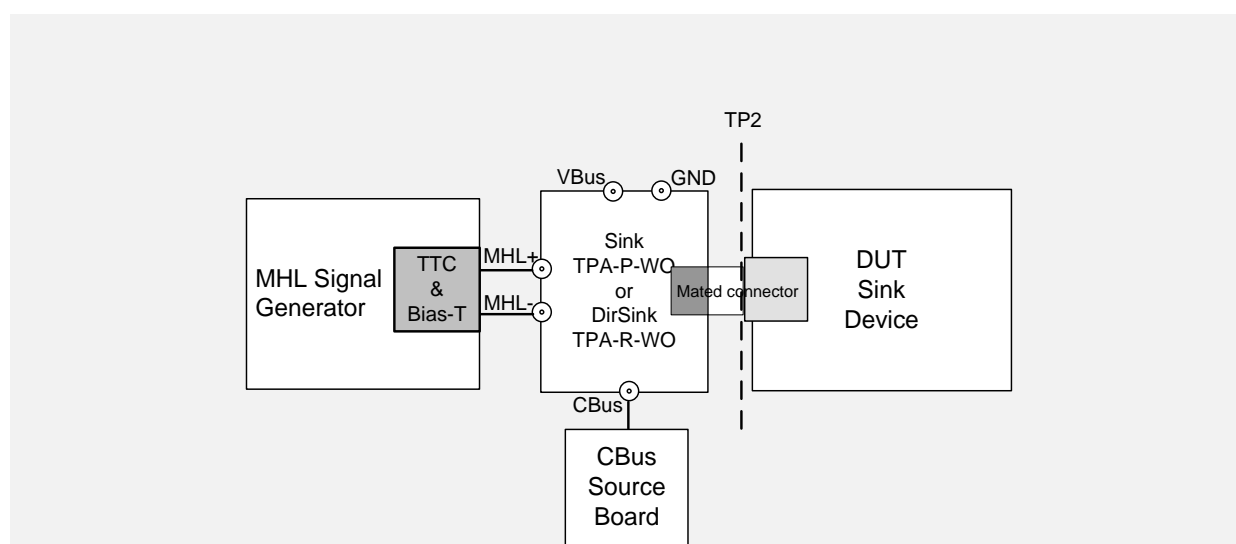


Figure 34. Sink – Input Signal Min and Max Swing Voltage Tolerance Test Setup

4.1.1.3 Intra-Pair Skew Tolerance

Application: Sink

4.1.1.3.1 Test Objective

This test confirms that the sink device can tolerate the maximum intra-pair skew allowed by the specification.

4.1.1.3.2 References

[MHL] Table 13-8 Receiver AC Characteristics at TP2:

$$T_{\text{SKEW_DF}} \leq \text{Min}(0.25T_{\text{BIT}}, 93\text{ps})$$

$$T_{\text{SKEW_CM}} \leq \text{Min}(0.25T_{\text{BIT}}, 93\text{ps})$$

4.1.1.3.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

Transition Time Converter (TTC)

Bias Tee

MHL Signal Generator

Sink TPA-R-WT

Sink TPA-P-WO or DirSink TPA-R-WO

CBUS Source Board

4.1.1.3.4 Required Methodology

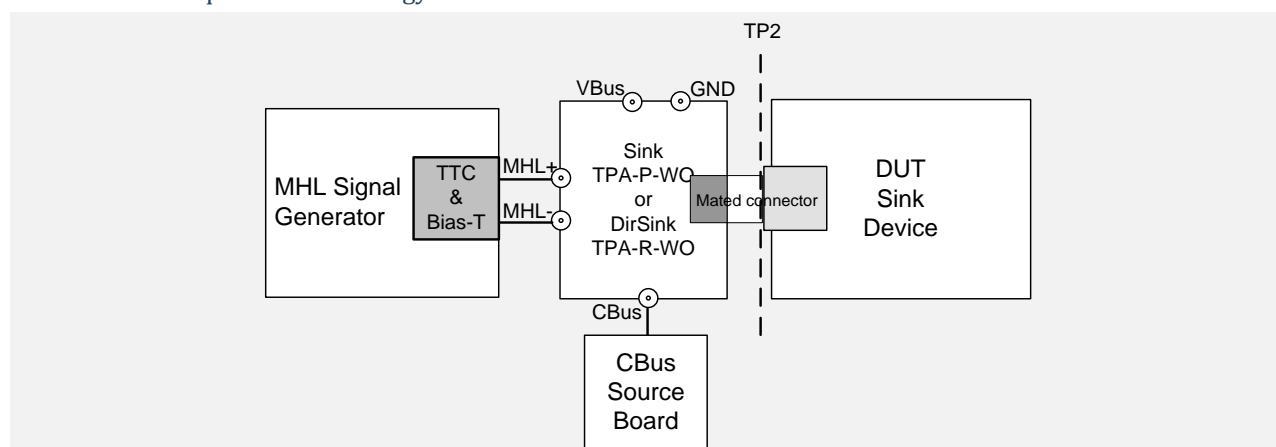


Figure 35. Sink – Intra-Pair Skew Tolerance Test Setup

1. Connect the equipment as shown in Figure 31 for single-ended signals calibration and Figure 32 for differential and common-mode signals. Set $V_{\text{TERM}} = 3.3\text{V}$.
2. Calibrate the Signal Generator output for the following settings.
 - a. Frequency: Highest MHL data bit rate supported by the Sink DUT
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps
Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)

- c. Swing voltages: Differential signal 800mV, Common-mode signal 540mV
 - d. DC level of the output single-ended signal: $V_{\text{TERM}} - 750\text{mV}$
 - e. Intra-pair skew: Differential skew 0ps, Common-mode skew 0ps
3. Connect the equipment as shown in Figure 35.
4. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
5. Examine the output of the Sink through the display image.
6. Confirm that there is no error on the display. If there is any error, STOP. The test FAILED.
7. Increase the intra-pair skew of differential signal and common-mode signal by the same amount to the maximum where error appears on the display. Make the positive channel leads the negative channel in time, which is called positive skew direction.
8. Record the maximum intra-pair skew for the given MHL data bit rate and skew direction.
9. Set the differential and common-mode intra-pair skews to 0ps.
10. Increase the intra-pair skew of differential signal and common-mode signal by the same amount to the maximum where error appears on the display. Make the positive channel lags the negative channel in time, which is called negative skew direction.
11. Record the maximum intra-pair skew for the given MHL data bit rate and skew direction.
12. If the all recorded maximum intra-pair skew value is larger than the spec limit, then PASS. Otherwise FAIL.

4.1.1.4 Jitter Tolerance in Normal Mode

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.1.1.4.1 Test Objective

This test confirms that the sink device can tolerate the maximum clock and data jitter amounts allowed by the specification in Normal Mode.

4.1.1.4.2 References

[MHL] Figure 13-16 MHL Clock Jitter at TP2

[MHL] Figure 13-17 MHL Data Eye Mask at TP2

[MHL] Figure 13-18 MHL Data Jitter at TP2

4.1.1.4.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

Jitter and Eye Analyzer

Clock Jitter Generator

MHL Cable Emulator

Bias Tee

Transition Time Converter (TTC)

MHL Signal Generator

Sink TPA-R-WT or Source TPA-P-WT

Sink TPA-P-WO or DirSink TPA-R-WO

CBUS Source Board

4.1.1.4.4 Required Methodology

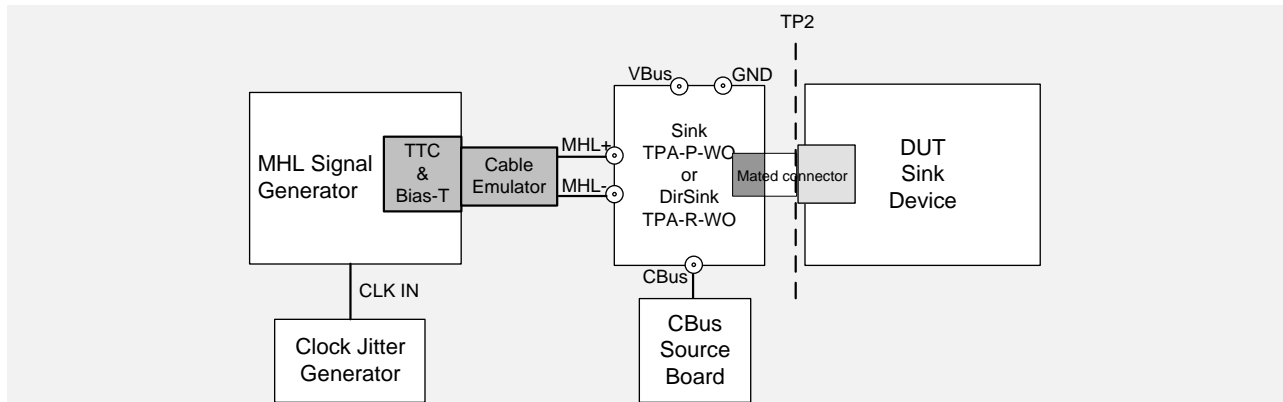


Figure 36. Sink – Jitter Tolerance Test Setup

1. Connect the equipment as shown in Figure 31 for single-ended signals calibration and Figure 32 for differential and common-mode signals. Set $V_{TERM} = 3.3V$.
2. Calibrate the Signal Generator output for the following settings.
 - a. Frequency: Lowest MHL Normal Mode data bit rate supported by the Sink DUT
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps
Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - c. Swing voltages: Differential signal 800mV, Common-mode signal 360mV
 - d. DC level of the output single-ended signal: $V_{TERM} - 540mV$
3. Set the clock jitter frequency to 10MHz and the data jitter frequency to 500KHz.
4. Calibrate the Signal Generator and Clock Jitter Generator to get data eye diagram and clock jitter which are marginal to TP1 specification.
 - a. Adjust clock jitter and data jitter to the TP1 maximum amount specified in MHL Spec Figures 13-13 and 13-15 for the given clock frequency.
 - b. Adjust the differential signal swing to get the marginal eye diagram specified in MHL Spec Figure 13-14.
5. Connect the equipment as shown in Figure 36.
6. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
7. Select a video mode which uses Normal Mode.
8. Examine the output of the Sink through the display image.
9. Record error on the display if any for the given clock frequency, jitter frequency and clock edge location.
10. Change the common-mode clock edge location by $0.1T_{bit}$ step with respect the differential data edge for 10 times.
11. Examine the display at each step and record error on the display if any for the given clock frequency, jitter frequency and clock edge location.
12. Connect the equipment as shown in Figure 32.
13. Set the clock jitter frequency to 7MHz and the data jitter frequency to 1MHz.
14. Repeat steps 4 to 11 for clock jitter frequency 7MHz and data jitter frequency 1MHz.

15. Repeat steps 1 to 14 for the highest supported MHL data bit rate in 24-Bit-per-pixel mode.
16. Disconnect Cable Emulator and connect MHL signal generator directly to the TPA board.
17. Repeat steps 1-15 for jitter tolerance test without Cable Emulator.
18. If there is no error on the display in all recorded results, then PASS. Otherwise FAIL.

4.1.1.5 Differential Impedance

Application: Sink

4.1.1.5.1 Test Objective

This test confirms that the differential input impedance of the sink device is within the range allowed by the specification.

4.1.1.5.2 References

[MHL] Table 13-9 Sink Device Impedance Characteristics at TP2:

Through Connection: 100 ohms +/- 15 ohms*

* A single excursion is permitted out to 100 ohms +/- 25 ohms, no wider than 250ps

At termination: 100 ohms +/- 10 ohms

4.1.1.5.3 Required Test Equipment

TDR Scope or VNA-base TDR

Sink TPA-P-WO or DirSink TPA-R-WO

CBUS Source Board

SMA Coaxial Cable

DC Block

4.1.1.5.4 Required Methodology

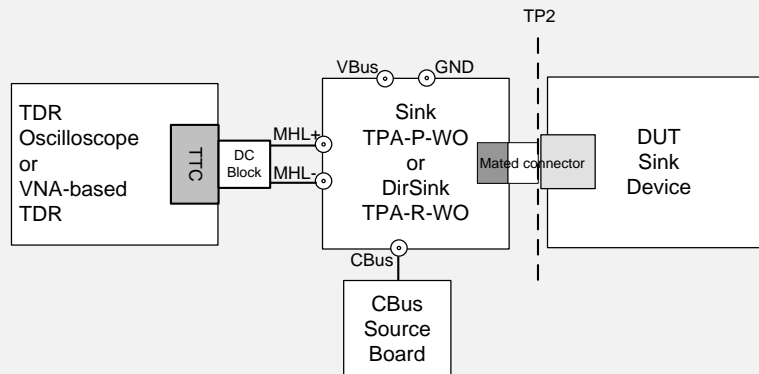


Figure 37. Sink – Differential Impedance Test Setup

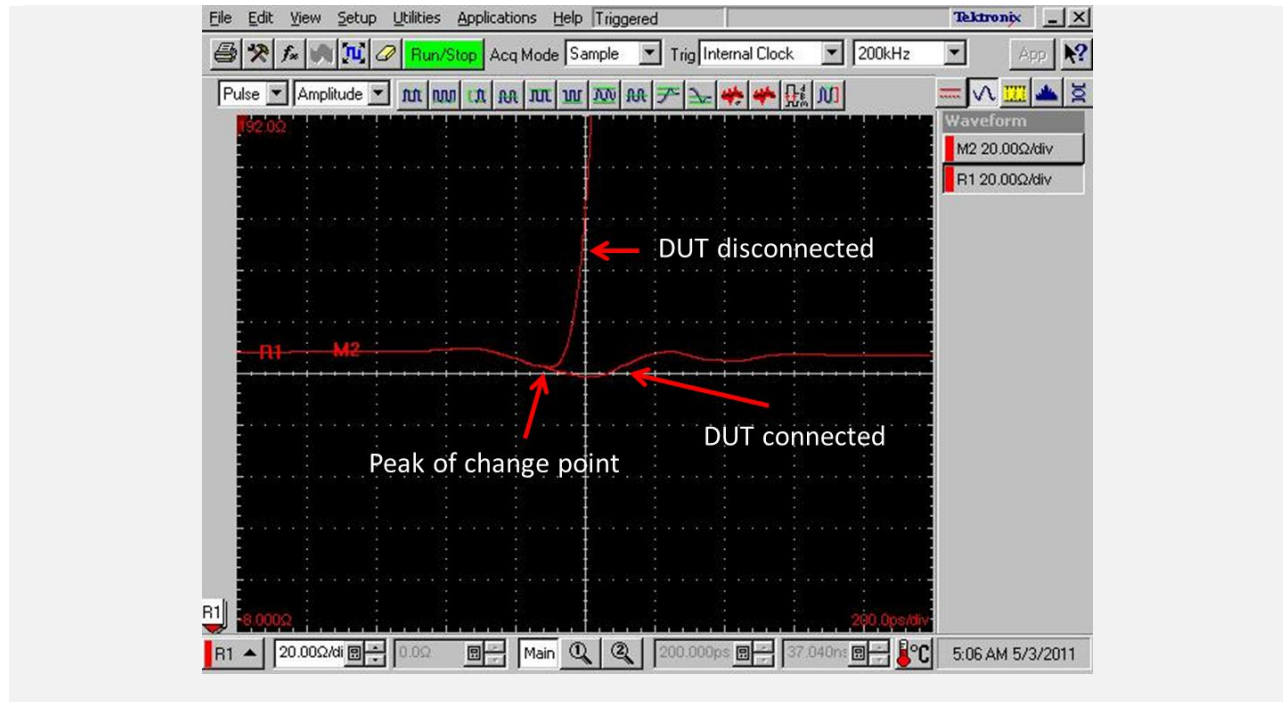


Figure 38. Set reference point for differential mode

Before testing, compensate the differential intra-pair skew caused by the test fixtures (TPA boards and co-ax cables).

1. Connect Sink TPA-P-WO or DirSink TPA-R-WO to the TDR oscilloscope through a DC Block.
2. Set the effective rise time of the differential TDR pulse to 200ps (20 – 80%).
3. Record the time at the peak of the change point in Figure 38. This is the start point of the Sink DUT input connector (T_{IN}).
4. Calculate $T_{TERM} = T_{IN} + CDF_SINK_DF_TERM_DISTANCE$. T_{TERM} is the sink termination point for differential signals. **CDF_SINK_DF_TERM_DISTANCE** is obtained from CDF of the Sink DUT.
5. Connect the equipment as shown in Figure 37.
6. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
7. If DUT is Direct Attached Sink, go to step 9.
8. Measure the differential impedance along the MHL differential signal path from T_{IN} to $T_{TERM} + 1$ ns.
 - a. The differential impedance from T_{IN} to T_{TERM} is $Z_{DTHROUGH}$ and the differential impedance from T_{TERM} to $T_{TERM} + 1$ ns is Z_{DT} .
 - b. If $(85 \text{ ohms} \leq Z_{DTHROUGH} \leq 115 \text{ ohms})$ AND $(90 \text{ ohms} \leq Z_{DT} \leq 110 \text{ ohms})$, then PASS. Else if $(75 \text{ ohms} \leq Z_{DTHROUGH} < 85 \text{ ohms})$ OR $(115 \text{ ohms} < Z_{DTHROUGH} \leq 125 \text{ ohms})$, the impedance is in the excursion range. If the excursion occurs single time AND the duration of the single excursion is less than 250ps AND $(90 \text{ ohms} \leq Z_{DT} \leq 110 \text{ ohms})$, then PASS.
 - c. For all other cases, FAIL.
9. Calculate $T_{CABLESTART}$ and $T_{CABLEEND}$. $T_{CABLESTART}$ and $T_{CABLEEND}$ are the start and end points of the cable portion in Direct Attached Sink for differential signals.
 - a. If $(T_{TERM} \leq T_{IN} + 6\text{ns})$, $T_{CABLESTART} = T_{CABLEEND} = T_{IN} + (T_{TERM} - T_{IN}) / 2$.

- b. If ($T_{\text{TERM}} > T_{\text{IN}} + 6\text{ns}$), $T_{\text{CABLESTART}} = T_{\text{IN}} + \text{CDF_DirSINK_DF_CABLE_START_DISTANCE}$ and $T_{\text{CABLEEND}} = T_{\text{IN}} + \text{CDF_DirSINK_DF_CABLE_END_DISTANCE}$. $\text{CDF_DirSINK_DF_CABLE_START_DISTANCE}$ and $\text{CDF_DirSINK_DF_CABLE_END_DISTANCE}$ are obtained from CDF of the Direct Attached Sink DUT.
10. Record the differential impedance increment $Z_{\text{DINCREMENT_A}}$ from $T_{\text{CABLESTART}}$ to T_{CABLEEND} .
11. Calculate $Z_{\text{DINCREMENT_B}}$.
 - a. If ($Z_{\text{DINCREMENT_A}} < 0$), $Z_{\text{DINCREMENT_B}} = 0$.
 - b. If ($Z_{\text{DINCREMENT_A}} \geq 0$), $Z_{\text{DINCREMENT_B}} = Z_{\text{DINCREMENT_A}}$.
12. Measure the differential impedance $Z_{\text{DTHROUGH1}}$ from T_{IN} to $T_{\text{CABLESTART}}$ and $Z_{\text{DTHROUGH2_WITHCABLE}}$ from T_{CABLEEND} to T_{TERM} .
13. Calculate $Z_{\text{DTHROUGH2}} = Z_{\text{DTHROUGH2_WITHCABLE}} - Z_{\text{DINCREMENT_B}}$.
14. Measure the differential impedance $Z_{\text{DT_WITHCABLE}}$ from T_{TERM} to $T_{\text{TERM}} + 1\text{ns}$.
15. Calculate $Z_{\text{DT}} = Z_{\text{DT_WITHCABLE}} - Z_{\text{DINCREMENT_B}}$.
16. Evaluate $Z_{\text{DTHROUGH1}}$, $Z_{\text{DTHROUGH2}}$, and Z_{DT} .
 - a. If ($85\text{ ohms} \leq [Z_{\text{DTHROUGH1}}, Z_{\text{DTHROUGH2}}] \leq 115\text{ ohms}$) AND ($90\text{ ohms} \leq Z_{\text{DT}} \leq 110\text{ ohms}$), it is PASS
 - b. If ($65\text{ ohms} \leq [Z_{\text{DTHROUGH1}}, Z_{\text{DTHROUGH2}}] < 85\text{ ohms}$) OR ($115\text{ ohms} < [Z_{\text{DTHROUGH1}}, Z_{\text{DTHROUGH2}}] \leq 125\text{ ohms}$), the impedance is in the excursion range. If an excursion occurs no more than one time for $Z_{\text{DTHROUGH1}}$ and no more than one time for $Z_{\text{DTHROUGH2}}$ AND the duration of each excursion is less than 350ps AND ($90\text{ ohms} \leq Z_{\text{DT}} \leq 110\text{ ohms}$), it is PASS.
 - c. All other cases are FAIL.

4.1.1.6 Common-Mode Impedance

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.1.1.6.1 Test Objective

This test confirms that the common-mode input impedance of the sink device is within the range allowed by the specification.

4.1.1.6.2 References

[MHL] Table 13-9 Sink Device Impedance Characteristics at TP2

Through Connection: 30 ohms +/- 6 ohms*

* A single excursion is permitted out to 30 ohms +/- 10 ohms, no wider than 500ps

At termination: 30 ohms +/- 5 ohms

4.1.1.6.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR

Sink TPA-P-WO or DirSink TPA-R-WO

CBUS Source Board

SMA Coaxial Cable

DC Block

4.1.1.6.4 Required Methodology

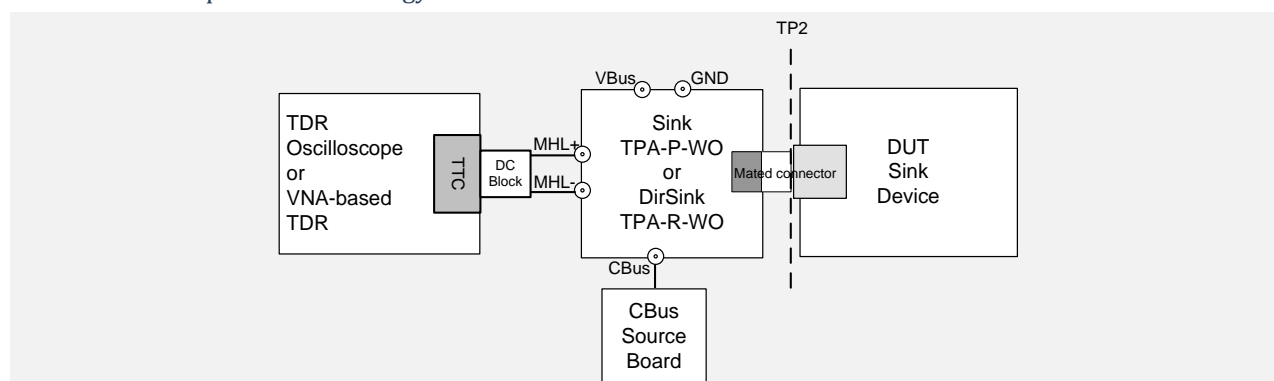


Figure 39. Sink – Common-mode Impedance Test Setup

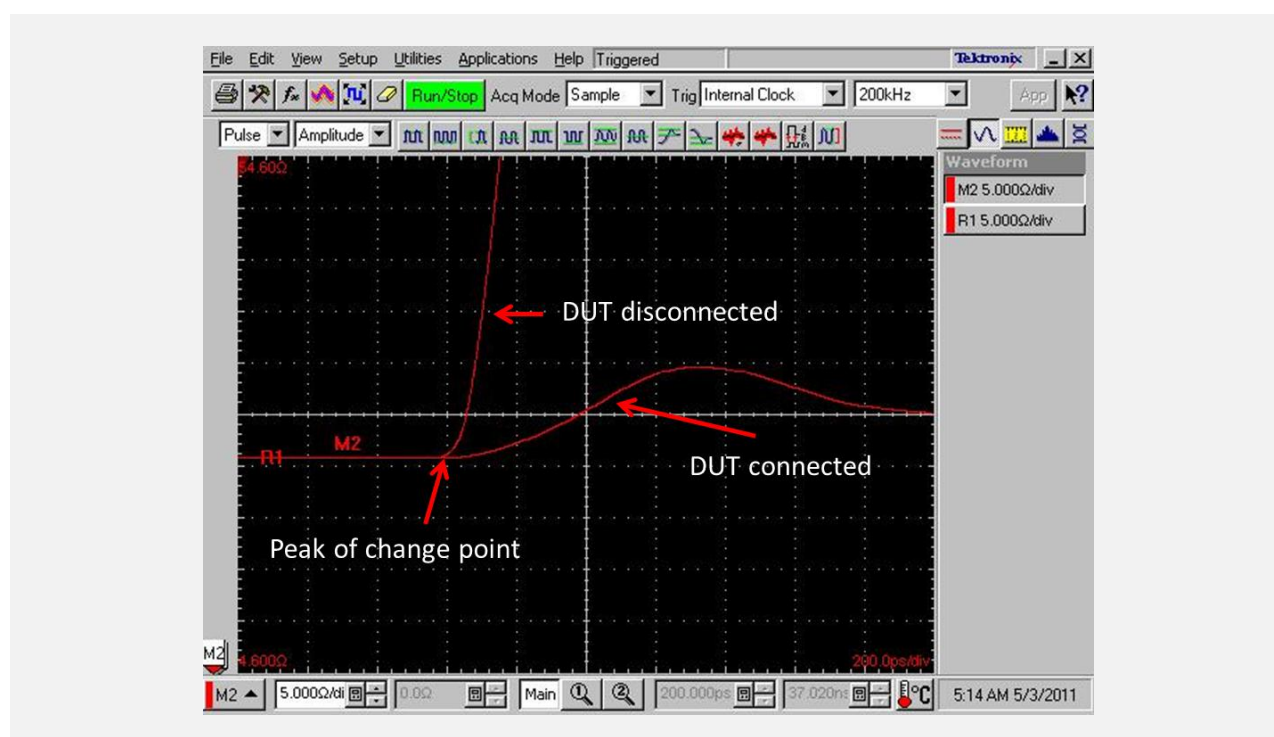


Figure 40. Set reference point for common-mode

Before testing, compensate the common-mode intra-pair skew caused by the test fixtures (TPA boards and co-ax cables).

1. Connect Sink TPA-P-WO or DirSink TPA-R-WO to the TDR oscilloscope through a DC block.
2. Set the effective rise time of the common-mode TDR pulse to 600 ps (20 – 80%).
3. Record the time at the peak of change point in Figure 40. This is the start point of the Sink DUT input connector (T_{IN}).
4. Calculate $T_{TERM} = T_{IN} + CDF_SINK_CM_TERM_DISTANCE$. T_{TERM} is the sink termination point for common-mode signals. $CDF_SINK_CM_TERM_DISTANCE$ is obtained from CDF of the Sink DUT.
5. Connect the equipment as shown in Figure 39.

6. Turn on the DUT and enable the MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
7. If DUT is Direct Attached Sink, go to 9.
8. Measure the common-mode impedance along the MHL common-mode signal path from T_{IN} to $T_{TERM} + 3$ ns.
 - a. The common-mode impedance from T_{IN} to T_{TERM} is $Z_{CTHROUGH}$ and the common-mode impedance from T_{TERM} to $T_{TERM} + 3$ ns is Z_{CT} .
 - b. If $(24 \text{ ohms} \leq Z_{CTHROUGH} \leq 36 \text{ ohms})$ AND $(25 \text{ ohms} \leq Z_{CT} \leq 35 \text{ ohms})$, then PASS.
 - c. Otherwise
 - i. If $(20 \text{ ohms} \leq Z_{CTHROUGH} < 24 \text{ ohms})$ OR $(36 \text{ ohms} < Z_{CTHROUGH} \leq 40 \text{ ohms})$, the impedance is in the excursion range. If the excursion occurs single time AND the duration of the single excursion is less than 500ps AND $(25 \text{ ohms} \leq Z_{CT} \leq 35 \text{ ohms})$, then PASS.
 - ii. For all other cases, FAIL.
9. Calculate $T_{CABLESTART}$ and $T_{CABLEEND}$. $T_{CABLESTART}$ and $T_{CABLEEND}$ are the start and end points of the cable portion in Direct Attached Sink for common-mode signals.
 - a. If $(T_{TERM} \leq T_{IN} + 10\text{ns})$, $T_{CABLESTART} = T_{CABLEEND} = T_{IN} + (T_{TERM} - T_{IN}) / 2$.
 - b. If $(T_{TERM} > T_{IN} + 10\text{ns})$, $T_{CABLESTART} = T_{IN} + \text{CDF_DirSINK_CM_CABLE_START_DISTANCE}$ and $T_{CABLEEND} = T_{IN} + \text{CDF_DirSINK_CM_CABLE_END_DISTANCE}$. CDF_DirSINK_CM_CABLE_START_DISTANCE and CDF_DirSINK_CM_CABLE_END_DISTANCE are obtained from CDF of the Direct Sink DUT.
10. Record the common-mode impedance increment $Z_{CINCREMENT_A}$ from $T_{CABLESTART}$ to $T_{CABLEEND}$.
11. Calculate $Z_{CINCREMENT_B}$.
 - a. If $(Z_{CINCREMENT_A} < 0)$, $Z_{CINCREMENT_B} = 0$.
 - b. If $(Z_{CINCREMENT_A} \geq 0)$, $Z_{CINCREMENT_B} = Z_{CINCREMENT_A}$.
12. Measure the common-mode impedance $Z_{CTHROUGH1}$ from T_{IN} to $T_{CABLESTART}$ and $Z_{CTHROUGH2_WITHCABLE}$ from $T_{CABLEEND}$ to T_{TERM} .
13. Calculate $Z_{CTHROUGH2} = Z_{CTHROUGH2_WITHCABLE} - Z_{CINCREMENT_B}$.
14. Measure the common-mode impedance $Z_{CT_WITHCABLE}$ from T_{TERM} to $T_{TERM} + 3$ ns.
15. Calculate $Z_{CT} = Z_{CT_WITHCABLE} - Z_{CINCREMENT_B}$.
16. Evaluate $Z_{CTHROUGH1}$, $Z_{CTHROUGH2}$, and Z_{CT} .
 - a. If $(24 \text{ ohms} \leq [Z_{CTHROUGH1}, Z_{CTHROUGH2}] \leq 36 \text{ ohms})$ AND $(25 \text{ ohms} \leq Z_{CT} \leq 35 \text{ ohms})$, it is PASS
 - b. If $(20 \text{ ohms} \leq [Z_{CTHROUGH1}, Z_{CTHROUGH2}] < 24 \text{ ohms})$ OR $(36 \text{ ohms} < [Z_{CTHROUGH1}, Z_{CTHROUGH2}] \leq 40 \text{ ohms})$, the impedance is in the excursion range. If an excursion occurs no more than one time for $Z_{CTHROUGH1}$ and no more than one time for $Z_{CTHROUGH2}$ AND the duration of each excursion is less than 500ps AND $(25 \text{ ohms} \leq Z_{CT} \leq 35 \text{ ohms})$, it is PASS.
17. All other cases are FAIL.

4.1.1.7 RxSense Impedance

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.1.1.7.1 Test Objective

This test confirms that the input RxSense impedance of the sink device is within the range allowed by the specification.

NOTE: Voltages, currents and impedances in the Required and Recommended Methodologies of this test subscripted with the suffix "1" apply to the MHL-pin; voltages subscripted with the suffix "2" apply to the MHL+ pin. Refer to Figure 15.

4.1.1.7.2 References

[MHL] Table 13-9 Sink Device Impedance Characteristics at TP2

RxSense Active: $Z_{RXSENSE_TERM} \leq 100K \text{ ohms}$

TMD5 Termination: $Z_{RXSENSE_TMD5} \leq 70 \text{ ohms}$

RxSense Open (leakage current): $I_{RXSENSE_LEAK} \leq 2 \text{ uA}$

4.1.1.7.3 Required Test Equipment

Low-bandwidth Digital Oscilloscope

Passive Probe

Sink/Dongle TPA-RSEN

CBUS Tester (Source Emulation)

DC Power Supply

4.1.1.7.4 Required Methodology

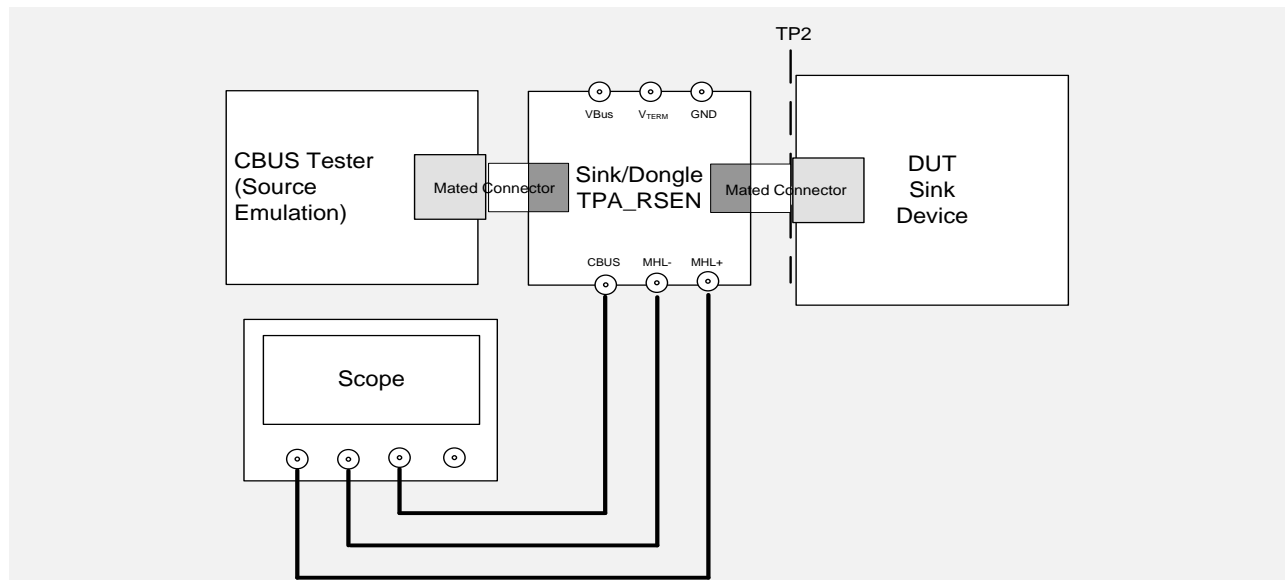


Figure 41. Sink – RxSense Impedance Test Setup

1. Connect the CBUS Tester (Source Emulation) to the Sink DUT as shown in Figure 41.
2. Turn on Sink DUT and CBUS Tester (Source Emulation).
3. Measure $I_{RXSENSE_LEAK}$ current. FAIL if $I_{RXSENSE_LEAK}\{DUT\} > I_{RXSENSE_LEAK}\{max\}$.
 - a. Connect $R_{REF1 \& 2_ILEAK}$ as shown in Figure 15 and measure voltage between $R_{REF1 \& 2_ILEAK}$ and GND.
 - b. This voltage is $V_{RSEN_OPEN1 \& 2}$.
 - c. Calculate $I_{RXSENSE_LEAK1 \& 2} = V_{RSEN_OPEN1 \& 2} / R_{REF1 \& 2_ILEAK}$.
 - d. Fail if $I_{RXSENSE_LEAK1 \& 2}\{DUT\} > I_{RXSENSE_LEAK}\{max\}$.
4. Set CBUS Tester (Source Emulation) to perform CBUS discovery to Sink DUT.
5. Wait for Sink DUT to change CBUS impedance from $Z_{CBUS_SINK_DISCOVER}$ to $Z_{CBUS_SINK_ON} + T_{SINK_CONN} (max) + T_{SINK_RxSense_EN}\{max\}$.
6. Measure Sink DUT TMD5 impedance ($Z_{RXSENSE_TERM}$). FAIL if $Z_{RXSENSE_TERM}\{DUT\} > Z_{RXSENSE_TERM}\{max\}$.

- a. Measure voltage between $R_{REF1 \& 2}$ and GND without resistors. This is $V_{TERM1 \& 2}$.
- b. Connect $R_{REF1 \& 2_TERM}$ as shown in Figure 15 and measure voltage between $R_{REF1 \& 2_TERM}$ and GND.
- c. This voltage is $V_{RXSENSE_TERM1 \& 2}$.
- d. Calculate $Z_{RXSENSE_TERM1 \& 2} = R_{REF1 \& 2_TERM} * (V_{TERM1 \& 2} - V_{RXSENSE_TERM1 \& 2}) / V_{RXSENSE_TERM1 \& 2}$.
- e. Fail if $Z_{RXSENSE_TERM1 \& 2\{DUT\}} > Z_{RXSENSE_TERM\{max\}}$.
7. Configure Sink DUT to select testing port as input (ready to accept video).
8. Wait for Sink DUT to send SET_HPDP and PATH_EN=1{SINK} packets to CBUS Tester (Source Emulation).
9. Measure Sink DUT TMDs impedance ($Z_{RXSENSE_TMDs}$) changes at $T_{SRC_RXSENSE_DEGLITCH\{min\}}$ after PATH_EN{SINK} is detected. FAIL if $Z_{RXSENSE_TMDs\{DUT\}} > Z_{RXSENSE_TMDs\{max\}}$.
 - a. Connect $R_{REF1 \& 2_TMDs}$ as shown in Figure 15 and measure voltage between $R_{REF1 \& 2_TMDs}$ and GND.
 - b. This voltage is $V_{RXSENSE_TMDs1 \& 2}$.
 - c. Calculate $Z_{RXSENSE_TMDs1 \& 2} = R_{REF1 \& 2_TMDs} * (V_{TERM1 \& 2} - V_{RXSENSE_TMDs1 \& 2}) / V_{RXSENSE_TMDs1 \& 2}$.
 - d. Fail if $Z_{RXSENSE_TMDs1 \& 2\{DUT\}} > Z_{RXSENSE_TMDs\{max\}}$.
10. If all preceding steps pass, then PASS; else FAIL.

| State | R_{REF1} | R_{REF2} |
|-------|------------|------------|
| ILEAK | 1M | 1M |
| TERM | 300K | 300K |
| TMDs | 210 | 210 |

4.1.1.8 Jitter Tolerance in PackedPixel Mode

Application: Sink

4.1.1.8.1 Test Objective

This test confirms that the sink device can tolerate the maximum clock and data jitter amounts allowed by the specification in PackedPixel Mode.

4.1.1.8.2 References

[MHL] Figure 13-16 MHL Clock Jitter at TP2

[MHL] Figure 13-17 MHL Data Eye Mask at TP2

[MHL] Figure 13-18 MHL Data Jitter at TP2

4.1.1.8.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

Jitter and Eye Analyzer

Clock Jitter Generator

MHL Cable Emulator

Bias Tee

Transition Time Converter (TTC)

MHL Signal Generator

Sink TPA-R-WT or Source TPA-P-WT

Sink TPA-P-WO or DirSink TPA-R-WO

CBUS Source Board

4.1.1.8.4 Required Methodology

1. Connect the equipment as shown in Figure 31 for single-ended signals calibration and Figure 32 for differential and common-mode signals. Set $V_{TERM} = 3.3V$.
2. Calibrate the Signal Generator output for the following settings.
 - a. Frequency: Highest MHL PackedPixel data bit rate supported by the Sink DUT
 - b. Pattern: MHL Gray Ramp data streams
 - d. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps
Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - c. Swing voltages: Differential signal 800mV, Common-mode signal 360mV
 - d. DC level of the output single-ended signal: $V_{TERM} - 540mV$
3. Set the clock jitter frequency to 10MHz and the data jitter frequency to 500KHz.
4. Calibrate the Signal Generator and Clock Jitter Generator to get data eye diagram and clock jitter which are marginal to TP1 specification.
 - a. Adjust clock jitter and data jitter to the TP1 maximum amount specified in MHL Spec Figures 13-13 and 13-15 for the given clock frequency.
 - b. Adjust the differential signal swing to get the marginal eye diagram specified in MHL Spec Figure 13-14.
5. Connect the equipment as shown in Figure 36.
6. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
7. Select one video mode which uses PackedPixel Mode.
8. Examine the output of the Sink through the display image.
9. Record error on the display if any for the given clock frequency, jitter frequency and clock edge location.
10. Change the common-mode clock edge location by $0.1T_{bit}$ step with respect the differential data edge for 10 times.
11. Examine the display at each step and record error on the display if any for the given clock frequency, jitter frequency and clock edge location.
12. Connect the equipment as shown in Figure 32.
13. Set the clock jitter frequency to 7MHz and the data jitter frequency to 1MHz.
14. Repeat steps 4 to 11 for clock jitter frequency 7MHz and data jitter frequency 1MHz.
15. Disconnect Cable Emulator and connect MHL signal generator directly to the TPA board.
16. Repeat steps 1-14 for jitter tolerance test without Cable Emulator.
17. If there is no error on the display in all recorded results, then PASS. Otherwise FAIL.

4.2 System Tests

4.2.1 TMDs Coding Tests

4.2.1.1 Character Synchronization in Normal Mode

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.1.1.1 Test Objective

Confirm that Sink DUT synchronizes if the Normal Mode data stream provides only minimum length Control Periods. At least one of either 720x480p 59.94/60Hz or 720x576p 50Hz must be supported, optionally both can be supported.

4.2.1.1.2 Reference

[MHL] 4.1.4 Character Synchronization

[MHL] 4.2.1.1.1 Guard Bands

[MHL] 4.4.3 Control Period Coding

[MHL] 4.4.4 TERC4 Coding

[MHL] 4.4.5 Video Data Coding

4.2.1.1.3 Required Test Equipment

MHL Signal Generator

4.2.1.1.4 Required Methodology

1. Configure MHL Signal Generator to transmit a 720x480p 59.94/60Hz Normal Mode video stream with every horizontal and vertical blanking interval completely filled with one or more Data Islands and with Control Periods 12 characters in length. If 60Hz video is not supported per CDF, then skip to step 4.
2. Connect MHL Signal Generator to Sink DUT.
3. If the DUT correctly shows the transmitted signal, then continue to test, else FAIL.
4. Repeat the test from step 1 for 720x576p 50Hz Normal Mode video stream.
5. If all supported video modes listed in the previous steps are verified, then PASS, else FAIL.

Note. Use following Data Islands and Control Periods arrangement for HBLANK:

720x480p 59.94/60Hz : 12+12+2+32+32+32+2+12+2 (138 pixels)

720x576p 50Hz : 12+18+2+32+32+32+2+12+2 (144 pixels)

4.2.1.2 Packet Types in Normal Mode

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.1.2.1 Test Objective

Confirm that Sink DUT accepts all valid packet types in Normal Mode.

4.2.1.2.2 Reference

[MHL] 4.3. Data Island Packet Definitions

[CEA 861] 6 Auxiliary Information Carried from Source to Sink

4.2.1.2.3 Required Test Equipment

MHL Signal Generator

4.2.1.2.4 Required Methodology

1. If 60Hz video modes are not supported per CDF, then skip to step 5.
2. Configure MHL Signal Generator to output a 720x480p 59.94/60Hz 24 bit per pixel mode video stream, while also transmitting 2-channel PCM audio. The video stream shall have the following characteristics; During VBLANK, one or more Data Islands contain a valid:
 - a. Null Packet
 - b. Content Mute Packet with both MUTE and UNMUTE bits unset (0)
 - c. AVI InfoFrame Packet
 - d. Source Product Description InfoFrame Packet
 - e. Audio InfoFrame Packet
 - f. MPEG Source InfoFrame Packet
 - g. a Packet with Packet Type in the range between 0x04 and 0x6F, with arbitrary payload
 - h. a Packet with Packet Type in the range between 0x71 and 0x7F, with arbitrary payload
 - i. a Packet with Packet Type 0x80, with arbitrary payload
3. Connect MHL Signal Generator to Sink DUT.
4. If the Sink DUT correctly shows the signal, then continue to test, else FAIL.
5. Repeat the test from step 2 for 720x576p 50Hz 24 bit per pixel mode.
6. If all video modes listed in the previous steps are verified, then PASS, else FAIL.

4.2.1.3 Character Synchronization in PackedPixel Mode

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.1.3.1 Test Objective

Confirm that Sink DUT synchronizes if the data stream provides only minimum length Control Periods.

4.2.1.3.2 Reference

[MHL] 4.1.4 Character Synchronization

[MHL] 4.2.2.1.1 Guard Bands (PackedPixel Mode)

[MHL] 4.4.3 Control Period Coding

[MHL] 4.4.4 TERC4 Coding

[MHL] 4.4.5 Video Data Coding

4.2.1.3.3 Required Test Equipment

MHL Signal Generator

4.2.1.3.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** in the CDF is "Yes", then continue to test; else end test with PASS (SKIP).
2. Configure MHL Signal Generator to transmit a 1920x1080p 59.94/60Hz or 50Hz PackedPixel Mode video stream with every horizontal and vertical blanking interval completely filled with one or more Data Islands and with Control Periods 12 characters in length.
3. Connect MHL Signal Generator to Sink DUT.
4. If the DUT correctly shows the transmitted signal, then PASS, else FAIL.

Note. Use following Data Islands and Control Periods arrangement for HBLANK:

1920x1080p 60 Hz: 12+2+32+32+2+16+2+32+32+32+32+32+2+18+2 (280 pixels)

1920x1080p 50hz:

12+2+32+32+32+32+32+32+32+32+32+32+32+32+32+32+2+12+2+32+32+32+32+32+2+14+2
(720 pixels)

4.2.1.4 Packet Types in PackedPixel Mode

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.2.1.4.1 Test Objective

Confirm that Sink DUT accepts all valid packet types in PackedPixel Mode.

4.2.1.4.2 Reference

[MHL] 4.3. Data Island Packet Definitions

[CEA 861] 6 Auxiliary Information Carried from Source to Sink

4.2.1.4.3 Required Test Equipment

MHL Signal Generator

4.2.1.4.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** in the CDF is “Yes”, then continue to test; else end test with PASS (SKIP).
2. Configure MHL Signal Generator to output a 1920x1080p 59.94/60Hz or 1920x1080p 50Hz (if supported) PackedPixel Mode video stream, while also transmitting 2-channel PCM audio, with the following characteristics; During VBLANK, one or more Data Islands contain a valid:
 - a. Null Packet
 - b. Content Mute Packet with both MUTE and UNMUTE bits unset (0)
 - c. AVI InfoFrame Packet
 - d. Source Product Description InfoFrame Packet
 - e. Audio InfoFrame Packet
 - f. MPEG Source InfoFrame Packet
 - g. a Packet with Packet Type in the range between 0x04 and 0x6F, with arbitrary payload
 - h. a Packet with Packet Type in the range between 0x71 and 0x7F, with arbitrary payload
 - i. a Packet with Packet Type 0x80, with arbitrary payload
3. Connect MHL Signal Generator to Sink DUT.
4. If the Sink DUT correctly shows the signal, then continue to test, else FAIL.
5. If all video modes listed in the previous steps are verified, then PASS, else FAIL.

4.2.2 Video Test

4.2.2.1 Video Formats in Normal Mode

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.2.2.1.1 Test Objective

Verify that sink DUT supports the video formats with no distortion.

4.2.2.1.2 References

[MHL] Section 5.2.1.2 MHL Sink Video Format Support Requirements

4.2.2.1.3 Required Test Equipment

MHL Signal Generator

4.2.2.1.4 Required Methodology

1. If DUT supports enhanced definition video format on any other input and if **CDF_VIDEO_480p_60** field and **CDF_VIDEO_576p_50** field in the CDF is "NO", then FAIL, else continue to test.
2. If DUT accepts 59.94/60Hz video formats and if **CDF_VIDEO_480p_60** field in the CDF is "NO", then FAIL, else continue to test.
3. If DUT accepts 50Hz video formats and if **CDF_VIDEO_576p_50** field in the CDF is "NO", then FAIL, else continue to test.
4. If DUT supports 640x480(VGA) format on any other input and if **CDF_VIDEO_VGA** field in the CDF in "NO" then FAIL, else continue to test.
5. If DUT is HDTV capable and if **CDF_VIDEO_480p_60** field in the CDF is "YES" and if both of **CDF_VIDEO_720p_60** and **CDF_VIDEO_1080i_60** field in the CDF is "NO", then FAIL, else continue to test.
6. If DUT is HDTV capable and if **CDF_VIDEO_576p_50** field in the CDF is "YES" and if both of **CDF_VIDEO_720p_50** and **CDF_VIDEO_1080i_50** is "NO", then FAIL, else continue to test.
7. Configure MHL Signal Generator to send out test patterns in one of the video formats in the CDF.
8. If DUT displays the patterns with no distortion, then continue to test, else FAIL.
9. Repeat the test from step 7 for all video formats indicated in the CDF.
10. If none of the video format fails, then PASS.

Note: Use Ramp pattern.

4.2.2.2 Pixel Encoding in Normal Mode

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.2.2.2.1 Test Objective

Verify that sink supports RGB, YCCr 4:4:4 and YCbCr 4:2:2 pixel encodings in Normal Mode.

4.2.2.2.2 References

[MHL] Section 5.2.3 24-Bit Pixel Encoding Requirement.

[MHL] Section 5.5 Pixel Encodings.

4.2.2.2.3 Required Test Equipment

MHL Signal Generator

4.2.2.2.4 Required Methodology

1. If **CDF_VIDEO_RGB** field in the CDF is "YES", then continue to test, else FAIL.
2. If **CDF_VIDEO_YCBCR_444** field and **CDF_VIDEO_YCBCR_422** field in the CDF is "NO" and DUT is capable of supporting YCbCr color space on any other analog or digital video input, then FAIL, else continue to test.
3. Configure MHL Signal Generator to send out test patterns in 720x480p 59.94/60Hz or 720x 576p 50Hz using Normal Mode, with one of the pixel encoding indicated in the CDF.

4. If DUT displays the patterns with no distortion, then PASS, else FAIL.
5. Repeat the test from step 3 for all pixel encoding formats indicated in the CDF which use Normal Mode.
6. If any of Pixel Encoding indicated in the CDF for Normal Mode fails, then FAIL, else PASS.

Note: Use Black/White Ramp pattern.

4.2.2.3 Video Quantization Range

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.2.3.1 Test Objective

Verify that Sink has correct QY and QS bits in the Video Capability Data Block and Sink displays with no distortion.

4.2.2.3.2 References

[MHL] Section 5.6 Video Quantization Ranges.

4.2.2.3.3 Required Test Equipment

MHL Signal Generator

4.2.2.3.4 Required Methodology

1. If either **CDF_VIDEO_YCBCR_444** or **CDF_VIDEO_YCBCR_422** field in the CDF is "YES", and if **CDF_VIDEO_YCC_FULL** is "YES" then continue to test, else PASS (SKIP).
2. Read sink device QY bit in the EDID's Video Capability Data Block. If Video Capability Data Block is not available in the DUT's EDID, then PASS (SKIP); else continue this test.
 - a. If QY bit is '0' then FAIL, else continue to test.
 - b. Configure MHL signal generator to send out the test patterns in 720x480p59.94/60Hz or 720x576p50Hz with either YCbCr4:2:2 or YCbCr4:4:4, and set YQ bits to '1' in AVI InfoFrame.
 - c. If the DUT displays the test patterns with no distortion, then PASS, else FAIL.

4.2.2.4 Video Formats in PackedPixel Mode

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.2.4.1 Test Objective

Verify that sink DUT supports the video formats in PackedPixel Mode with no distortion.

4.2.2.4.2 References

[MHL] Section 5.2.1.2 MHL Sink Video Format Support Requirements

4.2.2.4.3 Required Test Equipment

MHL Signal Generator

4.2.2.4.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** in the CDF is "Yes", then continue test; else end test with PASS (SKIP).
2. Configure MHL Signal Generator to send out test patterns in one of the video formats in the CDF which use PackedPixel Mode.

3. If A/V Display with DUT attached displays the test patterns with no distortion, then continue to test, else FAIL.
4. Repeat the test from step 1 for all video formats indicated in the CDF which use PackedPixel Mode.
5. If none of the video formats fails, then PASS; else FAIL.

4.2.2.5 Pixel Encoding in PackedPixel Mode

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.2.5.1 Test Objective

Verify that sink supports YCbCr4:2:2 pixel encoding in PackedPixel Mode.

4.2.2.5.2 References

[MHL] Section 5.2.4 PackedPixel Encoding Requirement.

[MHL] Section 5.5 Pixel Encodings.

4.2.2.5.3 Required Test Equipment

MHL Signal Generator

4.2.2.5.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** in the CDF is "Yes", then continue to test; else end test with PASS (SKIP).
2. Configure MHL Signal Generator to send out test patterns in 1920x1080p 59.94/60Hz or 1920x1080p 50Hz with PackedPixel Mode, as indicated in the CDF.
3. If DUT displays the patterns with no distortion, then PASS, else FAIL.

Note: Use Black/White Ramp pattern.

4.2.3 Audio Test

4.2.3.1 IEC 60958 / IEC61937

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.3.1.1 Test Objective

Verify that Sink supports the required Audio formats and reproduces audio properly.

4.2.3.1.2 References

[MHL] Section 6.3 Audio Sample Rates and Support Requirements.

4.2.3.1.3 Required Test Equipment

MHL Signal Generator

4.2.3.1.4 Required Methodology

1. If **CDF_AUDIO_2CH_32kHz** and **CDF_AUDIO_2CH_44kHz** and **CDF_AUDIO_2CH_48kHz** fields in the CDF are all three set to "NO", and if the DUT has no other audio input, then end test with PASS (SKIP).
2. If **CDF_AUDIO_2CH_32kHz** and **CDF_AUDIO_2CH_44kHz** and **CDF_AUDIO_2CH_48kHz** fields in the CDF are all three set to "NO", and if the DUT has any other audio input, then FAIL.

3. If any of **CDF_AUDIO_2CH_32kHz**, **CDF_AUDIO_2CH_44kHz** and **CDF_AUDIO_2CH_48kHz** field in the CDF is “NO”, then FAIL, else continue to test.
4. Configure the MHL Signal Generator as 720x480p 59.94/60Hz or 720x576p 50Hz and one of PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz.
5. Configure the MHL Signal Generator to output 1 KHz sine wave.
6. If DUT regenerates the 1 kHz sound with no distortion, then continue the test, else FAIL.
7. Repeat the test from step 2 for PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz. If any of the Audio formats fails the test, then FAIL, else PASS.

4.2.3.2 Audio Clock Regeneration

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.3.2.1 Test Objective

Verify that Sink properly regenerates audio when different Audio Clock Regeneration parameters are used.

4.2.3.2.2 References

[MHL] Section 6.2 Audio Sample Clock Capture and Regeneration.

4.2.3.2.3 Required Test Equipment

MHL Signal Generator

4.2.3.2.4 Required Methodology

1. If any of **CDF_AUDIO_2CH_32kHz**, **CDF_AUDIO_2CH_44kHz** and **CDF_AUDIO_2CH_48kHz** field in the CDF is “YES”, then continue to test, else end test with PASS (SKIP).
2. Configure MHL Signal Generator as one of PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz mode.
3. Configure MHL Signal Generator to send out the test patterns with 1kHz sine wave using minimum N parameter.
 - a. If DUT reproduces 1kHz sine wave with no distortion, then continue to test, else FAIL.
4. Configure MHL Signal Generator to send out test patterns with 1kHz sine wave using maximum N parameter.
 - a. If DUT reproduces 1kHz sine wave with no distortion, then continue to test, else FAIL.
5. Repeat the test from step 2 for PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz.
6. If any of the Audio-mode fails the test, then FAIL, else PASS.

4.2.4 HDCP Test

4.2.4.1 HDCP

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.4.1.1 Test Objective

Verify that Sink is complied with HDCP.

4.2.4.1.2 References

[MHL] Section 11.3 HDCP Compliance.

4.2.4.1.3 Required Test Equipment

HDCP Emulator

4.2.4.1.4 Required Methodology

1. If the **CDF_HDCP_SUPPORT** field in CDF is “YES”, then continue to test, else end test with PASS (SKIP).
2. Follow the HDCP CTS procedure. If the **CDF_HDCP_REPEATER** field in the CDF is “YES”, then include all HDCP Repeater tests.
3. If all tests in the HDCP CTS procedure pass, then PASS; else FAIL.

*Note: HDCP test tool must support clock stretch.

4.2.5 EDID Test and Device Capability / Device Status Register Test

4.2.5.1 EDID Test

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.5.1.1 Test Objective

Verify that Sink EDID is accessible and accurate.

4.2.5.1.2 References

[MHL] Section 10.3 EDID Access.

4.2.5.1.3 Required Test Equipment

MHL Signal Generator

4.2.5.1.4 Required Methodology

1. Connect MHL EDID Reader to DUT.
2. Wait until SET_HPD command is received.
3. Read Block0. If Extension Flag(byte 0x7E) is 0, then FAIL, else read all extended blocks.
4. If check_sum is correct, then continue to test, else FAIL.
5. Read Block0 and extended blocks again and compare.
6. If they do not match, then FAIL.
7. Examine EDID:
 - a. For each video mode in the EDID which has an MHL clock frequency less than or equal to the MHL clock frequency limit in the **CDF_CR_BANDWIDTH** field of the CDF:
 - i. If all Video Modes in the CDF are found in the EDID, then continue to test, else FAIL.
 - b. If all Pixel Encodings in the CDF are found in the EDID, then continue to test, else FAIL.
 - c. If all audio modes in the CDF are found in the EDID, then PASS, else FAIL.

4.2.5.2 Device Capability Register Test

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.5.2.1 Test Description

Verify that Device Capability Registers have accurate value.

4.2.5.2.2 Reference

[MHL] Section 7.8.1 Device Capability Registers

4.2.5.2.3 Required Test Equipment

MHL Signal Generator

4.2.5.2.4 Required Methodology

Perform the test in the following steps:

1. Connect DUT to MHL Signal Generator.
2. Configure MHL Signal Generator read DUT's MHL Capability Register.
3. Examine the values:
 - a. If the values in MHL_VERSION(offset:0x01) register does not match the **CDF_CR_MHL_VER_MAJOR** and **CDF_CR_MHL_VER_MINOR** fields **CDF_CR_MHL_VERSION** in the CDF, then FAIL.
 - b. If DEV_TYPE in the DEV_CAT(Offset:0x02) register is not 0b0001:Sink, then FAIL.
 - c. If POW in the DEV_CAT(offset:0x02) register does not match the **CDF_CR_POW** field in the CDF, then FAIL.
 - d. If ADOPTER_ID_H(Offset:0x03) or ADOPTER_ID_L(offset:0x04) registers do not match the corresponding **CDF_CR_ADOPTER_ID_H** and **CDF_CR_ADOPTER_ID_L** field in the CDF, then FAIL.
 - e. If SUPP_RGB444, SUPP_YCBCR444, SUPP_YCBCR422, SUPP_PPIXEL, SUPP_ISLANDS or SUPP_VGA bits in the VID_LINK_MODE(offset:0x05) register do not match the corresponding **CDF_CR_SUPP_RGB444**, **CDF_CR_SUPP_YCBCR444**, **CDF_CR_SUPP_YCBCR422**, **CDF_CR_SUPP_PPIXEL**, **CDF_CR_SUPP_ISLANDS** and **CDF_CR_SUPP_VGA** fields in the CDF, then FAIL.
 - f. If AUD_2CH and AUD_8CH bits in the AUD_LINK_MODE(offset:0x06) do not match the corresponding **CDF_CR_AUD_2CH** and **CDF_CR_AUD_8CH** fields in the CDF, then FAIL.
 - g. If SUPP_VT in VIDEO_TYPE is 0b1 and if VT_GRAPHICS, VT_PHOTO, VT_CINEMA or VT_GAME bits in VIDEO_TYPE register(offset:0x07) do not match the corresponding **CDF_CR_VT_GRAPHICS**, **CDF_CR_VT_PHOTO**, **CDF_CR_VT_CINEMA** and **CDF_CR_VT_GAME** fields in the CDF, then FAIL.
 - h. If LD_DISPLAY, LD_VIDEO, LD_AUDIO, LD_MEDIA, LD_TUNER, LD_RECORD, LD_SPEAKER, or LD_GUI bits in the LOG_DEV_MAP(offset:0x08) register do not match the corresponding **CDF_CR_LD_DISPLAY**, **CDF_CR_LD_VIDEO**, **CDF_CR_LD_AUDIO**, **CDF_CR_LD_MEDIA**, **CDF_CR_LD_TUNER**, **CDF_CR_LD_RECORD**, **CDF_CR_LD_SPEAKER**, **CDF_CR_LD_GUI** fields in the CDF, then FAIL.
 - i. If BANDWIDTH(offset:0x09) register, which is expressed in terms of 5's of megahertz, does not match the **CDF_CR_BANDWIDTH** field in the CDF, then FAIL.
 - j. Check the following support bits in the FEATURE_FLAG register against the CDF:
 - If RCP_SUPPORT does not match **CDF_CR_RCP_SUPPORT**, then FAIL.
 - If RAP_SUPPORT does not match **CDF_CR_RAP_SUPPORT**, then FAIL.
 - If SP_SUPPORT does not match **CDF_CR_SP_SUPPORT**, then FAIL.
 - If UCP_SEND_SUPPORT does not match **CDF_CR_UCP_SEND_SUPPORT**, then FAIL.
 - If UCP_RECV_SUPPORT does not match **CDF_CR_UCP_RECV_SUPPORT**, then FAIL.
 - k. If DEVICE_ID_H(Offset:0x0b) or DEVICE_ID_L(offset:0x0c) do not match the corresponding **CDF_CR_DEVICE_ID_H** and **CDF_CR_DEVICE_ID_L** fields in the CDF, then FAIL.

- I. If SP_SUPPORT bit is 0b1 and if SCRATCHPAD_SIZE(offset:0x0d) register does not match the **CDF_CR_SCRATCHPAD_SIZE** field in the CDF, then FAIL.
 - m. If INT_SIZE or STAT_SIZE fields in INT_STAT_SIZE register(offset:0x0e) do not match the corresponding **CDF_CR_INT_SIZE** and **CDF_CR_STAT_SIZE** fields in the CDF, then FAIL.
4. If all register values match the corresponding CDF field, then PASS.

4.2.5.3 Device Status Registers Test

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.5.3.1 Test Description

Verify that Device Status Registers have proper values.

4.2.5.3.2 Reference

[MHL] Section 7.8.1 Device Capability Registers

4.2.5.3.3 Required Test Equipment

MHL Signal Generator

4.2.5.3.4 Required Methodology

1. Connect DUT to MHL Signal Generator while let the MHL Signal Generator monitoring LINK_MODE status register.
2. Wait until DUT writes PATH_EN =1.
3. When PATH_EN =1, examine TMDS line. If TMDS lines are terminated to Z_{RXSENSE_TMDS}, then FAIL, else continue to test.
4. Write PATH_EN = 1 into DUT.
5. Examine TMDS line. If TMDS lines are terminated to Z_{RXSENSE_TMDS}, then PASS, else FAIL.

4.2.6 RCP Sub-Commands Tests

4.2.6.1 RCP Sub-Commands Receiving Test

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.6.1.1 Test Objective

Verify that Sink DUT responds to RCP sub-commands with the expected behavior based on the definitions in the MHL Specification, for each Logical Device claimed to be supported by the Sink DUT.

4.2.6.1.2 References

[MHL] Section 7.7.8.1 Support when Receiving RCP Command

[MHL] Table 7-17 Key Code Receiving Support Map

[MHL] Table 7-16 Key Code Behaviors

4.2.6.1.3 Required Test Equipment

MHL Signal Generator

4.2.6.1.4 Required Methodology

1. If **CDF_RCP_RECEIVE** field in the CDF is "YES", then continue to test, else FAIL.

2. Examine the Logical Device Type field in the CDF.
 - a. If no Logical Device Type is defined, then FAIL, else continue to test.
3. For each logical device type marked in the CDF:
 - a. Send each RCP command (with both PRESS and RELEASE key flag, per Spec section 7.7.4) marked in for the Logical Device in MHL Specification Table 7-17.
 - b. If Sink DUT does not demonstrate proper behavior (described in MHL Specification Table 7-16), then FAIL.
4. If any additional RCP sub-commands are defined in the CDF, repeat the test in step 3 for the additional RCP sub-commands.
5. If **CDF_LOG_DEV_MAP_CHANGE** in the CDF is "YES", then follow instructions in the CDF to change the DUT's LOG_DEV_MAP setting to each supported value, and repeat steps 3-4.
6. If all RCP sub-commands are verified, then PASS.

4.2.6.2 RCP Sub-Commands Transmitting Test

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.6.2.1 Test Objective

Verify that the Sink DUT outputs each RCP sub-command supported as identified in the CDF, demonstrating the proper opcode and sub-command.

| |
|---|
| Note that the Sink DUT is required to output those RCP commands for which it has means to initiate the command. Information on which commands have such means is to be provided in the completed CDF. |
|---|

4.2.6.2.2 References

[MHL] Section 7.7.8.2 Sink Support Sending RCP Command

[MHL] Table 7-18 Key Code Sending Support Map for Sink

4.2.6.2.3 Required Test Equipment

MHL Signal Generator

4.2.6.2.4 Required Methodology

1. If **CDF_RCP_SEND** field in the CDF is "YES", then continue to test, else FAIL.
2. Configure DUT to send each of RCP sub-commands (with both PRESS and RELEASE key flag, per Spec section 7.7.4) marked in the CDF Table 12, and for which means to initiate the command is identified in the CDF, and verify the opcode.
3. Repeat from step 2 for all RCP sub-commands marked in the CDF Table 12.
4. If all opcodes of the RCP sub-commands match, then continue to test, else FAIL.
5. If **CDF_LOG_DEV_MAP_CHANGE** in the CDF is "YES" then follow instructions in the CDF to change the DUT's LOG_DEV_MAP setting to each supported value, and repeat steps 2-4.
6. If all iterations of the test pass, then PASS; else FAIL.

4.2.7 RAP Sub-Commands Test

4.2.7.1 RAP and RAPK Sub-Commands Test

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.7.1.1 Test Objective

Verify that the Sink DUT supports RAP sub-commands and responds with the proper behavior.

4.2.7.1.2 References

[MHL] Section 7.6 Request Action Protocol Sub-commands.

[MHL] Table 7-11 MSC Request Action Protocol Sub-Command Action Codes.

[MHL] Table 7-12 MSC Request Action Protocol Status Codes.

4.2.7.1.3 Required Test Equipment

MHL Signal Generator

4.2.7.1.4 Required Methodology

1. If **CDF_RAP_SUPPORT** field in the CDF is YES, then continue to test, else FAIL.
2. Send "Poll": OPcode(0x00) to the DUT,
 - a. If the DUT responds with RAPK and Status Code "No Error": 0x00, then continue to test, else FAIL.
 - b. If the DUT responds with RAPK and Status Code "Unrecognized Action Code": 0x01 or "Unsupported Action Code": 0x02, then FAIL, else continue to test.
 - c. If the DUT responds with RAPK and Status Code "Responder Busy": 0x03 then, wait 1 second and repeat from the beginning of step 2.
3.
 - a.
 - b.
 - c.
4. Send "CONTENT_OFF": OPcode(0x11) to the DUT,
 - a. If the DUT responds with RAPK and Status Code "No Error": 0x00 and if the DUT stops rendering test patterns then PASS, else FAIL.
 - b. If the DUT responds with RAPK and Status Code "Unrecognized Action Code": 0x01 or "Unsupported Action Code": 0x02, then FAIL, else continue to test.
 - c. If the DUT responds with RAPK and Status Code "Responder Busy": 0x03 then wait 1 second then repeat from the beginning of step 3.
5. Send "CONTENT_ON": OPcode(0x10) to the DUT,
 - a. If the DUT responds with RAPK and Status Code "No Error": 0x00 and if the DUT renders test patterns, then continue to test, else FAIL.
 - b. If the DUT responds with RAPK and Status Code "Unrecognized Action Code": 0x01 or "Unsupported Action Code": 0x02, then FAIL, else continue to test.
 - c. If the DUT responds with RAPK and Status Code "Responder Busy": 0x03 then wait 1 second then repeat from the beginning of step 4.
6. If DUT passes all step 2-4, then PASS, else FAIL.

4.2.8 3D Video Test

4.2.8.1 3D Video Mode Support Data

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.8.1.1 Test Objective

Verify that the Sink DUT responds to 3D_REQ and sends the list of 3D video modes supported by the DUT.

4.2.8.1.2 References

[MHL] Section 5.9.1 3D Video Support Detection

[MHL] Table 5-3 3D Video Format Data Structure in WRITE_BURST

4.2.8.1.3 Required Test Equipment

MHL Signal Generator

4.2.8.1.4 Required Methodology

1. If **CDF_VIDEO_3D** field in the CDF is “Yes”, then continue to test, else PASS (SKIP).
2. After discovery, wait until **DCAP_RDY** in the DUT becomes 0b1, then write 3D_REQ in Interrupt Register4 to 0b1.
3. If DUT transmits the list of 3D video modes that DUT supports by a sequence of WRITE_BURST, then continue to test, else FAIL.
4. If a byte-wide sum of all data bytes of each WRITE_BURST, including the checksum itself, equals zero, then continue to test, else FAIL.
5. Examine the list of 3D video modes. The list may contain additional modes not specified in the CDF, but all modes listed in the CDF must be contained in the list. If this is true, then PASS; else FAIL.

4.2.8.2 3D Video Format in Normal Mode

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.8.2.1 Test Objective

Verify that the Sink DUT supports required and optional 3D Video Modes which use Normal Mode.

4.2.8.2.2 References

[MHL] Section 5.9.4 3D Video Support Requirements

[MHL] Section 5.9.4.2 MHL Sink 3D Video Mode Support Requirements

4.2.8.2.3 Required Test Equipment

MHL Signal Generator

4.2.8.2.4 Required Methodology

1. If **CDF_VIDEO_3D** field in the CDF is “NO” and If DUT supports 3D video on any other analog or digital input port, then FAIL, else continue to test.
2. If **CDF_VIDEO_3D** field in the CDF is “YES”, then continue to test, else PASS (SKIP).
3. If DUT accepts 59.94/60Hz video formats from any other analog or digital video input, and if any of **CDF_VIDEO_1280x720P_60_3D_Top_Bottom** and **CDF_VIDEO_1920x1080i_60_3D_Left_Right** fields in the CDF is “No”, then FAIL, else continue to test.

4. If DUT accepts 50Hz video formats from any other analog or digital video input, and if any of **CDF_VIDEO_1280x720P_50_3D_Top_Bottom** and **CDF_VIDEO_1920x1080i_50_3D_Left_Right** fields in the CDF is “No”, then FAIL, else continue to test.
5. If **CDF_VIDEO_1920x1080p_24_Top_Bottom** field in the CDF is “No”, then FAIL, else continue to test.
6. Configure MHL Signal Generator to send out test patterns in one of the 3D video formats indicated in the CDF which use Normal Mode.
7. If DUT displays the patterns with no distortion, then continue to test, else FAIL.
8. Repeat test steps 6-7 for all 3D video formats indicated in the CDF which use Normal Mode.
9. If none of the video formats fails, then PASS, else FAIL.

4.2.8.3 3D Video Format in PackedPixel Mode

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.8.3.1 Test Objective

Verify that the Sink DUT supports required and optional 3D Video Modes which use PackedPixel Mode.

4.2.8.3.2 References

[MHL] Section 5.9.4 3D Video Support Requirements

[MHL] Section 5.9.4.2 MHL Sink 3D Video Mode Support Requirements

4.2.8.3.3 Required Test Equipment

MHL Signal Generator

4.2.8.3.4 Required Methodology

1. If **CDF_VIDEO_3D** field in the CDF is “YES”, then continue to test, else end test with PASS (SKIP).
2. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is “Yes”, and if DUT accepts 59.94/60Hz video formats from any other analog or digital video input, and if **CDF_VIDEO_1280x720P_60_3D_Frame** field in the CDF is “No”, then FAIL, else continue to test.
3. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is “Yes”, and if DUT accepts 50Hz video formats from any other analog or digital video input, and if **CDF_VIDEO_1280x720P_50_3D_Frame** field in the CDF is “No”, then FAIL, else continue to test.
4. If **CDF_VIDEO_PACKEDPIXEL** field in the CDF is “Yes”, and if **CDF_VIDEO_1920x1080p_24_3D_Frame** field in the CDF is “No”, then FAIL, else continue to test.
5. Configure MHL Signal Generator to send out test patterns in one of the 3D video formats indicated in the CDF which use PackedPixel Mode.
6. If DUT displays the patterns with no distortion, then continue to test, else FAIL.
7. Repeat test steps 5-6 for all 3D video formats indicated in the CDF which use PackedPixel Mode.
8. If none of the video formats fails, then PASS.

4.2.9 UCP Sub-Command Tests

The test implementation shall include UTF character sequences for all lengths of characters (1 to 4 8-bit bytes or octets), with both invalid and valid values in each of those lengths.

4.2.9.1 UCP Sub-Commands Receiving Test

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.9.1.1 Test Objective

Verify that DUT responds to valid UCP sub-commands by displaying the character or characters sent in the UCP command, or response to invalid UCP sub-commands by displaying an error message.

4.2.9.1.2 References

[MHL] Section 7.8.4 MSC UTF-8 Character Protocol Support Requirements

4.2.9.1.3 Required Test Equipment

MHL Signal Generator

The test implementation shall include UTF character sequences for all lengths of characters, with both invalid and valid values in those lengths.

4.2.9.1.4 Required Methodology

1. If CDF indicates **CDF_UCP_RECV_SUPPORT**=1 then continue test, else PASS (SKIP).
2. Connect the Test Equipment to the DUT.
3. Perform discover sequence between the Test Equipment and the DUT.
 - a. If DUT fails to discover properly, then FAIL.
4. For each UCP command described in the CDF, perform the following steps:
 - a. Enter into the test equipment the expected character or character sequence to send to the DUT, according to the information in the CDF entry.
 - b. Configure the DUT to receive the selected character or character sequence. Use the Application or other software on the DUT as described in the CDF with field **CDF_UCP_RECV_APPLICATION**.
 - c. Trigger the sending of the character or character sequence by Test Equipment action to the DUT.
 - d. Test Equipment displays all bits and interpretation of the UCP payload(s) it sends.
 - e. DUT displays all bits and interpretation of the UCP payload(s) it receives.
 - f. Test Engineer compares the DUT displayed data with the Test Equipment data.
 - g. If the payload received by the DUT does not match that sent by the Test Equipment, then FAIL.
5. If DUT passes all iterations of the test, then PASS. Else FAIL.

4.2.9.2 UCP Sub-Commands Transmitting Test

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.2.9.2.1 Test Objective

Verify that DUT sends valid UCP sub-commands by initiating the sending of UTF-8 characters in various formats through the user interface on the DUT.

4.2.9.2.2 References

[MHL] Section 7.8.4 MSC UTF-8 Character Protocol Support Requirements

4.2.9.2.3 Required Test Equipment

TBD

4.2.9.2.4 Required Methodology

1. If CDF indicates **CDF_UCP_SEND_SUPPORT**=1 then continue test, else PASS (SKIP).
2. Connect the Test Equipment to the DUT.
3. Perform discover sequence between the Test Equipment and the DUT.
 - a. If DUT fails to discover properly, then FAIL.
4. For each UCP command described in the CDF, perform the following steps:
 - a. Enter into the test equipment the expected character or character sequence from the DUT, according to the information in the CDF entry.
 - b. Configure the DUT to send the selected character or character sequence. Use the Application or other software on the DUT as described in the CDF with field **CDF_UCP_SEND_APPLICATION**.
 - c. Trigger the sending of the character or character sequence by user action on the DUT.
 - d. Test Equipment displays all bits and interpretation of the UCP payload(s) it receives.
 - e. If the Test Equipment displays “Mismatch in received UCP information”, then FAIL.
 - f.
5. If DUT passes all iterations of the test, then PASS. Else FAIL.

4.3 CBUS Tests

Test the CBUS timing and functionality of the Sink DUT. Many of the tests in this section use common test equipment setups and common methodologies (listed in 4.3.1). These common elements are delineated before the individual tests.

Note 1: Preparation: Sinks may be in an Unpowered or Standby state prior to testing. Except where indicated, testing assumes that a suitable method, as specified in the CDF, is used to put the Sink into an Active state before testing begins. This further assumes that all Sinks, including DC-powered Sinks which are capable of driving VBUS should have their external power connections attached to meet the testing requirement.

Note 2: Typical resistances, capacitances, and timings: Where called out, use $Z_{CBUS_SOURCE_DISCOVER\{typ\}}$, $Z_{CBUS_SOURCE_ON\{typ\}}$, $C_{CBUS}=180pF$ ($C_{SOURCE_CBUS}+150pF$ Cable) and $T_{BIT_CBUS\{typ\}}$.

Note 3: Continuous monitoring: Continuous monitoring assumes that a parameter measurement is checked continuously for the duration of the testing. Continuous monitoring is usually applied to detect out of range voltage levels and timings.

Note 4: Direct Attach Sink DUT testing: If possible, Direct Attach Sink DUTs should be connected directly to the MHL Tester's Source output port. In cases where the mechanical design of the DUT or Tester does not allow a direct connection, the use of a “Direct Attach Sink Extension Cable” is allowed.

4.3.1 CBUS Sink DUT Common Test Equipment Setups

Each of the tests in Section 4.3 uses one of the common test setups in this section or defines a unique test setup. Refer back to this section for the list of equipment for a particular test using the hyperlink if one is shown in the individual test definition.

4.3.2 CBUS Sink DUT Common Required Methodologies

Many of the tests in Section 3.3 use a common methodology, followed by test steps specific to the individual test definition. Refer back to this section for the initial test steps using the hyperlink if one is shown in the individual test definition.

4.3.2.1 Tester_Source_Discovery Procedure

- 1) Start Continuous Monitor procedures.
- 2) Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
- 3) Set Tester Source port to not drive the VBUS.
- 4) Set Tester Source to drive high impedance pull-downs on MHL+ and MHL- for observability.
- 5) Set Tester Source to set the CBUS to high impedance for $T_{SRC_CBUS_FLOAT}$.
- 6) Set Tester Source to apply $Z_{CBUS_SRC_DISCOVER\{typ\}}$ pull-up resistor to CBUS.
- 7) Tester Source will pause until it thinks it sees a $Z_{CBUS_SINK_DISCOVER}$ resistor.
- 8) If Tester Source does not detect Sink DUT driving VBUS, then Tester Source will drive VBUS itself, then wait $T_{SRC_VBUS_OUT_TO_STABLE}$.
- 9) Set Tester Source to start normal Device Discovery by sending wake pulses.
- 10) Tester Source then drives Device Discovery pulses.
- 11) After discovery completes, Tester Source will change its CBUS pull-up to $Z_{CBUS_SRC_ON\{typ\}}$.

4.3.2.2 Tester_Source_R_Discovery Procedure

- 1) Start Continuous Monitor procedures.
- 2) Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
- 3) Set Tester Source port to not drive the VBUS.
- 4) Set Tester Source to drive high impedance pull-downs on MHL+ and MHL- for observability.
- 5) Set Tester Source to set the CBUS to high impedance for $T_{SRC_CBUS_FLOAT}$.
- 6) Set Tester Source to apply $Z_{CBUS_SRC_DISCOVER\{typ\}}$ pull-up resistor to CBUS.
- 7) Tester Source will pause until it thinks it sees a $Z_{CBUS_SINK_DISCOVER}$ resistor.
- 8) If Tester Source does not detect Sink DUT driving VBUS, then Tester Source will drive VBUS itself, then wait $T_{SRC_VBUS_OUT_TO_STABLE}$.
- 9) Set Tester Source to start normal Device Discovery by sending wake pulses.
- 10) Tester Source then drives Device Discovery pulses.
- 11) After discovery completes, Tester Source will change its CBUS pull-up to TEST_RESISTOR.

4.3.2.3 Tester_Source_Modified_Discovery Procedure

- 1) Start Continuous Monitor procedures.
- 2) Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
- 3) Set Tester Source port to not drive the VBUS.
- 4) Set Tester Source to drive high impedance pull-downs on MHL+ and MHL- for observability.
- 5) Set Tester Source Wake Pulse HIGH time to TEST_WAKE_PULSE_HIGH.
- 6) Set Tester Source Wake Pulse LOW time to TEST_WAKE_PULSE_LOW.
- 7) Set Tester Source Wake Pulse gap time to TEST_WAKE_PULSE_GAP.
- 8) Set Tester Source Wake Pulse discover time to TEST_WAKE_PULSE_DISCOVER.
- 9) Set Tester Source discovery pulse HIGH time to TEST_DISCOVERY_PULSE_HIGH.
- 10) Set Tester Source discovery pulse LOW time to TEST_DISCOVERY_PULSE_LOW.
- 11) Set first discovery pulse to have HIGH width of TEST_FIRST_DISCOVERY_PULSE_HIGH.
- 12) Set first discovery pulse to have LOW width of TEST_FIRST_DISCOVERY_PULSE_LOW.
- 13) Set fifth discovery pulse to have HIGH width of TEST_FIFTH_DISCOVERY_PULSE_HIGH.
- 14) Set fifth discovery pulse to have LOW width of TEST_FIFTH_DISCOVERY_PULSE_LOW.
- 15) If called out, set Tester Source Discovery Engine to send TEST_EXTRA_PULSES discovery pulses.
The default sends 6 discovery pulses.
- 16) Set Tester Source to set CBUS to high impedance for $T_{SRC_CBUS_FLOAT}$.

- 17) Set Tester Source to apply $Z_{CBUS_SRC_DISCOVER\{typ\}}$ pull-up resistor to CBUS.
- 18) Tester Source will pause until it thinks it sees a $Z_{CBUS_SINK_DISCOVER}$ resistor.
- 19) If Tester Source does not detect Sink DUT driving VBUS, then Tester Source will drive VBUS itself, then wait $T_{SRC_VBUS_OUT_TO_STABLE}$.
- 20) Set Tester Source to start normal device discovery by sending wake pulses.
- 21) Tester Source then drives Device Discovery pulses.
- 22) After discovery completes, Tester Source will change its CBUS pull-up to $Z_{CBUS_SRC_ON\{typ\}}$.

4.3.2.4 Tester Source Device Capability Registers

Unless changed in specific tests, the Tester will exhibit the following Device Capability Register contents:

Table 4-1. Tester Source Device Capability Registers

| | Register Name | Register Value | Description |
|----|-----------------|-------------------------------|--|
| 0 | DEV_STATE | 0x00 | Reserved |
| 1 | MHL_VERSION | 0x20 | Level of MHL Spec supported |
| 2 | DEV_CAT | 0x02 | Identify the type of MHL system (Source) |
| 3 | ADOPTER_ID_H | Set by Equipment Manufacturer | High-order byte of Adopter ID |
| 4 | ADOPTER_ID_L | Set by Equipment Manufacturer | Low-order byte of Adopter ID |
| 5 | VID_LINK_MODE | 0x37 | List of link modes supported for Video |
| 6 | AUD_LINK_MODE | 0x03 | List of link modes supported for Audio |
| 7 | VIDEO_TYPE | 0x80 | Video Type Support |
| 8 | LOG_DEV_MAP | 0x06 | Logical Device Map |
| 9 | BANDWIDTH | 0x0F | Link Bandwidth limit |
| 10 | FEATURE_FLAG | 0x07 | Flags for MHL Optional feature |
| 11 | DEVICE_ID_H | Set by Equipment Manufacturer | High-order byte of system identifier |
| 12 | DEVICE_ID_L | Set by Equipment Manufacturer | Low-order byte of system identifier |
| 13 | SCRATCHPAD_SIZE | 0x10 | Count of Scratchpad registers |
| 14 | INT_STAT_SIZE | 0x33 | Count of interrupt and status registers |
| 15 | No Name | 0x00 | Reserved |

4.3.3 Link Layer Electrical – Sink: Absolute Maximum Voltages

Monitor the voltages on VBUS and CBUS wires continuously to make sure they never exceed their absolute maximum values.

4.3.3.1 Common Test Environment

4.3.3.1.1 Test Objective

4.3.3.1.2 References

4.3.3.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.3.1.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. Set Tester Source VBUS Limit comparator to appropriate voltage.
2. Set Tester Source CBUS Limit comparator to appropriate voltage.
3. Set Tester Source to report VBUS Limit errors. If VBUS limits are violated, then FAIL.
4. Set Tester Source to report CBUS Limit errors. If CBUS limits are violated, then FAIL.
5. If no limits are violated, then PASS, else FAIL.

4.3.3.2 CBE-Sink: VBUS Absolute Maximum Positive Voltage

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.3.2.1 Test Objective

Continuous monitor the VBUS wire to make sure it never exceeds the absolute maximum positive voltage.

4.3.3.2.2 References

[MHL] Section 13.9.1.1; Table 13-21: $V_{VBUS_TP2_SINK_DRV}$; $V_{VBUS_TP1_SRC_DRV}$

4.3.3.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.3.2.4 Required Methodology

Continuous monitor runs in the background.

FAIL if VBUS positive limit is exceeded.

PASS if VBUS limit is not exceeded.

4.3.3.3 CBE-Sink: CBUS Absolute Maximum Positive Voltage

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.3.3.1 Test Objective

Continuous monitor the CBUS wire to make sure it never exceeds the absolute maximum positive voltage.

4.3.3.3.2 References

[MHL] Section 13.10.1; Table 13-23: CBUS Absolute Max Voltage

4.3.3.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.3.3.4 Required Methodology

Continuous monitor runs in the background.

FAIL if CBUS positive limit is exceeded.

PASS if CBUS limit is not exceeded.

4.3.4 Link Layer Electrical – Sink DUT Output: Standby Discovery Impedance

Verify that a Sink DUT in standby mode in SINK4 state exhibits a valid $Z_{CBUS_SINK_DISCOVER}$.

4.3.4.1 CBE-Sink: Powered-Off $Z_{CBUS_SINK_DISCOVER}$

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.4.1.1 Test Objective

Verify that Sink DUT $Z_{CBUS_SINK_DISCOVER}$ is within proper limits.

4.3.4.1.2 References

[MHL] Section 8.2.2; Table 8-5; SINK4

[MHL] Section 13.10.1; Table 13-26; $Z_{CBUS_SINK_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-30: $T_{SRC_CBUS_FLOAT}$

4.3.4.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.4.1.4 Required Methodology

This procedure is executed only on a Sink DUT which supports Standby mode.

Execute the following test procedure once for each row in Table 4-2, substituting TEST_CBUS_VOLTAGE:

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. If procedure **CDF_PROC_SET_STANDBY** is empty in the CDF, then PASS (SKIP).
4. If procedure **CDF_PROC_SET_STANDBY** does not put the DUT into state SINK4, then PASS (SKIP).
5. Set Sink DUT into Standby State based on the CDF procedure **CDF_PROC_SET_STANDBY**.
6. Set Tester Source port to disconnected state.
7. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
8. Tester does not drive VBUS.
9. Tester Source applies TEST_CBUS_VOLTAGE to CBUS through a suitable test impedance $Z_{CBUS_SRC_TEST}$.
10. Tester Source measures the Sink DUT discovery resistance.
11. FAIL if DUT Pull-down resistor is not in valid range ($800\text{ Ohm} \leq Z_{CBUS_SINK_DISCOVER} \leq 1200\text{ Ohm}$).
12. If all preceding steps pass, then PASS; else FAIL.

Table 4-2. Standby $Z_{CBUS_SINK_DISCOVER}$

| | TEST_CBUS_VOLTAGE | Purpose |
|---|-------------------|---|
| 1 | 1.9V | $Z_{CBUS_SINK_DISCOVER}$ @ VBUS == 0V, $800 \leq \text{CBUS measured ohms} \leq 1200$ |

4.3.5 Link Layer Timing – Sink DUT Output: Pre-Discovery

Measure the timing of the Sink DUT as it optionally signals the Source that a discovery sequence is invited.

4.3.5.1 CBT-Sink: Time from Sink-side MHL Cable Detect until Sink CBUS Leaves HIGH-Z

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.5.1.1 Test Objective

Verify that Sink DUT notices that an MHL Cable has been inserted, and exhibits a valid $Z_{CBUS_SINK_DISCOVER}$.

Behavior is required for a Sink, but it only leaves HIGHZ mode when it can meet other timing specs.

4.3.5.1.2 References

[MHL] Section 8.2.2 (SINK1 -> SINK5) Table 8-6 note 1

[MHL] Section 13.10.1.1; Table 13-30; $T_{\text{SINK_CBUS_FLOAT}}$

[MHL] Section 13.10.1; Table 13-29; $T_{\text{SINK_READY_TO_DISCOVER}}$

4.3.5.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.5.1.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Tester pulls the CBUS HIGH with a large resistance (100K) for observability.
6. Set Sink DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
7. Go from no MHL Cable inserted to MHL Cable inserted at input to Sink DUT.
8. Wait until Sink DUT assert VBUS.
9. PASS and end the test if DUT never goes through a HIGH-Z state, indicating that it can meet all timing requirements.
10. FAIL if DUT does enter HIGH-Z state but does not remain there for at least $T_{\text{SINK_CBUS_FLOAT}}$.
11. FAIL if Sink DUT does not assert a valid $Z_{\text{CBUS_SINK_DISCOVER}}$ within the time specified in the CDF value **CDF_SINK_CABLE_DETECT_TO_R_DISCOVER**.
12. FAIL if Sink DUT does not drive VBUS earlier than $T_{\text{SINK_VBUS_EN}}$ after asserting $Z_{\text{CBUS_SINK_DISCOVER}}$ the final time.
13. Note the presentation of $Z_{\text{CBUS_SINK_DISCOVER}}$ can be delayed arbitrarily by the Sink not being ready.
14. If all preceding steps pass, then PASS; else FAIL.

4.3.6 Link Layer Electrical – Sink DUT Output: Arbitration/Sync/Data Signaling

Measure the electrical behavior of the Sink DUT as it drives Arbitration, Sync, and Data pulses.

4.3.6.1 CBE-Sink: Post-Discovery Passive Pulldown $Z_{\text{CBUS_SINK_ON}}$ Resistance

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.6.1.1 Test Objective

Verify that Sink DUT $Z_{\text{CBUS_SINK_ON}}$ has correct value.

4.3.6.1.2 References

[MHL] Section 13.10.1; Table 13-26; $Z_{\text{CBUS_SINK_ON}}$

4.3.6.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.6.1.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.

5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Wait $T_{\text{SINK_CONN}}\{\text{max}\}$ from the rising edge of the last discovery pulse.
8. Measure impedance on sink side CBUS as $Z_{\text{CBUS_MEAS}}$.
9. FAIL if $Z_{\text{CBUS_MEAS}}$ is not within the valid $Z_{\text{CBUS_SINK_ON}}$ range.
10. If all preceding steps pass, then PASS; else FAIL.

4.3.6.2 CBE-Sink: CBUS Capacitance

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.6.2.1 Test Objective

Verify that the Sink DUT has a low-enough input capacitance

4.3.6.2.2 References

[MHL] Section 13.10.1; Table 13-26: $C_{\text{SINK_CBUS}}$

4.3.6.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.6.2.4 Required Methodology

Characterize the capacitance of the Test Equipment, so that it can be subtracted from any measurement of DUT + Test Equipment.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Measure the CBUS Capacitance ($C_{\text{CBUS_MEAS}}$) after DISCOVERY completes.
8. FAIL if the calculated Sink capacitance is greater than $C_{\text{SINK_CBUS}}\{\text{max}\}$.
9. If all preceding steps pass, then PASS; else FAIL.

NOTE: The capacitance measurement may depend on compliant termination of the CBUS line in the DUT. The test may result in a FAIL if the resistance $Z_{\text{CBUS_SINK_ON}}$ is out of range.

4.3.6.3 CBE-Sink: Arbitrate/Sync/Data Drive LOW Voltage

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.6.3.1 Test Objective

Verify that Sink DUT drives Arbitration, Sync, and Data Pulses with the correct DRIVE LOW voltage.

4.3.6.3.2 References

[MHL] Section 13.10.1; Table 13-23: $V_{\text{OL_CBUS}}$

4.3.6.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.6.3.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source $Z_{CBUS_SRC_ON}\{min\}$ (4K).
8. Observe the LOW levels in the waveform while the Sink DUT arbitrates for the bus.
9. Set Tester Source to send a GET_STATE to Sink DUT.
10. FAIL if DUT does not eventually arbitrate for the bus to send a data packet to the Tester Source.
11. FAIL if DUT does not drive CBUS LOW during the Arbitration, Sync, and Data pulses to a voltage below $V_{OL_CBUS}\{max\}$.
12. If all preceding steps pass, then PASS; else FAIL.

4.3.6.4 CBE-Sink: Arbitrate/Sync/Data Drive HIGH Voltage

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.6.4.1 Test Objective

Verify that Sink DUT drives Arbitration, Sync, and Data Pulses with correct DRIVE HIGH voltage.

4.3.6.4.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OH_CBUS}

4.3.6.4.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.6.4.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source $Z_{CBUS_SRC_ON}\{max\}$ to maximum value (6K).
8. Observe the HIGH levels in the waveform while the Sink DUT arbitrates for the bus.
9. Set Tester Source to send a GET_STATE to Sink DUT.
10. FAIL if DUT does not eventually arbitrate for the bus to send a data packet to the Tester Source.
11. FAIL if DUT does not drive CBUS HIGH after the Arbitration, Sync and Data pulses to a voltage above $V_{OH_CBUS}\{min\}$ and below $V_{OH_CBUS}\{max\}$.
12. If all preceding steps pass, then PASS; else FAIL.

Note: $V_{OH_CBUS}\{max\}$ should be measured at the end of the CBUS drive high time $T_{DRV_HI_CBUS}\{max\}$ to avoid failing devices due to impedance/driver signal overshoot.

4.3.7 Link Layer Timing – Sink DUT Output: Arbitration/Sync/Data in Nanoseconds

Measure the timing of the Sink DUT as it drives Arbitration and data pulses.

The Sink DUT output Bit Time must be measured.

4.3.7.1 CBT-Sink: Arbitration/Sync/Data Active Drive HIGH Duration

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.7.1.1 Test Objective

Verify that Sink DUT drives CBUS HIGH, then releases bus, when changing the CBUS from LOW to HIGH. (The HIGH voltage will be sustained by a Resistor pull-up at the Source end.)

4.3.7.1.2 References

[MHL] Section 7.2.4

[MHL] Section 13.10.1; Table 13-23: $T_{DRV_HI_CBUS}$

4.3.7.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.7.1.4 Required Methodology

Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Adjust the Tester resistances and voltages so that the DUT Drive High time is observable.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a command from the DUT, reply with a link level ACK.
9. Measure the timing of the packets from the Sink DUT in nanoseconds.
10. FAIL if DUT has problems doing Discovery.
11. FAIL if DUT does not arbitrate for access to CBUS to send a command.
12. FAIL if the CBUS Drive HIGH time (the period starting when the CBUS goes from LOW to HIGH and lasting until only the Source resistor pull-up drives the CBUS) is outside the limits set for $T_{DRV_HI_CBUS}$.
13. If all preceding steps pass, then PASS; else FAIL.

Note: This test covers driving HIGH at the end of the Arbitration bit time, the Sync bit time, and at rising edges throughout the data period including the rising edge transition at the mid-point of a bit time when a DATA 1 is being sent on the wire.

Note this test covers driving HIGH at the end of the Arbitration bit time, the Sync bit time, and at rising edges throughout the data period including the rising edge transition at the mid-point of a bit time when a DATA 1 is being sent on the wire.

This test does not check the Drive HIGH time for ACK pulses, since this test does not result in Source-driven ACKs.

4.3.7.2 CBT-Sink: Arbitration/Sync/Data Edge Rate

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.7.2.1 Test Objective

Verify that Sink DUT drives Arbitration/Sync/Data and ACK Pulses which have valid Rise or Fall times..

4.3.7.2.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OH_CBUS}

[MHL] Section 13.10.1; Table 13-23: V_{OL_CBUS}

[MHL] Section 13.10.1; Table 13-23: T_{F_CBUS}

[MHL] Section 13.10.1; Table 13-23: T_{R_CBUS}

4.3.7.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.7.2.4 Required Methodology

Execute the following test procedure once for each row in Table 4-3, substituting TEST_CAPACITOR and TEST_RESISTOR. Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Connect Oscilloscope to Tester to allow it to monitor the Sink DUT CBUS wire.
4. Set Oscilloscope to trigger on a rising edge at about 1V.
5. Set Tester Source port to disconnected state.
6. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
7. Set Tester Source capacitance to TEST_CAPACITOR.
8. Set Tester Source to use its Source Packet engine to ACK everything.
9. Execute the Tester_Source_R_Discovery procedure.
10. When the Tester Source receives a command, reply with a link level ACK.
11. FAIL if DUT has problems doing Discovery.
12. FAIL if DUT does not arbitrate for access to CBUS to send a command.
13. For Test 1 FAIL if Arbitration/Sync/Data pulses have a rise time of less than $T_{R_CBUS}\{\min\}$.
14. For Test 2 FAIL if Arbitration/Sync/Data pulses have a fall time of less than $T_{F_CBUS}\{\min\}$.
15. For Test 3 FAIL if Arbitration/Sync/Data pulses have a rise time of greater than $T_{R_CBUS}\{\max\}$.
16. For Test 4 FAIL if Arbitration/Sync/Data pulses have a fall time of greater than $T_{F_CBUS}\{\max\}$.
17. If rise times and fall times differ by more than the allowed amount, ΔT_{RF} , then FAIL, else PASS.
18. If all preceding steps pass, then PASS; else FAIL.

Execute the local test procedure once for every row in the following table, substituting variables from the table for each execution.

Table 4-3. Arbitration/Sync/Data Edge Rate

| | TEST_CAPACITOR | TEST_RESISTOR | Purpose |
|---|--|--------------------------------|-------------------------------------|
| 1 | minimum cable + minimum tester cap | $Z_{CBUS_SOURCE_ON}\{\max\}$ | Arbitration/Sync/Data Rise time MIN |
| 2 | minimum cable + minimum tester cap | $Z_{CBUS_SOURCE_ON}\{\min\}$ | Arbitration/Sync/Data Fall time MIN |
| 3 | $C_{CBUS}\{\max\} + C_{SRC_ON_CBUS}\{\max\}^1$ | $Z_{CBUS_SOURCE_ON}\{\min\}$ | Arbitration/Sync/Data Rise time MAX |
| 4 | $C_{CBUS}\{\max\} + C_{SRC_ON_CBUS}\{\max\}^1$ | $Z_{CBUS_SOURCE_ON}\{\max\}$ | Arbitration/Sync/Data Fall time MAX |

Notes on Table 4-3:

1. Capacitance load for testing is calculated from cable and source capacitance. Test tool shall adjust its capacitive load to match this total

4.3.8 Link Layer Timing – Sink DUT Output: Arbitration/Sync/Data in Bit Times

Measure the timing of the Sink DUT as it drives Data pulses.

The Sink DUT output Bit Time must be measured based on observing CBUS edges.

4.3.8.1 CBT-Sink: Arb, Sync, Data HIGH and LOW Timing

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.8.1.1 Test Objective

Verify that the Sink DUT does not arbitrate to send a first Packet over the CBUS until $T_{\text{SINK_ARBITRATE}}$ after discovery completes.

Verify that the Sink DUT drives a CBUS LOW arbitration pulse for $T_{\text{ARB_SINK}}$. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Sink DUT drives CBUS HIGH before Sync pulse after winning arbitration. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Sink DUT drives CBUS LOW to make a Sync pulse after winning arbitration. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Sink DUT drives CBUS HIGH after the Sync pulse until it starts driving data. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Check the timing for Arbitration pulses in terms of “Bit Times”, and verify that it meets the required spec.

Check the Idle period between Arbitration and the Sync pulse in terms of “Bit Times”, and verify that it meets the required spec.

Check the Sync LOW pulse in terms of “Bit Times”, and verify that it meets the required spec.

Check the delay between the Sync LOW end and the first Data transition in terms of “Bit Times”, and verify that it meets the required spec.

Check the timing of 0 and 1 bits, taking care to observe duty-cycle and edge rate differences.

Measure the timings using the method described in Section 2.5.1.1.

4.3.8.1.2 References

[MHL] Section 8.5.1

[MHL] Section 13.10.1.1; Table 13-31; $T_{\text{SINK_ARBITRATE}}$

[MHL] Section 7.2.2

[MHL] Section 13.10.2; Table 12-33; $T_{\text{ARB_SINK}}$

[MHL] Section 7.2.4

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$

[MHL] Section 13.10.2; Table 13-33; $T_{\text{ARB_SRC}}$

[MHL] Section 13.10.2; Table 13-33; $T_{\text{REQ_OPP}}$

[MHL] Section 13.10.2; Table 12-33; $T_{\text{ARB_TOT}} - T_{\text{ARB_SINK}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{SYNC_TOTAL}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{SYNC_DUTY}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{CBUS_DUTY}}$

[MHL] Section 13.10.1; Table 13-23; ΔT_{RF}

4.3.8.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.8.1.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a command, reply with a link level ACK.
9. Measure the timing of the packets from the Sink DUT in nanoseconds.
10. Measure timing information in nanoseconds and assign values to a set of named timing variables:
 - a. ARB_LOW time
 - b. Idle_Time
 - c. Sync_LOW time
 - d. Sync_HIGH time
 - e. HIGH_0 time
 - f. LOW_0 time
 - g. Starting_LOW_1 timings
 - h. Starting_HIGH_1 timings
11. FAIL if Sink does CBUS arbitration sooner than $T_{SINK_ARBITRATE}$ from the end of Discovery.
12. FAIL if CBUS ARB LOW time is not between (minimum or maximum T_{BIT_CBUS}) * T_{ARB_SINK} .
13. FAIL if CBUS HIGH time after the arbitration LOW time but before Sync LOW time is not between (minimum or maximum T_{BIT_CBUS}) * $(3 + T_{REQ_OPP})$.
14. FAIL if CBUS LOW time during Sync is not between (minimum or maximum T_{BIT_CBUS}) * (minimum or maximum T_{SYNC_DUTY}).
15. FAIL if CBUS HIGH time between Sync LOW time and falling edge of first Data bit is not between (minimum or maximum T_{BIT_CBUS}) * $(T_{SYNC_TOTAL} - (\text{minimum or maximum } T_{SYNC_DUTY}))$.
16. FAIL if time between Sync going LOW time and falling edge of first Data bit is not between (minimum or maximum T_{BIT_CBUS}) * (T_{SYNC_TOTAL}) .
17. FAIL if DUT does not drive Data 0 bits which are asserted entirely LOW on the CBUS for the required period (in nanoseconds).
18. FAIL if DUT does not drive Data 0 bits which are asserted entirely HIGH on the CBUS for the required period (in nanoseconds).
19. FAIL if DUT does not drive Data 1 bits which start HIGH for the first half of a bit time for the required HIGH period (in nanoseconds).
20. FAIL if DUT does not drive Data 1 bits which start HIGH for the first half of a bit time for the following required LOW period (in nanoseconds).
21. FAIL if DUT does not drive Data 1 bits which start LOW for the first half of a bit time for the required LOW period (in nanoseconds).
22. FAIL if DUT does not drive Data 1 bits which start LOW for the first half of a bit time for the following required HIGH period (in nanoseconds).
23. FAIL if DUT does not drive Data 1 bits with required symmetry.
24. Calculate Bit Time range based on Arbitration pulse LOW.
25. Calculate Bit Time range based on HIGH period after the arbitration pulse.

26. Calculate Bit Time range based on Sync pulse LOW.
27. Calculate Bit Time range based on "1" bit timing.
28. Calculate Bit Time range based on pairs of "0" bit timings.
29. FAIL if the bit time ranges based on Arbitration, Sync, and Data bits are not consistent.
30. If all preceding steps pass, then PASS; else FAIL.

4.3.8.2 CBT-Sink: Bit Timing Variation within a Packet

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.8.2.1 Test Objective

Continuously monitor Sink DUT individual received packets to verify that Bit Timing does not change too quickly.

4.3.8.2.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_VARY_PACKET}}$

4.3.8.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.8.2.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

This continuous monitor will be disabled during any test which sets the CBUS LOW Limit comparator to a value different than the CDF value .

1. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
2. Continuously monitor the CBUS to look for failures of this spec.
3. FAIL if the bit timing of any bit varies from the mean bit time of the packet by more than $T_{\text{BIT_VARY_PACKET}}$.
4. If all preceding steps pass, then PASS; else FAIL.

4.3.9 Link Layer Timing – Sink DUT Output: Link Level NACK

Verify that Sink DUT is willing to retransmit data the required number of times when it receives link level NACKs.

4.3.9.1 CBT-Sink: Response to Link Level NACK

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.9.1.1 Test Objective

Verify that Sink DUT is willing to retransmit data the required number of times when it receives NACKs.

4.3.9.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.2; Table 13-33; N_{RETRY}

4.3.9.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.9.1.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to NACK the first $N_{\text{RETRY}}\{\text{min}\}-1$ packets from the Sink DUT, then to ACK all subsequent packets.
6. Execute the Tester_Source_Discovery procedure.
7. When the Tester Source receives a command, it responds with a link layer NACK up to $N_{\text{RETRY}}\{\text{min}\}-1$ times.
8. FAIL if DUT does not eventually issue a command.
9. FAIL if DUT does not release the bus based on the link level NACK.
10. FAIL if DUT does not retry the command the required minimum number of times.
11. FAIL if DUT does not stop retrying the command based on the link level ACK.
12. If all preceding steps pass, then PASS; else FAIL.

Note that the Sink DUT might NACK the ACK Packet. Tester needs to re-issue this command up to $N_{\text{RETRY}}\{\text{min}\}$ times to give the Sink DUT a chance to ACK it.

4.3.10 Link Layer Timing – Sink DUT Output: Link Level ACK

Measure the timing of the Sink DUT as it drives link level ACK pulses.

The Sink DUT output Bit Time must be measured.

4.3.10.1 CBT-Sink: ACK Output Timing in Nanoseconds

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.10.1.1 Test Objective

Measure when Sink DUT drives link level ACK when Tester uses various Source Bit timing and various Parities.

4.3.10.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 7.2.4

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$ Max-Min

[MHL] Section 7.2.4.1; Table 13-33; $T_{\text{REQ_CONT}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{CBUS_ACK_FALL}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{CBUS_ACK_0}}\{\text{min}\}$

4.3.10.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.10.1.4 Required Test Methodology

Execute the following test procedure once for every row in Table 4-4, substituting TEST_IDLE_PERIOD and TEST_DATA from the table for each execution. Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.

5. Set Tester Source to use its Source Packet engine to link level ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
8. When the Tester Source receives a command, reply with link level ACK.
9. Tester Source does SET_INT to Sink DUT, releases the bus.
10. Tester Source sends TEST_DATA for an offset, releases the bus.
11. Tester Source sends Data of all 0's.
12. FAIL if Sink does not eventually send a command.
13. FAIL if DUT does not link level ACK all 3 packets sent from the Tester.
14. FAIL if DUT does not drive the link level ACK LOW starting within $T_{CBUS_ACK_FALL}$ of the end of the Tester-driven packet parity bit.
15. FAIL if DUT drives an ACK LOW pulse which is thinner or wider than $T_{CBUS_ACK_0}$.
16. If all preceding steps pass, then PASS; else FAIL.

Note that the Sink DUT might NACK the packets it is receiving. Tester needs to re-issue this command up to N times to give the Sink DUT a chance to ACK it.

Table 4-4. ACK Output Timing in Nanoseconds

| | TEST_IDLE_PERIOD | TEST_DATA | Purpose |
|---|-----------------------|-----------|--------------------|
| 1 | $T_{REQ_OPP}\{min\}$ | 0x20 | Measure ACK timing |
| 2 | $T_{REQ_OPP}\{min\}$ | 0x21 | Measure ACK timing |

4.3.10.2 CBT-Sink: ACK Drive HIGH Duration.

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.10.2.1 Test Objective

Measure the time the Sink DUT drives ACK HIGH.

4.3.10.2.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.1; Table 13-23; $T_{DRV_HI_CBUS}$

4.3.10.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.10.2.4 Required Methodology

Connect the scope to observe the Tester Source CBUS.

Either use the scope to examine the CBUS HIGH period, or adjust the CBUS HIGH and CBUS LOW Limit comparators, to separate out the period at the beginning of a HIGH period where the Sink DUT is actively pulling HIGH to signal ACK.

This HIGH voltage should be higher than the voltage throughout the rest of the CBUS HIGH period, where only a Source-side resistor is pulling the bus up.

Execute the following test procedure once for every row in Table 4-5, substituting TEST_IDLE_PERIOD and TEST_DATA.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Adjust the Tester resistances and voltages so that the DUT Drive High time is observable.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a command, reply with a link level ACK.
9. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
10. Tester Source does SET_INT to Sink DUT, releases the bus.
11. Tester Source sends TEST_DATA for an offset, releases the bus.
12. Tester Source sends Data of all 0's.
13. Observe post-discovery Sink DUT link-level ACK activity as it replies to the Tester Source command.
14. Observe that the Sink DUT replies with an ACK packet.
15. FAIL if the CBUS Drive HIGH time during the ACK period (the period starting when the CBUS goes from LOW to HIGH and lasting until only the Source resistor pull-up drives the CBUS) is outside the limits set for $T_{DRV_HL_CBUS}$.
16. If all preceding steps pass, then PASS; else FAIL.

Table 4-5. ACK Drive HIGH Duration.

| | TEST_IDLE_PERIOD | TEST_DATA | Purpose |
|---|-----------------------|-----------|--------------------|
| 1 | $T_{REQ_OPP}\{min\}$ | 0x20 | Measure ACK timing |

4.3.11 Link Layer Timing – Sink DUT Output: Bus Re-Arbitration

Measure the behavior and timing of the Sink DUT as it re-arbitrates for the bus.

The Sink DUT output Bit Time must be measured.

4.3.11.1 CBT-Sink: Sink uses Case 2 Regular Arbitration after NACK

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.11.1.1 Test Objective

Verify that Sink DUT backs off after a link-level NACK, and uses Case 2 regular arbitration timing to reacquire the bus.

4.3.11.1.2 References

[MHL] Section 7.2.4; Case 2

[MHL] Section 13.10.2; Table 13-33; T_{REQ_HOLD}

4.3.11.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.11.1.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.

5. Set Tester Source to use its Source Packet engine to NACK each received packet $N_{RETRY}\{min\}$ -1 times, then ACK it.
6. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
7. Execute the Tester_Source_Discovery procedure.
8. Set Tester Source to send a GET_STATE to Sink DUT.
9. FAIL if DUT does not eventually arbitrate for the bus to send a data packet to the Tester Source.
10. Carefully measure the Sink DUT T_{BIT_CBUS} during the retransmissions of the data packet.
11. Finally accept the data packet and link level ACK it.
12. FAIL if DUT does not eventually send a data packet.
13. FAIL if DUT does not release the bus based on the NACK.
14. FAIL if DUT does arbitration (after a NACK) in less than $T_{REQ_HOLD}\{min\}$ (expressed in DUT Bit Times of the subsequent packet).
15. FAIL if DUT does not eventually complete its command after retransmission.
16. If all preceding steps pass, then PASS; else FAIL.

4.3.11.2 CBT-Sink: Sink Case 3 Long Re-arbitration when it Gives up the Bus

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.11.2.1 Test Objective

Verify that Sink DUT uses Case 3 Long Arbitration whenever it gives up the bus and later re-acquires it.

4.3.11.2.2 References

4.3.11.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.11.2.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK all packets.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source to send a GET_STATE to Sink DUT.
8. When DUT sends the data packet back, link level ACK it.
9. Use the timing of the observed waveforms to calculate the Sink DUT T_{BIT_CBUS} .
10. FAIL if DUT does not eventually send a data packet.
11. If DUT never sends a packet, then pauses for re-arbitration, then sends another within 10 seconds, end test with PASS (SKIP).
12. FAIL if DUT does re-arbitration (after an ACK) sooner than T_{WAIT} (expressed in Source Bit Times of the subsequent packet).
13. If all preceding steps pass, then PASS; else FAIL.

4.3.11.3 CBT-Sink: Sink Uses Case 1 Back-to-Back Timing (No Re-arbitration)

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.11.3.1 Test Objective

Verify that Sink DUT uses correct delay from ACK period start to Sync falling edge on Case 1 back-to-back packet sends.

4.3.11.3.2 References

[MHL] Section 7.2.4; Case 1

[MHL] Section 13.10.2; Table 13-33; T_{REQ_CONT}

[MHL] Section 13.10.2; Table 13-23; T_{SYNC_TOTAL} , T_{SYNC_DUTY}

[MHL] Section 13.10.1; Table 13-23: $T_{DRV_HI_CBUS}$

4.3.11.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.11.3.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK all packets.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source to send READ_DEVCAP with offset of 0 to Sink DUT.
8. After READ_DEVCAP + offset packet is sent, Sink DUT sends an ACK packet followed by a data packet.
9. Use the timing of the observed waveforms to calculate the Sink DUT T_{BIT_CBUS} .
10. FAIL if DUT does not respond with an ACK packet followed by a data packet.
11. If DUT never sends back-to-back packs within 10 seconds, end test with PASS (SKIP). (Sink might always allow arbitration after sending every packet.)
12. FAIL if DUT does not drive a Sync pulse for the data packet within T_{REQ_CONT} (in Sink DUT bit times) after the 2-bit-time (measured using the Sink DUT Bit Time) ACK period.
13. FAIL if CBUS LOW time during Sync is not between (minimum or maximum T_{BIT_CBUS}) * (minimum or maximum T_{SYNC_DUTY}).
14. FAIL if CBUS HIGH time between Sync LOW time and falling edge of first Data bit is not between (minimum or maximum T_{BIT_CBUS}) * ($T_{SYNC_TOTAL} - (\text{minimum or maximum } T_{SYNC_DUTY})$).
15. If all preceding steps pass, then PASS; else FAIL.

4.3.11.4 CBT-Sink: Sink Never Sends Too Many Back-to-Back Packets

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.11.4.1 Test Objective

Continuously monitor the CBUS to verify that the Sink DUT does not send too many packets back-to-back.

4.3.11.4.2 References

[MHL] Section 7.2.4; Case 4

[MHL] Section 13.10.2; Table 13-33; N_{MAX}

4.3.11.4.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.11.4.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. Continuous monitor the CBUS wire after Discovery to detect Sink DUT arbitration and packet activity.
2. FAIL if DUT ever sends more than N_{MAX} back-to-back packets without giving up the bus.
3. If all preceding steps pass, then PASS; else FAIL.

4.3.12 Link Layer Timing – Sink DUT Output: Ill-formed packets

Verify that the Sink does not send ill-formed packets (for instance, packets shorter than 16 bit-times long).

4.3.12.1 CBT-Sink: Sink Never Sends Impulse Noise

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.12.1.1 Test Objective

Continuously monitor Sink DUT to check that it does not drive CBUS LOW for short periods.

4.3.12.1.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; T_{BIT_CBUS}

4.3.12.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.12.1.4 Required Methodology

Continuous monitor runs in the background.

FAIL if DUT seems to be responsible for LOW periods on the CBUS of less than (maybe) 25% of T_{BIT_CBUS} .
If all preceding steps pass, then PASS; else FAIL.

4.3.12.2 CBT-Sink: Sink Never Sends Partial Packets

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.12.2.1 Test Objective

Continuously monitor Sink DUT to check that it does not arbitrate for the CBUS, then send a short packet.

4.3.12.2.2 References

4.3.12.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.12.2.4 Required Methodology

Continuous monitor runs in the background.

FAIL if DUT drives arbitration pulse, then Sync pulse, then less than a full packet consisting of Control, Type, Data, and Parity bits.

If all preceding steps pass, then PASS; else FAIL.

4.3.13 Link Layer Electrical – Sink DUT Input: Discovery

Verify that Sink DUT sees Discovery pulses with varying voltage levels.

4.3.13.1 CBE-Sink: Discovery Sensitivity to Input Voltages

Application: Sink

4.3.13.1.1 Test Objective

Verify that Sink DUT responds to Wake and Discovery pulses with different voltage settings.

4.3.13.1.2 References

[MHL] Section 8.4

[MHL] Section 13.10.1; Table 13-23; V_{OH_CBUS} , V_{OL_CBUS}

[MHL] Section 13.10.1; Table 13-24; V_{IH_CBUS} , V_{IL_CBUS}

4.3.13.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.13.1.4 Required Methodology

Execute the following test procedure once for each row in Table 4-6, substituting TEST_VOH_CBUS and TEST_VOL_CBUS.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source HIGH output voltage to TEST_VOH_CBUS.
6. Set Tester Source LOW output voltage to TEST_VOL_CBUS.
7. Execute the Tester_Source_Discovery procedure.
8. FAIL if discovery fails.
9. If all preceding steps pass, then PASS; else FAIL.

Table 4-6. Sensitivity to VIH/VIL

| | TEST_VOH_CBUS | TEST_VOL_CBUS | Purpose |
|---|------------------------------|-------------------------------------|--------------------|
| 1 | $V_{IH_CBUS}\{\text{min}\}$ | $V_{OL_CBUS}\{\text{min}\}$ (0.0V) | Sensitivity to VIH |
| 2 | $V_{OH_CBUS}\{\text{max}\}$ | $V_{IL_CBUS}\{\text{max}\}$ | Sensitivity to VIL |

4.3.14 Link Layer Timing – Sink DUT Input: Discovery OK

Observe the response of the Sink DUT as it is presented with Discovery pulses with different speeds and widths.

4.3.14.1 CBT-Sink: Valid Wake Pulse Timing

Application: Sink

4.3.14.1.1 Test Objective

Verify that Sink DUT responds to valid Wake pulses.

4.3.14.1.2 References

NOTE that Sink DUT may not immediately respond to Wake pulses if it is in SINK4 state. This test may need to be repeated after waiting for the DUT to enter SINK1 state.

[MHL] Section 13.8.1; Table 13-13; C_{CBUS}

[MHL] Section 13.10.1; Table 13-23; T_{R_CBUS} , T_{F_CBUS}

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_1}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_2}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_TO_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-30; $T_{SRC_PULSE_WIDTH}$

[MHL] Section 13.10.1.1; Table 13-30; $T_{SINK_PULSE_WIDTH}$

[MHL] Section 13.10.1; Table 13-30; T_{SINK_CONN}

[MHL] Section 13.10.1; Table 13-31; $T_{SINK_RXSENSE_EN}$

[MHL] Section 13.6.3; Table 13-9; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$

[MHL] Section 13.7.3; Table 13-12; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$

4.3.14.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.14.1.4 Required Methodology

Execute the following test procedure once for each test case #1 - #7 in Table 4-7 and Table 4-8, substituting for the variables in the leftmost column.

1. If the CDF value **CDF_SINK_WAKE_FROM_STANDBY** indicates that Sink DUT awakens from standby, then continue, else end test with PASS (SKIP).
2. If necessary, connect Sink DUT to Tester Source port.
3. Apply power to the Sink DUT.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Tester Source to these variables: $C_{CBUS}\{\text{min}\}$, TEST_WAKE_PULSE_HIGH, TEST_WAKE_PULSE_LOW, TEST_WAKE_PULSE_GAP, TEST_WAKE_PULSE_DISCOVER, TEST_DISCOVERY_PULSE_HIGH, TEST_DISCOVERY_PULSE_LOW, TEST_FIRST_DISCOVERY_PULSE_HIGH, TEST_FIRST_DISCOVERY_PULSE_LOW, TEST_FIFTH_DISCOVERY_PULSE_HIGH, and TEST_FIFTH_DISCOVERY_PULSE_LOW
7. Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
8. Set Sink DUT into Standby State based on the CDF procedure **CDF_PROC_SET_STANDBY**.
9. Execute the Tester_Source_Modified_Discovery procedure.
10. FAIL if Source Tester detects that the CBUS does not go LOW after each Wake pulse.
11. FAIL if Source Tester detects that the CBUS does not go LOW after each of the first 5 Discovery pulses.
12. NOTE that the CBUS might stay HIGH after the 6th Discovery Pulse. This indicates that the DUT transitioned from SINK1 to SINK5, and is delaying Discovery. In this case, wait for the CDF value **CDF_SINK_MAX_STANDBY_TO_ACTIVE** and restart the test. The DUT should NOT transition to SINK5 the second time the test is run.
13. FAIL if Source Tester does not see CBUS stay HIGH after the 6th Discovery pulse.
14. FAIL if DUT does not change its pulldown resistor from $Z_{CBUS_SINK_DISCOVER}$ to $Z_{CBUS_SINK_ON}$ after the rising edge of the 6th Discovery Pulse plus T_{SINK_CONN} at latest.

15. FAIL if DUT does not pull MHL+ and MHL- up with between $Z_{RXSENSE_TERM}$ and $Z_{RXSENSE_TMDS}$ after the rising edge of the 6th Discovery Pulse plus T_{SINK_CONN} plus $T_{SINK_RXSENSE_EN}$ at latest.
16. If all preceding steps pass, then PASS; else FAIL.

Table 4-7. Wake Pulse Timing 1

| | 1 | 2 | 3 | 4 |
|---------------------------------|---|---|---|---|
| TEST_WAKE_PULSE_HIGH | $T_{SRC_WAKE_PULSE_WIDTH_1}\{max\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{max\}$ |
| TEST_WAKE_PULSE_LOW | $T_{SRC_WAKE_PULSE_WIDTH_1}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{max\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{max\}$ |
| TEST_WAKE_PULSE_GAP | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{min\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| TEST_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| Purpose | Minimum Wake Pulse LOW width | Minimum Wake Pulse HIGH width | Minimum Wake Pulse period | Maximum Wake Pulse period |

Table 4-8. Wake Pulse Timing 2

| | 5 | 6 | 7 |
|---------------------------|---|---|---|
| TEST_WAKE_PULSE_HIGH | $T_{SRC_WAKE_PULSE_WIDTH_1}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{typ\}$ |
| TEST_WAKE_PULSE_LOW | $T_{SRC_WAKE_PULSE_WIDTH_1}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{typ\}$ |
| TEST_WAKE_PULSE_GAP | $T_{SRC_WAKE_PULSE_WIDTH_2}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{max\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{max\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |

| | 5 | 6 | 7 |
|---------------------------------|---|---|---|
| TEST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| Purpose | Minimum inter-Wake Pulse gap | Maximum inter-Wake Pulse gap | Maximum Wake Pulse to Discovery Pulse gap |

4.3.14.2 CBT-Sink: Valid Discovery Pulse Timing

Application: Sink

4.3.14.2.1 Test Objective

Verify that Sink DUT responds to valid Discovery pulses.

Tester actively drives Discovery pulses HIGH, but depends on Sink discovery resistor pulling the bus LOW. The Source does not actively drive LOW during discovery pulses.

Tester Source can make the first Discovery pulse or the fifth Discovery pulse with programmable timing.

4.3.14.2.2 References

NOTE that Sink DUT may not immediately respond to Wake pulses if it is in SINK4 state. This test may need to be repeated after waiting for the DUT to enter SINK1 state.

[MHL] Section 13.8.1; Table 13-13; C_{CBUS}

[MHL] Section 13.10.1; Table 13-23; $T_{\text{R_CBUS}}$, $T_{\text{F_CBUS}}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{\text{SRC_WAKE_TO_DISCOVER}}$

[MHL] Section 13.10.1.1; Table 13-30; $T_{\text{SRC_PULSE_WIDTH}}$

[MHL] Section 13.10.1.1; Table 13-30; $T_{\text{SINK_PULSE_WIDTH}}$

[MHL] Section 13.10.1; Table 13-30; $T_{\text{SINK_CONN}}$

[MHL] Section 13.10.1; Table 13-31; $T_{\text{SINK_RXSENSE_EN}}$

[MHL] Section 13.6.3; Table 13-9; $Z_{\text{RXSENSE_TERM}}$, $Z_{\text{RXSENSE_TMDS}}$

[MHL] Section 13.7.3; Table 13-12; $Z_{\text{RXSENSE_TERM}}$, $Z_{\text{RXSENSE_TMDS}}$

4.3.14.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.14.2.4 Required Methodology

Execute the following test procedure once for each test case #1 - #10 in Table 4-9 to Table 4-11, substituting for the variables in the leftmost column.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to these variables: $C_{CBUS}\{\text{min}\}$, TEST_WAKE_PULSE_HIGH, TEST_WAKE_PULSE_LOW, TEST_WAKE_PULSE_GAP, TEST_WAKE_PULSE_DISCOVER, TEST_DISCOVERY_PULSE_HIGH, TEST_DISCOVERY_PULSE_LOW, TEST_FIRST_DISCOVERY_PULSE_HIGH, TEST_FIRST_DISCOVERY_PULSE_LOW, TEST_FIFTH_DISCOVERY_PULSE_HIGH, and TEST_FIFTH_DISCOVERY_PULSE_LOW
6. Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
7. Set Sink DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
8. Execute the Tester_Source_Modified_Discovery procedure.
9. FAIL if Source Tester detects that the CBUS does not go LOW after each Wake pulse.
10. FAIL if Source Tester detects that the CBUS does not go LOW after each of the first 5 Discovery pulses.
11. FAIL if Source Tester does not see CBUS stay HIGH after the 6th Discovery pulse.
12. FAIL if DUT does not change its pulldown resistor from $Z_{CBUS_SINK_DISCOVER}$ to $Z_{CBUS_SINK_ON}$ after the rising edge of the 6th Discovery Pulse plus T_{SINK_CONN} at latest.
13. FAIL if DUT does not pull MHL+ and MHL- up with between $Z_{RXSENSE_TERM}$ and $Z_{RXSENSE_TMDS}$ after the rising edge of the 6th Discovery Pulse plus T_{SINK_CONN} plus $T_{SINK_RXSENSE_EN}$ at latest.
14. If all preceding steps pass, then PASS; else FAIL.

Table 4-9. Discovery Pulse Timing 1

| | 1 | 2 | 3 | 4 |
|---------------------------------|--|--|--|--|
| TEST_WAKE_PULSE_HIGH | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_LOW | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_GAP | $T_{SRC_WAKE_PULSE_WIDTH_2}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{SRC_WAKE_TO_DISCOVER}\{\text{min}\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{\text{min}\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{\text{min}\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{\text{min}\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}$ | $T_{SINK_PULSE_WIDTH}\{\text{max}\}$ |
| TEST_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}\{\text{min}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{max}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{min}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{max}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}\{\text{min}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{max}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{min}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{max}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ |

| | 1 | 2 | 3 | 4 |
|--------------------------------|---|---|---|---|
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ |
| Purpose | Minimum Discovery Pulse LOW width | Maximum Discovery Pulse LOW width | Minimum Discovery Pulse HIGH width | Maximum Discovery Pulse HIGH width |

Table 4-10. Discovery Pulse Timing 2

| | 5 | 6 | 7 | 8 |
|---------------------------------|---|---|---|---|
| TEST_WAKE_PULSE_HIGH | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_LOW | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_GAP | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{min}\}$ | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{max}\}$ | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{min}\}$ | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{max}\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ |
| TEST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ |
| Purpose | Minimum Discovery Pulse width | Maximum Discovery Pulse width | Minimum, then Maximum Discovery Pulse width | Maximum, then Minimum Discovery Pulse width |

Table 4-11. Discovery Pulse Timing 3

| | 9 | 10 |
|---------------------------|---|---|
| TEST_WAKE_PULSE_HIGH | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_LOW | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_GAP | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{min}\}$ | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{max}\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |

| | | |
|---------------------------------|---------------------------------------|---------------------------------------|
| TEST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| Purpose | Discovery Pulses with fast Edge Rates | Discovery Pulses with slow Edge Rates |

4.3.14.3 CBT-Sink: Sink in Standby Discovers on Wake plus Discovery Pulse Sequence

Application: Sink

4.3.14.3.1 Test Objective

Verify that Sink DUT in Standby State responds to a valid Wake plus Discovery Pulse sequence.

4.3.14.3.2 References

[MHL] Section 8.2.2; Table 8-5; SINK0, SINK1, SINK2

[MHL] Section 8.2.2; Table 8-5; SINK3, SINK4, SINK5

[MHL] Section 13.10.1; Table 13-26; Z_{CBUS_SINK_DISCOVER}

[MHL] Section 13.10.1.1; Table 13-29; T_{SINK_VBUS_EN}

[MHL] Section 13.10.1.1; Table 13-29; T_{SINK_READY_TO_DISCOVER}

[MHL] Section 8.2.2; Table 8-6; Note1

4.3.14.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.14.3.4 Required Methodology

1. If the CDF value **CDF_SINK_WAKE_FROM_STANDBY** indicates that Sink DUT awakens from standby, then continue, else end test with PASS (SKIP).
2. If necessary, connect Sink DUT to Tester Source port.
3. Apply power to the Sink DUT.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. If the CDF indicates that the DUT does not implement Standby mode (procedure **CDF_PROC_SET_STANDBY** is empty), then PASS (SKIP).
7. Set Sink DUT into Standby State based on the CDF procedure **CDF_PROC_SET_STANDBY**.
8. FAIL if Sink Discovery resistor is not a valid Z_{CBUS_SINK_DISCOVER}.
9. Execute the Tester_Source_Discovery procedure.
10. If device becomes successfully Connected by the first Wake and Discovery Pulse sequence, PASS the test and terminate early.
11. FAIL if CBUS does not become HIGH-Z during SINK5 (Sink Port transitions to I_{CBUS_LEAK_SINK}) within the shorter of T_{SINK_VBUS_EN} and T_{SINK_READY_TO_DISCOVER} for a minimum of T_{sink_CBUS_FLOAT}.
12. FAIL if Sink DUT does not assert a valid Z_{CBUS_SINK_DISCOVER} within the time specified in the CDF value **CDF_SINK_MAX_STANDBY_TO_ACTIVE** to allow a second Discovery attempt.
13. FAIL if Sink DUT does not enable VBUS within T_{SINK_VBUS_EN}{max} of asserting Z_{CBUS_SINK_DISCOVER}.
14. Tester Source executes Tester_Source_discovery procedure in response to Sink transition back to Z_{CBUS_SINK_DISCOVER}.
15. FAIL if Sink DUT is not successfully become Connected as a result of the Wake and Discovery Pulse sequence.

16. If all preceding steps pass, then PASS; else FAIL.

Please note that the Sink DUT shall enable VBUS during the state transition from unpowered to powered state. The enabling of VBUS and the assertion of $Z_{CBUS_SINK_DISCOVER}$ do not follow any particular order and both shall follow their individual timing constraints. This test uses the assertion of $Z_{CBUS_SINK_DISCOVER}$ as a trigger to check the timing of VBUS enabling, because the Sink internal state transition from unpowered to powered state is not observable directly.

4.3.15 Link Layer Timing – Sink DUT Input: Discovery Reject

Observe the response of the Sink DUT as it is presented with Discovery pulses with illegal widths.

4.3.15.1 CBT-Sink: First Discovery Pulse should be Ignored

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.15.1.1 Test Objective

Observe the Sink DUT rejects first illegal Discovery pulses.

4.3.15.1.2 References

[MHL] Section 13.8.1; Table 13-13; C_{CBUS}
[MHL] Section 13.10.1; Table 13-23; T_{R_CBUS} , T_{F_CBUS}
[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_1}$
[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_2}$
[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_TO_DISCOVER}$
[MHL] Section 13.10.1.1; Table 13-30; $T_{SRC_PULSE_WIDTH}$
[MHL] Section 13.10.1.1; Table 13-30; $T_{SINK_PULSE_WIDTH}$
[MHL] Section 13.10.1; Table 13-30; T_{SINK_CONN}
[MHL] Section 13.10.1; Table 13-31; $T_{SINK_RXSENSE_EN}$
[MHL] Section 13.6.3; Table 13-9; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$
[MHL] Section 13.7.3; Table 13-12; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$
[MHL] Section 13.10.1; Table 13-30; $T_{SRC_CBUS_FLOAT}$
[MHL] Section 13.10.1; Table 13-30; $T_{SINK_PULSE_REJECT_MIN}$
[MHL] Section 13.10.1; Table 13-30; $T_{SINK_PULSE_REJECT_MAX}$

4.3.15.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.15.1.4 Required Methodology

Execute the following test procedure once for each test case #1 - #8 in Table 4-12 and Table 4-13, substituting for the variables in the leftmost column.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source port capacitance to minimum $C_{CBUS}\{\min\}$.

6. Set Tester Source to use settings for these variables from the tables: TEST_WAKE_PULSE_HIGH, TEST_WAKE_PULSE_LOW, TEST_WAKE_PULSE_GAP, TEST_WAKE_PULSE_DISCOVER, TEST_DISCOVERY_PULSE_HIGH, TEST_DISCOVERY_PULSE_LOW, TEST_FIRST_DISCOVERY_PULSE_HIGH, TEST_FIRST_DISCOVERY_PULSE_LOW, TEST_FIFTH_DISCOVERY_PULSE_HIGH, TEST_FIFTH_DISCOVERY_PULSE_LOW, and TEST_EXTRA_PULSES
7. Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
8. Set Sink DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
9. Execute the Tester_Source_Modified_Discovery procedure.
10. If Discovery fails, HIGH_Z the CBUS wire for $T_{SRC_CBUS_FLOAT}$.
11. If Discovery succeeds, also HIGH_Z the CBUS wire for $T_{SRC_CBUS_FLOAT}$.
12. FAIL if Source Tester detects that the CBUS does not go LOW after each Wake pulse, up to the point of the intended failure.
13. FAIL if the CBUS stays HIGH after the 6th Discovery Pulse, indicating unexpected Sink Discovery Completion.
14. If TEST_EXTRA_PULSES is non-zero, FAIL if Sink does not complete Discovery after the last extra Discovery pulse.
15. If TEST_EXTRA_PULSES is non-zero, FAIL if Sink does not pull MHL+ and MHL- up after the rising edge of the last extra Discovery Pulse plus T_{SINK_CONN} plus $T_{SINK_RXSENSE_EN}$ at latest.
16. FAIL if DUT does not reset back to $Z_{CBUS_SINK_DISCOVER}$ and High-Z on MHL+/- wires in response to the CBUS becoming HIGH_Z.
17. Set for a nominal Discovery, with typical Wake pulses and typical Discovery pulses
18. Direct Tester Source to initiate discovery.
19. FAIL if second Discovery procedure does not succeed.
20. If all preceding steps pass, then PASS; else FAIL.

Table 4-12. Discovery Pulse Reject Timing 1

| | 1 | 2 | 3 | 4 |
|---------------------------------|---|---|---|---|
| TEST_WAKE_PULSE_HIGH | $T_{SRC_WAKE_PULSE_WIDTH_1}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_1}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_1}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_1}$ {typ} |
| TEST_WAKE_PULSE_LOW | $T_{SRC_WAKE_PULSE_WIDTH_1}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_1}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_1}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_1}$ {typ} |
| TEST_WAKE_PULSE_GAP | $T_{SRC_WAKE_PULSE_WIDTH_2}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_2}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_2}$ {typ} | $T_{SRC_WAKE_PULSE_WIDTH_2}$ {typ} |
| TEST_WAKE_PULSE_DISCOVER | $T_{SRC_WAKE_TO_DISCOVER}$ {min} | $T_{SRC_WAKE_TO_DISCOVER}$ {min} | $T_{SRC_WAKE_TO_DISCOVER}$ {min} | $T_{SRC_WAKE_TO_DISCOVER}$ {min} |
| TEST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_WIDTH}$ {typ} |
| TEST_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_WIDTH}$ {typ} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_REJECT_MIN}$ | $T_{SINK_PULSE_REJECT_MAX}$ | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_WIDTH}$ {typ} |
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_WIDTH}$ {typ} | $T_{SINK_PULSE_REJECT_MIN}$ | $T_{SINK_PULSE_REJECT_MAX}$ |

| | 1 | 2 | 3 | 4 |
|---------------------------------|--|--|---|---------------------------------------|
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_EXTRA_PULSES | 0 | 0 | 0 | 0 |
| Purpose | First Discovery HIGH pulse is too narrow | First Discovery HIGH pulse is too wide | First Discovery LOW pulse is too narrow | First Discovery LOW pulse is too wide |

Table 4-13. Discovery Pulse Reject Timing 2

| | 5 | 6 | 7 | 8 |
|---------------------------------|---|---|--|--|
| TEST_WAKE_PULSE_HIGH | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_LOW | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_GAP | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} |
| TEST_WAKE_PULSE_DISCOVER | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} |
| TEST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_EXTRA_PULSES | 1 | 1 | 1 | 1 |
| Purpose | First Discovery HIGH pulse is too narrow; finally succeed | First Discovery HIGH pulse is too wide; finally succeed | First Discovery LOW pulse is too narrow; finally succeed | First Discovery LOW pulse is too wide; finally succeed |

4.3.15.2 CBT-Sink: Last Discovery Pulse should be Ignored

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.15.2.1 Test Objective

Observe the Sink DUT rejects last illegal Discovery pulses.

4.3.15.2.2 References

[MHL] Section 13.8.1; Table 13-13; C_{CBUS}
[MHL] Section 13.10.1; Table 13-23; T_{R_CBUS} , T_{F_CBUS}
[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_1}$
[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_2}$
[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_TO_DISCOVER}$
[MHL] Section 13.10.1.1; Table 13-30; $T_{SRC_PULSE_WIDTH}$
[MHL] Section 13.10.1.1; Table 13-30; $T_{SINK_PULSE_WIDTH}$
[MHL] Section 13.10.1; Table 13-30; T_{SINK_CONN}
[MHL] Section 13.10.1; Table 13-31; $T_{SINK_RXSENSE_EN}$
[MHL] Section 13.6.3; Table 13-9; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$
[MHL] Section 13.7.3; Table 13-12; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$
[MHL] Section 13.10.1; Table 13-30; $T_{SRC_CBUS_FLOAT}$
[MHL] Section 13.10.1; Table 13-30; $T_{SINK_PULSE_REJECT_MIN}$
[MHL] Section 13.10.1; Table 13-30; $T_{SINK_PULSE_REJECT_MAX}$

4.3.15.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.15.2.4 Required Methodology

Execute the following test procedure once for each test case #1 - #8 in Table 4-14 and Table 4-15, substituting for the variables in the leftmost column.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source port capacitance to minimum $C_{CBUS}\{min\}$.
6. Set Tester Source to use settings for these variables from the tables: TEST_WAKE_PULSE_HIGH, TEST_WAKE_PULSE_LOW, TEST_WAKE_PULSE_GAP, TEST_WAKE_PULSE_DISCOVER, TEST_DISCOVERY_PULSE_HIGH, TEST_DISCOVERY_PULSE_LOW, TEST_FIRST_DISCOVERY_PULSE_HIGH, TEST_FIRST_DISCOVERY_PULSE_LOW, TEST_FIFTH_DISCOVERY_PULSE_HIGH, TEST_FIFTH_DISCOVERY_PULSE_LOW, and TEST_EXTRA_PULSES
7. Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
8. Set Sink DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
9. Execute the Tester_Source_Modified_Discovery procedure.
10. If Discovery fails, HIGH_Z the CBUS wire for $T_{SRC_CBUS_FLOAT}$.
11. If Discovery succeeds, also HIGH_Z the CBUS wire for $T_{SRC_CBUS_FLOAT}$.

12. FAIL if Source Tester detects that the CBUS does not go LOW after each Wake pulse, up to the point of the intended failure.
13. FAIL if the CBUS stays HIGH after the 6th Discovery Pulse, indicating unexpected Sink Discovery Completion.
14. If TEST_EXTRA_PULSES is non-zero, FAIL if Sink does not complete Discovery using the last extra Discovery pulse.
15. If TEST_EXTRA_PULSES is non-zero, FAIL if Sink does not pull MHL+ and MHL- up after the rising edge of the last extra Discovery Pulse plus $T_{\text{SINK_CONN}}$ plus $T_{\text{SINK_RXSENSE_EN}}$ at latest.
16. FAIL if DUT does not reset back to $Z_{\text{CBUS_SINK_DISCOVER}}$ and High-Z on MHL+/- wires in response to the CBUS being HIGH_Z.
17. Set for a nominal Discovery, with typical Wake pulses and typical Discovery pulses
18. Direct Tester Source to initiate discovery.
19. FAIL if second Discovery procedure does not succeed.
20. If all preceding steps pass, then PASS; else FAIL.

Table 4-14. Discovery Pulse Reject Timing 1

| | 1 | 2 | 3 | 4 |
|---------------------------------|--|--|--|--|
| TEST_WAKE_PULSE_HIGH | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$ {typ} |
| TEST_WAKE_PULSE_LOW | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$ {typ} |
| TEST_WAKE_PULSE_GAP | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}$ {typ} | $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}$ {typ} |
| TEST_WAKE_PULSE_DISCOVER | $T_{\text{SRC_WAKE_TO_DISCOVER}}$ {min} | $T_{\text{SRC_WAKE_TO_DISCOVER}}$ {min} | $T_{\text{SRC_WAKE_TO_DISCOVER}}$ {min} | $T_{\text{SRC_WAKE_TO_DISCOVER}}$ {min} |
| TEST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} |
| TEST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} |
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_REJECT_MIN}}$ | $T_{\text{SINK_PULSE_REJECT_MAX}}$ | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_WIDTH}}$ {typ} | $T_{\text{SINK_PULSE_REJECT_MIN}}$ | $T_{\text{SINK_PULSE_REJECT_MAX}}$ |
| TEST_EXTRA_PULSES | 0 | 0 | 0 | 0 |
| Purpose | Fifth Discovery HIGH pulse is too narrow | Fifth Discovery HIGH pulse is too wide | Fifth Discovery LOW pulse is too narrow | Fifth Discovery LOW pulse is too wide |

Table 4-15. Discovery Pulse Reject Timing 2

| | 5 | 6 | 7 | 8 |
|---------------------------------|---|---|--|--|
| TEST_WAKE_PULSE_HIGH | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_LOW | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_GAP | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} |
| TEST_WAKE_PULSE_DISCOVER | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} |
| TEST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} |
| TEST_EXTRA_PULSES | 6 | 6 | 6 | 6 |
| Purpose | Fifth Discovery HIGH pulse is too narrow; finally succeed | Fifth Discovery HIGH pulse is too wide; finally succeed | Fifth Discovery LOW pulse is too narrow; finally succeed | Fifth Discovery LOW pulse is too wide; finally succeed |

4.3.16 Link Layer Electrical – Sink DUT Input: Arbitration/Sync/Data Signaling

Measure the electrical behavior of the Sink DUT as it receives Arbitration, Sync, and Data pulses.

4.3.16.1 CBE-Sink: Sensitivity to VIH/VIL

Application: Sink

4.3.16.1.1 Test Objective

Verify that the Sink DUT correctly receives data pulses above V_{IH_CBUS} and below V_{IL_CBUS}

4.3.16.1.2 References

[MHL] Section 8-4

[MHL] Section 13.10.1; Table 13-24; V_{OH_CBUS}, V_{OL_CBUS}

[MHL] Section 13.10.1; Table 13-24; V_{IH_CBUS}, V_{IL_CBUS}

4.3.16.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.16.1.4 Required Methodology

Execute the following test procedure once for each row in Table 4-16, substituting TEST_VOH_CBUS and TEST_VOL_CBUS.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source HIGH output voltage to TEST_VOH_CBUS.
6. Set Tester Source LOW output voltage to TEST_VOL_CBUS.
7. Set Tester Source to use its Source Packet engine to link level ACK everything.
8. Execute the Tester_Source_Discovery procedure.
9. When the Tester Source receives a command, reply with an ACK packet.
10. FAIL if DUT does not eventually arbitrate for the bus to send a command to the Tester Source.
11. FAIL if DUT does not link level ACK the ACK packet with fewer than $N_{RETRY}\{min\}$ retries.
12. The Sink DUT will probably NACK the first several attempts to do the ACK packet. Retransmit.
13. If all preceding steps pass, then PASS; else FAIL.

Note: The Sink DUT might send a command that requires a data packet instead of an ACK packet in response. For this test, the contents of the response packet may be either a data packet or a control packet.

Table 4-16. Sensitivity to VIH/VIL

| | TEST_VOH_CBUS | TEST_VOL_CBUS | Purpose |
|---|-----------------------|------------------------------|--------------------|
| 1 | $V_{IH_CBUS}\{min\}$ | $V_{OL_CBUS}\{min\}$ (0.0V) | Sensitivity to VIH |
| 2 | $V_{OH_CBUS}\{max\}$ | $V_{IL_CBUS}\{max\}$ | Sensitivity to VIL |

4.3.17 Link Layer Timing – Sink DUT Input: Arbitration

Measure the behavior of the Sink DUT as it receives Arbitration pulses with different timings.

4.3.17.1 CBT-Sink: End of Discovery to Early Source-side Arbitration

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.17.1.1 Test Objective

Verify that the Sink DUT correctly responds to a packet if a Source sends one before the minimum Discovery-to-Arbitration holdoff.

4.3.17.1.2 References

[MHL] Section 8.5.1

[MHL] Section 13.10.1.1; Table 13-31: T_{SINK_CONN}

[MHL] Section 13.10.1.1; Table 13-31: $T_{SINK_ARBITRATE}$

4.3.17.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.17.1.4 Required Methodology

Execute the following test procedure once for each row in Table 4-17, substituting TEST_BIT_TIME, TEST_IDLE_PERIOD and TEST_FIRST_SINK_COMMAND_DELAY.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. Set Tester Source to operate using TEST_BIT_TIME for T_{BIT_CBUS} .
9. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
10. Wait TEST_FIRST_SINK_COMMAND_DELAY.
11. Tester Source does GET_STATE to Sink DUT.
12. FAIL if DUT does not respond to the GET_STATE command from the Tester with a Data packet.
13. If all preceding steps pass, then PASS; else FAIL.

Note that the Sink DUT might NACK the GET_STATE command. It needs to be re-issued up to N_{RETRY} times to give the Sink DUT a chance to ACK it.

Table 4-17. End of Discovery to Early Source-side Arbitration

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_FIRST_SINK_COMMAND_DELAY | Purpose |
|---|------------------------|-----------------------|-------------------------------|-----------------------------|
| 1 | $T_{BIT_CBUS}\{typ\}$ | $T_{REQ_OPP}\{min\}$ | $T_{SINK_CONN}\{max\}$ | Very Early Sink arbitration |

4.3.17.2 CBT-Sink: Sink Loses Arbitration Collision Correctly

Application: Sink

4.3.17.2.1 Test Objective

Verify that the Sink DUT loses when it collides with the Source during Arbitration.

4.3.17.2.2 References

[MHL] Section 7.2.4; Case 1

4.3.17.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.17.2.4 Required Methodology

Execute the following test procedure once for each row in Table 4-18, substituting TEST_BIT_TIME.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.

6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. Set Tester Source to operate using the TEST_BIT_TIME for $T_{\text{BIT_CBUS}}$.
9. Prepare the Tester Source to do a GET_VENDOR_ID, but do not issue it yet.
10. Set Tester Source to artificially force an Arbitration collision when it sees the first Sink DUT arbitration.
11. The Tester Source will win arbitration because of the collision. Issue a GET_VENDOR_ID.
12. Once the GET_VENDOR_ID is issued, set its Source Packet engine to ACK the response.
13. FAIL if DUT does not collide with the Tester Source during Arbitration and back off, but instead continues on to do its command.
14. FAIL if DUT does not reply to the GET_VENDOR_ID with the expected data.
15. FAIL if DUT does not re-issue the command it was sending before it lost the Arbitration.
16. If all preceding steps pass, then PASS; else FAIL.

Table 4-18. Loses Arbitration Collision Correctly

| | TEST_BIT_TIME | Purpose |
|---|--------------------------------------|--|
| 1 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | Sink loses Arbitration Collision correctly, nominal Sink bit clock |
| 2 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | Sink loses Arbitration Collision correctly, fast Sink bit clock |
| 3 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | Sink loses Arbitration Collision correctly, slow Sink bit clock |

4.3.18 Link Layer Timing – Sink DUT Input: Data

Measure the behavior of the Sink DUT as it receives Data pulses with different timings.

4.3.18.1 CBT-Sink: Bit Timing Sensitivity

| |
|--------------------------|
| Application: Sink |
|--------------------------|

4.3.18.1.1 Test Objective

Verify that Sink DUT can receive data from a Source with the fastest/slowest/most legal bit timings.

4.3.18.1.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}} \pm T_{\text{CBUS_DUTY}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}} \pm \Delta T_{\text{RF}}$

4.3.18.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.18.1.4 Required Methodology

Execute the following test procedure once for each row in Table 4-19, substituting TEST_BIT_TIME, TEST_IDLE_PERIOD, TEST_CENTER_OFFSET and TEST_RISE_OFFSET.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.

6. Set Tester Source to operate using TEST_BIT_TIME for T_{BIT_CBUS} .
7. Set Tester Source to TEST_IDLE_PERIOD delay from Arbitration pulse to Sync pulse.
8. Set Tester Source to offset "1" bit center-bit transitions by TEST_CENTER_OFFSET
9. Set Tester Source to offset rising edges TEST_RISE_OFFSET compared to falling edges.
10. Set Tester Source to use its Source Packet engine to ACK everything.
11. Execute the Tester_Source_Discovery procedure.
12. When the Tester Source receives a SET_HPD, reply with an ACK packet.
13. Tester Source switches to responding with ACK to everything.
14. FAIL if DUT does not ACK the ACK packet from the Tester.
15. If all preceding steps pass, then PASS; else FAIL.

Table 4-19. Bit Timing Sensitivity

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_CENTER_OFFSET | TEST_RISE_OFFSET | Purpose |
|----|------------------------|-----------------------|-----------------------|--------------------|--------------------------------|
| 1 | $T_{BIT_CBUS}\{min\}$ | $T_{REQ_OPP}\{min\}$ | 0 | 0 | Fast; Bits symmetrical |
| 2 | $T_{BIT_CBUS}\{typ\}$ | $T_{REQ_OPP}\{typ\}$ | 0 | 0 | Typical; Bits symmetrical |
| 3 | $T_{BIT_CBUS}\{max\}$ | $T_{REQ_OPP}\{max\}$ | 0 | 0 | Slow; Bits symmetrical |
| 4 | $T_{BIT_CBUS}\{min\}$ | $T_{REQ_OPP}\{min\}$ | $-TEST_BIT_TIME/10$ | 0 | Fast; midpoint early |
| 5 | $T_{BIT_CBUS}\{typ\}$ | $T_{REQ_OPP}\{typ\}$ | $-TEST_BIT_TIME/10$ | 0 | Typical; midpoint early |
| 6 | $T_{BIT_CBUS}\{max\}$ | $T_{REQ_OPP}\{max\}$ | $-TEST_BIT_TIME/10$ | 0 | Slow; midpoint early |
| 7 | $T_{BIT_CBUS}\{min\}$ | $T_{REQ_OPP}\{min\}$ | $+TEST_BIT_TIME/10$ | 0 | Fast; midpoint late |
| 8 | $T_{BIT_CBUS}\{typ\}$ | $T_{REQ_OPP}\{typ\}$ | $+TEST_BIT_TIME/10$ | 0 | Typical; midpoint late |
| 9 | $T_{BIT_CBUS}\{max\}$ | $T_{REQ_OPP}\{max\}$ | $+TEST_BIT_TIME/10$ | 0 | Slow; midpoint late |
| 10 | $T_{BIT_CBUS}\{min\}$ | $T_{REQ_OPP}\{min\}$ | 0 | $-\Delta T_{RF}/2$ | Fast; rise faster than fall |
| 11 | $T_{BIT_CBUS}\{typ\}$ | $T_{REQ_OPP}\{typ\}$ | 0 | $-\Delta T_{RF}/2$ | Typical; rise faster than fall |
| 12 | $T_{BIT_CBUS}\{max\}$ | $T_{REQ_OPP}\{max\}$ | 0 | $-\Delta T_{RF}/2$ | Slow; rise faster than fall |
| 13 | $T_{BIT_CBUS}\{min\}$ | $T_{REQ_OPP}\{min\}$ | 0 | $+\Delta T_{RF}/2$ | Fast; fall faster than rise |
| 14 | $T_{BIT_CBUS}\{typ\}$ | $T_{REQ_OPP}\{typ\}$ | 0 | $+\Delta T_{RF}/2$ | Typical; fall faster than rise |
| 15 | $T_{BIT_CBUS}\{max\}$ | $T_{REQ_OPP}\{max\}$ | 0 | $+\Delta T_{RF}/2$ | Slow; fall faster than rise |

4.3.19 Link Layer Timing – Sink DUT Input: NACK

Verify that Sink DUT can detect bad parity. Bad parity is defined as a Packet with a parity bit with a low to high transition at its end. Refer to Section 3.3.2.12.

4.3.19.1 CBT-Sink: Response to Packet with Bad Parity

Application: Sink

4.3.19.1.1 Test Objective

Verify that Sink DUT can detect bad parity.

4.3.19.1.2 References

[MHL] Section 7.2.3.5

4.3.19.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.19.1.4 Required Methodology

Execute the following test procedure once for each row in Table 4-20, substituting TEST_BIT_RATE, TEST_IDLE_PERIOD, and TEST_BAD_PACKET.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to operate using the TEST_BIT_RATE for T_{BIT_CBUS}.
6. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
7. Set Tester Source to use its Source Packet engine to ACK everything.
8. Execute the Tester_Source_Discovery procedure.
9. Tester Source sends TEST_BAD_PACKET N_{RETRY}{min}-1 times.
10. FAIL if DUT does not NACK the bad Packet from the Tester N_{RETRY}{min}-1 times.
11. If all preceding steps pass, then PASS; else FAIL.

Table 4-20. Check Response to Bad Parity Packet

| | TEST_BIT_RATE | TEST_IDLE_PERIOD | TEST_BAD_PACKET | Purpose |
|---|-----------------------------|----------------------------|-----------------|---------------------------|
| 1 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {max} | Error-Packet-1 | bad parity (Nominal Sink) |
| 2 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | Error-Packet-1 | bad parity (Slow Sink) |
| 3 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | Error-Packet-1 | bad parity (Fast Sink) |
| 4 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {max} | Error-Packet-2 | bad parity (Nominal Sink) |
| 5 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | Error-Packet-2 | bad parity (Slow Sink) |
| 6 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | Error-Packet-2 | bad parity (Fast Sink) |

4.3.20 Link Layer Timing – Sink DUT Input: ACK

Measure the behavior of the Sink DUT as it receives ACK pulses with different timings.

4.3.20.1 CBT-Sink: Response to ACKs with Different Timings

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.20.1.1 Test Objective

Sink DUT receives Tester ACK at various times. Verify that Sink DUT recognizes ACK.

4.3.20.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.1; Table 13-23, T_{CBUS_ACK_0}

4.3.20.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.20.1.4 Required Methodology

1. Execute the following test procedure once for each row in Table 4-21, substituting TEST_BIT_TIME, TEST_ACK_TIME and TEST_ACK_DURATION.
2. If necessary, connect Sink DUT to Tester Source port.
3. Apply power to the Sink DUT.
4. Set Tester Source port to disconnected state.

5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
7. Set Tester Source to operate using TEST_BIT_TIME for T_{BIT_CBUS} .
8. Set Tester Source to place ACK pulses at the TEST_ACK_TIME after expected ACK period starts.
9. Set Tester Source to drive the ACK pulse LOW for TEST_ACK_DURATION.
10. Execute the Tester_Source_Discovery procedure.
11. Set Tester Source to send a GET_STATE to Sink DUT.
12. The Sink DUT may NACK the GET_STATE command. The tester shall re-issue this command up to N_{RETRY} times, to provide time for the Sink DUT to ACK the command.
13. FAIL if DUT does not eventually arbitrate for the bus to send a data packet to the Tester Source.
14. ACK the DUT using the calculated bit timing and the previously set ACK pulse placement.
15. FAIL if DUT does not try to reply with a data packet.
16. FAIL if DUT retries sending the data packet after the tester link level ACK. This indicates that the DUT is sensitive to the link level ACK where placed.
17. If all preceding steps pass, then PASS; else FAIL.

Table 4-21. Response to ACKs with Different Timings

| | TEST_BIT_TIME | TEST_ACK_TIME | TEST_ACK_DURATION | Purpose |
|----|------------------------|------------------------------|---------------------------|-----------------------------|
| 1 | $T_{BIT_CBUS}\{typ\}$ | $T_{CBUS_ACK_FALL}\{min\}$ | $T_{CBUS_ACK_O}\{min\}$ | Earliest possible thin ACK |
| 2 | $T_{BIT_CBUS}\{typ\}$ | $T_{CBUS_ACK_FALL}\{max\}$ | $T_{CBUS_ACK_O}\{min\}$ | Latest possible thin ACK |
| 3 | $T_{BIT_CBUS}\{typ\}$ | $T_{CBUS_ACK_FALL}\{min\}$ | $T_{CBUS_ACK_O}\{max\}$ | Earliest possible thick ACK |
| 4 | $T_{BIT_CBUS}\{typ\}$ | $T_{CBUS_ACK_FALL}\{max\}$ | $T_{CBUS_ACK_O}\{max\}$ | Latest possible thick ACK |
| 5 | $T_{BIT_CBUS}\{max\}$ | $T_{CBUS_ACK_FALL}\{min\}$ | $T_{CBUS_ACK_O}\{min\}$ | Earliest possible thin ACK |
| 6 | $T_{BIT_CBUS}\{max\}$ | $T_{CBUS_ACK_FALL}\{max\}$ | $T_{CBUS_ACK_O}\{min\}$ | Latest possible thin ACK |
| 7 | $T_{BIT_CBUS}\{max\}$ | $T_{CBUS_ACK_FALL}\{min\}$ | $T_{CBUS_ACK_O}\{max\}$ | Earliest possible thick ACK |
| 8 | $T_{BIT_CBUS}\{max\}$ | $T_{CBUS_ACK_FALL}\{max\}$ | $T_{CBUS_ACK_O}\{max\}$ | Latest possible thick ACK |
| 9 | $T_{BIT_CBUS}\{min\}$ | $T_{CBUS_ACK_FALL}\{min\}$ | $T_{CBUS_ACK_O}\{min\}$ | Earliest possible thin ACK |
| 10 | $T_{BIT_CBUS}\{min\}$ | $T_{CBUS_ACK_FALL}\{max\}$ | $T_{CBUS_ACK_O}\{min\}$ | Latest possible thin ACK |
| 11 | $T_{BIT_CBUS}\{min\}$ | $T_{CBUS_ACK_FALL}\{min\}$ | $T_{CBUS_ACK_O}\{max\}$ | Earliest possible thick ACK |
| 12 | $T_{BIT_CBUS}\{min\}$ | $T_{CBUS_ACK_FALL}\{max\}$ | $T_{CBUS_ACK_O}\{max\}$ | Latest possible thick ACK |

4.3.21 Link Layer Timing – Sink DUT Input: Bus Re-Arbitration

Measure the behavior of the Sink DUT as it responds to Source re-arbitration with different timings.

4.3.21.1 CBT-Sink: Sink receives Case 1 Back-to-Back Transfers

Application: Sink

4.3.21.1.1 Test Objective

Verify that Sink DUT can receive Case 1 Back-to-Back packets when connected to various speed Sinks using variable ARB-to-Sync Idle timing.

4.3.21.1.2 References

[MHL] Section 7.2.4; Case 1

4.3.21.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.21.1.4 Required Methodology

Execute the following test procedure once for each row in Table 4-22, substituting TEST_BIT_TIME and TEST_IDLE_PERIOD.

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to operate using TEST_BIT_TIME as $T_{\text{BIT_CBUS}}$.
6. Set Tester Source to use TEST_IDLE_PERIOD as delay from Arbitration pulse to Sync pulse.
7. Set Tester Source to use its Source Packet engine to ACK everything.
8. Execute the Tester_Source_Discovery procedure.
9. When the Tester Source receives a command, reply with a link level ACK.
10. Tester Source does SET_INT to Sink DUT, followed by Offset 0x20, followed by Data 0x00. The Offset packet has Parity == 0, the Data packet has Parity == 1. The last 2 Packets are sent back-to-back.
11. Tester Source switches to responding with ACK to everything.
12. Observe post-discovery Sink DUT link-level ACK activity as it replies to the Tester Source command.
13. Observe that the Sink DUT replies with an ACK packet.
14. FAIL if DUT does not eventually send a command.
15. FAIL if DUT does not link level ACK all 3 packets sent from the Tester.
16. FAIL if DUT does not respond with an ACK packet.
17. If all preceding steps pass, then PASS; else FAIL.

Note that the Sink DUT might NACK the packets it receives. Tester needs to re-issue this command up to N_{RETRY} times to give the Sink DUT a chance to ACK it.

Table 4-22. Sink Receives Back-to-Back Transfers

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | Purpose |
|---|--------------------------------------|--------------------------------------|--|
| 1 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | nominal Source Bit Timing, shortest ARB to SYNC |
| 2 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | nominal Source Bit Timing, longest ARB to SYNC delay |
| 3 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | minimum Source Bit Timing, shortest ARB to SYNC |
| 4 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | minimum Source Bit Timing, longest ARB to SYNC delay |
| 5 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | maximum Source Bit Timing, shortest ARB to SYNC |
| 6 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | maximum Source Bit Timing, longest ARB to SYNC delay |

4.3.22 Link Layer Timing – Sink DUT Input: Ill-formed Packets

Verify that the Sink does not have problems when the Source sends ill-formed packets.

4.3.22.1 CBT-Sink: Source Wins Arbitration, then No Packet, when Sink is Idle

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.22.1.1 Test Objective

Verify that Sink DUT is not confused if the Source arbitrates for the CBUS but then does not use it.

4.3.22.1.2 References

[MHL] Section 7.2.4

4.3.22.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.22.1.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a SET_HPD, reply with an ACK packet.
9. Set Tester Source to occasionally drive a total of 100 CBUS LOW pulses for periods of 4000 nanoseconds (4 nominal bit times), followed by drive HIGH as usual. These are valid Sink-side arbitration pulses, but are not followed by Sync pulses.
10. Set Tester Source to wait until the bogus arbitration test period ends, then do a READ_DEVCAP followed by Offset 0x00.
11. FAIL if DUT responds in any way to the bad arbitration activity.
12. FAIL if DUT cannot respond to the valid READ_DEVCAP command from the Tester.
13. If all preceding steps pass, then PASS; else FAIL.

4.3.23 Link Layer Timing – Sink DUT Input: Disconnect

Verify that the Sink responds appropriately when the MHL bus is disconnected (or when the Source pretends that the bus is disconnected).

4.3.23.1 CBT-Sink: CBUS LOW for Less than Minimum $T_{SINK_CBUS_DISCONN}$ Reject Time

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.23.1.1 Test Objective

Verify that Sink DUT ignores LOW-asserted pulses less than $T_{SINK_CBUS_DISCONN}\{min\}$ on the CBUS.

4.3.23.1.2 References

[MHL] Section 13.10.1.2; Table 13-32; $T_{SINK_CBUS_DISCONN}$

4.3.23.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.23.1.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.

6. Execute the Tester_Source_Discovery procedure.
7. When the Tester Source receives a SET_HPDP, reply with an ACK packet.
8. Service other packets from the Sink DUT if they arrive.
9. When bus is idle, set Tester Source to drive CBUS LOW for $T_{\text{SINK_CBUS_DISCONN}}\{\text{min}\} - 1 \text{ uSec}$.
10. FAIL if DUT has trouble doing Discovery.
11. FAIL if DUT does not eventually send SET_HPDP.
12. FAIL if DUT does not eventually become idle within 10 seconds of Discovery, leaving a constant CBUS HIGH.
13. FAIL if DUT detects the Glitch as a Disconnect event. Sink DUT disconnect is inferred by the Source detecting the MHL+ and MHL- wires changing to a LOW voltage level, under the influence of the Source-side pull down resistors.
14. If all preceding steps pass, then PASS; else FAIL.

4.3.23.2 CBT-Sink: CBUS LOW for Greater than Maximum $T_{\text{SINK_CBUS_DISCOVER}}$ Reject Time

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.23.2.1 Test Objective

Verify that Sink DUT detects wide LOW-asserted pulses on the CBUS as a disconnect event.

Verify that Sink DUT quickly disables its MHL+/- pull-ups when it sees a CBUS disconnect.

Verify that Sink DUT quickly changes its pulldown to $Z_{\text{CBUS_SINK_DISCOVER}}$ when it sees a CBUS disconnect.

4.3.23.2.2 References

[MHL] Section 13.10.1.2; Table 13-32; $T_{\text{SINK_CBUS_DISCONN}}$

[MHL] Section 13.10.1.2; Table 13-32; $T_{\text{SINK_RXSENSE_DIS}}$

[MHL] Section 13.10.1.2; Table 13-32; $T_{\text{SINK:CBUS_SINK_CONN}}$

4.3.23.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.23.2.4 Required Methodology

1. If necessary, connect Sink DUT to Tester Source port.
2. Apply power to the Sink DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. When the Tester Source receives a SET_HPDP, reply with an ACK packet.
8. Service other packets from the Sink DUT if they arrive.
9. Set Tester Source to drive CBUS LOW for $T_{\text{SINK_CBUS_DISCONN}}\{\text{max}\} + 1 \text{ uSec}$.
10. Sink DUT disconnect is inferred by seeing the MHL+/- wires go LOW under the influence of the Source-side pulldowns.
11. FAIL if DUT does not eventually send SET_HPDP.
12. FAIL if DUT does not detect the Pulse LOW as a Disconnect event.
13. FAIL if DUT does not remove its pull-ups on MHL+/- within $T_{\text{SINK_RXSENSE_DIS}}$ (5 mSec).
14. FAIL if DUT does not switch to $Z_{\text{CBUS_SINK_DISCOVER}}$ within $T_{\text{SINK:CBUS_SINK_CONN}}$ (5 mSec).
15. If all preceding steps pass, then PASS; else FAIL.

4.3.24 Link Layer Electrical – Sink DUT VBUS Output

Verify that a Sink drives the VBUS as required.

4.3.24.1 CBE-Sink: Sink DUT VBUS Output

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.24.1.1 Test Objective

Verify that a Sink supplies power over the VBUS as required.

4.3.24.1.2 References

[MHL] Section 9.1

[MHL] Section 13.9.1.1; Table 13-21; $V_{VBUS_TP1_SINK_DRV}$

[MHL] Section 13.9.1.2; Table 13-30: $I_{VBUS_PREDISCOVERY}$, $I_{VBUS_SINK_TO_SOURCE}$

4.3.24.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.24.1.4 Required Methodology

1. If the CDF value CDF_SINK_DC indicates that the Sink is a DC powered Sink and if the CDF value CDF_SINK_POWERED also indicates that Sink DUT provides no VBUS output, then end the test with PASS (SKIP); else continue.
2. If the Sink DUT has an external power source accessory, connect the DUT to active power now.
3. Disconnect Sink DUT from Tester Source port.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Tester does not drive VBUS.
7. Set Tester Source to apply an $I_{VBUS_PREDISCOVERY}\{max\}$ load (100mA) to the VBUS.
8. Connect Sink DUT to Tester Source port using 20cm MHL Test Cable.
9. Set Sink DUT into Active State based on CDF procedure CDF_PROC_SET_ACTIVE.
10. Tester Source measures the VBUS driven by the Sink and loaded by the Tester Source.
11. If VBUS Voltage is not within the range $V_{VBUS_TP1_SINK_DRV}\{min\}$ to $V_{VBUS_TP1_SINK_DRV}\{max\}$, then FAIL.
12. Execute the Tester_Source_Discovery procedure.
13. Execute the Tester_Wait_Capability_Registers_Valid procedure.
14. If the procedure indicates TIMEOUT, then FAIL.
15. Tester Source reads Sinks DEV_CAT register.
16. FAIL if DEV_CAT register not read as SINK, and immediately terminate the test.
17. FAIL if DEV_CAT POW bit is not '1'.
18. If {PLIM1, PLIM0} are {0, 1}, set Tester Source to apply $I_{VBUS_SINK_TO_SOURCE}\{min\}$ load (900mA) to the VBUS.
 - a. Else if {PLIM1, PLIM0} are {1,0}, set Tester Source to apply a load of 1500mA to the VBUS.
 - b. Else if the CDF value CDF_SINK_DC indicates that Sink DUT is DC powered, then continue; else end test with a FAIL.
19. If {PLIM1, PLIM0} are {0,0}, set Tester Source to apply a load of 500mA to the VBUS.
20. If {PLIM1, PLIM0} are {1,1}, set Tester Source to apply a load of 100mA to the VBUS.
21. If VBUS Voltage is within the range $V_{VBUS_TP1_SINK_DRV}\{min\}$ to $V_{VBUS_TP1_SINK_DRV}\{max\}$, then continue; else end test with FAIL.
22. If all preceding steps pass, then PASS; else FAIL.

4.3.25 Link Layer Timing – Sink DUT VBUS Turn On Transition

Verify that a connected Sink DUT VBUS behaves correctly when a source is connected.

4.3.25.1 CBT-Sink: Sink DUT VBUS Turn On Transition Powered Sink

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

4.3.25.1.1 Test Objective

Verify that an active powered Sink DUT VBUS timing behaves correctly when connected.

4.3.25.1.2 References

[MHL] Section 8.2.2; Table 8-5; SINK0, SINK1, SINK2

[MHL] Section 13.10.1; Table 13-26; $Z_{CBUS_SINK_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SINK_VBUS_EN}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SRC_VBUS_CBUS_STABLE}$

4.3.25.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

4.3.25.1.4 Required Methodology

1. If the CDF value CDF_SINK_POWERED indicates that Sink DUT provides VBUS output, then continue; else end test with PASS (SKIP).
2. Disconnect Sink DUT from Tester Source port MHL cable at Sink side TP4-TP2.
3. Put Sink into non-standby state (SINK0) as specified in the CDF procedure **CDF_PROC_SET_ACTIVE**.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Tester Source does not drive VBUS.
7. Set Tester Source to apply an $I_{VBUS_PREDISCOVERY}\{max\}$ load (100mA) to the VBUS.
8. Execute the Tester_Source_Discovery procedure.
9. Connect Sink DUT to Tester Source port by reconnecting MHL cable at TP4-TP2.
10. FAIL if Sink DUT does not drive VBUS earlier than $T_{SINK_VBUS_EN}$ after asserting $Z_{CBUS_SINK_DISCOVER}$ the final time.
11. Tester then waits $T_{SRC_VBUS_CBUS_STABLE}\{max\}$.
12. FAIL if VBUS Voltage is not within the range $V_{VBUS_TP1_SINK_DRV}\{min\}$ to $V_{VBUS_TP1_SINK_DRV}\{max\}$.
13. If all preceding steps pass, then PASS; else FAIL.

5 Dongle Test

Dongles may have a male (Type-M) or female (Type-F) connector on the downstream, non-MHL side but a male type connector (plug) is attached on the upstream side for both types of Dongle device. Thus the same type of TPA board will be used for both Type-M and Type F Dongle devices.

5.1 Electrical Tests

5.1.1 TMD5 Electrical Tests

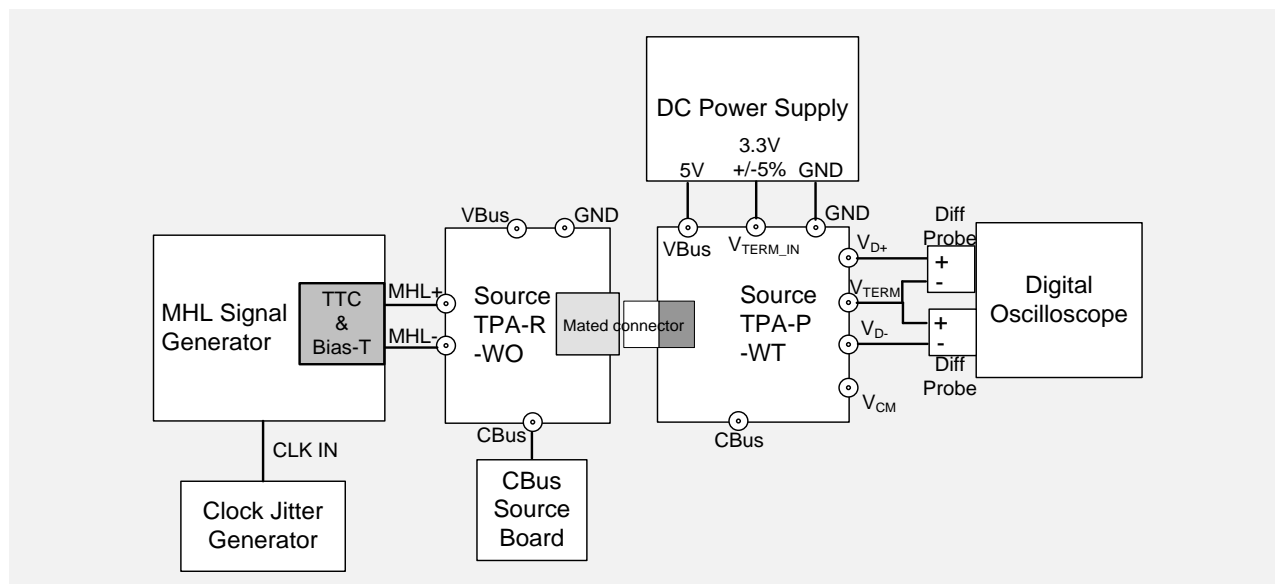


Figure 42. Single-Ended Signals Calibration Setup for Dongle Tests

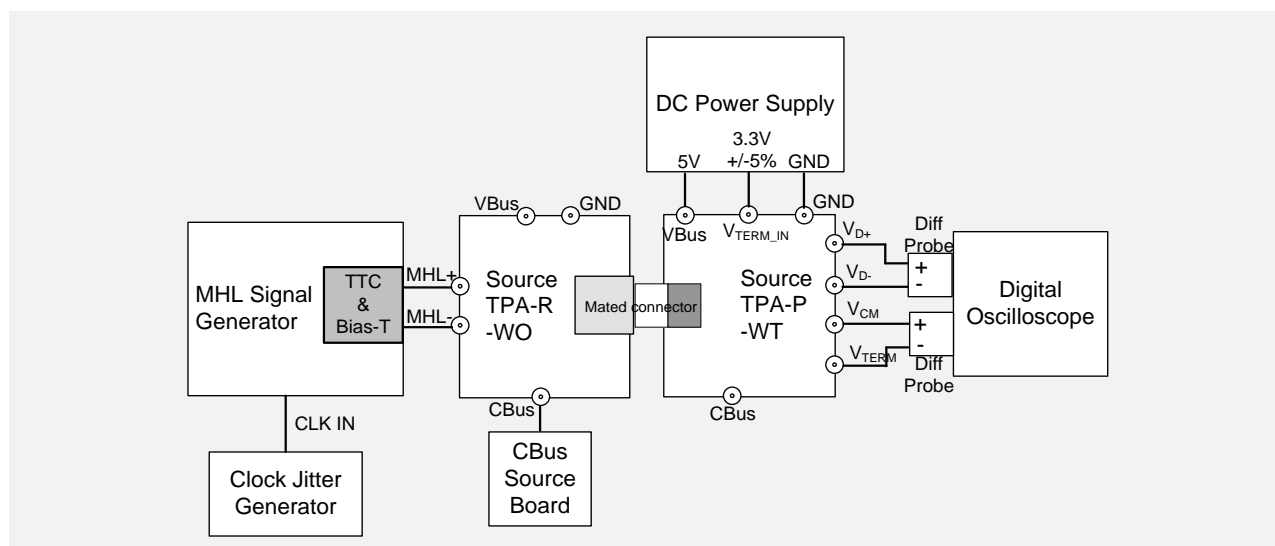


Figure 43. Differential & Common-Mode Signals Calibration Setup for Dongle Tests

5.1.1.1 Input Signal Single-Ended Voltage Level Tolerance

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.1.1.1.1 Test Objective

This test confirms that the dongle device supports input signal single-ended voltage level allowed by the specification.

5.1.1.1.2 References

[MHL] Section 13.7.1 Table 13-10 Dongle DC Characteristics at TP3:

$$V_{\text{TERM}} - 540\text{mV} \leq V_{\text{SE_HIGH}} \leq V_{\text{TERM}} + 10\text{mV}$$

$$V_{\text{TERM}} - 1760\text{mV} \leq V_{\text{SE_LOW}} \leq V_{\text{TERM}} - 700\text{mV}$$

5.1.1.1.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

Differential Probe

Transition Time Converter (TTC)

Bias-Tee

MHL Signal Generator

CBUS Source Board

Source TPA-P-WT

Source TPA-R-WO

DC Power Supply

A/V Display

5.1.1.1.4 Required Methodology

1. Connect the equipment as shown in Figure 43 and set $V_{\text{TERM}} = 3.3\text{V}$.
2. Calibrate the MHL Signal Generator output for the following settings.
 - a. Frequency: Highest MHL data bit rate supported by the Dongle DUT
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps
Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - c. Swing voltages: Differential signal 800mV, Common-mode signal 540mV
3. Connect the equipment as shown in Figure 42. Adjust the DC level of the MHL Signal Generator so that $V_{\text{SE_HIGH}} = V_{\text{TERM}} + 10\text{ mV}$, $V_{\text{SE_HIGH}} = V_{\text{TERM}} - 540\text{ mV}$, $V_{\text{SE_LOW}} = V_{\text{TERM}} - 700\text{ mV}$ and $V_{\text{SE_LOW}} = V_{\text{TERM}} - 1760\text{ mV}$. Record the DC level setting for each single-ended value setting.
4. Connect the equipment as shown in Figure 44. Provide VBus if DUT is unpowered dongle. Do not provide VBus if DUT is powered dongle.
5. Turn on the DUT and enable the MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
6. Adjust the DC level of the MHL Signal Generator output so that $V_{\text{SE_HIGH}} = V_{\text{TERM}} + 10\text{ mV}$.
7. Examine the output of the Dongle DUT on the A/V Display.
8. Record errors on the display if any for the given V_{TERM} , MHL data bit rate and single-ended voltage.

9. Adjust the DC level of the output so that $V_{SE_HIGH} = V_{TERM} - 540 \text{ mV}$.
10. Examine the output of the Dongle DUT on the A/V Display.
11. Record errors on the display if any for the given V_{TERM} , MHL data bit rate and single-ended voltage.
12. Adjust the DC level of the output so that $V_{SE_LOW} = V_{TERM} - 700 \text{ mV}$.
13. Examine the output of the Dongle DUT on the A/V Display.
14. Record errors on the display if any for the given V_{TERM} , MHL data bit rate and single-ended voltage.
15. Adjust the DC level of the output so that $V_{SE_LOW} = V_{TERM} - 1760 \text{ mV}$.
16. Examine the output of the Dongle DUT on the A/V Display.
17. Record errors on the display if any for the given V_{TERM} , MHL data bit rate and single-ended voltage.
18. If there is no errors on the display in all recorded results, then PASS. Otherwise FAIL.

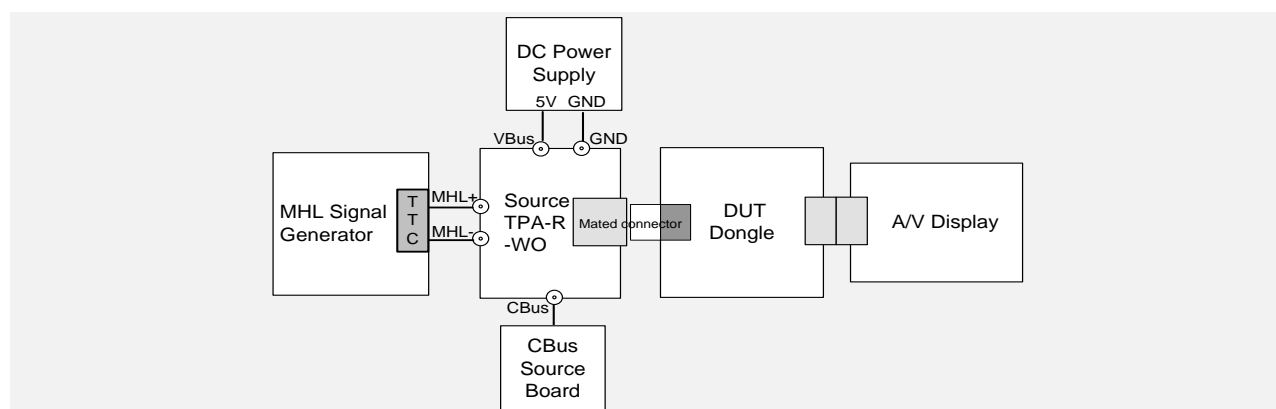


Figure 44. Dongle – V_{SE_HIGH} and V_{SE_LOW} Tolerance Test Setup

5.1.1.2 Input Signal Minimum and Maximum Swing Voltages Tolerance

Application: Dongle

5.1.1.2.1 Test Objective

This test confirms that the dongle device supports input signal minimum and maximum swing voltages allowed by the specification.

5.1.1.2.2 References

[MHL] Section 13.7.1 Table 13-10 Dongle DC Characteristics at TP3:

$$600\text{mV} \leq V_{DFSWING} \leq 1000\text{mV}$$

$$360\text{mV} \leq V_{CMSWING} \leq \text{Min} (720\text{mV}, 0.85V_{IDF})$$

5.1.1.2.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

Differential Probe

Transition Time Converter (TTC)

Bias-Tee

MHL Signal Generator

CBUS Source Board

Source TPA-P-WT

Source TPA-R-WO

DC Power Supply

A/V Display

5.1.1.2.4 Required Methodology

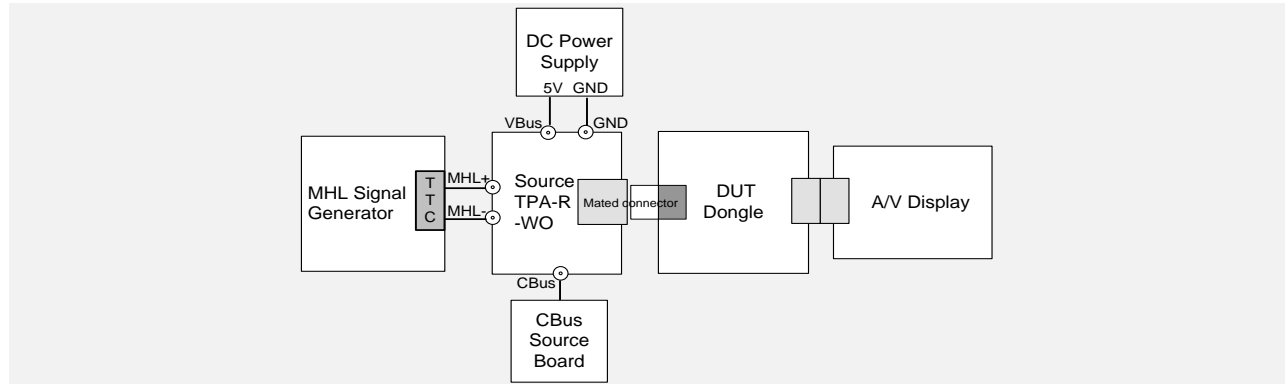


Figure 45. Dongle – V_{DFSWING} and V_{CMSWING} Tolerance Test Setup

1. Connect the equipment as shown in Figure 42 for single-ended voltage calibration and Figure 43 for differential mode and common mode voltages calibration. Set $V_{\text{TERM}} = 3.3\text{V}$.
2. Calibrate the MHL Signal Generator output for the following settings.
 - a. Frequency: Highest MHL data bit rate supported by the Dongle DUT
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps
Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
3. Adjust the DC level and swing amplitudes of the MHL Signal Generator so that $V_{\text{SE_HIGH}} = V_{\text{TERM}} + 10\text{ mV}$, $V_{\text{DFSWING}} = 600\text{ mV}$ and $V_{\text{CMSWING}} = 360\text{ mV}$. Record the DC level and swing amplitude settings.
4. Adjust the DC level and swing amplitudes of the MHL Signal Generator so that $V_{\text{SE_LOW}} = V_{\text{TERM}} - 1760\text{ mV}$, $V_{\text{DFSWING}} = 1000\text{ mV}$ and $V_{\text{CMSWING}} = 720\text{ mV}$. Record the DC level and swing amplitude settings.
5. Connect the equipment as shown in Figure 45. Provide VBus if DUT is unpowered dongle. Do not provide VBus if DUT is powered dongle.
6. Turn on the DUT and enable the MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
7. Adjust the DC level and swing amplitudes of the MHL Signal Generator so that $V_{\text{SE_HIGH}} = V_{\text{TERM}} + 10\text{ mV}$, $V_{\text{DFSWING}} = 600\text{ mV}$ and $V_{\text{CMSWING}} = 360\text{ mV}$.
8. Examine the output of the Dongle DUT on the A/V Display.
 - Record errors on the display if any for the given V_{TERM} , MHL data bit rate and swing voltages.
9. Adjust the DC level and swing amplitudes of the MHL Signal Generator so that $V_{\text{SE_LOW}} = V_{\text{TERM}} - 1760\text{ mV}$, $V_{\text{DFSWING}} = 1000\text{ mV}$ and $V_{\text{CMSWING}} = 720\text{ mV}$.
10. Examine the output of the Dongle DUT on the A/V Display.
 - Record errors on the display if any for the given V_{TERM} , MHL data bit rate and swing voltages.
11. If there is no errors on the display in all recorded results, then PASS. Otherwise FAIL.

5.1.1.3 Intra-Pair Skew Tolerance

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.1.1.3.1 Test Objective

This test confirms that the dongle device can tolerate the maximum intra-pair skew allowed by the specification.

5.1.1.3.2 References

[MHL] Section 13.7.2 Table 13-11 Dongle AC Characteristics at TP3:

$$T_{\text{SKEW_DF}} \leq \text{Min}(0.12T_{\text{BIT}}, 50\text{ps})$$

$$T_{\text{SKEW_CM}} \leq \text{Min}(0.12T_{\text{BIT}}, 50\text{ps})$$

5.1.1.3.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

Differential Probe

Transition Time Converter (TTC)

Bias-Tee

MHL Signal Generator

CBUS Source Board

Source TPA-P-WT

Source TPA-R-WO

DC Power Supply

A/V Display

5.1.1.3.4 Required Methodology

1. Connect the equipment as shown in Figure 42 for single-ended voltage calibration and Figure 43 for differential mode and common mode voltages calibration. Set $V_{\text{TERM}} = 3.3\text{V}$.
2. Calibrate the Signal Generator output for the following settings.
 - a. Frequency: Highest MHL data bit rate supported by the Dongle DUT
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps
Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - c. Swing voltages: Differential signal 800mV, Common-mode signal 540mV
 - d. DC level of the output single-ended signal: $V_{\text{TERM}} - 750\text{mV}$
 - e. Intra-pair skew: Differential skew 0ps, Common-mode skew 0ps
3. Connect the equipment as shown in Figure 46. Provide VBus if DUT is unpowered dongle. Do not provide VBus if DUT is powered dongle.
4. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
5. Examine the output of the Dongle DUT on the A/V Display.
6. Confirm that there is no error on the display. If there is any error, STOP. The test FAILED.

7. Increase the intra-pair skew of differential signal and common-mode signal by the same amount to the maximum where error appears on the display. Make the positive channel leads the negative channel in time, which is called positive skew direction.
8. Record the maximum intra-pair skew for the given MHL data bit rate and skew direction.
9. Set the differential and common-mode intra-pair skews to 0ps.
10. Increase the intra-pair skew of differential signal and common-mode signal by the same amount to the maximum where error appears on the display. Make the positive channel lags the negative channel in time, which is called negative skew direction.
11. Record the maximum intra-pair skew for the given MHL data bit rate and skew direction.

If the all recorded maximum intra-pair skew value is larger than the spec limit, then PASS. Otherwise FAIL.

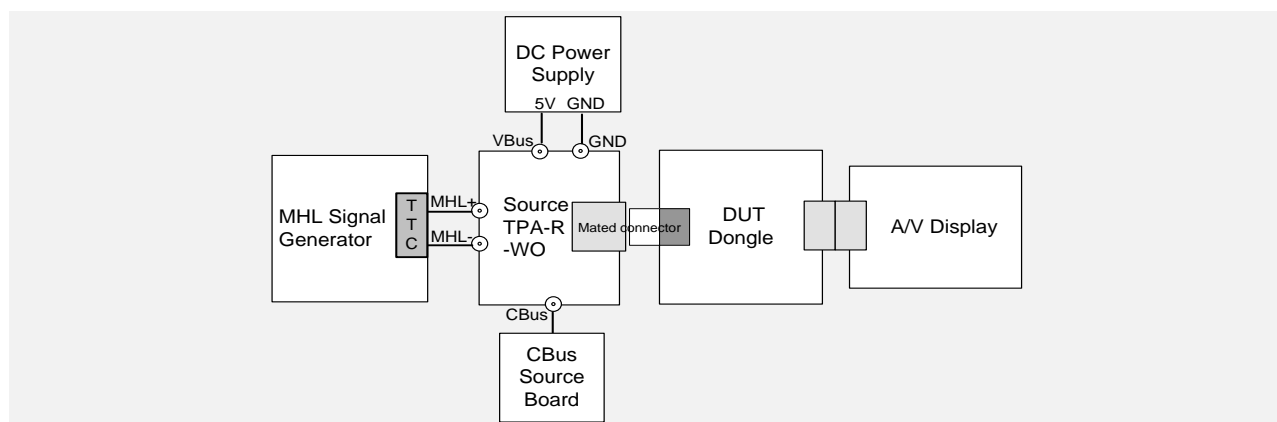


Figure 46. Dongle – T_{SKEW_DF} and T_{SKEW_CM} Tolerance Test Setup

5.1.1.4 Jitter Tolerance in Normal Mode

Application: Dongle

5.1.1.4.1 Test Objective

This test confirms that the dongle device can tolerate the maximum clock and data jitter amounts allowed by the specification in Normal Mode.

5.1.1.4.2 References

- [MHL] Section 13.5.2 Figure 13-13 MHL Clock Jitter at TP1
- [MHL] Section 13.5.3 Figure 13-14 Differential Eye Diagram Mask at TP1
- [MHL] Section 13.5.3 Figure 13-15 MHL Data Jitter at TP1

5.1.1.4.3 Required Test Equipment

High-bandwidth Digital Oscilloscope
Differential Probe
Jitter and Eye Analyzer
Clock Jitter Generator
Transition Time Converter (TTC)
Bias-Tee
MHL Signal Generator
CBUS Source Board

Source TPA-P-WT
Source TPA-R-WO
DC Power Supply
A/V Display

5.1.1.4.4 Required Methodology

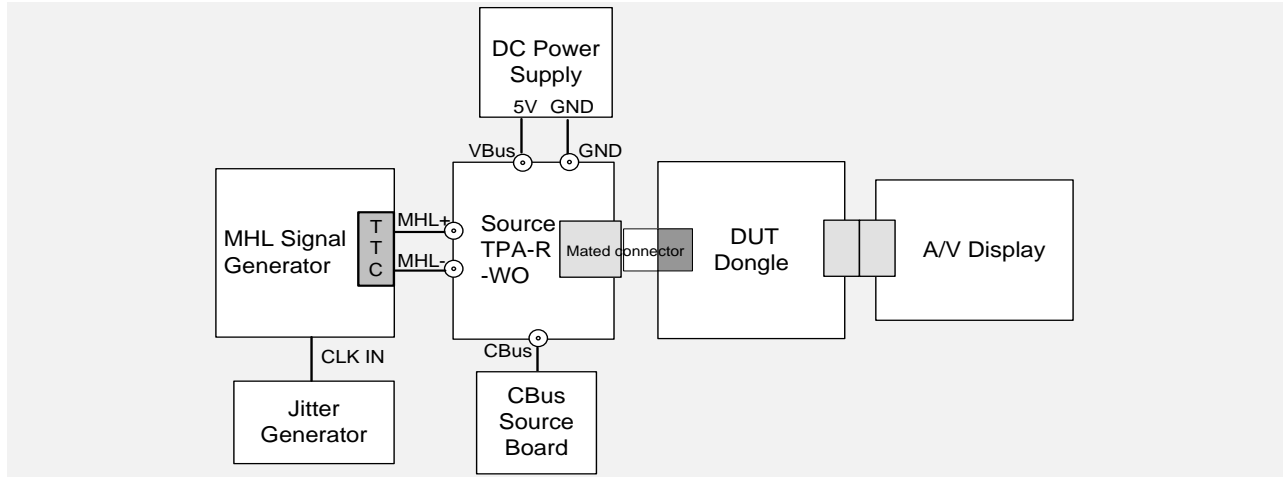


Figure 47. Dongle – Jitter Tolerance Test Setup

1. Connect the equipment as shown in Figure 42 for single-ended voltage calibration and Figure 43 for differential mode and common mode voltages calibration. Set $V_{\text{TERM}} = 3.3\text{V}$.
2. Calibrate the Signal Generator output for the following settings.
 - a. Frequency: Lowest MHL data bit rate supported by the Dongle DUT in Normal Mode
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps
Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - d. Swing voltages: Differential signal 800mV, Common-mode signal 360mV
 - d. DC level of the output single-ended signal: $V_{\text{TERM}} - 540\text{mV}$
3. Set the clock jitter frequency to 10MHz and the data jitter frequency to 500KHz.
4. Calibrate the Signal Generator and Clock Jitter Generator to get data eye diagram and clock jitter which are marginal to TP1 specification.
 - a. Adjust clock jitter and data jitter to the TP1 maximum amount specified in MHL Spec Figures 13-13 and 13-15 for the given clock frequency.
 - b. Adjust the differential signal swing to the marginal eye opening specified in MHL Spec Figure 13-14.
5. Connect the equipment as shown in Figure 47. Provide VBus if DUT is unpowered dongle. Do not provide VBus if DUT is powered dongle.
6. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
7. Set the video mode to a mode which uses Normal Mode.
8. Examine the output of the Dongle DUT on the A/V Display and record error on the display if any.

9. Change the common-mode clock edge location by $0.1T_{\text{bit}}$ step with respect the differential data edge for 10 times and examine the A/V Display at each step.
10. Record error on the display if any for the given V_{TERM} , clock frequency, jitter frequency and clock edge location.
11. Connect the equipment as shown in Figure 43.
12. Set the clock jitter frequency to 7MHz and the data jitter frequency to 1MHz.
13. Repeat steps 4 to 10 for clock jitter frequency 7MHz and data jitter frequency 1MHz.
14. Repeat steps 2 to 13 for the highest supported MHL data bit rate in Normal Mode.
15. If there is no error on the display in all recorded results, then PASS. Otherwise FAIL.

5.1.1.5 Differential Impedance

Application: Dongle

5.1.1.5.1 Test Objective

This test confirms that the differential input impedance of the dongle device is within the range allowed by the specification.

5.1.1.5.2 References

[MHL] Section 13.7.3 Table 13-12 Dongle Device Impedance Characteristics at TP3:

5.1.1.5.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR
CBUS Source Board
Source TPA-R-WO
DC Block
DC Power Supply
SMA Coaxial Cable

5.1.1.5.4 Required Methodology

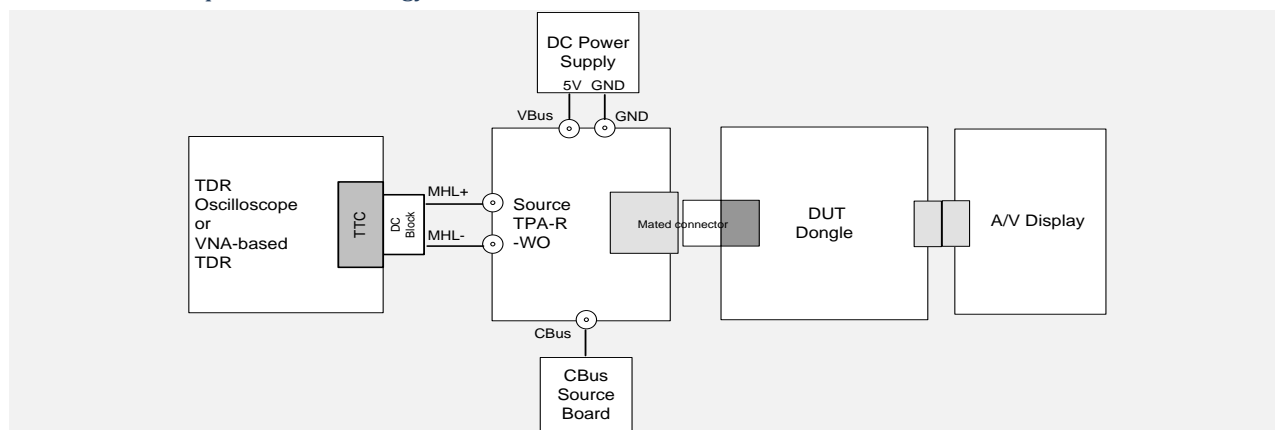


Figure 48. Dongle – Differential Impedance Test Setup

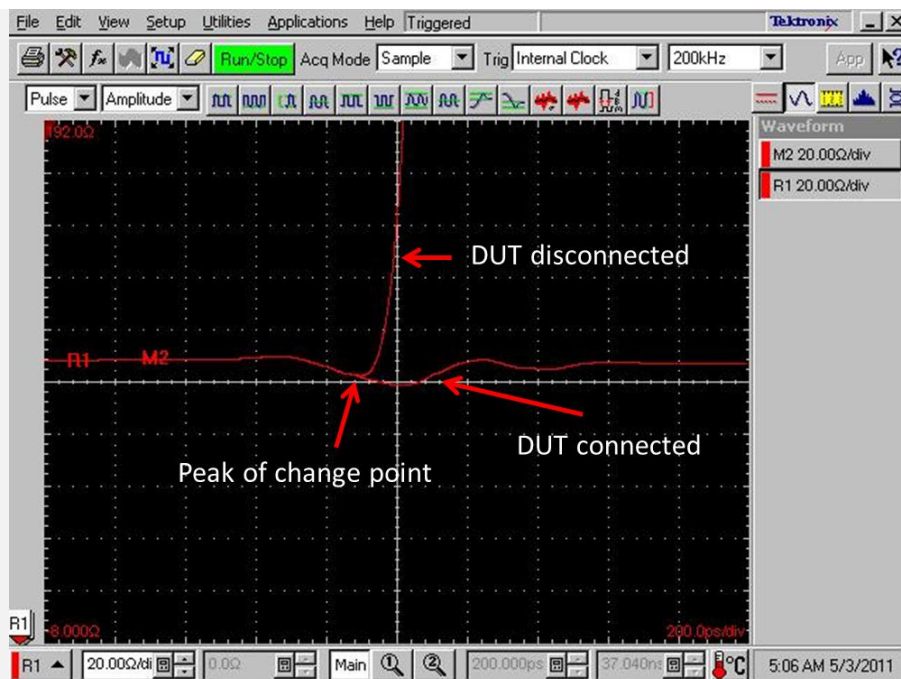


Figure 49. Set reference point for differential mode

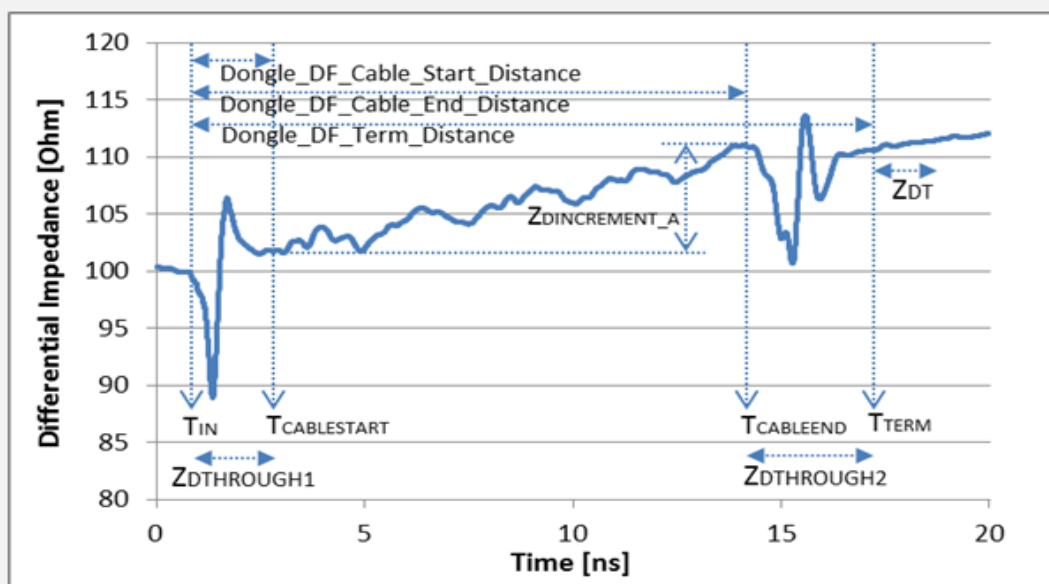


Figure 50. Dongle – Differential Impedance Test Symbolology

Before testing, compensate the differential intra-pair skew caused by the test fixtures (TPA boards and co-ax cables).

1. Connect the Source TPA-R-WO to the TDR oscilloscope through a DC block.
2. Set the effective rise time of the differential TDR pulse to 200ps (20 – 80%).

3. Record the time at the peak of change point in Figure 49. This is the start point of the Dongle DUT input connector (T_{IN}).
4. Calculate $T_{TERM} = T_{IN} + CDF_D_DF_TERM_DISTANCE$. T_{TERM} is the dongle termination point for differential signals. **CDF_D_DF_TERM_DISTANCE** is obtained from CDF of the Dongle DUT.
5. Calculate $T_{CABLESTART}$ and $T_{CABLEEND}$. $T_{CABLESTART}$ and $T_{CABLEEND}$ are the start and end points of the cable portion in Dongle for differential signals.
 - a. If ($T_{TERM} \leq T_{IN} + 6ns$), $T_{CABLESTART} = T_{CABLEEND} = T_{IN} + (T_{TERM} - T_{IN}) / 2$.
 - b. If ($T_{TERM} > T_{IN} + 6ns$), $T_{CABLESTART} = T_{IN} + CDF_D_DF_CABLE_START_DISTANCE$ and $T_{CABLEEND} = T_{IN} + CDF_D_DF_CABLE_END_DISTANCE$. **CDF_D_DF_CABLE_START_DISTANCE** and **CDF_D_DF_CABLE_END_DISTANCE** are obtained from CDF of the Dongle DUT.
6. Connect the equipment as shown in Figure 48. Provide VBus if DUT is unpowered dongle. Do not provide VBus if DUT is powered dongle.
7. Turn on the Dongle DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
8. Record the differential impedance increment $Z_{DINCREMENT_A}$ from $T_{CABLESTART}$ to $T_{CABLEEND}$.
9. Calculate $Z_{DINCREMENT_B}$.
 - a. If ($Z_{DINCREMENT_A} < 0$), $Z_{DINCREMENT_B} = 0$.
 - b. If ($Z_{DINCREMENT_A} \geq 0$), $Z_{DINCREMENT_B} = Z_{DINCREMENT_A}$.
10. Measure the differential impedance $Z_{DTHROUGH1}$ from T_{IN} to $T_{CABLESTART}$ and $Z_{DTHROUGH2_WITHCABLE}$ from $T_{CABLEEND}$ to T_{TERM} .
11. Calculate $Z_{DTHROUGH2} = Z_{DTHROUGH2_WITHCABLE} - Z_{DINCREMENT_B}$.
12. Measure the differential impedance $Z_{DT_WITHCABLE}$ from T_{TERM} to $T_{TERM} + 1ns$.
13. Calculate $Z_{DT} = Z_{DT_WITHCABLE} - Z_{DINCREMENT_B}$.
14. Evaluate $Z_{DTHROUGH1}$, $Z_{DTHROUGH2}$, and Z_{DT} .
 - a. If ($85\text{ ohms} \leq [Z_{DTHROUGH1}, Z_{DTHROUGH2}] \leq 115\text{ ohms}$) AND ($90\text{ ohms} \leq Z_{DT} \leq 110\text{ ohms}$), it is PASS
 - b. If ($65\text{ ohms} \leq [Z_{DTHROUGH1}, Z_{DTHROUGH2}] < 85\text{ ohms}$) OR ($115\text{ ohms} < [Z_{DTHROUGH1}, Z_{DTHROUGH2}] \leq 125\text{ ohms}$), the impedance is in the excursion range. If an excursion occurs no more than one time for $Z_{DTHROUGH1}$ and no more than one time for $Z_{DTHROUGH2}$ AND the duration of each excursion is less than 350ps AND ($90\text{ ohms} \leq Z_{DT} \leq 110\text{ ohms}$), it is PASS.
 - c. All other cases are FAIL.

5.1.1.6 Common-Mode Impedance

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.1.1.6.1 Test Objective

This test confirms that the common-mode input impedance of the dongle device is within the range allowed by the specification.

5.1.1.6.2 References

[MHL] Section 13.7.3 Table 13-12 Dongle Device Impedance Characteristics at TP3:

5.1.1.6.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR

CBUS Source Board

Source TPA-R-WO

DC Block
DC Power Supply
SMA Coaxial Cable

5.1.1.6.4 Required Methodology

Before testing, compensate the common-mode intra-pair skew caused by the test fixtures (TPA boards and co-ax cables).

1. Connect the Source TPA-R-WO to the TDR oscilloscope through a DC block.
2. Set the effective rise time of the common-mode TDR pulse to 600ps (20 – 80%).
3. Record the time at the peak of change point in Figure 51. This is the start point of the Dongle DUT input connector (T_{IN}).
4. Calculate $T_{TERM} = T_{IN} + CDF_D_CM_TERM_DISTANCE$. T_{TERM} is the dongle termination point for common-mode signals. **CDF_D_CM_TERM_DISTANCE** is obtained from CDF of the Dongle DUT.
5. Calculate $T_{CABLESTART}$ and $T_{CABLEEND}$. $T_{CABLESTART}$ and $T_{CABLEEND}$ are the start and end points of the cable portion in Dongle for common-mode signals.
 - a. If ($T_{TERM} \leq T_{IN} + 10ns$), $T_{CABLESTART} = T_{CABLEEND} = T_{IN} + (T_{TERM} - T_{IN}) / 2$.
 - b. If ($T_{TERM} > T_{IN} + 10ns$), $T_{CABLESTART} = T_{IN} + CDF_D_CM_CABLE_START_DISTANCE$ and $T_{CABLEEND} = T_{IN} + CDF_D_CM_CABLE_END_DISTANCE$. **CDF_D_CM_CABLE_START_DISTANCE** and **CDF_D_CM_CABLE_END_DISTANCE** are obtained from CDF of the Dongle DUT.
6. Connect the equipment as shown in Figure 52. Provide VBus if DUT is unpowered dongle. Do not provide VBus if DUT is powered dongle.
7. Turn on the Dongle DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
8. Record the common-mode impedance increment $Z_{CINCREMENT_A}$ from $T_{CABLESTART}$ to $T_{CABLEEND}$.
9. Calculate $Z_{CINCREMENT_B}$.
 - a. If ($Z_{CINCREMENT_A} < 0$), $Z_{CINCREMENT_B} = 0$.
 - b. If ($Z_{CINCREMENT_A} \geq 0$), $Z_{CINCREMENT_B} = Z_{CINCREMENT_A}$.
10. Measure the common-mode impedance $Z_{CTHROUGH1}$ from T_{IN} to $T_{CABLESTART}$ and $Z_{CTHROUGH2_WITHCABLE}$ from $T_{CABLEEND}$ to T_{TERM} .
11. Calculate $Z_{CTHROUGH2} = Z_{CTHROUGH2_WITHCABLE} - Z_{CINCREMENT_B}$.
12. Measure the common-mode impedance $Z_{CT_WITHCABLE}$ from T_{TERM} to $T_{TERM} + 3 ns$.
13. Calculate $Z_{CT} = Z_{CT_WITHCABLE} - Z_{CINCREMENT_B}$.
14. Evaluate $Z_{CTHROUGH1}$, $Z_{CTHROUGH2}$, and Z_{CT} .
 - a. If ($24\text{ ohms} \leq [Z_{CTHROUGH1}, Z_{CTHROUGH2}] \leq 36\text{ ohms}$) AND ($25\text{ ohms} \leq Z_{CT} \leq 35\text{ ohms}$), it is PASS
 - b. If ($20\text{ ohms} \leq [Z_{CTHROUGH1}, Z_{CTHROUGH2}] < 24\text{ ohms}$) OR ($36\text{ ohms} < [Z_{CTHROUGH1}, Z_{CTHROUGH2}] \leq 40\text{ ohms}$), the impedance is in the excursion range. If an excursion occurs no more than one time for $Z_{CTHROUGH1}$ and no more than one time for $Z_{CTHROUGH2}$ AND the duration of each excursion is less than 500ps AND ($25\text{ ohms} \leq Z_{CT} \leq 35\text{ ohms}$), it is PASS.
 - c. If input captive cable length does not exceed 30 cm AND ($24\text{ ohms} \leq [Z_{CTHROUGH1}, Z_{CTHROUGH2}] \leq 140\text{ ohms}$) AND ($25\text{ ohms} \leq Z_{CT} \leq 35\text{ ohms}$), it is PASS.
 - d. All other cases are FAIL.

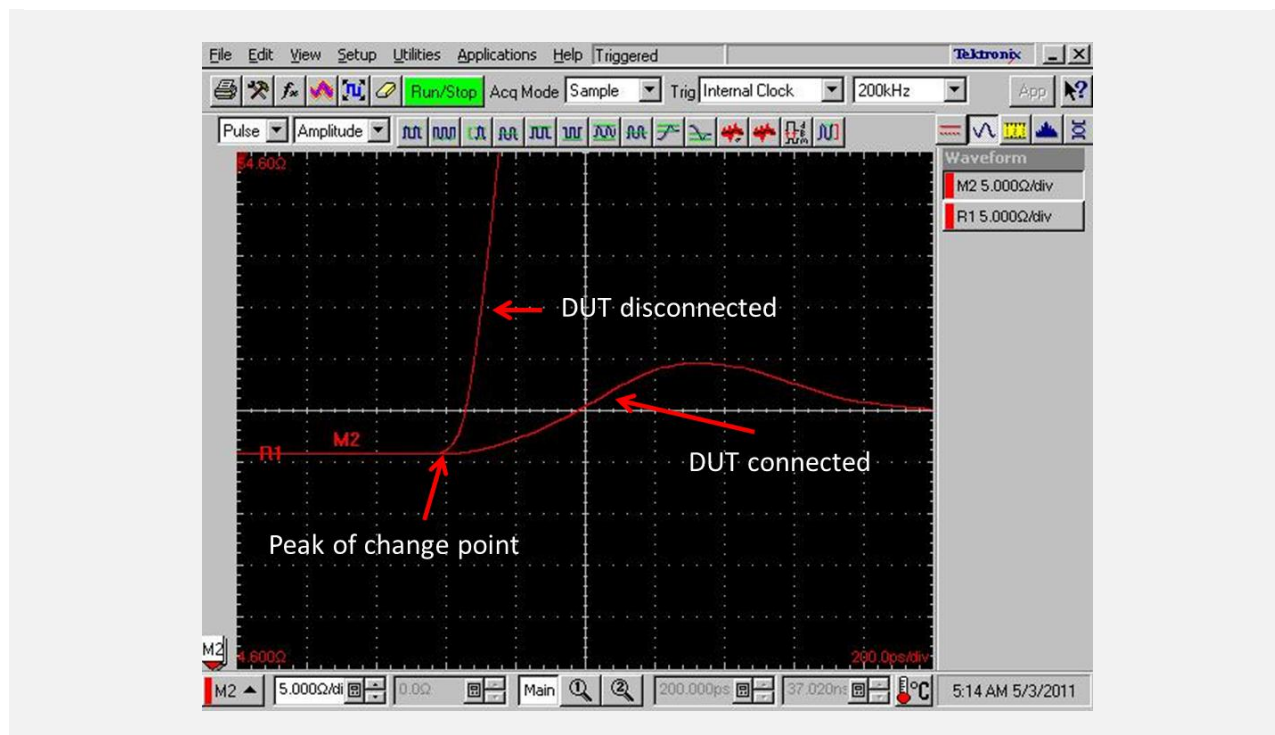


Figure 51. Set reference point for common mode

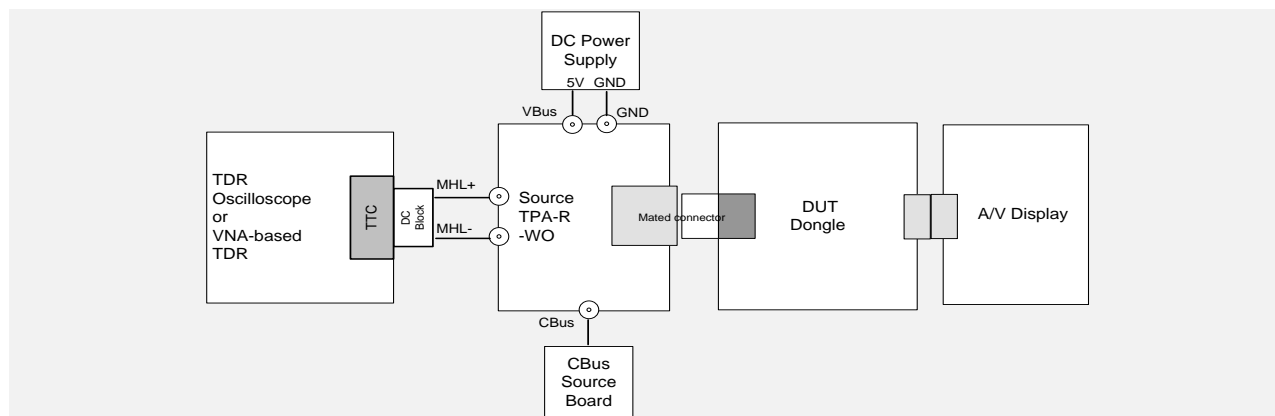


Figure 52. Dongle – Common-mode Impedance Test Setup

5.1.1.7 RxSense Impedance for Powered Dongle

Application: Dongle

5.1.1.7.1 Test Objective

This test confirms that the input RxSense impedance of the dongle device is within the range allowed by the specification.

NOTE: Voltages, currents and impedances in the Required and Recommended Methodologies of this test subscripted with the suffix "1" apply to the MHL-pin; voltages subscripted with the suffix "2" apply to the MHL+ pin. Refer to Figure 15.

5.1.1.7.2 References

[MHL] Section 13.7.3 Table 13-12 Dongle Device Impedance Characteristics at TP3:

RxSense Active: $Z_{RXSENSE_TERM} \leq 100K \text{ ohms}$

TMDS Termination: $Z_{RXSENSE_TMDS} \leq 70 \text{ ohms}$

RxSense Open (leakage current): $I_{RXSENSE_LEAK} \leq 2 \text{ uA}$

5.1.1.7.3 Required Test Equipment

Low-Bandwidth Digital Oscilloscope

Passive Probe

Sink/Dongle TPA-RSEN

CBUS Tester (Source Emulation)

5.1.1.7.4 Required Methodology

1. Connect the equipment as shown in Figure 53.
2. Turn on Dongle DUT and CBUS Tester.
3. Wait for CBUS to settle to "low" (initial wakeup from Dongle may float CBUS)
4. Measure $I_{RxSense_LEAK}$ current. FAIL if $I_{RxSense_LEAK}\{DUT\} > I_{RxSense_LEAK}\{max\}$.
 - a. Connect $R_{REF1 \& 2_ILEAK}$ as shown in Figure 15 and measure voltage between $R_{REF1 \& 2_ILEAK}$ and GND.
 - b. This voltage is $V_{RSEN_OPEN1 \& 2}$.
 - c. Calculate $I_{RxSense_LEAK1 \& 2} = V_{RSEN_OPEN1 \& 2} / R_{REF1 \& 2_ILEAK}$.
 - d. Fail if $I_{RxSense_LEAK1 \& 2}\{DUT\} > I_{RxSense_LEAK}\{max\}$.
5. Set CBUS Tester (Source) to perform CBUS discovery to Dongle DUT.
6. Wait for Dongle DUT to change CBUS impedance from $Z_{CBUS_DONGLE_DISCOVER}$ to $Z_{CBUS_DONGLE_ON} + T_{SINK:CONN}(max) + T_{SINK:RxSense_EN}(max)$.
7. Measure Dongle DUT TMDS impedance ($Z_{RXSENSE_TERM}$). FAIL if $Z_{RXSENSE_TERM}\{DUT\} > Z_{RXSENSE_TERM}\{max\}$.
 - a. Measure voltage between $R_{REF1 \& 2}$ and GND without resistors. This is $V_{TERM1 \& 2}$.
 - b. Connect $R_{REF1 \& 2_TERM}$ as shown in Figure 15 and measure voltage between $R_{REF1 \& 2_TERM}$ and GND.
 - c. This voltage is $V_{RXSENSE_TERM1 \& 2}$.
 - d. Calculate $Z_{RXSENSE_TERM1 \& 2} = R_{REF1 \& 2_TERM} * (V_{TERM1 \& 2} - V_{RXSENSE_TERM1 \& 2}) / V_{RXSENSE_TERM1 \& 2}$
 - e. Fail if $Z_{RXSENSE_TERM1 \& 2}\{DUT\} > Z_{RXSENSE_TERM}\{max\}$.
8. Connect Dongle DUT to downstream AV display device and select Dongle DUT as input source.
9. Wait for Dongle DUT to send SET_HPD and PATH_EN=1{SINK} packets to CBUS Tester.
10. Measure Dongle DUT TMDS impedance ($Z_{RXSENSE_TMDS}$) changes at $T_{SRC_RXSENSE_DEGLITCH}\{min\}$ after PATH_EN{SINK} is detected. FAIL if $Z_{RXSENSE_TMDS}\{DUT\} > Z_{RXSENSE_TMDS}\{max\}$.
 - a. Connect $R_{REF1 \& 2_TMDS}$ as shown in Figure 15 and measure voltage between $R_{REF1 \& 2_TMDS}$ and GND.
 - b. This voltage is $V_{RXSENSE_TMDS1 \& 2}$.
 - c. Calculate $Z_{RXSENSE_TMDS1 \& 2} = R_{REF1 \& 2_TMDS} * (V_{TERM1 \& 2} - V_{RXSENSE_TMDS1 \& 2}) / V_{RXSENSE_TMDS1 \& 2}$
 - d. Fail if $Z_{RXSENSE_TMDS1 \& 2}\{DUT\} > Z_{RXSENSE_TMDS}\{max\}$.
11. Disconnect Dongle (DUT) from AV Display to force CLR_HPD & PATH_EN{SINK} change from 1->0.
12. FAIL if Dongle (DUT) doesn't send a CLR_HPD.
13. FAIL if Dongle (DUT) doesn't send a PATH_EN(SINK) 1->0.
14. Use the CBUS Tester (Source) to write PATH_EN=0{SOURCE} (Source Content ON-> OFF) to Dongle DUT.
15. Measure Dongle DUT TMDS impedance ($Z_{RXSENSE_TERM}$). Verify Dongle DUT switches $Z_{RXSENSE}$ to $Z_{RXSENSE_TERM}$. FAIL if $Z_{RXSENSE}\{DUT\} > Z_{RXSENSE_TERM}\{max\}$.
 - a. Connect $R_{REF1 \& 2_TERM}$ as shown in Figure 15 and measure voltage between $R_{REF1 \& 2_TERM}$ and GND.
 - b. This voltage is $V_{RXSENSE_TERM1 \& 2}$.
 - c. Calculate $Z_{RXSENSE_TERM1 \& 2} = R_{REF1 \& 2_TERM} * (V_{TERM1 \& 2} - V_{RXSENSE_TERM1 \& 2}) / V_{RXSENSE_TERM1 \& 2}$

- d. Fail if $Z_{RXSENSE_TERM1 \& 2\{DUT\}} > Z_{RXSENSE_TERM\{max\}}$.
16. If all preceding steps pass, then PASS; else FAIL.

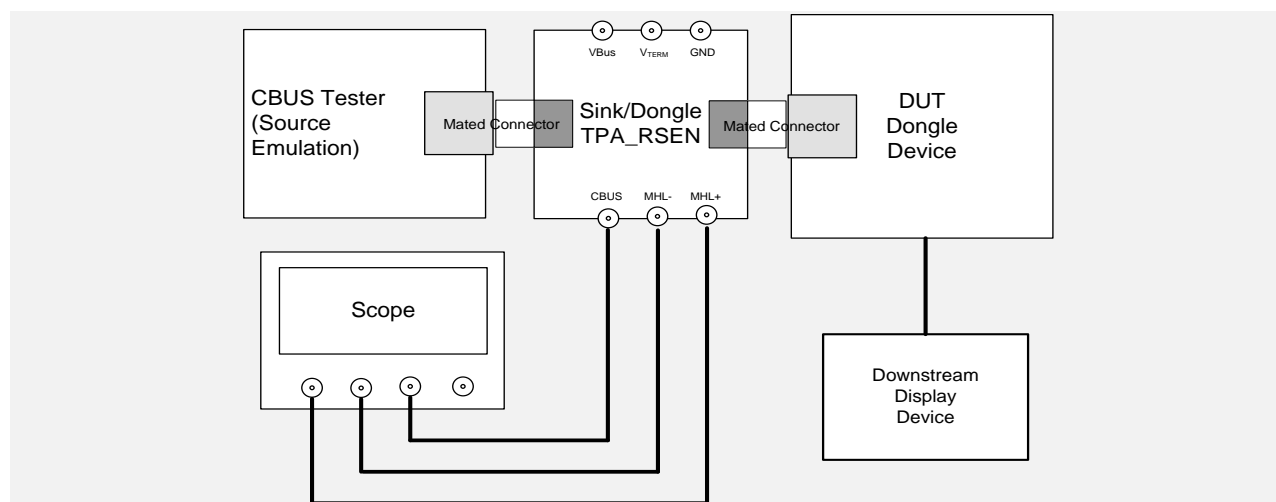


Figure 53. Dongle – RxSense Impedance Test Setup

5.1.1.8 RxSense Impedance for Unpowered Dongle

Application: Dongle

5.1.1.8.1 Test Objective

This test confirms that the input RxSense impedance of the dongle device is within the range allowed by the specification. Check that the dongle device is designed to accept power from the Source through VBUS.

NOTE: Voltages, currents and impedances in the Required and Recommended Methodologies of this test subscripted with the suffix "1" apply to the MHL-pin; voltages subscripted with the suffix "2" apply to the MHL+ pin. Refer to Figure 15.

5.1.1.8.2 References

[MHL] Section 13.7.3 Table 13-12 Dongle Device Impedance Characteristics at TP3:

RxSense Active: $Z_{RXSENSE_TERM} \leq 100K$ ohms

TMDS Termination: $Z_{RXSENSE_TMDS} \leq 70$ ohms

RxSense Open (leakage current): $I_{RXSENSE_LEAK} \leq 2$ uA

5.1.1.8.3 Required Test Equipment

Low-Bandwidth Digital Oscilloscope

Passive Probe

Sink/Dongle TPA-RSEN

CBUS Tester (Source Emulation)

5.1.1.8.4 Required Methodology

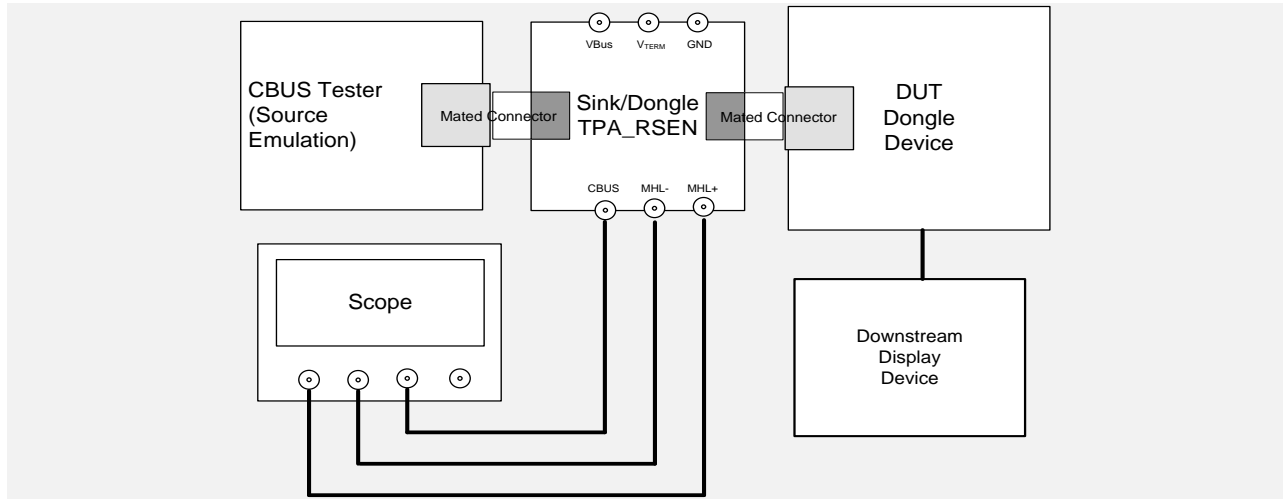


Figure 54. Dongle – RxSense Impedance Test Setup

1. If **CDF_D_ACCEPTS_POWER_FROM_SOURCE** indicates that the DUT can accept power, then proceed with this test. Else PASS (SKIP).
2. Connect the equipment as shown in Figure 54.
3. Turn on Source (CBUS tester) VBUS power to Dongle DUT.
4. Wait for CBUS to settle to “low” (initial wakeup from Dongle may float CBUS).
5. Measure $I_{\text{RxSense_LEAK}}$ current. FAIL if $I_{\text{RxSense_LEAK}}\{\text{DUT}\} > I_{\text{RxSense_LEAK}}\{\text{max}\}$.
 - a. Connect $R_{\text{REF1 \& 2_ILEAK}}$ as shown in Figure 15 and measure voltage between $R_{\text{REF1 \& 2_ILEAK}}$ and GND.
 - b. This voltage is $V_{\text{RSEN_OPEN1 \& 2}}$.
 - c. Calculate $I_{\text{RxSense_LEAK1 \& 2}} = V_{\text{RSEN_OPEN1 \& 2}} / R_{\text{REF1 \& 2_ILEAK}}$.
 - d. Fail if $I_{\text{RxSense_LEAK1 \& 2}}\{\text{DUT}\} > I_{\text{RxSense_LEAK}}\{\text{max}\}$.
6. Set CBUS Tester (Source) to perform CBUS discovery to Dongle DUT.
7. Wait for Dongle DUT to change CBUS impedance from $Z_{\text{CBUS_DONGLE_DISCOVER}}$ to $Z_{\text{CBUS_DONGLE_ON}} + T_{\text{SINK:CONN}}(\text{max}) + T_{\text{SINK:RxSense_EN}}(\text{max})$.
8. Measure Dongle DUT TMDS impedance ($Z_{\text{RXSENSE_TERM}}$). FAIL if $Z_{\text{RXSENSE_TERM}}\{\text{DUT}\} > Z_{\text{RXSENSE_TERM}}\{\text{max}\}$.
 - a. Measure voltage between $R_{\text{REF1 \& 2}}$ and GND without resistors. This is $V_{\text{TERM1 \& 2}}$.
 - b. Connect $R_{\text{REF1 \& 2_TERM}}$ as shown in Figure 15 and measure voltage between $R_{\text{REF1 \& 2_TERM}}$ and GND.
 - c. This voltage is $V_{\text{RXSENSE_TERM1 \& 2}}$.
 - d. Calculate $Z_{\text{RXSENSE_TERM1 \& 2}} = R_{\text{REF1 \& 2_TERM}} * (V_{\text{TERM1 \& 2}} - V_{\text{RXSENSE_TERM1 \& 2}}) / V_{\text{RXSENSE_TERM1 \& 2}}$.
 - e. Fail if $Z_{\text{RXSENSE_TERM1 \& 2}}\{\text{DUT}\} > Z_{\text{RXSENSE_TERM}}\{\text{max}\}$.
9. Connect Dongle DUT to downstream AV display device and select Dongle DUT as input source.
10. Wait for Dongle DUT to send SET_HPD and PATH_EN=1{SINK} packets to CBUS Tester.
11. Measure Dongle DUT TMDS impedance ($Z_{\text{RXSENSE_TMDs}}$) changes at $T_{\text{SRC_RXSENSE_DEGLITCH}}\{\text{min}\}$ after PATH_EN{SINK} is detected. FAIL if $Z_{\text{RXSENSE_TMDs}}\{\text{DUT}\} > Z_{\text{RXSENSE_TMDs}}\{\text{max}\}$.
 - a. Connect $R_{\text{REF1 \& 2_TMDs}}$ as shown in Figure 15 and measure voltage between $R_{\text{REF1 \& 2_TMDs}}$ and GND.
 - b. This voltage is $V_{\text{RXSENSE_TMDs1 \& 2}}$.
 - c. Calculate $Z_{\text{RXSENSE_TMDs1 \& 2}} = R_{\text{REF1 \& 2_TMDs}} * (V_{\text{TERM1 \& 2}} - V_{\text{RXSENSE_TMDs1 \& 2}}) / V_{\text{RXSENSE_TMDs1 \& 2}}$.
 - d. Fail if $Z_{\text{RXSENSE_TMDs1 \& 2}}\{\text{DUT}\} > Z_{\text{RXSENSE_TMDs}}\{\text{max}\}$.
12. Disconnect Dongle (DUT) from AV Display to force CLR_HPD & PATH_EN{SINK} change from 1->0.
13. FAIL if Dongle (DUT) doesn't send a CLR_HPD.
14. FAIL if Dongle (DUT) doesn't send a PATH_EN(SINK) 1->0.

15. Use the CBUS Tester (Source) to write PATH_EN=0{SOURCE} (Source Content ON→ OFF) to Dongle DUT.
16. Measure Dongle DUT TMDs impedance ($Z_{RXSENSE_TERM}$). Verify DONGLE DUT switches $Z_{RXSENSE}$ to $Z_{RXSENSE_TERM}$.
FAIL if $Z_{RXSENSE}\{DUT\} > Z_{RXSENSE_TERM}\{max\}$.
 - a. Connect $R_{REF1 \& 2_TERM}$ as shown in Figure 15 and measure voltage between $R_{REF1 \& 2_TERM}$ and GND.
 - b. This voltage is $V_{RXSENSE_TERM1 \& 2}$.
 - c. Calculate $Z_{RXSENSE_TERM1 \& 2} = R_{REF1 \& 2_TERM} * (V_{TERM1 \& 2} - V_{RXSENSE_TERM1 \& 2}) / V_{RXSENSE_TERM1 \& 2}$.
 - d. Fail if $Z_{RXSENSE_TERM1 \& 2}\{DUT\} > Z_{RXSENSE_TERM}\{max\}$.
17. If all preceding steps pass, then PASS; else FAIL.

5.1.1.9 Jitter Tolerance in PackedPixel Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.1.1.9.1 Test Objective

This test confirms that the dongle device can tolerate the maximum clock and data jitter amounts allowed by the specification in PackedPixel Mode.

5.1.1.9.2 References

[MHL] Section 13.5.2 Figure 13-13 MHL Clock Jitter at TP1
 [MHL] Section 13.5.3 Figure 13-14 Differential Eye Diagram Mask at TP1
 [MHL] Section 13.5.3 Figure 13-15 MHL Data Jitter at TP1

5.1.1.9.3 Required Test Equipment

High-bandwidth Digital Oscilloscope
 Differential Probe
 Jitter and Eye Analyzer
 Clock Jitter Generator
 Transition Time Converter (TTC)
 Bias-Tee
 MHL Signal Generator
 CBUS Source Board
 Source TPA-P-WT
 Source TPA-R-WO
 DC Power Supply
 A/V Display

5.1.1.9.4 Required Methodology

1. If CDF_VIDEO_PACKEDPIXEL in the CDF is “Yes”, then continue test; else end test with PASS (SKIP).
2. Connect the equipment as shown in Figure 42 for single-ended voltage calibration and Figure 43 for differential mode and common mode voltages calibration. Set $V_{TERM} = 3.3V$.
3. Calibrate the Signal Generator output for the following settings.
 - e. Frequency: Highest MHL data bit rate supported by the Dongle DUT in PackedPixel Mode
 - f. Pattern: MHL Gray Ramp data streams
 - d. Rise/fall times:
 - Less than or equal to 2.25Gbps
Differential signal 200ps (20% - 80%), Common-mode signal 600ps (20% - 80%)
 - Above 2.25Gbps

Differential signal 100ps (20% - 80%), Common-mode signal 600ps (20% - 80%)

- g. Swing voltages: Differential signal 800mV, Common-mode signal 360mV
- h. DC level of the output single-ended signal: $V_{TERM} - 540mV$
4. Set the clock jitter frequency to 10MHz and the data jitter frequency to 500KHz.
5. Calibrate the Signal Generator and Clock Jitter Generator to get data eye diagram and clock jitter which are marginal to TP1 specification.
 - a. Adjust clock jitter and data jitter to the TP1 maximum amount specified in MHL Spec Figures 13-13 and 13-15 for the given clock frequency.
 - b. Adjust the differential signal swing to the marginal eye opening specified in MHL Spec Figure 13-14.
6. Connect the equipment as shown in Figure 47. Provide VBus if DUT is unpowered dongle. Do not provide VBus if DUT is powered dongle.
7. Turn on the DUT and enable MHL mode through Device Discovery and CBUS Information Exchange with the CBUS Source board.
8. Set the video mode to a mode which uses PackedPixel Mode.
9. Examine the output of the Dongle DUT on the A/V Display and record error on the display if any.
10. Change the common-mode clock edge location by $0.1T_{bit}$ step with respect the differential data edge for 10 times and examine the A/V Display at each step.
11. Record error on the display if any for the given V_{TERM} , clock frequency, jitter frequency and clock edge location.
12. Connect the equipment as shown in Figure 43.
13. Set the clock jitter frequency to 7MHz and the data jitter frequency to 1MHz.
14. Repeat steps 4 to 10 for clock jitter frequency 7MHz and data jitter frequency 1MHz.
15. If there is no error on the display in all recorded results, then PASS. Otherwise FAIL.

5.2 System Tests

5.2.1 TMDS Coding

5.2.1.1 Character Synchronization in Normal Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.1.1.1 Test Objective

Confirm that Dongle DUT synchronizes if the Normal Mode data stream provides only minimum length Control Periods.

5.2.1.1.2 References

[MHL] 4.1.4 Character Synchronization

[MHL] 4.2.1.1.1 Guard Bands

[MHL] 4.4.3 Control Period Coding

[MHL] 4.4.4 TERC4 Coding

[MHL] 4.4.5 Video Data Coding

5.2.1.1.3 Required Test Equipment

MHL Signal Generator

Downstream display device which supports both 50Hz and 60Hz video modes.

5.2.1.1.4 Required Methodology

1. Configure MHL Signal Generator to transmit a 720x480p 59.94/60Hz Normal Mode video stream with every horizontal and vertical blanking interval completely filled with one or more Data Islands and with Control Periods 12 characters.
2. Connect Dongle DUT to an AV Monitor. Connect MHL Signal Generator to Dongle DUT.
3. If AV Monitor connected to the DUT correctly shows the transmitted signal, then continue to test, else FAIL.
4. Repeat the test from step 1 for 720x576p 50Hz Normal Mode.
5. If all video modes listed in the previous steps are verified, then PASS, else FAIL.

Note. Use following Data Islands and Control Periods arrangement for HBLANK:

720x480p 59.94/60Hz : 12+12+2+32+32+32+2+12+2 (138 pixels)

720x576p 50Hz : 12+18+2+32+32+32+2+12+2 (144 pixels)

5.2.1.2 Packet Types in Normal Mode

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.2.1.2.1 Test Objective

Confirm that Dongle DUT accepts all valid packet types in Normal Mode.

5.2.1.2.2 References

[MHL] Section 4.3. Data Island Packet Definitions

[CEA 861] 6 Auxiliary Information Carried from Source to Sink

5.2.1.2.3 Required Test Equipment

MHL Signal Generator

Downstream display device which supports both 50Hz and 60Hz video modes.

5.2.1.2.4 Required Methodology

1. If 60Hz video is not supported, as indicated in the CDF, then skip to step #5; else continue the test.
2. Configure MHL Signal Generator to output a 720x480p 59.94/60Hz Normal Mode video stream, while also transmitting 2-channel PCM audio, with the following characteristics.
During VBLANK, one or more Data Islands contain a valid:
 - a. Null Packet
 - b. Content Mute Packet with both MUTE and UNMUTE bits unset (0)
 - c. AVI InfoFrame Packet
 - d. Source Product Description Packet
 - e. Audio InfoFrame Packet
 - f. MPEG Source InfoFrame Packet
 - g. a Packet with Packet Type in the range between 0x04 and 0x6F, with arbitrary payload
 - h. a Packet with Packet Type in the range between 0x71 and 0x7F, with arbitrary payload
 - i. a Packet with Packet Type 0x80, with arbitrary payload
3. Connect Dongle DUT to a AV Monitor. Connect MHL Signal Generator to Dongle DUT.
4. If AV monitor connect to the DUT correctly shows the transmitted signal, then continue to test, else FAIL.
5. Repeat the test from step 2 for 720x576p 50Hz in Normal Mode.

6. If all video modes listed in the previous steps are verified, then PASS, else FAIL.

5.2.1.3 Character Synchronization in PackedPixel Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.1.3.1 Test Objective

Confirm that Dongle DUT synchronizes if the PackedPixel Mode data stream provides only minimum length Control Periods.

5.2.1.3.2 References

[MHL] 4.1.4 Character Synchronization
[MHL] 4.2.2.1.1 Guard Bands (PackedPixel Mode)
[MHL] 4.4.3 Control Period Coding
[MHL] 4.4.4 TERC4 Coding
[MHL] 4.4.5 Video Data Coding

5.2.1.3.3 Required Test Equipment

MHL Signal Generator

5.2.1.3.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** in the CDF is “Yes”, then continue test; else end test with PASS (SKIP).
2. Configure MHL Signal Generator to transmit a 1920x1080p 59.94/60Hz PackedPixel Mode video stream with every horizontal and vertical blanking interval completely filled with one or more Data Islands and with Control Periods 12 characters.
3. Connect Dongle DUT to an AV Monitor. Connect MHL Signal Generator to Dongle DUT.
4. If AV Monitor connected to the DUT correctly shows the transmitted signal, then continue to test, else FAIL.
5. Repeat the test from step 1 for 1920x1080p 50Hz PackedPixel Mode.
6. If all video modes listed in the previous steps are verified, then PASS, else FAIL.

Note. Use following Data Islands and Control Periods arrangement for HBLANK:

1920x1080p 60 Hz: 12+2+32+32+2+16+2+32+32+32+32+32+2+18+2 (280 pixels)

1920x1080p 50hz:

12+2+32+32+32+32+32+32+32+32+32+32+32+32+32+2+12+2+32+32+32+32+32+2+14+2
(720 pixels)

5.2.1.4 Packet Types in PackedPixel Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.1.4.1 Test Objective

Confirm that Dongle DUT accepts all valid packet types in PackedPixel Mode.

5.2.1.4.2 References

[MHL] Section 4.3. Data Island Packet Definitions

[CEA 861] 6 Auxiliary Information Carried from Source to Sink

5.2.1.4.3 Required Test Equipment

MHL Signal Generator

5.2.1.4.4 Required Methodology

1. Configure MHL Signal Generator to output a 1920x1080p 59.94/60Hz or 1920x1080p 50Hz (if supported) PackedPixel Mode video stream, while also transmitting 2-channel PCM audio, with the following characteristics.

During VBLANK, one or more Data Islands contain a valid:

- a. Null Packet
 - b. Content Mute Packet with both MUTE and UNMUTE bits unset (0)
 - c. AVI InfoFrame Packet
 - d. Source Product Description Packet
 - e. Audio InfoFrame Packet
 - f. MPEG Source InfoFrame Packet
 - g. a Packet with Packet Type in the range between 0x04 and 0x6F, with arbitrary payload
 - h. a Packet with Packet Type in the range between 0x71 and 0x7F, with arbitrary payload
 - i. a Packet with Packet Type 0x80, with arbitrary payload
2. Connect Dongle DUT to a AV Monitor. Connect MHL Signal Generator to Dongle DUT.
 3. If AV monitor connect to the DUT correctly shows the transmitted signal, then continue to test, else FAIL.
 4. If all video modes listed in the previous steps are verified, then PASS, else FAIL.

5.2.2 Video Tests

5.2.2.1 Video Formats in Normal Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.2.1.1 Test Objective

Verify that dongle DUT supports the video formats in Normal Mode with no distortion.

5.2.2.1.2 References

[MHL] Section 5.2.1.2 MHL Sink Video Format Support Requirements

5.2.2.1.3 Required Test Equipment

MHL Signal Generator

5.2.2.1.4 Required Methodology

1. Configure MHL Signal Generator to send out test patterns in one of the video formats in the CDF which use Normal Mode.
2. If A/V Display with DUT attached displays the test patterns with no distortion, then continue to test, else FAIL.
3. Repeat the test from step 1 for all video formats indicated in the CDF which use Normal Mode.
4. If none of the video format fails, then PASS.

5.2.2.2 Pixel Encoding in Normal Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.2.2.1 Test Objective

Verify that Dongle supports various pixel encodings in Normal Mode.

5.2.2.2.2 References

[MHL] Section 5.2.3 24-Bit Pixel Encoding Requirement.

[MHL] Section 5.5 Pixel Encodings.

5.2.2.2.3 Required Test Equipment

MHL Signal Generator

5.2.2.2.4 Required Methodology

1. If **CDF_VIDEO_RGB** field in the CDF is "YES", then continue to test, else FAIL.
2. Configure MHL Signal Generator to send out test patterns in 720x480p 59.94/60Hz or 720x 576p 50Hz with one of the pixel encoding indicated in the CDF.
3. If A/V Display with DUT attached displays the test patterns with no distortion, then continue to test, else FAIL.
4. Repeat the test from step 2 for all pixel encoding formats indicated in the CDF.
5. If any of Normal Mode Pixel Encoding indicated in the CDF fails, then FAIL, else PASS.

Note: Use Black/White Ramp pattern.

5.2.2.3 Video Quantization Range

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.2.3.1 Test Objective

Verify that Dongle has correct QY and QS bits in the Video Capability Data Block and A/V Display with Dongle DUT attached displays with no distortion.

5.2.2.3.2 References

[MHL] Section 5.6 Video Quantization Ranges.

5.2.2.3.3 Required Test Equipment

MHL Signal Generator

5.2.2.3.4 Required Methodology

1. If either **CDF_VIDEO_YCBCR_444** or **CDF_VIDEO_YCBCR_422** field in the CDF is "YES" and if **CDF_VIDEO_YCC_FULL** is "YES", then continue to test else PASS (SKIP).
2. Read QY bit in the EDID's Video Capability Data Block. If Video Capability Data Block is not available in the DUT's EDID or the EDID read through the DUT, then PASS (SKIP); else continue this test.
 - a. If QY bit is "0" then FAIL, else continue to test.
 - b. Configure MHL signal generator to send out the test patterns in 720x480p59.94/60Hz or 720x576p50Hz with either YCbCr4:2:2 or YCbCr4:4:4, and set YQ bits to '1' in the AVI InfoFrame.

- c. If A/V Display attached to the DUT displays the test patterns with no distortion, then PASS, else FAIL.

5.2.2.4 Video Formats in PackedPixel Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.2.4.1 Test Objective

Verify that dongle DUT supports the video formats in PackedPixel Mode with no distortion.

5.2.2.4.2 References

[MHL] Section 5.2.1.2 MHL Sink Video Format Support Requirements

5.2.2.4.3 Required Test Equipment

MHL Signal Generator

5.2.2.4.4 Required Methodology

6. If **CDF_VIDEO_PACKEDPIXEL** in the CDF is “Yes”, then continue test; else end test with PASS (SKIP).
7. Configure MHL Signal Generator to send out test patterns in one of the video formats in the CDF which use PackedPixel Mode.
8. If A/V Display with DUT attached displays the test patterns with no distortion, then continue to test, else FAIL.
9. Repeat the test from step 1 for all video formats indicated in the CDF which use PackedPixel Mode.
10. If none of the video format fails, then PASS.

5.2.2.5 Pixel Encoding in PackedPixel Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.2.5.1 Test Objective

Verify that Dongle supports various pixel encodings in PackedPixel Mode.

5.2.2.5.2 References

[MHL] Section 5.2.4 PackedPixel Encoding Requirement.

[MHL] Section 5.5 Pixel Encodings.

5.2.2.5.3 Required Test Equipment

MHL Signal Generator

5.2.2.5.4 Required Methodology

1. If **CDF_VIDEO_PACKEDPIXEL** in the CDF is “Yes”, then continue test; else end test with PASS (SKIP).
2. Configure MHL Signal Generator to send out test patterns in 1920x1080p 59.94/60Hz or 1920x1080p 50Hz , according to which is indicated in the CDF as a supported video mode.
3. If A/V Display with DUT attached displays the test patterns with no distortion, then continue to test, else FAIL.

5.2.3 Audio Test

5.2.3.1 IEC 60958 / IEC61937

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.3.1.1 Test Objective

Verify that Dongle converts audio properly.

5.2.3.1.2 References

[MHL] Section 6.3 Audio Sample Rates and Support Requirements.

5.2.3.1.3 Required Test Equipment

MHL Signal Generator

5.2.3.1.4 Required Methodology

1. If any of **CDF_AUDIO_2CH_32kHz**, **CDF_AUDIO_2CH_44kHz** and **CDF_AUDIO_2CH_48kHz** field in the CDF is "YES", then continue to test, else end test with PASS (SKIP).
2. Configure MHL Signal Generator as one of PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz mode indicated in the CDF.
3. Configure the MHL Signal Generator to send out 1 kHz sine wave.
4. If DUT regenerates the 1 kHz sound with no distortion, then continue the test, else FAIL.
5. Repeat the test from step 2 for PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz indicated in the CDF. If any of the Audio formats fails the test, then FAIL, else PASS.

5.2.3.2 Audio Clock Regeneration

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.3.2.1 Test Objective

Verify that Dongle properly regenerate audio when different Audio Clock Regeneration parameters are used.

5.2.3.2.2 References

[MHL] Section 6.2 Audio Sample Clock Capture and Regeneration.

5.2.3.2.3 Required Test Equipment

MHL Signal Generator

5.2.3.2.4 Required Methodology

1. If any of **CDF_AUDIO_2CH_32kHz**, **CDF_AUDIO_2CH_44kHz** and **CDF_AUDIO_2CH_48kHz** field in the CDF is "YES", then continue to test, else end test with PASS (SKIP).
2. Configure MHL Signal Generator as one of PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz mode indicated in the CDF.
3. Configure MHL Signal Generator to send out the test patterns with 1kHz sine wave using minimum N parameter.
 - a. If A/V Display with DUT attached reproduces 1kHz sine wave with no distortion, then continue to test, else FAIL.

4. Configure MHL Signal Generator to send out test patterns with 1kHz sine wave using maximum N parameter.
 - a. If A/V Display with DUT attached reproduces 1kHz sine wave with no distortion, then continue to test, else FAIL.
 - b. The corresponding audio format selection is under the "Audio Format" pull-down menu.
5. Repeat the test from step 2 for PCM_2CH_32kHz, PCM_2CH_44.1kHz and PCM_2CH_48kHz indicated in the CDF.
6. If any of the Audio modes fails the test, then FAIL, else PASS.

5.2.4 HDCP Test

5.2.4.1 HDCP

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.4.1.1 Test Objective

Verify that Dongle complies with HDCP.

5.2.4.1.2 References

[MHL] Section 11.3 HDCP Compliance.

5.2.4.1.3 Required Test Equipment

HDCP Emulator

5.2.4.1.4 Required Methodology

1. If **CDF_HDCP_SUPPORT** field in CDF is "YES", then continue to test, else end test with PASS (SKIP).
2. Follow the HDCP CTS procedure. If the **CDF_HDCP_REPEATER** field in the CDF is "YES", then include all HDCP Repeater tests.
3. If all tests in HDCP CTS pass, then PASS; else FAIL.

*Note: HDCP test tool must support clock stretch.

5.2.5 EDID Test and Device Capability / Device Status Register Test

5.2.5.1 EDID Test

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.5.1.1 Test Objective

Verify that downstream Sink EDID is accessible through Dongle.

5.2.5.1.2 References

[MHL] Section 10.3 EDID Access.

5.2.5.1.3 Required Test Equipment

MHL Signal Generator

Downstream display device which supports both 50Hz and 60Hz video modes.

5.2.5.1.4 Required Methodology

1. Connect MHL Sink test equipment to DUT.
2. Wait until SET_HPD command is received.
3. Read Block0. If Extension Flag(byte 0x7E) is 0, then FAIL, else read all extended blocks.
4. If check_sum is correct, then continue to test, else FAIL.
5. Read Block0 and extended blocks again and compare.
6. If they do not match, then FAIL.
7. Examine EDID:
 - a. For each video mode in the EDID which has an MHL clock frequency less than or equal to the MHL clock frequency limit in the **CDF_CR_BANDWIDTH** field of the CDF:
 - i. If all Video Modes in the CDF are found in the EDID, then continue to test, else FAIL.
 - b. If all Pixel Encodings in the CDF are found in EDID, then continue to test, else FAIL.
 - c. If all audio modes in the CDF are found in EDID, then PASS, else FAIL.

5.2.5.2 Device Capability Register

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.5.2.1 Test Description

Verify that Device Capability Registers have accurate value.

5.2.5.2.2 Reference

[MHL] Section 7.8.1 Device Capability Registers

5.2.5.2.3 Required Test Equipment

MHL Signal Generator

5.2.5.2.4 Required Methodology

Perform the test in the following steps:

1. Connect DUT to MHL Signal Generator.
2. Configure MHL Signal Generator read DUT's MHL Capability Register.
3. Examine the values:
 - a. If the values in MHL_VERSION(offset:0x01) register does not match the **CDF_CR_MHL_VERSION** field in the CDF, then FAIL.
 - b. If DEV_TYPE in the DEV_CAT(Offset:0x02) register is not 0b0001:Sink, then FAIL.
 - c. If POW in the DEV_CAT(offset:0x02) register does not match the **CDF_CR_POW** field in the CDF, then FAIL.
 - d. If ADOPTER_ID_H(Offset:0x03) or ADOPTER_ID_L(offset:0x04) registers do not match the corresponding **CDF_CR_ADOPTER_ID_H** and **CDF_CR_ADOPTER_ID_L** fields in the CDF, then FAIL.
 - e. If SUPP_RGB444, SUPP_YCBCR444, SUPP_YCBCR422, SUPP_PPIXEL, SUPP_ISLANDS or SUPP_VGA bits in the VID_LINK_MODE(offset:0x05) register do not match the corresponding **CDF_CR_SUPP_RGB444**, **CDF_CR_SUPP_YCBCR444**, **CDF_CR_SUPP_YCBCR422**, **CDF_CR_SUPP_PPIXEL**, **CDF_CR_SUPP_ISLANDS** and **CDF_CR_SUPP_VGA** fields in the CDF, then FAIL.

- f. If SUPP_VT in VIDEO_TYPE is 0b1 and if VT_GRAPHICS, VT_PHOTO, VT_CINEMA or VT_GAME bits in VIDEO_TYPE register(offset:0x07) do not match the corresponding **CDF_CR_VT_GRAPHICS**, **CDF_CR_VT_PHOTO**, **CDF_CR_VT_CINEMA** or **CDF_CR_VT_GAME** fields in the CDF, then FAIL.
 - g. If LD_DISPLAY, LD_VIDEO, LD_AUDIO, LD_MEDIA, LD_TUNER, LD_RECORD, LD_SPEAKER, or LD_GUI bits in the LOG_DEV_MAP(offset:0x08) register do not match the corresponding **CDF_CR_LD_DISPLAY**, **CDF_CR_LD_VIDEO**, **CDF_CR_LD_AUDIO**, **CDF_CR_LD_MEDIA**, **CDF_CR_LD_TUNER**, **CDF_CR_LD_RECORD**, **CDF_CR_LD_SPEAKER**, **CDF_CR_LD_GUI** fields in the CDF, then FAIL.
 - h. If BANDWIDTH(offset:0x09) register, which is expressed in terms of 5's of megahertz, does not match the **CDF_CR_BANDWIDTH** field in the CDF, then FAIL.
 - i. Check the following support bits in the FEATURE_FLAG register against the CDF:
 - If RCP_SUPPORT does not match **CDF_CR_RCP_SUPPORT**, then FAIL.
 - If RAP_SUPPORT does not match **CDF_CR_RAP_SUPPORT**, then FAIL.
 - If SP_SUPPORT does not match **CDF_CR_SP_SUPPORT**, then FAIL.
 - If UCP_SEND_SUPPORT does not match **CDF_CR_UCP_SEND_SUPPORT**, then FAIL.
 - If UCP_RECV_SUPPORT does not match **CDF_CR_UCP_RECV_SUPPORT**, then FAIL.
 - j. If DEVICE_ID_H(Offset:0x0b) or DEVICE_ID_L(offset:0x0c) do not match the corresponding **CDF_CR_DEVICE_ID_H** and **CDF_CR_DEVICE_ID_L** fields in the CDF, then FAIL.
 - k. If SP_SUPPORT bit is 0b1 and if SCRATCHPAD_SIZE(offset:0x0d) register does not match the **CDF_CR_SCRATCHPAD_SIZE** field in the CDF, then FAIL.
 - l. If INT_SIZE or STAT_SIZE fields in INT_STAT_SIZE register(offset:0x0e) do not match the corresponding **CDF_CR_INT_SIZE** and **CDF_CR_STAT_SIZE** fields in the CDF, then FAIL.
4. If all register values match the corresponding CDF field, then PASS.

5.2.5.3 Device Status Registers Test

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.5.3.1 Test Description

Verify that Device Status Registers have proper values.

5.2.5.3.2 Reference

[MHL] Section 7.8.1 Device Capability Registers

5.2.5.3.3 Required Test Equipment

MHL Signal Generator

5.2.5.3.4 Required Methodology

1. Connect DUT to MHL Signal Generator while let the MHL Signal Generator monitoring LINK_MODE status register.
2. Wait until DUT writes PATH_EN =1.
3. When PATH_EN =1, examine TMDS line. If TMDS lines are terminated to Z_{RXSENSE_TMDS}, then FAIL, else continue to test.
4. Write PATH_EN = 1 into DUT.
5. Examine TMDS line. If TMDS lines are terminated to Z_{RXSENSE_TMDS}, then PASS, else FAIL.

5.2.6 RCP Sub-Commands Tests

5.2.6.1 RCP Sub-Command Receiving Test

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.6.1.1 Test Objective

Verify that DUT responds properly to all incoming RCP sub-commands, according to the logical device type of types indicated in the CDF.

5.2.6.1.2 References

[MHL] Section 7.7.8.1 Support when Receiving RCP Command

[MHL] Table 7-17 Key Code Receiving Support Map

[MHL] Section 7.7.8.2 Sink Support Sending RCP Command

[MHL] Table 7-18 Key Code Sending Support Map for Sink

[MHL] Table 7-16 Key Code Behaviors

5.2.6.1.3 Required Test Equipment

MSC Test Tool

5.2.6.1.4 Required Methodology

1. If **CDF_RCP_RECEIVE** field in CDF is "YES", then continue to test, else end test with PASS (SKIP).
2. Examine the Logical Device Type field in the CDF.
 - a. If no Logical Device Type is defined, then FAIL, else continue to test.
3. For each logical device type marked in the CDF:
 - a. Send each RCP command marked in for the Logical Device in MHL Specification Table 7-17.
 - b. If A/V Display with DUT attached does not demonstrate proper behavior as listed in the MHL Specification Table 7-16, then FAIL.
4. If any additional RCP sub-commands are defined in the CDF, repeat the test in step 3 for the additional RCP sub-commands.
5. If **CDF_LOG_DEV_MAP_CHANGE** in the CDF is "YES", then follow instructions in the CDF to change the DUT's LOG_DEV_MAP setting to each supported value, and repeat steps 3-4.
6. If all RCP sub-commands are verified, then PASS.

5.2.6.2 RCP Sub-Command Transmitting Test

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.6.2.1 Test Objective

Verify that DUT transmits properly each required RCP sub-command, according to the logical device type of types indicated for the Dongle DUT in the CDF.

5.2.6.2.2 References

[MHL] Section 7.7.8.1 Support when Receiving RCP Command

[MHL] Table 7-17 Key Code Receiving Support Map

[MHL] Section 7.7.8.2 Sink Support Sending RCP Command

[MHL] Table 7-18 Key Code Sending Support Map for Sink

[MHL] Table 7-16 Key Code Behaviors

5.2.6.2.3 Required Test Equipment

MHL Signal Generator

5.2.6.2.4 Required Methodology

1. If **CDF_RCP_SEND** field in CDF is “YES”, then continue to test, else end test with PASS (SKIP).
2. Configure A/V Display with attached DUT to send each of RCP sub-commands marked in the CDF Table 12, and verify the opcode.
3. If opcode does not match, then FAIL.
4. Repeat from step 2 for all RCP commands marked in the CDF Table 12.
5. If **CDF_LOG_DEV_MAP_CHANGE** in the CDF is “YES” then follow instructions in the CDF to change the DUT’s LOG_DEV_MAP setting to each supported value, and repeat steps 2-4.
6. If all opcode match, then PASS.

5.2.7 RAP Sub-Commands Test

5.2.7.1 RAP and RAPK Sub-Commands Test

| | |
|---------------------|-------------|
| Application: | Sink |
|---------------------|-------------|

5.2.7.1.1 Test Objective

Verify that the Sink DUT supports RAP sub-commands and responds with the proper behavior.

5.2.7.1.2 References

[MHL] Section 7.6 Request Action Protocol Sub-commands.

[MHL] Table 7-11 MSC Request Action Protocol Sub-Command Action Codes.

[MHL] Table 7-12 MSC Request Action Protocol Status Codes.

5.2.7.1.3 Required Test Equipment

MHL Signal Generator

5.2.7.1.4 Required Methodology

1. If **CDF_RAP_SUPPORT** field in the CDF is YES, then continue to test, else PASS(SKIP).
2. Send “Poll”: Opcode(0x00) to the DUT.
 - a. If the DUT responds with RAPK and Status Code “No Error”: 0x00, then continue to test, else FAIL.
 - b. If the DUT responds with RAPK and Status Code “Unrecognized Action Code”: 0x01 or “Unsupported Action Code”: 0x02, then FAIL, else continue to test.
 - c. If the DUT responds with RAPK and Status Code “Responder Busy”: 0x03 then, wait 1 second and repeat from the beginning of step 2.
3.
 - a.
 - b.
 - c.
4. Send “CONTENT_OFF”: Opcode(0x11) to the DUT.
 - a. If the DUT responds with RAPK and Status Code “No Error”: 0x00 and if the DUT stops rendering test patterns then PASS, else FAIL.

- b. If the DUT responds with RAPK and Status Code “Unrecognized Action Code”: 0x01 or “Unsupported Action Code”: 0x02, then FAIL, else continue to test.
 - c. If the DUT responds with RAPK and Status Code “Responder Busy”: 0x03 then wait 1 second then repeat from the beginning of step 3.
 5. Send “CONTENT_ON”: Opcode(0x10) to the DUT.
 - a. If the DUT responds with RAPK and Status Code “No Error”: 0x00 and if the DUT renders test patterns, then continue to test, else FAIL.
 - b. If the DUT responds with RAPK and Status Code “Unrecognized Action Code”: 0x01 or “Unsupported Action Code”: 0x02, then FAIL, else continue to test.
 - c. If the DUT responds with RAPK and Status Code “Responder Busy”: 0x03 then wait 1 second then repeat from the beginning of step 4.
 6. If DUT passes all step 2-4, then PASS, else FAIL.

5.2.8 3D Video Test

5.2.8.1 3D Video Mode Support Data

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.8.1.1 Test Objective

Verify that the Dongle DUT responds to 3D_REQ and sends the list of 3D video modes supported by the DUT.

5.2.8.1.2 References

[MHL] Section 5.9.1 3D Video Support Detection

[MHL] Table 5-3 3D Video Format Data Structure in WRITE_BURST

5.2.8.1.3 Required Test Equipment

MHL Signal Generator

5.2.8.1.4 Required Methodology

1. If **CDF_VIDEO_3D** field in the CDF is YES, then continue to test, else PASS(SKIP).
2. After discovery, wait until DCAP_RDY in the DUT becomes 0b1, then write 3D_REQ in Interrupt Register4 to 0b1.
3. If DUT transmits the list of 3D video modes that DUT supports by a sequence of WRITE_BURST, then continue to test, else FAIL.
4. If a byte-wide sum of all data bytes of each WRITE_BURST, including the checksum itself, equals zero, then continue to test, else FAIL.
5. Examine the list of 3D video modes. The list may contain additional modes not specified in the CDF but all modes listed in the CDF must be contained in the list. If TRUE, then PASS, else FAIL.
- 6.

5.2.8.2 3D Video Format

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.8.2.1 Test Objective

Verify that the Dongle DUT supports the required and optional 3D Video Modes.

5.2.8.2.2 References

[MHL] Section 5.9.4 3D Video Support Requirements

[MHL] Section 5.9.4.2 MHL Sink 3D Video Mode Support Requirements

5.2.8.2.3 Required Test Equipment

MHL Signal Generator

5.2.8.2.4 Required Methodology

1. If CDF_VIDEO_3D field in the CDF is YES, then continue to test, else PASS(SKIP).
2. Configure MHL Signal Generator to send out test patterns in one of the 3D video formats in the CDF.
3. If DUT displays the patterns with no distortion, then continue to test, else FAIL.
4. Repeat the test from step 2 for all 3D video formats indicated in the CDF.
5. If none of the video format fails, then PASS.

5.2.8.3 3D Video Format in PackedPixel Mode

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.8.3.1 Test Objective

Verify that the Dongle DUT supports required and optional 3D Video Modes which use PackedPixel Mode.

5.2.8.3.2 References

[MHL] Section 5.9.4 3D Video Support Requirements

[MHL] Section 5.9.4.2 MHL Sink 3D Video Mode Support Requirements

5.2.8.3.3 Required Test Equipment

MHL Signal Generator

Sink device which supports 3D PackedPixel Modes

5.2.8.3.4 Required Methodology

1. If CDF_VIDEO_3D field in the CDF is "YES", then continue to test, else end test with PASS (SKIP).
2. If CDF_VIDEO_PACKEDPIXEL field in the CDF is "Yes", and if DUT accepts 59.94/60Hz video formats from any other analog or digital video input, and if CDF_VIDEO_1280x720P_60_3D_Frame field in the CDF is "No", then FAIL, else continue to test.
3. If CDF_VIDEO_PACKEDPIXEL field in the CDF is "Yes", and if DUT accepts 50Hz video formats from any other analog or digital video input, and if CDF_VIDEO_1280x720P_50_3D_Frame field in the CDF is "No", then FAIL, else continue to test.
4. If CDF_VIDEO_PACKEDPIXEL field in the CDF is "Yes", and if CDF_VIDEO_1920x1080p_24_3D_Frame field in the CDF is "No", then FAIL, else continue to test.
5. Configure MHL Signal Generator to send out test patterns in one of the 3D video formats indicated in the CDF which use PackedPixel Mode.
6. If DUT displays the patterns with no distortion, then continue to test, else FAIL.
7. Repeat test steps 5-6 for all 3D video formats indicated in the CDF which use PackedPixel Mode.
8. If none of the video formats fails, then PASS.

5.2.9 UCP Sub-Command Tests

The test implementation shall include UTF character sequences for all lengths of characters (1 to 4 8-bit bytes or octets), with both invalid and valid values in each of those lengths.

5.2.9.1 UCP Sub-Commands Receiving Test

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.9.1.1 Test Objective

Verify that DUT responds to valid UCP sub-commands by displaying the character or characters sent in the UCP command, or response to invalid UCP sub-commands by displaying an error message.

5.2.9.1.2 References

[MHL] Section 7.8.4 MSC UTF-8 Character Protocol Support Requirements

5.2.9.1.3 Required Test Equipment

TBD.

The test implementation shall include UTF character sequences for all lengths of characters, with both invalid and valid values in those lengths.

5.2.9.1.4 Required Methodology

1. If CDF indicates **CDF_UCP_RECV_SUPPORT**=1 then continue test, else PASS (SKIP).
2. Connect the Test Equipment to the DUT.
3. Perform discover sequence between the Test Equipment and the DUT.
 - a. If DUT fails to discover properly, then FAIL.
4. For each UCP command described in the CDF, perform the following steps:
 - a. Enter into the test equipment the expected character or character sequence to send to the DUT, according to the information in the CDF entry.
 - b. Configure the DUT to receive the selected character or character sequence. Use the Application or other software on the DUT as described in the CDF with field **CDF_UCP_RECV_APPLICATION**.
 - c. Trigger the sending of the character or character sequence by Test Equipment action to the DUT.
 - d. Test Equipment displays all bits and interpretation of the UCP payload(s) it sends.
 - e. DUT displays all bits and interpretation of the UCP payload(s) it receives.
 - f. Test Engineer compares the DUT displayed data with the Test Equipment data.
 - g. If the payload received by the DUT does not match that sent by the Test Equipment, then FAIL.
5. If DUT passes all iterations of the test, then PASS. Else FAIL.

5.2.9.2 UCP Sub-Commands Transmitting Test

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.2.9.2.1 Test Objective

Verify that DUT sends valid UCP sub-commands by initiating the sending of UTF-8 characters in various formats through the user interface on the DUT.

5.2.9.2.2 References

[MHL] Section 7.8.4 MSC UTF-8 Character Protocol Support Requirements

5.2.9.2.3 Required Test Equipment

TBD

5.2.9.2.4 Required Methodology

1. If CDF indicates **CDF_UCP_SEND_SUPPORT**=1 then continue test, else PASS (SKIP).
2. Connect the Test Equipment to the DUT.
3. Perform discover sequence between the Test Equipment and the DUT.
 - a. If DUT fails to discover properly, then FAIL.
4. For each UCP command described in the CDF, perform the following steps:
 - a. Enter into the test equipment the expected character or character sequence from the DUT, according to the information in the CDF entry.
 - b. Configure the DUT to send the selected character or character sequence. Use the Application or other software on the DUT as described in the CDF with field **CDF_UCP_SEND_APPLICATION**.
 - c. Trigger the sending of the character or character sequence by user action on the DUT.
 - d. Test Equipment displays all bits and interpretation of the UCP payload(s) it receives.
 - e. If the Test Equipment displays "Mismatch in received UCP information", then FAIL.
 - f. .
5. If DUT passes all iterations of the test, then PASS. Else FAIL.

5.3 CBUS Tests

Test the CBUS timing and functionality of the Dongle DUT. Many of the tests in this section use common test equipment setups and common methodologies (listed in 5.3.1). These common elements are delineated before the individual tests.

Note 1: Preparation: Dongles may be in an Unpowered or Standby state prior to testing. Except where indicated, testing assumes that a suitable method, as specified in the CDF, is used to put the Dongle into an Active state before testing begins.

Note 2: Typical resistances, capacitances, and timings: Where called out, use $Z_{CBUS_SOURCE_DISCOVER\{typ\}}$, $Z_{CBUS_SOURCE_ON\{typ\}}$, C_{CBUS} =minimum possible tester capacitance and $T_{BIT_CBUS\{typ\}}$.

Note 3: Continuous monitoring: Continuous monitoring assumes that a parameter measurement is checked continuously for the duration of the testing. Continuous monitoring is usually applied to detect out of range voltage levels and timings.

5.3.1 CBUS Dongle DUT Common Test Equipment Setups

Each of the tests in Section 4.3 uses one of the common test setups in this section or defines a unique test setup. Refer back to this section for the list of equipment for a particular test using the hyperlink if one is shown in the individual test definition.

5.3.2 CBUS Dongle DUT Common Required Methodologies

Many of the tests in Section 3.3 use a common methodology, followed by test steps specific to the individual test definition. Refer back to this section for the initial test steps using the hyperlink if one is shown in the individual test definition.

5.3.2.1 Tester_Source_Discovery Procedure

- 1) Start Continuous Monitor procedures.
- 2) Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
- 3) Set Tester Source port to not drive the VBUS.
- 4) Set Tester Source to drive high impedance pull-downs on MHL+ and MHL- for observability.
- 5) Set Tester Source to set the CBUS to high impedance for $T_{SRC_CBUS_FLOAT}$.
- 6) Set Tester Source to apply $Z_{CBUS_SRC_DISCOVER\{typ\}}$ pull-up resistor to CBUS.
- 7) Tester Source will pause until it thinks it sees a $Z_{CBUS_DONGLE_DISCOVER}$ resistor.
- 8) If Tester Source does not detect Dongle DUT driving VBUS, then Tester Source will drive VBUS itself, then wait $T_{SRC_VBUS_OUT_TO_STABLE}$.
- 9) Set Tester Source to start normal Device Discovery by sending wake pulses.
- 10) Tester Source then drives Device Discovery pulses.
- 11) After discovery completes, Tester Source will change its CBUS pull-up to $Z_{CBUS_SRC_ON\{typ\}}$.

5.3.2.2 Tester_Source_R_Discovery Procedure

- 1) Start Continuous Monitor procedures.
- 2) Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
- 3) Set Tester Source port to not drive the VBUS.
- 4) Set Tester Source to drive high impedance pull-downs on MHL+ and MHL- for observability.
- 5) Set Tester Source to set the CBUS to high impedance for $T_{SRC_CBUS_FLOAT}$.
- 6) Set Tester Source to apply $Z_{CBUS_SRC_DISCOVER\{typ\}}$ pull-up resistor to CBUS.
- 7) Tester Source will pause until it thinks it sees a $Z_{CBUS_DONGLE_DISCOVER}$ resistor.
- 8) If Tester Source does not detect Dongle DUT driving VBUS, then Tester Source will drive VBUS itself, then wait $T_{SRC_VBUS_OUT_TO_STABLE}$.

- 9) Set Tester Source to start normal Device Discovery by sending wake pulses.
- 10) Tester Source then drives Device Discovery pulses.
- 11) After discovery completes, Tester Source will change its CBUS pull-up to TEST_RESISTOR.

5.3.2.3 *Tester_Source_Modified_Discovery_Procedure*

- 1) Start Continuous Monitor procedures.
- 2) Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
- 3) Set Tester Source port to not drive the VBUS.
- 4) Set Tester Source to drive high impedance pull-downs on MHL+ and MHL- for observability.
- 5) Set Tester Source Wake Pulse HIGH time to TEST_WAKE_PULSE_HIGH.
- 6) Set Tester Source Wake Pulse LOW time to TEST_WAKE_PULSE_LOW.
- 7) Set Tester Source Wake Pulse gap time to TEST_WAKE_PULSE_GAP.
- 8) Set Tester Source Wake Pulse discover time to TEST_WAKE_PULSE_DISCOVER.
- 9) Set Tester Source discovery pulse HIGH time to TEST_DISCOVERY_PULSE_HIGH.
- 10) Set Tester Source discovery pulse LOW time to TEST_DISCOVERY_PULSE_LOW.
- 11) Set first discovery pulse to have HIGH width of TEST_FIRST_DISCOVERY_PULSE_HIGH.
- 12) Set first discovery pulse to have LOW width of TEST_FIRST_DISCOVERY_PULSE_LOW.
- 13) Set fifth discovery pulse to have HIGH width of TEST_FIFTH_DISCOVERY_PULSE_HIGH.
- 14) Set fifth discovery pulse to have LOW width of TEST_FIFTH_DISCOVERY_PULSE_LOW.
- 15) If called out, set Tester Source Discovery Engine to send TEST_EXTRA_PULSES discovery pulses.
The default sends 6 discovery pulses.
- 16) Set Tester Source to set CBUS to high impedance for $T_{SRC_CBUS_FLOAT}$.
- 17) Set Tester Source to apply $Z_{CBUS_SRC_DISCOVER\{typ\}}$ pull-up resistor to CBUS.
- 18) Tester Source will pause until it thinks it sees a $Z_{CBUS_DONGLE_DISCOVER}$ resistor.
- 19) If Tester Source does not detect Dongle DUT driving VBUS, then Tester Source will drive VBUS itself, then wait $T_{SRC_VBUS_OUT_TO_STABLE}$.
- 20) Set Tester Source to start normal device discovery by sending wake pulses.
- 21) Tester Source then drives Device Discovery pulses.
- 22) After discovery completes, Tester Source will change its CBUS pull-up to $Z_{CBUS_SRC_ON\{typ\}}$.

5.3.2.4 Tester Source Device Capability Registers

Unless changed in specific tests, the Tester will exhibit the following Device Capability Register contents:

Table 5-1. Tester Source Device Capability Registers

| | Register Name | Register Value | Description |
|----|-----------------|-------------------------------|--|
| 0 | DEV_STATE | 0x00 | Reserved |
| 1 | MHL_VERSION | 0x20 | Level of MHL Spec supported |
| 2 | DEV_CAT | 0x02 | Identify the type of MHL system (Source) |
| 3 | ADOPTER_ID_H | Set by Equipment Manufacturer | High-order byte of Adopter ID |
| 4 | ADOPTER_ID_L | Set by Equipment Manufacturer | Low-order byte of Adopter ID |
| 5 | VID_LINK_MODE | 0x37 | List of link modes supported for Video |
| 6 | AUD_LINK_MODE | 0x03 | List of link modes supported for Audio |
| 7 | VIDEO_TYPE | 0x80 | Video Type Support |
| 8 | LOG_DEV_MAP | 0x06 | Logical Device Map |
| 9 | BANDWIDTH | 0x0F | Link Bandwidth limit |
| 10 | FEATURE_FLAG | 0x07 | Flags for MHL Optional feature |
| 11 | DEVICE_ID_H | Set by Equipment Manufacturer | High-order byte of system identifier |
| 12 | DEVICE_ID_L | Set by Equipment Manufacturer | Low-order byte of system identifier |
| 13 | SCRATCHPAD_SIZE | 0x10 | Count of Scratchpad registers |
| 14 | INT_STAT_SIZE | 0x33 | Count of interrupt and status registers |
| 15 | No Name | 0x00 | Reserved |

5.3.3 Link Layer Electrical – Dongle: Absolute Maximum Voltages

Monitor the voltages on VBUS and CBUS wires continuously to make sure they never exceed their absolute maximum values.

5.3.3.1 Common Test Environment

5.3.3.1.1 Test Objective

5.3.3.1.2 References

5.3.3.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.3.1.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. Set Tester Source VBUS Limit comparator to appropriate voltage.
2. Set Tester Source CBUS Limit comparator to appropriate voltage.
3. Set Tester Source to report VBUS Limit errors. If VBUS limit is violated, then FAIL.

4. Set Tester Source to report CBUS Limit errors. If CBUS limit is violated, then FAIL.
5. If no limits are violated, then PASS, else FAIL.

5.3.3.2 CBE-Dongle: VBUS Absolute Maximum Positive Voltage

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.3.2.1 Test Objective

Continuous monitor the VBUS wire to make sure it never exceeds the absolute maximum positive voltage.

5.3.3.2.2 References

[MHL] Section 13.9.1.1; Table 13-21: $V_{VBUS_TP2_SINK_DRV}$; $V_{VBUS_TP1_SRC_DRV}$

5.3.3.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.3.2.4 Required Methodology

Continuous monitor runs in the background.

FAIL if VBUS positive limit is exceeded.

If all preceding steps pass, then PASS; else FAIL.

5.3.3.3 CBE-Dongle: CBUS Absolute Maximum Positive Voltage

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.3.3.1 Test Objective

Continuous monitor the CBUS wire to make sure it never exceeds the absolute maximum positive voltage.

5.3.3.3.2 References

[MHL] Section 13.10.1; Table 13-23: CBUS Absolute Max Voltage

5.3.3.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.3.3.4 Required Methodology

Continuous monitor runs in the background.

FAIL if CBUS positive limit is exceeded.

If all preceding steps pass, then PASS; else FAIL.

5.3.4 Link Layer Electrical – Dongle DUT Output: Discovery Impedance

Verify that a Dongle DUT in DONGLE0 state which is supposed to have a valid $Z_{CBUS_DONGLE_DISCOVER}$ while $VBUS == 0$ exhibits the correct resistance at various CBUS voltages.

5.3.4.1 CBE-Dongle: Powered-Off $Z_{CBUS_DONGLE_DISCOVER}$

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.4.1.1 Test Objective

Verify that Dongle DUT $Z_{CBUS_DONGLE_DISCOVER}$ is within proper limits when no power is applied.

5.3.4.1.2 References

[MHL] Section 8.2.3; Table 8-5; DONGLE0

[MHL] Section 13.10.1; Table 13-27; $Z_{CBUS_DONGLE_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-30: $T_{SRC_CBUS_FLOAT}$

5.3.4.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.4.1.4 Required Methodology

Execute the following test procedure once for each row in Table 5-2, substituting TEST_CBUS_VOLTAGE:

1. If procedure **CDF_PROC_SET_STANDBY** is empty in the CDF, then PASS (SKIP).
2. If procedure **CDF_PROC_SET_STANDBY** does not put the DUT into state DONGLE9, then PASS (SKIP).
3. If necessary, connect Dongle DUT to Tester Source port.
4. For the duration of this test, disconnect the Dongle DUT from the downstream device, and disconnect power from the Dongle DUT.
5. Set Tester Source port to disconnected state.
6. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
7. Tester does not drive VBUS.
8. Tester Source applies TEST_CBUS_VOLTAGE to CBUS through $Z_{CBUS_SRC_DISCOVER}\{typ\}$.
9. Tester Source measures the Dongle DUT discovery resistance.
10. FAIL if DUT Pull-down resistor is not in valid range ($800\text{ Ohm} \leq Z_{CBUS_DONGLE_DISCOVER} \leq 1200\text{ Ohm}$).
11. If all preceding steps pass, then PASS; else FAIL.

Table 5-2. Powered-Off $Z_{CBUS_DONGLE_DISCOVER}$

| | TEST_CBUS_VOLTAGE | Purpose |
|---|-------------------|--|
| 1 | 1.9V | $Z_{CBUS_DONGLE_DISCOVER}$ @ VBUS == 0V, $0.262V \leq$ CBUS measured voltage $\leq 0.367V$ |

5.3.4.2 CBE-Dongle: VBUS-Powered $Z_{CBUS_DONGLE_DISCOVER}$

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.4.2.1 Test Objective

Verify that Dongle DUT $Z_{CBUS_DONGLE_DISCOVER}$ is within proper limits when powered by the Source VBUS.

5.3.4.2.2 References

[MHL] Section 8.2.3; Table 8-5; DONGLE0

[MHL] Section 13.10.1; Table 13-27; $Z_{CBUS_DONGLE_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-30: $T_{SRC_CBUS_FLOAT}$

5.3.4.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.4.2.4 Required Methodology

Execute the following test procedure once for each row in Table 5-3, substituting TEST_CBUS_VOLTAGE:

1. If the CDF value CDF_D_ACCEPTS_POWER_FROM_SOURCE indicates that the DUT cannot be powered by the Tester Source, then end the test with PASS (SKIP).
2. If the CDF value **CDF_D_POWERED** indicates that the DUT is a Powered Dongle, disconnect power from the Dongle DUT.
3. If necessary, connect Dongle DUT to Tester Source port.
4. If necessary, connect the Dongle DUT to the downstream device.
5. Set Tester Source port to disconnected state.
6. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
7. Tester drives the VBUS, then waits $T_{SRC_VBUS_OUT_TO_STABLE}\{min\}$.
8. Tester Source applies TEST_CBUS_VOLTAGE to CBUS through $Z_{CBUS_SRC_DISCOVER}\{typ\}$.
9. Tester Source measures the Dongle DUT discovery resistance.
10. FAIL if DUT Pull-down resistor is not in valid range ($800\text{ Ohm} \leq Z_{CBUS_DONGLE_DISCOVER} \leq 1200\text{ Ohm}$).
11. If all preceding steps pass, then PASS; else FAIL.

Table 5-3. VBUS-Powered $Z_{CBUS_DONGLE_DISCOVER}$

| | TEST_CBUS_VOLTAGE | Purpose |
|---|-------------------|---|
| 1 | 1.9V | $Z_{CBUS_DONGLE_DISCOVER}$ @ VBUS == 0V, $0.262V \leq \text{CBUS measured voltage} \leq 0.367V$ |

5.3.4.3 CBE-Dongle: Locally-Powered $Z_{CBUS_DONGLE_DISCOVER}$

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.4.3.1 Test Objective

Verify that Dongle DUT $Z_{CBUS_DONGLE_DISCOVER}$ is within proper limits when powered by its own power supply.

5.3.4.3.2 References

[MHL] Section 8.2.3; Table 8-5; DONGLE0

[MHL] Section 13.10.1; Table 13-27; $Z_{CBUS_DONGLE_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-30: $T_{SRC_CBUS_FLOAT}$

5.3.4.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.4.3.4 Required Methodology

Execute the following test procedure once for each row in Table 5-4, substituting TEST_CBUS_VOLTAGE:

1. If the CDF value **CDF_D_POWERED** indicates that the DUT is a Powered Dongle, then continue, else end test with PASS (SKIP).
2. If necessary, connect Dongle DUT to Tester Source port.
3. If necessary, connect the Dongle DUT to the downstream device, and connect power to the Dongle DUT.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Tester does not drive VBUS.
7. Tester Source applies TEST_CBUS_VOLTAGE to CBUS through $Z_{CBUS_SRC_DISCOVER}\{typ\}$.
8. Tester Source measures the Dongle DUT discovery resistance.

9. FAIL if DUT Pull-down resistor is not in valid range ($800 \text{ Ohm} \leq Z_{\text{CBUS_DONGLE_DISCOVER}} \leq 1200 \text{ Ohm}$).
10. If all preceding steps pass, then PASS; else FAIL.

Table 5-4. Self-Powered $Z_{\text{CBUS_DONGLE_DISCOVER}}$

| | TEST_CBUS_VOLTAGE | Purpose |
|---|-------------------|---|
| 1 | 1.9V | $Z_{\text{CBUS_DONGLE_DISCOVER}}$ @ $\text{VBUS} == 0\text{V}$, $0.262\text{V} \leq \text{CBUS measured voltage} \leq 0.367\text{V}$ |

5.3.5 Link Layer Timing – Dongle DUT Output: Pre-Discovery

Measure the timing of the Dongle DUT as it optionally signals the Source that a discovery sequence is invited.

5.3.5.1 CBT-Dongle: Time from Dongle Power applied until Dongle CBUS leaves HIGH-Z

Application: Dongle

5.3.5.1.1 Test Objective

Verify that Dongle DUT notices that it has transitioned from No Dongle Power applied to Dongle Power applied, and exhibits first $I_{\text{CBUS_LEAK_SINK}}$, then a valid $Z_{\text{CBUS_DONGLE_DISCOVER}}$.

Behavior is required for a Dongle, but it only leaves HIGHZ mode when it can meet other timing specs.

5.3.5.1.2 References

- [MHL] Section 7.8.1.3; Table 7-25; POW
- [MHL] Section 8.2.3 (DONGLE0 -> DONGLE4)
- [MHL] Section 13.10.1.1; Table 13-30; $T_{\text{SINK_CBUS_FLOAT}}$
- [MHL] Section 13.10.1.2; Table 13-32; $T_{\text{DONGLE_DCAP_CHG}}$
- [MHL] Section 13.10.1; Table 13-27; $I_{\text{CBUS_LEAK_SINK}}$
- [MHL] Section 13.10.1; Table 13-29; $T_{\text{SINK_READY_TO_DISCOVER}}$

5.3.5.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.5.1.4 Required Methodology

1. If the CDF value **CDF_D_POWERED** indicates that the DUT is a Powered Dongle, then continue, else end test with PASS (SKIP).
2. If necessary, connect Dongle DUT to Tester Source port.
3. If necessary, connect the Dongle DUT to the Downstream device.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Tester pulls the CBUS HIGH with a large resistance (100K) for observability.
7. Remove the Dongle's Power from the Dongle DUT for at least the Dongle Power-down time in the CDF value **CDF_D_MAX_POWER_DOWN**.
8. Apply external power source to the Dongle DUT.
9. FAIL if DUT CBUS does not enter HIGH-Z state within the Dongle Power-up time in the CDF value **CDF_D_MAX_POWER_UP**.
10. FAIL if DUT does enter HIGH-Z state but does not remain there for at least $T_{\text{SINK_CBUS_FLOAT}}$.
11. FAIL if Dongle DUT does not assert a valid $Z_{\text{CBUS_DONGLE_DISCOVER}}$ after the float period, but within the time specified in the CDF value **CDF_D_MAX_STANDBY_TO_ACTIVE**.
12. Set Tester Source to use its Source Packet engine to ACK everything.

13. Execute the Tester_Source_Discovery procedure.
14. FAIL if Dongle DUT does not post a DCAP_CHG interrupt within $T_{\text{DONGLE_DCAP_CHG}}$ of the re-assertion of $Z_{\text{CBUS_DONGLE_DISCOVER}}$.
15. Execute the Tester_Wait_Capability_Registers_Valid procedure.
16. FAIL if the procedure indicates TIMEOUT.
17. Tester Source does READ_DEVCAP of register DEV_CAT.
18. FAIL if READ_DEVCAP fails.
19. FAIL if POW bit is not 1.
20. FAIL if VBUS does not stay HIGH, driven by the Powered Dongle.
21. Note the presentation of $Z_{\text{CBUS_DONGLE_DISCOVER}}$ can be delayed arbitrarily by the Dongle not being ready.
22. If all preceding steps pass, then PASS; else FAIL.

5.3.6 Link Layer Electrical – Dongle DUT Output: Arbitration/Sync/Data Signaling

Measure the electrical behavior of the Dongle DUT as it drives Arbitration, Sync, and Data pulses.

5.3.6.1 CBE-Dongle: Post-Discovery Passive Pulldown $Z_{\text{CBUS_DONGLE_ON}}$ Resistance

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.6.1.1 Test Objective

Verify that Dongle DUT $Z_{\text{CBUS_DONGLE_ON}}$ has correct value.

5.3.6.1.2 References

[MHL] Section 13.10.1; Table 13-27; $Z_{\text{CBUS_DONGLE_ON}}$

5.3.6.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.6.1.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Wait $T_{\text{SINK_CONN}}\{\text{max}\}$ from the rising edge of the last discovery pulse.
8. Measure impedance on dongle side CBUS as $Z_{\text{CBUS_MEAS}}$.
9. FAIL if $Z_{\text{CBUS_MEAS}}$ is not within the valid $Z_{\text{CBUS_DONGLE_ON}}$ range.
10. If all preceding steps pass, then PASS; else FAIL.

5.3.6.2 CBE-Dongle: CBUS Capacitance

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.6.2.1 Test Objective

Verify that the Dongle DUT has a low-enough input capacitance.

5.3.6.2.2 References

[MHL] Section 13.10.1; Table 13-27: $C_{\text{DONGLE_CBUS}}$

5.3.6.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.6.2.4 Required Methodology

The “Dongle Expected Capacitance” is found in the CDF value **CDF_D_MAX_CBUS_CAP**.

Characterize the capacitance of the Test Equipment, so that it can be subtracted from any measurement of DUT + Test Equipment.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Measure the CBUS Capacitance (C_{CBUS_MEAS}) after DISCOVERY Completes.
8. FAIL if the calculated Dongle capacitance is greater than C_{DONGLE_CBUS} .
9. If all preceding steps pass, then PASS; else FAIL.

NOTE: The capacitance measurement may depend on compliant termination of the CBUS line in the DUT. The test may result in a FAIL if the resistance $Z_{CBUS_SINK_ON}$ is out of range.

5.3.6.3 CBE-Dongle: Arbitrate/Sync/Data Drive LOW Voltage

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.6.3.1 Test Objective

Verify that Dongle DUT drives Arbitration, Sync, and Data Pulses with the correct DRIVE LOW voltage.

5.3.6.3.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OL_CBUS}

5.3.6.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.6.3.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source $Z_{CBUS_SRC_ON\{min\}}$ (4K).
8. Observe the LOW levels in the waveform while the Dongle DUT arbitrates for the bus.
9. Set Tester Source to send a GET_STATE to Dongle DUT.
10. FAIL if DUT does not eventually arbitrate for the bus to send a data packet to the Tester Source.
11. FAIL if DUT does not drive CBUS LOW during the Arbitration, Sync, and Data pulses to a voltage below $V_{OL_CBUS\{max\}}$.
12. If all preceding steps pass, then PASS; else FAIL.

5.3.6.4 CBE-Dongle: Arbitrate/Sync/Data Drive HIGH Voltage

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.6.4.1 Test Objective

Verify that Dongle DUT drives Arbitration, Sync, and Data Pulses with correct DRIVE HIGH voltage.

5.3.6.4.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OH_CBUS}

5.3.6.4.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.6.4.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source $Z_{CBUS_SRC_ON}$ to maximum value (6K).
8. Observe the HIGH levels in the waveform while the Dongle DUT arbitrates for the bus.
9. Set Tester Source to send a GET_STATE to Dongle DUT.
10. FAIL if DUT does not eventually arbitrate for the bus to send a data packet to the Tester Source.
11. FAIL if DUT does not drive CBUS HIGH after the Arbitration, Sync and Data pulses to a voltage above $V_{OH_CBUS}\{min\}$ and below $V_{OH_CBUS}\{max\}$.
12. If all preceding steps pass, then PASS; else FAIL.

Note: $V_{OH_CBUS}\{max\}$ should be measured at the end of the CBUS drive high time $T_{DRV_HI_CBUS}\{max\}$ to avoid failing devices due to impedance/driver signal overshoot.

5.3.7 Link Layer Timing – Dongle DUT Output: Arbitration/Sync/Data in Nanoseconds

Measure the timing of the Dongle DUT as it drives Arbitration and data pulses.

The Dongle DUT output Bit Time must be measured.

5.3.7.1 CBT-Dongle: Arbitration/Sync/Data Active Drive HIGH Duration

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.7.1.1 Test Objective

Verify that Dongle DUT drives CBUS HIGH, then releases bus, when changing the CBUS from LOW to HIGH. (The HIGH voltage will be sustained by a Resistor pull-up at the Source end.)

5.3.7.1.2 References

[MHL] Section 7.2.4

[MHL] Section 13.10.1; Table 13-23: $T_{DRV_HI_CBUS}$

5.3.7.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.7.1.4 Required Methodology

Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Adjust the Tester resistances and voltages so that the DUT Drive High time is observable.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a command from the DUT, reply with a link level ACK.
9. Measure the timing of the packets from the Dongle DUT in nanoseconds.
10. FAIL if DUT has problems doing Discovery.
11. FAIL if DUT does not arbitrate for access to CBUS to send a command.
12. FAIL if the CBUS Drive HIGH time (the period starting when the CBUS goes from LOW to HIGH and lasting until only the Source resistor pull-up drives the CBUS) is outside the limits set for $T_{DRV_HI_CBUS}$.
13. Note this test covers driving HIGH at the end of the Arbitration bit time, the Sync bit time, and at rising edges throughout the data period including the rising edge transition at the mid-point of a bit time when a DATA 1 is being sent on the wire.
14. If all preceding steps pass, then PASS; else FAIL.

This test does not check the Drive HIGH time for ACK pulses, since this test does not result in Source-driven ACKs.

5.3.7.2 CBT-Dongle: Arbitration/Sync/Data Edge Rate

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.7.2.1 Test Objective

Verify that Dongle DUT drives Arbitration/Sync/Data and ACK Pulses which have valid Rise or Fall times.

5.3.7.2.2 References

[MHL] Section 13.10.1; Table 13-23: V_{OH_CBUS}

[MHL] Section 13.10.1; Table 13-23: V_{OL_CBUS}

[MHL] Section 13.10.1; Table 13-23: T_{F_CBUS}

[MHL] Section 13.10.1; Table 13-23: T_{R_CBUS}

5.3.7.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.7.2.4 Required Methodology

Execute the following test procedure once for each row in Table 5-5, substituting TEST_CAPACITOR and TEST_RESISTOR. Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Connect Oscilloscope to Tester to allow it to monitor the Dongle DUT CBUS wire.
4. Set Oscilloscope to trigger on a rising edge at about 1V.
5. Set Tester Source port to disconnected state.
6. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
7. Set Tester Source capacitance to TEST_CAPACITOR.

8. Set Tester Source to use its Source Packet engine to ACK everything.
9. Execute the Tester_Source_R_Discovery procedure.
10. When the Tester Source receives a command, reply with a link level ACK.
11. FAIL if DUT has problems doing Discovery.
12. FAIL if DUT does not arbitrate for access to CBUS to send a command.
13. For Test 1 FAIL if Arbitration/Sync/Data pulses have a rise time of less than $T_{R_CBUS}\{\min\}$.
14. For Test 2 FAIL if Arbitration/Sync/Data pulses have a fall time of less than $T_{F_CBUS}\{\min\}$.
15. For Test 3 FAIL if Arbitration/Sync/Data pulses have a rise time of greater than $T_{R_CBUS}\{\max\}$.
16. For Test 4 FAIL if Arbitration/Sync/Data pulses have a fall time of greater than $T_{F_CBUS}\{\max\}$.
17. FAIL if Rise times and Fall times differ by more than the allowed amount: ΔT_{RF} .
18. If all preceding steps pass, then PASS; else FAIL.

Table 5-5. Arbitration/Sync/Data Edge Rate

| | TEST_CAPACITOR | TEST_RESISTOR | Purpose |
|---|--------------------|--------------------------------|-------------------------------------|
| 1 | minimum Tester cap | $Z_{CBUS_SOURCE_ON}\{\max\}$ | Arbitration/Sync/Data Rise time MIN |
| 2 | minimum Tester cap | $Z_{CBUS_SOURCE_ON}\{\min\}$ | Arbitration/Sync/Data Fall time MIN |
| 3 | minimum Tester cap | $Z_{CBUS_SOURCE_ON}\{\min\}$ | Arbitration/Sync/Data Rise time MAX |
| 4 | minimum Tester cap | $Z_{CBUS_SOURCE_ON}\{\max\}$ | Arbitration/Sync/Data Fall time MAX |

5.3.8 Link Layer Timing – Dongle DUT Output: Arbitration/Sync/Data in Bit Times

Measure the timing of the Dongle DUT as it drives Data pulses.

The Dongle DUT output Bit Time must be measured based on observing CBUS edges.

5.3.8.1 CBT-Dongle: Arb, Sync, Data HIGH and LOW Timing

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.8.1.1 Test Objective

Verify that the Dongle DUT does not arbitrate to send a first Packet over the CBUS until $T_{SINK_ARBITRATE}$ after discovery completes.

Verify that the Dongle DUT drives a CBUS LOW arbitration pulse for T_{ARB_SINK} . Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Dongle DUT drives CBUS HIGH before Sync pulse after winning arbitration. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Dongle DUT drives CBUS LOW to make a Sync pulse after winning arbitration. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Verify that Dongle DUT drives CBUS HIGH after the Sync pulse until it starts driving data. Timings are expressed in nanoseconds here. They are expressed in Bit Times in a later test step.

Check the timing for Arbitration pulses in terms of “Bit Times”, and verify that it meets the required spec.

Check the Idle period between Arbitration and the Sync pulse in terms of “Bit Times”, and verify that it meets the required spec.

Check the Sync LOW pulse in terms of “Bit Times”, and verify that it meets the required spec.

Check the delay between the Sync LOW end and the first Data transition in terms of “Bit Times”, and verify that it meets the required spec.

Check the timing of 0 and 1 bits, taking care to observe duty-cycle and edge rate differences.
Measure the timings using the method described in Section 2.5.1.1.

5.3.8.1.2 References

[MHL] Section 8.5.1

[MHL] Section 13.10.1.1; Table 13-31; $T_{\text{SINK_ARBITRATE}}$

[MHL] Section 7.2.2

[MHL] Section 13.10.2; Table 12-33; $T_{\text{ARB_SINK}}$

[MHL] Section 7.2.4

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$

[MHL] Section 13.10.2; Table 13-33; $T_{\text{ARB_SRC}}$

[MHL] Section 13.10.2; Table 13-33; $T_{\text{REQ_OPP}}$

[MHL] Section 13.10.2; Table 12-33; $T_{\text{ARB_TOT}} - T_{\text{ARB_SINK}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{SYNC_TOTAL}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{SYNC_DUTY}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{CBUS_DUTY}}$

[MHL] Section 13.10.1; Table 13-23; ΔT_{RF}

5.3.8.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.8.1.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a command, reply with a link level ACK.
9. Measure the timing of the packets from the Dongle DUT in nanoseconds.
10. Measure timing information in nanoseconds and assign values to a set of named timing variables:
 - a. ARB_LOW time
 - b. Idle_Time
 - c. Sync_LOW time
 - d. Sync_HIGH time
 - e. HIGH_0 time
 - f. LOW_0 time
 - g. Starting_LOW_1 timings
 - h. Starting_HIGH_1 timings

11. FAIL if Dongle does CBUS arbitration sooner than $T_{\text{SINK_ARBITRATE}}$ from the end of Discovery.
12. FAIL if CBUS ARB LOW time is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * $T_{\text{ARB_SINK}}$.
13. FAIL if CBUS HIGH time after the arbitration LOW time but before Sync LOW time is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * $(3 + T_{\text{REQ_OPP}})$.
14. FAIL if CBUS LOW time during Sync is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * (minimum or maximum $T_{\text{SYNC_DUTY}}$).
15. FAIL if CBUS HIGH time between Sync LOW time and falling edge of first Data bit is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * $(T_{\text{SYNC_TOTAL}} - (\text{minimum or maximum } T_{\text{SYNC_DUTY}}))$.
16. FAIL if time between Sync going LOW time and falling edge of first Data bit is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * $(T_{\text{SYNC_TOTAL}})$.
17. FAIL if DUT does not drive Data 0 bits which are asserted entirely LOW on the CBUS for the required period (in nanoseconds).
18. FAIL if DUT does not drive Data 0 bits which are asserted entirely HIGH on the CBUS for the required period (in nanoseconds).
19. FAIL if DUT does not drive Data 1 bits which start HIGH for the first half of a bit time for the required HIGH period (in nanoseconds).
20. FAIL if DUT does not drive Data 1 bits which start HIGH for the first half of a bit time for the following required LOW period (in nanoseconds).
21. FAIL if DUT does not drive Data 1 bits which start LOW for the first half of a bit time for the required LOW period (in nanoseconds).
22. FAIL if DUT does not drive Data 1 bits which start LOW for the first half of a bit time for the following required HIGH period (in nanoseconds).
23. FAIL if DUT does not drive Data 1 bits with required symmetry.
24. Calculate Bit Time range based on Arbitration pulse LOW.
25. Calculate Bit Time range based on HIGH period after the arbitration pulse.
26. Calculate Bit Time range based on Sync pulse LOW.
27. Calculate Bit Time range based on "1" bit timing.
28. Calculate Bit Time range based on pairs of "0" bit timings.
29. FAIL if the bit time ranges based on Arbitration, Sync, and Data bits are not consistent.
30. If all preceding steps pass, then PASS; else FAIL.

5.3.8.2 CBT-Dongle: Bit Timing Variation within a Packet

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.8.2.1 Test Objective

Continuously monitor Dongle DUT individual received packets to verify that Bit Timing does not change too quickly.

5.3.8.2.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_VARY_PACKET}}$

5.3.8.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.8.2.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

This continuous monitor will be disabled during any test which sets the CBUS LOW Limit comparator to a value different than the CDF value .

1. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
2. Continuously monitor the CBUS to look for failures of this spec.
3. FAIL if the bit timing of any two consecutive bits within a packet differ by more than $T_{BIT_VARY_PACKET}$.
4. If all preceding steps pass, then PASS; else FAIL.

5.3.9 Link Layer Timing – Dongle DUT Output: Link Level NACK

Verify that Dongle DUT is willing to retransmit data the required number of times when it receives link level NACKs.

5.3.9.1 CBT-Dongle: Response to Link Level NACK

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.9.1.1 Test Objective

Verify that Dongle DUT is willing to retransmit data the required number of times when it receives NACKs.

5.3.9.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.2; Table 13-33; N_{RETRY}

5.3.9.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.9.1.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to NACK the first $N_{RETRY}\{min\}-1$ packets from the Dongle DUT, then to ACK all subsequent packets.
6. Execute the Tester_Source_Discovery procedure.
7. When the Tester Source receives a command, it responds with a link layer NACK up to $N_{RETRY}\{min\}-1$ times.
8. FAIL if DUT does not eventually issue a command.
9. FAIL if DUT does not release the bus based on the link level NACK.
10. FAIL if DUT does not retry the command the required minimum number of times.
11. FAIL if DUT does not stop retrying the command based on the link level ACK.
12. If all preceding steps pass, then PASS; else FAIL.

Note that the Dongle DUT might NACK the ACK Packet. Tester needs to re-issue this command up to $N_{RETRY}\{min\}$ times to give the Dongle DUT a chance to ACK it.

5.3.10 Link Layer Timing – Dongle DUT Output: Link Level ACK

Measure the timing of the Dongle DUT as it drives link level ACK pulses.

The Dongle DUT output Bit Time must be measured.

5.3.10.1 CBT-Dongle: ACK Output Timing in Nanoseconds

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.10.1.1 Test Objective

Measure when Dongle DUT drives link level ACK when Tester uses various Source Bit timing and various Parities.

5.3.10.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 7.2.4

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$ Max-Min

[MHL] Section 7.2.4.1; Table 13-33; $T_{\text{REQ_CONT}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{CBUS_ACK_FALL}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{CBUS_ACK_0}}\{\text{min}\}$

5.3.10.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.10.1.4 Required Methodology

Execute the following test procedure once for every row in Table 5-6, substituting TEST_IDLE_PERIOD and TEST_DATA from the table for each execution. Measure the timings using the method described in Section 2.5.1.1.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to link level ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
8. When the Tester Source receives a command, reply with link level ACK.
9. Tester Source does SET_INT to Dongle DUT, releases the bus.
10. Tester Source sends TEST_DATA for an offset, releases the bus.
11. Tester Source sends Data of all 0's.
12. FAIL if Dongle does not eventually send a command.
13. FAIL if DUT does not link level ACK all 3 packets sent from the Tester.
14. FAIL if DUT does not drive the link level ACK LOW starting within $T_{\text{CBUS_ACK_FALL}}$ of the end of the Tester-driven packet parity bit.
15. FAIL if DUT drives an ACK LOW pulse which is thinner or wider than $T_{\text{CBUS_ACK_0}}$.
16. If all preceding steps pass, then PASS; else FAIL.

Note that the Dongle DUT might NACK the packets it is receiving. Tester needs to re-issue this command up to N times to give the Dongle DUT a chance to ACK it.

Table 5-6. ACK Output Timing in Nanoseconds

| | TEST_IDLE_PERIOD | TEST_DATA | Purpose |
|---|----------------------------|-----------|--------------------|
| 1 | T _{REQ_OPP} {min} | 0x20 | Measure ACK timing |
| 2 | T _{REQ_OPP} {min} | 0x21 | Measure ACK timing |

5.3.10.2 CBT-Dongle: ACK Drive HIGH Duration.

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.10.2.1 Test Objective

Measure the time the Dongle DUT drives ACK HIGH.

5.3.10.2.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.1; Table 13-23; T_{DRV_HI_CBUS}

5.3.10.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.10.2.4 Required Methodology

Connect the scope to observe the Tester Dongle CBUS.

Either use the scope to examine the CBUS HIGH period, or adjust the CBUS HIGH and CBUS LOW Limit comparators, to separate out the period at the beginning of a HIGH period where the Dongle DUT is actively pulling HIGH to signal ACK.

This HIGH voltage should be higher than the voltage throughout the rest of the CBUS HIGH period, where only a Source-side resistor is pulling the bus up.

Execute the following test procedure once for every row in Table 5-7, substituting TEST_IDLE_PERIOD and TEST_DATA.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Adjust the Tester resistances and voltages so that the DUT Drive High time is observable.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a command, reply with a link level ACK.
9. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
10. Tester Source does SET_INT to Dongle DUT, releases the bus.
11. Tester Source sends TEST_DATA for an offset, releases the bus.
12. Tester Source sends Data of all 0's.
13. Observe post-discovery Dongle DUT link-level ACK activity as it replies to the Tester Dongle command.
14. Observe that the Dongle DUT replies with an ACK packet.
15. FAIL if the CBUS Drive HIGH time during the ACK period (the period starting when the CBUS goes from LOW to HIGH and lasting until only the Source resistor pull-up drives the CBUS) is outside the limits set for T_{DRV_HI_CBUS}.
16. If all preceding steps pass, then PASS; else FAIL.

Table 5-7. ACK Drive HIGH Duration.

| | TEST_IDLE_PERIOD | TEST_DATA | Purpose |
|---|----------------------------|-----------|--------------------|
| 1 | T _{REQ_OPP} {min} | 0x20 | Measure ACK timing |

5.3.11 Link Layer Timing – Dongle DUT Output: Bus Re-Arbitration

Measure the behavior and timing of the Dongle DUT as it re-arbitrates for the bus.

The Dongle DUT output Bit Time must be measured.

5.3.11.1 CBT-Dongle: Dongle uses Case 2 Regular Arbitration after NACK

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.11.1.1 Test Objective

Verify that Dongle DUT backs off after a link-level NACK, and uses Case 2 regular arbitration timing to reacquire the bus.

5.3.11.1.2 References

[MHL] Section 7.2.4; Case 2

[MHL] Section 13.10.2; Table 13-33; T_{REQ_HOLD}

5.3.11.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.11.1.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to NACK each received packet N_{RETRY}{min}-1 times, then link level ACK it.
6. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between V_{IH_CBUS}{min} and V_{IL_CBUS}{max}.
7. Execute the Tester_Source_Discovery procedure.
8. Set Tester Source to send a GET_STATE to Dongle DUT.
9. FAIL if DUT does not eventually arbitrate for the bus to send a data packet to the Tester Source.
10. Carefully measure the Dongle DUT T_{BIT_CBUS} during the retransmissions of the data packet.
11. Finally accept the data packet and link level ACK it.
12. FAIL if DUT does not eventually send a data packet.
13. FAIL if DUT does not release the bus based on the NACK.
14. FAIL if DUT does arbitration (after a NACK) in less than T_{REQ_HOLD}{min} (expressed in DUT Bit Times of the subsequent packet).
15. FAIL if DUT does not eventually complete its command after retransmission.
16. If all preceding steps pass, then PASS; else FAIL.

5.3.11.2 CBT-Dongle: Dongle uses Case 3 Long Re-arbitration when it Gives up the Bus

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.11.2.1 Test Objective

Verify that Dongle DUT uses Case 3 Long Arbitration whenever it gives up the bus and later re-acquires it.

5.3.11.2.2 References

5.3.11.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.11.2.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK all packets.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source to send a GET_STATE to Dongle DUT.
8. When DUT sends the data packet back, link level ACK it.
9. Use the timing of the observed waveforms to calculate the Dongle DUT $T_{\text{BIT_CBUS}}$.
10. FAIL if DUT does not eventually send a data packet.
11. If DUT never sends a packet, then pauses for re-arbitration, then sends another packet within 10 seconds, end test with PASS (SKIP).
12. FAIL if DUT does re-arbitration (after an ACK) sooner than T_{WAIT} (expressed in Source Bit Times of the subsequent packet).
13. If all preceding steps pass, then PASS; else FAIL.

5.3.11.3 CBT-Dongle: Dongle uses Case 1 Back-to-Back Timing (No Re-arbitration)

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.11.3.1 Test Objective

Verify that Dongle DUT uses correct delay from ACK period start to Sync falling edge on Case 1 back-to-back packet sends.

5.3.11.3.2 References

[MHL] Section 7.2.4; Case 1

[MHL] Section 13.10.2; Table 13-33; $T_{\text{REQ_CONT}}$

[MHL] Section 13.10.2; Table 13-23; $T_{\text{SYNC_TOTAL}}$, $T_{\text{SYNC_DUTY}}$

[MHL] Section 13.10.1; Table 13-23: $T_{\text{DRV_HI_CBUS}}$

5.3.11.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.11.3.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.

2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK all packets.
6. Execute the Tester_Source_Discovery procedure.
7. Set Tester Source to send READ_DEVCAP with offset of 0 to Dongle DUT.
8. After READ_DEVCAP + offset packet is sent, Dongle DUT sends an ACK packet followed by a data packet.
9. Use the timing of the observed waveforms to calculate the Dongle DUT $T_{\text{BIT_CBUS}}$.
10. FAIL if DUT does not respond with an ACK packet followed by a data packet.
11. If DUT does not send back-to-back packets within 10 seconds, end test with PASS (SKIP).
12. FAIL if DUT does not drive a Sync pulse for the data packet within $T_{\text{REQ_CONT}}$ (in Dongle DUT bit times) after the 2-bit-time (measured using the Dongle DUT Bit Time) ACK period.
13. FAIL if CBUS LOW time during Sync is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * (minimum or maximum $T_{\text{SYNC_DUTY}}$).
14. FAIL if CBUS HIGH time between Sync LOW time and falling edge of first Data bit is not between (minimum or maximum $T_{\text{BIT_CBUS}}$) * ($T_{\text{SYNC_TOTAL}} - (\text{minimum or maximum } T_{\text{SYNC_DUTY}})$).
15. If all preceding steps pass, then PASS; else FAIL.

5.3.11.4 CBT-Dongle: Dongle Never Sends Too Many Back-to-Back Packets

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.11.4.1 Test Objective

Continuously monitor the CBUS to verify that the Dongle DUT does not send too many packets back-to-back.

5.3.11.4.2 References

[MHL] Section 7.2.4; Case 4

[MHL] Section 13.10.2; Table 13-33; N_{MAX}

5.3.11.4.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.11.4.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. Continuous monitor the CBUS wire after Discovery to detect Dongle DUT arbitration and packet activity.
2. FAIL if DUT ever sends more than N_{MAX} back-to-back packets without giving up the bus.
3. If all preceding steps pass, then PASS; else FAIL.

5.3.12 Link Layer Timing – Dongle DUT Output: Ill-formed packets

Verify that the Dongle does not send ill-formed packets (for instance, packets shorter than 16 bit-times long).

This set of tests should be run continuously in parallel with many or all of the other tests in the CTS. As continuous monitors, they will detect Dongle misbehavior when a variety of stimulus is applied to the Dongle.

5.3.12.1 CBT-Dongle: Dongle Never Sends Impulse Noise

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.12.1.1 Test Objective

Continuously monitor Dongle DUT to check that it does not drive CBUS LOW for short periods.

5.3.12.1.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$

5.3.12.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.12.1.4 Required Methodology

Continuous monitor runs in the background.

FAIL if DUT seems to be responsible for LOW periods on the CBUS of less than (maybe) 25% of $T_{\text{BIT_CBUS}}$.

If all preceding steps pass, then PASS; else FAIL.

5.3.12.2 CBT-Dongle: Dongle Never Sends Partial Packets

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.12.2.1 Test Objective

Continuously monitor Dongle DUT to check that it does not arbitrate for the CBUS, then send a short packet.

5.3.12.2.2 References

5.3.12.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.12.2.4 Required Methodology

Continuous monitor runs in the background.

FAIL if DUT drives arbitration pulse, then Sync pulse, then less than a full packet consisting of Control, Type, Data, and Parity bits.

If all preceding steps pass, then PASS; else FAIL.

5.3.13 Link Layer Electrical – Dongle DUT Input: Discovery

Verify that Dongle DUT sees Discovery pulses with varying voltage levels.

5.3.13.1 CBE-Dongle: Discovery Sensitivity to Input Voltages

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.13.1.1 Test Objective

Verify that Dongle DUT responds to Wake and Discovery pulses with different voltage settings.

5.3.13.1.2 References

[MHL] Section 8.4

[MHL] Section 13.10.1; Table 13-23; $V_{\text{OH_CBUS}}$, $V_{\text{OL_CBUS}}$

[MHL] Section 13.10.1; Table 13-24; V_{IH_CBUS} , V_{IL_CBUS}

5.3.13.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.13.1.4 Required Methodology

Execute the following test procedure once for each row in Table 5-8, substituting TEST_VOH_CBUS and TEST_VOL_CBUS.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source HIGH output voltage to TEST_VOH_CBUS.
6. Set Tester Source LOW output voltage to TEST_VOL_CBUS.
7. Execute the Tester_Source_Discovery procedure.
8. FAIL if discovery fails.
9. If all preceding steps pass, then PASS; else FAIL.

Table 5-8. Sensitivity to VIH/VIL

| | TEST_VOH_CBUS | TEST_VOL_CBUS | Purpose |
|---|------------------------|-------------------------------|--------------------|
| 1 | $V_{IH_CBUS}\{\min\}$ | $V_{OL_CBUS}\{\min\}$ (0.0V) | Sensitivity to VIH |
| 2 | $V_{OH_CBUS}\{\max\}$ | $V_{IL_CBUS}\{\max\}$ | Sensitivity to VIL |

5.3.14 Link Layer Timing – Dongle DUT Input: Discovery OK

Observe the response of the Dongle DUT as it is presented with Discovery pulses with different speeds and widths.

5.3.14.1 CBT-Dongle: Valid Wake Pulse Timing

Application: Dongle

5.3.14.1.1 Test Objective

Verify that Dongle DUT responds to valid Wake pulses.

5.3.14.1.2 References

NOTE that Dongle DUT may not immediately respond to Wake pulses if it is in DONGLE9 state. This test may need to be repeated after waiting for the DUT to enter DONGLE5 state.

[MHL] Section 13.8.1; Table 13-13; C_{CBUS}

[MHL] Section 13.10.1; Table 13-23; T_{R_CBUS} , T_{F_CBUS}

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_1}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_2}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_TO_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-30; $T_{SRC_PULSE_WIDTH}$

[MHL] Section 13.10.1.1; Table 13-30; $T_{SINK_PULSE_WIDTH}$

[MHL] Section 13.10.1; Table 13-30; T_{SINK_CONN}

[MHL] Section 13.10.1; Table 13-31; $T_{SINK_RXSENSE_EN}$

[MHL] Section 13.6.3; Table 13-9; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$

[MHL] Section 13.7.3; Table 13-12; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$

5.3.14.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.14.1.4 Required Methodology

Execute the following test procedure once for each test case #1 - #7 in Table 5-9 and Table 5-10, substituting for the variables in the leftmost column.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to these variables: $C_{CBUS}\{min\}$, TEST_WAKE_PULSE_HIGH, TEST_WAKE_PULSE_LOW, TEST_WAKE_PULSE_GAP, TEST_WAKE_PULSE_DISCOVER, TEST_DISCOVERY_PULSE_HIGH, TEST_DISCOVERY_PULSE_LOW, TEST_FIRST_DISCOVERY_PULSE_HIGH, TEST_FIRST_DISCOVERY_PULSE_LOW, TEST_FIFTH_DISCOVERY_PULSE_HIGH, and TEST_FIFTH_DISCOVERY_PULSE_LOW
6. Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
7. Set Dongle DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
8. Execute the Tester_Source_Modified_Discovery procedure.
9. FAIL if Source Tester detects that the CBUS does not go LOW after each Wake pulse.
10. FAIL if Source Tester detects that the CBUS does not go LOW after each of the first 5 Discovery pulses.
11. FAIL if Source Tester does not see CBUS stay HIGH after the 6th Discovery pulse.
12. FAIL if DUT does not change its pulldown resistor from $Z_{CBUS_DONGLE_DISCOVER}$ to $Z_{CBUS_DONGLE_ON}$ after the rising edge of the 6th Discovery Pulse plus T_{SINK_CONN} at latest.
13. FAIL if DUT does not pull MHL+ and MHL- up with between $Z_{RXSENSE_TERM}$ and $Z_{RXSENSE_TMDS}$ after the rising edge of the 6th Discovery Pulse plus T_{SINK_CONN} plus $T_{SINK_RXSENSE_EN}$ at latest.
14. If all preceding steps pass, then PASS; else FAIL.

Table 5-9. Wake Pulse Timing 1

| | 1 | 2 | 3 | 4 |
|---------------------------------|---|---|---|---|
| TEST_WAKE_PULSE_HIGH | $T_{SRC_WAKE_PULSE_WIDTH_1}\{max\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{max\}$ |
| TEST_WAKE_PULSE_LOW | $T_{SRC_WAKE_PULSE_WIDTH_1}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{max\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{min\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{max\}$ |
| TEST_WAKE_PULSE_GAP | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{typ\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{min\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| TEST_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |
| TEST_FIRST_ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ | $T_{SINK_PULSE_WIDTH}\{typ\}$ |

| | 1 | 2 | 3 | 4 |
|---------------------------------|---|---|---|---|
| DISCOVERY_PULSE_LOW | | | | |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| Purpose | Minimum Wake Pulse LOW width | Minimum Wake Pulse HIGH width | Minimum Wake Pulse period | Maximum Wake Pulse period |

Table 5-10. Wake Pulse Timing 2

| | 5 | 6 | 7 |
|---------------------------------|--|--|--|
| TEST_WAKE_PULSE_HIGH | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_LOW | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_GAP | $T_{\text{SRC_WAKE_PULSE_WIDTH}_2}\{\text{min}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_2}\{\text{max}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_2}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{min}\}$ | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{min}\}$ | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{max}\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| Purpose | Minimum inter-Wake Pulse gap | Maximum inter-Wake Pulse gap | Maximum Wake Pulse to Discovery Pulse gap |

5.3.14.2 CBT-Dongle: Valid Discovery Pulse Timing

Application: Dongle

5.3.14.2.1 Test Objective

Verify that Dongle DUT responds to valid Discovery pulses.

Tester actively drives Discovery pulses HIGH, but depends on Dongle discovery resistor pulling the bus LOW. The Source does not actively drive LOW during discovery pulses.

Tester Source can make the first Discovery pulse or the fifth Discovery pulse with programmable timing.

5.3.14.2.2 References

NOTE that Dongle DUT may not immediately respond to Wake pulses if it is in DONGLE9 state. This test may need to be repeated after waiting for the DUT to enter DONGLE5 state.

[MHL] Section 13.8.1; Table 13-13; C_{CBUS}

[MHL] Section 13.10.1; Table 13-23; T_{R_CBUS} , T_{F_CBUS}

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_1}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_2}$

[MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_TO_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-30; $T_{SRC_PULSE_WIDTH}$

[MHL] Section 13.10.1.1; Table 13-30; $T_{SINK_PULSE_WIDTH}$

[MHL] Section 13.10.1; Table 13-30; T_{SINK_CONN}

[MHL] Section 13.10.1; Table 13-31; $T_{SINK_RXSENSE_EN}$

[MHL] Section 13.6.3; Table 13-9; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$

[MHL] Section 13.7.3; Table 13-12; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$

5.3.14.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.14.2.4 Required Methodology

Execute the following test procedure once for each test case #1 - #10 in Table 5-11, Table 5-12 and Table 5-13, substituting for the variables in the leftmost column.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to these variables: $C_{CBUS}\{\text{min}\}$, TEST_WAKE_PULSE_HIGH, TEST_WAKE_PULSE_LOW, TEST_WAKE_PULSE_GAP, TEST_WAKE_PULSE_DISCOVER, TEST_DISCOVERY_PULSE_HIGH, TEST_DISCOVERY_PULSE_LOW, TEST_FIRST_DISCOVERY_PULSE_HIGH, TEST_FIRST_DISCOVERY_PULSE_LOW, TEST_FIFTH_DISCOVERY_PULSE_HIGH, and TEST_FIFTH_DISCOVERY_PULSE_LOW
6. Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
7. Set Dongle DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
8. Execute the Tester_Source_Modified_Discovery procedure.
9. FAIL if Source Tester detects that the CBUS does not go LOW after each Wake pulse.
10. FAIL if Source Tester detects that the CBUS does not go LOW after each of the first 5 Discovery pulses.
11. FAIL if Source Tester does not see CBUS stay HIGH after the 6th Discovery pulse.
12. FAIL if DUT does not change its pulldown resistor from $Z_{CBUS_DONGLE_DISCOVER}$ to $Z_{CBUS_DONGLE_ON}$ after the rising edge of the 6th Discovery Pulse plus T_{SINK_CONN} at latest.
13. FAIL if DUT does not pull MHL+ and MHL- up with between $Z_{RXSENSE_TERM}$ and $Z_{RXSENSE_TMDS}$ after the rising edge of the 6th Discovery Pulse plus T_{SINK_CONN} plus $T_{SINK_RXSENSE_EN}$ at latest.
14. If all preceding steps pass, then PASS; else FAIL.

Table 5-11. Discovery Pulse Timing 1

| | 1 | 2 | 3 | 4 |
|---------------------------------|---|---|---|---|
| TEST_WAKE_PULSE_HIGH | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_LOW | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_GAP | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} |
| TEST_WAKE_PULSE_DISCOVER | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} |
| TEST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} |
| TEST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} |
| Purpose | Minimum Discovery Pulse LOW width | Maximum Discovery Pulse LOW width | Minimum Discovery Pulse HIGH width | Maximum Discovery Pulse HIGH width |

Table 5-12. Discovery Pulse Timing 2

| | 5 | 6 | 7 | 8 |
|---------------------------------|---|---|---|---|
| TEST_WAKE_PULSE_HIGH | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_LOW | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_GAP | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} |
| TEST_WAKE_PULSE_DISCOVER | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {max} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {max} |
| TEST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} |
| TEST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} | T _{SINK_PULSE_WIDTH} {min} | T _{SINK_PULSE_WIDTH} {max} |

| | 5 | 6 | 7 | 8 |
|---------------------------------|---|---|---|---|
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{max}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{min}\}$ |
| Purpose | Minimum Discovery Pulse width | Maximum Discovery Pulse width | Minimum, then Maximum Discovery Pulse width | Maximum, then Minimum Discovery Pulse width |

Table 5-13. Discovery Pulse Timing 3

| | 9 | 10 |
|---------------------------------|--|--|
| TEST_WAKE_PULSE_HIGH | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_LOW | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_1}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_GAP | $T_{\text{SRC_WAKE_PULSE_WIDTH}_2}\{\text{typ}\}$ | $T_{\text{SRC_WAKE_PULSE_WIDTH}_2}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{min}\}$ | $T_{\text{SRC_WAKE_TO_DISCOVER}}\{\text{max}\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| Purpose | Discovery Pulses with fast Edge Rates | Discovery Pulses with slow Edge Rates |

5.3.14.3 CBT-Dongle: Dongle in Standby Discovers on Wake plus Discovery Pulse Sequence

Application: Dongle

5.3.14.3.1 Test Objective

Verify that Dongle DUT in Standby State responds to a valid Wake plus Discovery Pulse sequence.

5.3.14.3.2 References

[MHL] Section 13.10.1.1; Table 13-29; $T_{\text{SINK:VBUS_EN}}$

[MHL] Section 13.10.1.1; Table 13-29; $T_{\text{SINK:READY_TO_DISCOVER}}$

[MHL] Section 8.2.2; Table 8-6; Note1

5.3.14.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.14.3.4 Required Methodology

1. If the CDF indicates that the DUT does not implement Standby mode (procedure **CDF_PROC_SET_STANDBY** is empty), then PASS (SKIP).
2. If necessary, connect Dongle DUT to Tester Source port.
3. Apply power to the Dongle DUT.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Dongle DUT into Standby State based on the CDF procedure **CDF_PROC_SET_STANDBY**.
7. FAIL if Dongle DUT drives VBUS.
8. FAIL if Dongle Discovery resistor is not a valid $Z_{CBUS_DONGLE_DISCOVER}$.
9. Execute the Tester_Source_Discovery procedure.
10. If device becomes successfully Connected by the first Wake and Discovery Pulse sequence, PASS the test and terminate early.
11. FAIL if CBUS does not become HIGH-Z during DONGLE8 (Dongle Port transitions to $I_{CBUS_LEAK_SINK}$) within the shorter of $T_{SINK_VBUS_EN}$ and $T_{SINK_READY_TO_DISCOVER}$ for a minimum of $T_{SINK_CBUS_FLOAT}$.
12. FAIL if Dongle DUT does not assert a valid $Z_{CBUS_DONGLE_DISCOVER}$ within the time specified in the CDF value **CDF_D_MAX_STANDBY_TO_ACTIVE** to allow a second Discovery attempt.
13. FAIL if Dongle DUT does not enable VBUS within $T_{SINK_VBUS_EN}\{max\}$ of asserting $Z_{CBUS_DONGLE_DISCOVER}$.
14. Tester Source executes Tester_Source_discovery procedure in response to Dongle transition back to $Z_{CBUS_DONGLE_DISCOVER}$.
15. FAIL if Dongle DUT is not successfully become Connected as a result of the Wake and Discovery Pulse sequence.
16. If all preceding steps pass, then PASS; else FAIL.

Please note that the Dongle DUT shall enable VBUS during the state transition from unpowered to powered state. The enabling of VBUS and the assertion of $Z_{CBUS_DONGLE_DISCOVER}$ do not follow any particular order and both shall follow their individual timing constraints. This test uses the assertion of $Z_{CBUS_DONGLE_DISCOVER}$ as a trigger to check the timing of VBUS enabling, because the Dongle internal state transition from unpowered to powered state is not observable directly.

5.3.15 Link Layer Timing – Dongle DUT Input: Discovery Reject

Observe the response of the Dongle DUT as it is presented with Discovery pulses with illegal widths.

5.3.15.1 CBT-Dongle: First Discovery Pulse Should be Ignored

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.15.1.1 Test Objective

Observe the Dongle DUT rejects first illegal Discovery pulses.

5.3.15.1.2 References

- [MHL] Section 13.8.1; Table 13-13; C_{CBUS}
- [MHL] Section 13.10.1; Table 13-23; T_{R_CBUS} , T_{F_CBUS}
- [MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_1}$
- [MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_PULSE_WIDTH_2}$
- [MHL] Section 13.10.1.1; Table 13-28; $T_{SRC_WAKE_TO_DISCOVER}$
- [MHL] Section 13.10.1.1; Table 13-30; $T_{SRC_PULSE_WIDTH}$
- [MHL] Section 13.10.1.1; Table 13-30; $T_{SINK_PULSE_WIDTH}$
- [MHL] Section 13.10.1; Table 13-30; T_{SINK_CONN}
- [MHL] Section 13.10.1; Table 13-31; $T_{SINK_RXSENSE_EN}$

[MHL] Section 13.6.3; Table 13-9; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$

[MHL] Section 13.7.3; Table 13-12; $Z_{RXSENSE_TERM}$, $Z_{RXSENSE_TMDS}$

[MHL] Section 13.10.1; Table 13-30; $T_{SRC_CBUS_FLOAT}$

[MHL] Section 13.10.1; Table 13-30; $T_{SINK_PULSE_REJECT_MIN}$

[MHL] Section 13.10.1; Table 13-30; $T_{SINK_PULSE_REJECT_MAX}$

5.3.15.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.15.1.4 Required Methodology

Execute the following test procedure once for each test case #1 - #8 in Table 5-14 and Table 5-15, substituting for the variables in the leftmost column.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source port capacitance to minimum $C_{CBUS\{min\}}$.
6. Set Tester Source to use settings for these variables from the tables: TEST_WAKE_PULSE_HIGH, TEST_WAKE_PULSE_LOW, TEST_WAKE_PULSE_GAP, TEST_WAKE_PULSE_DISCOVER, TEST_DISCOVERY_PULSE_HIGH, TEST_DISCOVERY_PULSE_LOW, TEST_FIRST_DISCOVERY_PULSE_HIGH, TEST_FIRST_DISCOVERY_PULSE_LOW, TEST_FIFTH_DISCOVERY_PULSE_HIGH, TEST_FIFTH_DISCOVERY_PULSE_LOW, and TEST_EXTRA_PULSES
7. Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
8. Set Dongle DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
9. Execute the Tester_Source_Modified_Discovery procedure.
10. If Discovery fails, HIGH_Z the CBUS wire for $T_{SRC_CBUS_FLOAT}$.
11. If Discovery succeeds, also HIGH_Z the CBUS wire for $T_{SRC_CBUS_FLOAT}$.
12. FAIL if Source Tester detects that the CBUS does not go LOW after each Wake pulse, up to the point of the intended failure.
13. FAIL if the CBUS stays HIGH after the 6th Discovery Pulse, indicating unexpected Sink Discovery Completion.
14. If TEST_EXTRA_PULSES is non-zero, FAIL if Dongle does not complete Discovery using the last extra Discovery pulse.
15. If TEST_EXTRA_PULSES is non-zero, FAIL if Sink does not pull MHL+ and MHL- up after the rising edge of the last extra Discovery Pulse plus T_{SINK_CONN} plus $T_{SINK_RXSENSE_EN}$ at latest.
16. FAIL if DUT does not reset back to $Z_{CBUS_DONGLE_DISCOVER}$ and High-Z on MHL+/- wires in response to the CBUS becoming HIGH_Z.
17. Set for a nominal Discovery, with typical Wake pulses and typical Discovery pulses
18. Direct Tester Source to initiate discovery.
19. FAIL if second Discovery procedure does not succeed.
20. If all preceding steps pass, then PASS; else FAIL.

Table 5-14. Discovery Pulse Reject Timing 1

| | 1 | 2 | 3 | 4 |
|--|---|---|---|---|
|--|---|---|---|---|

| | 1 | 2 | 3 | 4 |
|---------------------------------|---|---|---|---|
| TEST_WAKE_PULSE_HIGH | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_LOW | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_GAP | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} |
| TEST_WAKE_PULSE_DISCOVER | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} |
| TEST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_EXTRA_PULSES | 0 | 0 | 0 | 0 |
| Purpose | First Discovery HIGH pulse is too narrow | First Discovery HIGH pulse is too wide | First Discovery LOW pulse is too narrow | First Discovery LOW pulse is too wide |

Table 5-15. Discovery Pulse Reject Timing 2

| | 5 | 6 | 7 | 8 |
|---------------------------------|---|---|---|---|
| TEST_WAKE_PULSE_HIGH | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_LOW | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_GAP | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} |
| TEST_WAKE_PULSE_DISCOVER | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} |
| TEST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |

| | 5 | 6 | 7 | 8 |
|---------------------------------|---|---|--|--|
| TEST_FIRST_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_REJECT_MIN}}$ | $T_{\text{SINK_PULSE_REJECT_MAX}}$ |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ | $T_{\text{SINK_PULSE_WIDTH}}\{\text{typ}\}$ |
| TEST_EXTRA_PULSES | 1 | 1 | 1 | 1 |
| Purpose | First Discovery HIGH pulse is too narrow; finally succeed | First Discovery HIGH pulse is too wide; finally succeed | First Discovery LOW pulse is too narrow; finally succeed | First Discovery LOW pulse is too wide; finally succeed |

5.3.15.2 CBT-Dongle: Last Discovery Pulse Should be Ignored

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.15.2.1 Test Objective

Observe the Dongle DUT rejects last illegal Discovery pulses.

5.3.15.2.2 References

[MHL] Section 13.8.1; Table 13-13; C_{CBUS}
[MHL] Section 13.10.1; Table 13-23; $T_{\text{R_CBUS}}$, $T_{\text{F_CBUS}}$
[MHL] Section 13.10.1.1; Table 13-28; $T_{\text{SRC_WAKE_PULSE_WIDTH_1}}$
[MHL] Section 13.10.1.1; Table 13-28; $T_{\text{SRC_WAKE_PULSE_WIDTH_2}}$
[MHL] Section 13.10.1.1; Table 13-28; $T_{\text{SRC_WAKE_TO_DISCOVER}}$
[MHL] Section 13.10.1.1; Table 13-30; $T_{\text{SRC_PULSE_WIDTH}}$
[MHL] Section 13.10.1.1; Table 13-30; $T_{\text{SINK_PULSE_WIDTH}}$
[MHL] Section 13.10.1; Table 13-30; $T_{\text{SINK_CONN}}$
[MHL] Section 13.10.1; Table 13-31; $T_{\text{SINK_RXSENSE_EN}}$
[MHL] Section 13.6.3; Table 13-9; $Z_{\text{RXSENSE_TERM}}$, $Z_{\text{RXSENSE_TMDS}}$
[MHL] Section 13.7.3; Table 13-12; $Z_{\text{RXSENSE_TERM}}$, $Z_{\text{RXSENSE_TMDS}}$
[MHL] Section 13.10.1; Table 13-30; $T_{\text{SRC_CBUS_FLOAT}}$
[MHL] Section 13.10.1; Table 13-30; $T_{\text{SINK_PULSE_REJECT_MIN}}$
[MHL] Section 13.10.1; Table 13-30; $T_{\text{SINK_PULSE_REJECT_MAX}}$

5.3.15.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.15.2.4 Required Methodology

Execute the following test procedure once for each test case #1 - #8 in Table 5-16 and Table 5-17, substituting for the variables in the leftmost column.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.

3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source port capacitance to minimum $C_{CBUS}\{\min\}$.
6. Set Tester Source to use settings for these variables from the tables: TEST_WAKE_PULSE_HIGH, TEST_WAKE_PULSE_LOW, TEST_WAKE_PULSE_GAP, TEST_WAKE_PULSE_DISCOVER, TEST_DISCOVERY_PULSE_HIGH, TEST_DISCOVERY_PULSE_LOW, TEST_FIRST_DISCOVERY_PULSE_HIGH, TEST_FIRST_DISCOVERY_PULSE_LOW, TEST_FIFTH_DISCOVERY_PULSE_HIGH, TEST_FIFTH_DISCOVERY_PULSE_LOW, and TEST_EXTRA_PULSES
7. Set Tester Source to use its Source Discovery engine to enable normal MHL discovery.
8. Set Dongle DUT into Active State based on the CDF procedure **CDF_PROC_SET_ACTIVE**.
9. Execute the Tester_Source_Modified_Discovery procedure.
10. If Discovery fails, HIGH_Z the CBUS wire for $T_{SRC_CBUS_FLOAT}$.
11. If Discovery succeeds, also HIGH_Z the CBUS wire for $T_{SRC_CBUS_FLOAT}$.
12. FAIL if Source Tester detects that the CBUS does not go LOW after each Wake pulse, up to the point of the intended failure.
13. FAIL if the CBUS stays HIGH after the 6th Discovery Pulse, indicating unexpected Sink Discovery Completion.
14. If TEST_EXTRA_PULSES is non-zero, FAIL if Dongle does not complete Discovery using the last extra Discovery pulse.
15. If TEST_EXTRA_PULSES is non-zero, FAIL if Sink does not pull MHL+ and MHL- up after the rising edge of the last extra Discovery Pulse plus T_{SINK_CONN} plus $T_{SINK_RXSENSE_EN}$ at latest.
16. FAIL if DUT does not reset back to $Z_{CBUS_DONGLE_DISCOVER}$ and High-Z on MHL+/- wires in response to the CBUS being HIGH_Z.
17. Set for a nominal Discovery, with typical Wake pulses and typical Discovery pulses
18. Direct Tester Source to initiate discovery.
19. FAIL if second Discovery procedure does not succeed.
20. If all preceding steps pass, then PASS; else FAIL.

Table 5-16. Discovery Pulse Reject Timing 1

| | 1 | 2 | 3 | 4 |
|---------------------------------|--|--|--|--|
| TEST_WAKE_PULSE_HIGH | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_LOW | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_1}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_GAP | $T_{SRC_WAKE_PULSE_WIDTH_2}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{\text{typ}\}$ | $T_{SRC_WAKE_PULSE_WIDTH_2}\{\text{typ}\}$ |
| TEST_WAKE_PULSE_DISCOVER | $T_{SRC_WAKE_TO_DISCOVER}\{\text{min}\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{\text{min}\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{\text{min}\}$ | $T_{SRC_WAKE_TO_DISCOVER}\{\text{min}\}$ |
| TEST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ |
| TEST_DISCOVERY_PULSE_LOW | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ | $T_{SINK_PULSE_WIDTH}\{\text{typ}\}$ |

| | 1 | 2 | 3 | 4 |
|---------------------------------|--|--|---|---------------------------------------|
| TEST_FIRST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} |
| TEST_EXTRA_PULSES | 0 | 0 | 0 | 0 |
| Purpose | Fifth Discovery HIGH pulse is too narrow | Fifth Discovery HIGH pulse is too wide | Fifth Discovery LOW pulse is too narrow | Fifth Discovery LOW pulse is too wide |

Table 5-17. Discovery Pulse Reject Timing 2

| | 5 | 6 | 7 | 8 |
|---------------------------------|---|---|--|--|
| TEST_WAKE_PULSE_HIGH | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_LOW | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} | T _{SRC_WAKE_PULSE_WIDTH_1} {typ} |
| TEST_WAKE_PULSE_GAP | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} | T _{SRC_WAKE_PULSE_WIDTH_2} {typ} |
| TEST_WAKE_PULSE_DISCOVER | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} | T _{SRC_WAKE_TO_DISCOVER} {min} |
| TEST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIRST_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_HIGH | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} |
| TEST_FIFTH_DISCOVERY_PULSE_LOW | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_WIDTH} {typ} | T _{SINK_PULSE_REJECT_MIN} | T _{SINK_PULSE_REJECT_MAX} |
| TEST_EXTRA_PULSES | 6 | 6 | 6 | 6 |
| Purpose | Fifth Discovery HIGH pulse is too narrow; finally succeed | Fifth Discovery HIGH pulse is too wide; finally succeed | Fifth Discovery LOW pulse is too narrow; finally succeed | Fifth Discovery LOW pulse is too wide; finally succeed |

5.3.16 Link Layer Electrical – Dongle DUT Input: Arbitration/Sync/Data Signaling

Measure the electrical behavior of the Dongle DUT as it receives Arbitration, Sync, and Data pulses.

5.3.16.1 CBE-Dongle: Sensitivity to VIH/VIL

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.16.1.1 Test Objective

Verify that the Dongle DUT correctly receives data pulses above V_{IH_CBUS} and below V_{IL_CBUS}

5.3.16.1.2 References

[MHL] Section 8-4

[MHL] Section 13.10.1; Table 13-24; V_{OH_CBUS} , V_{OL_CBUS}

[MHL] Section 13.10.1; Table 13-24: V_{IH_CBUS} , V_{IL_CBUS}

5.3.16.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.16.1.4 Required Methodology

Execute the following test procedure once for each row in Table 5-18, substituting TEST_VO_H_CBUS and TEST_VOL_CBUS.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source HIGH output voltage to TEST_VO_H_CBUS.
6. Set Tester Source LOW output voltage to TEST_VOL_CBUS.
7. Set Tester Source to use its Source Packet engine to link level ACK everything.
8. Execute the Tester_Source_Discovery procedure.
9. When the Tester Source receives a command, reply with an ACK packet.
10. FAIL if DUT does not eventually arbitrate for the bus to send a command to the Tester Source.
11. FAIL if DUT does not link level ACK the ACK packet with fewer than $N_{RETRY}\{min\}$ retries.
12. The Dongle DUT will probably NACK the first several attempts to do the ACK packet. Retransmit.
13. If all preceding steps pass, then PASS; else FAIL.

Note: The Dongle DUT might send a command that requires a data packet instead of an ACK packet in response. For this test, the contents of the response packet may be either a data packet or a control packet.

Table 5-18. Sensitivity to VIH/VIL

| | TEST_VO _H _CBUS | TEST_VOL_CBUS | Purpose |
|---|----------------------------|------------------------------|--------------------|
| 1 | $V_{IH_CBUS}\{min\}$ | $V_{OL_CBUS}\{min\}$ (0.0V) | Sensitivity to VIH |
| 2 | $V_{OH_CBUS}\{max\}$ | $V_{IL_CBUS}\{max\}$ | Sensitivity to VIL |

5.3.17 Link Layer Timing – Dongle DUT Input: Arbitration

Measure the behavior of the Dongle DUT as it receives Arbitration pulses with different timings.

5.3.17.1 CBT-Dongle: Loses Arbitration Correctly

Application: Dongle

5.3.17.1.1 Test Objective

Verify that Dongle DUT loses arbitration to the Tester acting as Source using Fast Bit Timing.

5.3.17.1.2 References

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$

[MHL] Section 13.10.1.1; Table 13-31: $T_{\text{SRC_ARBITRATE}}$

5.3.17.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.17.1.4 Required Methodology

Execute the following test procedure once for each row in Table 5-19, substituting TEST_BIT_TIME, TEST_IDLE_PERIOD and TEST_FIRST_DONGLE_COMMAND_DELAY.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. Set Tester Source to operate using TEST_BIT_TIME for $T_{\text{BIT_CBUS}}$.
9. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
10. Wait TEST_FIRST_SRC_COMMAND_DELAY.
11. Tester Source does GET_STATE to Dongle DUT.
12. FAIL if DUT does not respond to the GET_STATE command from the Tester with a Data packet.
13. If all preceding steps pass, then PASS; else FAIL.

Note that the Dongle DUT might NACK the GET_STATE command. Tester needs to re-issue this command up to N times to give the Dongle DUT a chance to ACK it.

Table 5-19. Dongle Behaves Appropriately when Dongle Loses Arbitration

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_FIRST_DONGLE_COMMAND_DELAY | Purpose |
|---|--------------------------------------|-------------------------------------|--|---|
| 1 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$ | nominal Dongle Bit Timing, shortest ARB to SYNC delay |
| 2 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$ | nominal Dongle Bit Timing, longest ARB to SYNC delay |
| 3 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$ | fast Dongle Bit Timing, shortest ARB to SYNC delay |
| 4 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$ | fast Dongle Bit Timing, longest ARB to SYNC delay |
| 5 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$ | slow Dongle Bit Timing, shortest ARB to SYNC delay |

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_FIRST_DONGLE_COMMAND_DELAY | Purpose |
|---|--------------------------------------|-------------------------------------|--|---|
| 6 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | $T_{\text{SINK_ARBITRATE}}\{\text{min}\}$ | slow Dongle Bit Timing, longest ARB to SYNC delay |

5.3.17.2 CBT-Dongle: End of Discovery to Early Source-side Arbitration

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.17.2.1 Test Objective

Verify that the Dongle DUT correctly responds to a packet if a Source sends one before the minimum Discovery-to-Arbitration holdoff.

5.3.17.2.2 References

[MHL] Section 8.5.1

[MHL] Section 13.10.1.1; Table 13-31: $T_{\text{SRC_CONN}}$

[MHL] Section 13.10.1.1; Table 13-31: $T_{\text{SINK_ARBITRATE}}$

5.3.17.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.17.2.4 Required Methodology

Execute the following test procedure once for each row in Table 5-20, substituting TEST_BIT_TIME, TEST_IDLE_PERIOD and TEST_FIRST_DONGLE_COMMAND_DELAY.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. Set Tester Source to operate using TEST_BIT_TIME for $T_{\text{BIT_CBUS}}$.
9. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
10. Wait TEST_FIRST_SRC_COMMAND_DELAY.
11. Tester Source does GET_STATE to Dongle DUT.
12. FAIL if DUT does not respond to the GET_STATE command from the Tester with a Data packet.
13. If all preceding steps pass, then PASS; else FAIL.

Note that the Dongle DUT might NACK the GET_STATE command. It needs to be re-issued up to N_{RETRY} times to give the Dongle DUT a chance to ACK it.

Table 5-20. End of Discovery to Early Source-side Arbitration

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_FIRST_DONGLE_COMMAND_DELAY | Purpose |
|---|--------------------------------------|-------------------------------------|---------------------------------------|-------------------------------|
| 1 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | $T_{\text{SINK_CONN}}\{\text{max}\}$ | Very Early Dongle arbitration |

5.3.17.3 CBT-Dongle: Dongle Loses Arbitration Collision Correctly

Application: Dongle

5.3.17.3.1 Test Objective

Verify that the Dongle DUT loses when it collides with the Source during Arbitration.

5.3.17.3.2 References

[MHL] Section 7.2.4; Timing case 1

5.3.17.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.17.3.4 Required Methodology

Execute the following test procedure once for each row in Table 5-21, substituting TEST_BIT_TIME.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{IH_CBUS}\{min\}$ and $V_{IL_CBUS}\{max\}$.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. Set Tester Source to operate using the TEST_BIT_TIME for T_{BIT_CBUS} .
9. Prepare the Tester Source to do a GET_VENDOR_ID, but do not issue it yet.
10. Set Tester Source to artificially force an Arbitration collision when it sees the first Dongle DUT arbitration.
11. The Tester Source will win arbitration because of the collision. Issue a GET_VENDOR_ID.
12. Once the GET_VENDOR_ID is issued, set its Source Packet engine to ACK the response.
13. FAIL if DUT does not collide with the Tester Source during Arbitration and back off, but instead continues on to do its command.
14. FAIL if DUT does not reply to the GET_VENDOR_ID with the expected data.
15. FAIL if DUT does not re-issue the command it was sending before it lost the Arbitration.
16. If all preceding steps pass, then PASS; else FAIL.

Table 5-21. Loses Arbitration Collision Correctly

| | TEST_BIT_TIME | Purpose |
|---|------------------------|--|
| 1 | $T_{BIT_CBUS}\{typ\}$ | Dongle loses Arbitration Collision correctly, nominal Dongle bit clock |
| 2 | $T_{BIT_CBUS}\{min\}$ | Dongle loses Arbitration Collision correctly, fast Dongle bit clock |
| 3 | $T_{BIT_CBUS}\{max\}$ | Dongle loses Arbitration Collision correctly, slow Dongle bit clock |

5.3.18 Link Layer Timing – Dongle DUT Input: Data

Measure the behavior of the Dongle DUT as it receives Data pulses with different timings.

5.3.18.1 CBT-Dongle: Bit Timing Sensitivity

Application: Dongle

5.3.18.1.1 Test Objective

Verify that Dongle DUT can receive data from a Source with the fastest/slowest/most legal bit timings.

5.3.18.1.2 References

[MHL] Section 7.2.2

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}} \pm T_{\text{CBUS_DUTY}}$

[MHL] Section 13.10.1; Table 13-23; $T_{\text{BIT_CBUS}} \pm \Delta T_{\text{RF}}$

5.3.18.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.18.1.4 Required Methodology

Execute the following test procedure once for each row in Table 5-22, substituting TEST_BIT_TIME, TEST_IDLE_PERIOD, TEST_CENTER_OFFSET and TEST_RISE_OFFSET.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
6. Set Tester Source to operate using TEST_BIT_TIME for $T_{\text{BIT_CBUS}}$.
7. Set Tester Source to TEST_IDLE_PERIOD delay from Arbitration pulse to Sync pulse.
8. Set Tester Source to offset "1" bit center-bit transitions by TEST_CENTER_OFFSET.
9. Set Tester Source to offset rising edges TEST_RISE_OFFSET compared to falling edges.
10. Set Tester Source to use its Source Packet engine to ACK everything.
11. Execute the Tester_Source_Discovery procedure.
12. When the Tester Source receives a SET_HPD, reply with an ACK packet.
13. Tester Source switches to responding with ACK to everything.
14. FAIL if DUT does not ACK the ACK packet from the Tester.
15. If all preceding steps pass, then PASS; else FAIL.

Table 5-22. Bit Timing Sensitivity

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_CENTER_OFFSET | TEST_RISE_OFFSET | Purpose |
|---|--------------------------------------|-------------------------------------|------------------------------|------------------|---------------------------|
| 1 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | 0 | 0 | Fast; Bits symmetrical |
| 2 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{typ}\}$ | 0 | 0 | Typical; Bits symmetrical |
| 3 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | 0 | 0 | Slow; Bits symmetrical |
| 4 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | $-\text{TEST_BIT_TIME}/10$ | 0 | Fast; midpoint early |
| 5 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_OPP}}\{\text{typ}\}$ | $-\text{TEST_BIT_TIME}/10$ | 0 | Typical; midpoint early |
| 6 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_OPP}}\{\text{max}\}$ | $-\text{TEST_BIT_TIME}/10$ | 0 | Slow; midpoint early |
| 7 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_OPP}}\{\text{min}\}$ | $+\text{TEST_BIT_TIME}/10$ | 0 | Fast; midpoint late |

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | TEST_CENTER_OFFSET | TEST_RISE_OFFSET | Purpose |
|----|-----------------------------|----------------------------|--------------------|-----------------------|--------------------------------|
| 8 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {typ} | +TEST_BIT_TIME/10 | 0 | Typical; midpoint late |
| 9 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | +TEST_BIT_TIME/10 | 0 | Slow; midpoint late |
| 10 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | 0 | - ΔT _{RF} /2 | Fast; rise faster than fall |
| 11 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {typ} | 0 | - ΔT _{RF} /2 | Typical; rise faster than fall |
| 12 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | 0 | - ΔT _{RF} /2 | Slow; rise faster than fall |
| 13 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | 0 | + ΔT _{RF} /2 | Fast; fall faster than rise |
| 14 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {typ} | 0 | + ΔT _{RF} /2 | Typical; fall faster than rise |
| 15 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | 0 | + ΔT _{RF} /2 | Slow; fall faster than rise |

5.3.19 Link Layer Timing – Dongle DUT Input: NACK

Verify that Dongle DUT can detect bad parity.

Bad parity is defined as a Packet with a parity bit with a low to high transition at its end.

5.3.19.1 CBT-Dongle: Response to Packet with Bad Parity

Application: Dongle

5.3.19.1.1 Test Objective

Verify that Dongle DUT can detect bad parity.

5.3.19.1.2 References

[MHL] Section 7.2.3.5

5.3.19.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.19.1.4 Required Methodology

Execute the following test procedure once for each row in Table 5-23, substituting TEST_BIT_RATE, TEST_IDLE_PERIOD, and TEST_BAD_PACKET.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to operate using the TEST_BIT_RATE for T_{BIT_CBUS}.
6. Set Tester Source to use TEST_IDLE_PERIOD for delay from Arbitration pulse to Sync pulse.
7. Set Tester Source to use its Source Packet engine to ACK everything.
8. Execute the Tester_Source_Discovery procedure.
9. Tester Source sends TEST_BAD_PACKET N_{RETRY}{min}-1 times.
10. FAIL if DUT does not NACK the bad Packet from the Tester NRETRY{min}-1 times.
11. If all preceding steps pass, then PASS; else FAIL.

Table 5-23. Check Response to Bad Parity Packet

| | TEST_BIT_RATE | TEST_IDLE_PERIOD | TEST_BAD_PACKET | Purpose |
|---|-----------------------------|----------------------------|-----------------|---------------------------|
| 1 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {max} | Error-Packet-1 | bad parity (Nominal Sink) |

| | TEST_BIT_RATE | TEST_IDLE_PERIOD | TEST_BAD_PACKET | Purpose |
|---|-----------------------------|----------------------------|-----------------|---------------------------|
| 2 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | Error-Packet-1 | bad parity (Slow Sink) |
| 3 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | Error-Packet-1 | bad parity (Fast Sink) |
| 4 | T _{BIT_CBUS} {typ} | T _{REQ_OPP} {max} | Error-Packet-2 | bad parity (Nominal Sink) |
| 5 | T _{BIT_CBUS} {max} | T _{REQ_OPP} {max} | Error-Packet-2 | bad parity (Slow Sink) |
| 6 | T _{BIT_CBUS} {min} | T _{REQ_OPP} {min} | Error-Packet-2 | bad parity (Fast Sink) |

5.3.20 Link Layer Timing – Dongle DUT Input: ACK

Measure the behavior of the Dongle DUT as it receives ACK pulses with different timings.

5.3.20.1 CBT-Dongle: Response to ACKs with Different Timings

Application: Dongle

5.3.20.1.1 Test Objective

Dongle DUT receives Tester ACK at various times. Verify that Dongle DUT recognizes ACK.

5.3.20.1.2 References

[MHL] Section 7.2.3.6

[MHL] Section 13.10.1; Table 13-23, T_{CBUS_ACK_0}

5.3.20.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.20.1.4 Required Methodology

Execute the following test procedure once for each row in Table 5-24, substituting TEST_BIT_TIME, TEST_ACK_TIME and TEST_ACK_DURATION.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between V_{IH_CBUS}{min} and V_{IL_CBUS}{max}.
6. Set Tester Dongle to operate using TEST_BIT_TIME for T_{BIT_CBUS}.
7. Set Tester Source to place ACK pulses at the TEST_ACK_TIME after expected ACK period starts.
8. Set Tester Source to drive the ACK pulse LOW for TEST_ACK_DURATION.
9. Execute the Tester_Source_Discovery procedure.
10. Set Tester Source to send a GET_STATE to Dongle DUT.
11. The Dongle DUT may NACK the GET_STATE command. The tester shall re-issue this command up to N_{RETRY} times, to provide time for the Dongle DUT to ACK the command.
12. FAIL if DUT does not eventually arbitrate for the bus to send a data packet to the Tester Source.
13. ACK the DUT using the calculated bit timing and the previously set ACK pulse placement.
14. FAIL if DUT does not try to reply with a data packet.
15. FAIL if DUT retries sending the data packet after the tester link level ACK. This indicates that the DUT is sensitive to the link level ACK where placed.
16. If all preceding steps pass, then PASS; else FAIL.

Table 5-24. Response to ACKs with Different Timings

| | TEST_BIT_TIME | TEST_ACK_TIME | TEST_ACK_DURATION | Purpose |
|----|-----------------------------|----------------------------------|-------------------------------|-----------------------------|
| 1 | T _{BIT_CBUS} {typ} | T _{CBUS_ACK_FALL} {min} | T _{CBUS_ACK_O} {min} | Earliest possible thin ACK |
| 2 | T _{BIT_CBUS} {typ} | T _{CBUS_ACK_FALL} {max} | T _{CBUS_ACK_O} {min} | Latest possible thin ACK |
| 3 | T _{BIT_CBUS} {typ} | T _{CBUS_ACK_FALL} {min} | T _{CBUS_ACK_O} {max} | Earliest possible thick ACK |
| 4 | T _{BIT_CBUS} {typ} | T _{CBUS_ACK_FALL} {max} | T _{CBUS_ACK_O} {max} | Latest possible thick ACK |
| 5 | T _{BIT_CBUS} {max} | T _{CBUS_ACK_FALL} {min} | T _{CBUS_ACK_O} {min} | Earliest possible thin ACK |
| 6 | T _{BIT_CBUS} {max} | T _{CBUS_ACK_FALL} {max} | T _{CBUS_ACK_O} {min} | Latest possible thin ACK |
| 7 | T _{BIT_CBUS} {max} | T _{CBUS_ACK_FALL} {min} | T _{CBUS_ACK_O} {max} | Earliest possible thick ACK |
| 8 | T _{BIT_CBUS} {max} | T _{CBUS_ACK_FALL} {max} | T _{CBUS_ACK_O} {max} | Latest possible thick ACK |
| 9 | T _{BIT_CBUS} {min} | T _{CBUS_ACK_FALL} {min} | T _{CBUS_ACK_O} {min} | Earliest possible thin ACK |
| 10 | T _{BIT_CBUS} {min} | T _{CBUS_ACK_FALL} {max} | T _{CBUS_ACK_O} {min} | Latest possible thin ACK |
| 11 | T _{BIT_CBUS} {min} | T _{CBUS_ACK_FALL} {min} | T _{CBUS_ACK_O} {max} | Earliest possible thick ACK |
| 12 | T _{BIT_CBUS} {min} | T _{CBUS_ACK_FALL} {max} | T _{CBUS_ACK_O} {max} | Latest possible thick ACK |

5.3.21 Link Layer Timing – Dongle DUT Input: Bus Re-Arbitration

Measure the behavior of the Dongle DUT as it responds to Source re-arbitration with different timings.

5.3.21.1 CBT-Dongle: Dongle Receives Case 1 Back-to-Back Transfers

Application: Dongle

5.3.21.1.1 Test Objective

Verify that Dongle DUT can receive Case 1 Back-to-Back packets when connected to various speed Dongles using variable ARB-to-Sync Idle timing.

5.3.21.1.2 References

[MHL] Section 7.2.4; Case 1; Tester sends back-to-back; min Sync

5.3.21.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.21.1.4 Required Methodology

Execute the following test procedure once for each row in Table 5-25, substituting TEST_BIT_TIME and TEST_IDLE_PERIOD.

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to operate using TEST_BIT_TIME as T_{BIT_CBUS}.
6. Set Tester Source to use TEST_IDLE_PERIOD as delay from Arbitration pulse to Sync pulse.
7. Set Tester Source to use its Source Packet engine to ACK everything.
8. Execute the Tester_Source_Discovery procedure.
9. When the Tester Source receives a command, reply with a link level ACK.
10. Tester Source does SET_INT to Dongle DUT, followed by Offset 0x20, followed by Data 0x00. The Offset packet has Parity == 0, the Data packet has Parity == 1. The last 2 Packets are sent back-to-back.
11. Tester Source switches to responding with ACK to everything.

12. Observe post-discovery Dongle DUT link-level ACK activity as it replies to the Tester Source command.
13. Observe that the Dongle DUT replies with an ACK packet.
14. FAIL if DUT does not eventually send a command.
15. FAIL if DUT does not link level ACK all 3 packets sent from the Tester.
16. FAIL if DUT does not respond with an ACK packet.
17. If all preceding steps pass, then PASS; else FAIL.

Note that the Dongle DUT might NACK the packets it receives. Tester needs to re-issue this command up to N_{RETRY} times to give the Dongle DUT a chance to ACK it.

Table 5-25. Dongle Receives Back-to-Back Transfers

| | TEST_BIT_TIME | TEST_IDLE_PERIOD | Purpose |
|---|--------------------------------------|--------------------------------------|--|
| 1 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | nominal Source Bit Timing, shortest ARB to SYNC |
| 2 | $T_{\text{BIT_CBUS}}\{\text{typ}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | nominal Source Bit Timing, longest ARB to SYNC delay |
| 3 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | minimum Source Bit Timing, shortest ARB to SYNC |
| 4 | $T_{\text{BIT_CBUS}}\{\text{min}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | minimum Source Bit Timing, longest ARB to SYNC delay |
| 5 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_CONT}}\{\text{min}\}$ | maximum Source Bit Timing, shortest ARB to SYNC |
| 6 | $T_{\text{BIT_CBUS}}\{\text{max}\}$ | $T_{\text{REQ_CONT}}\{\text{max}\}$ | maximum Source Bit Timing, longest ARB to SYNC delay |

5.3.22 Link Layer Timing – Dongle DUT Input: Ill-formed Packets

Verify that the Dongle does not have problems when the Source sends ill-formed packets.

5.3.22.1 CBT-Dongle: Source Wins Arbitration, then No Packet, when Dongle is Idle

Application: Dongle

5.3.22.1.1 Test Objective

Verify that Dongle DUT is not confused if the Source arbitrates for the CBUS but then does not use it.

5.3.22.1.2 References

[MHL] Section 7.2.4

5.3.22.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.22.1.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Timing Measurement CBUS Voltage threshold to the 50% point between $V_{\text{IH_CBUS}}\{\text{min}\}$ and $V_{\text{IL_CBUS}}\{\text{max}\}$.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a SET_HPDP, reply with an ACK packet.

9. Set Tester Source to occasionally drive a total of 100 CBUS LOW pulses for periods of 4000 nanoseconds (4 nominal bit times), followed by drive HIGH as usual. These are valid Dongle-side arbitration pulses, but are not followed by Sync pulses.
10. Set Tester Source to wait until the bogus arbitration test period ends, then do a READ_DEVCAP followed by Offset 0x00.
11. FAIL if DUT responds in any way to the bad arbitration activity.
12. FAIL if DUT cannot respond to the valid READ_DEVCAP command from the Tester.
13. If all preceding steps pass, then PASS; else FAIL.

5.3.23 Link Layer Timing – Dongle DUT Input: Disconnect

Verify that the Dongle responds appropriately when the MHL bus is disconnected (or when the Source pretends that the bus is disconnected).

5.3.23.1 CBT-Dongle: CBUS LOW for Less than Minimum $T_{\text{SINK_CBUS_DISCONN}}$ Reject Time

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.23.1.1 Test Objective

Verify that Dongle DUT ignores LOW-asserted pulses less than $T_{\text{SINK_CBUS_DISCONN}}\{\text{min}\}$ on the CBUS.

5.3.23.1.2 References

[MHL] Section 13.10.1.2; Table 13-32; $T_{\text{SINK_CBUS_DISCONN}}$

5.3.23.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.23.1.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. When the Tester Source receives a SET_HPD, reply with an ACK packet.
8. Service other packets from the Dongle DUT if they arrive.
9. When bus is idle, set Tester Source to drive CBUS LOW for $T_{\text{SINK_CBUS_DISCONN}}\{\text{min}\} - 1 \mu\text{Sec}$.
10. FAIL if DUT has trouble doing Discovery.
11. FAIL if DUT does not eventually send SET_HPD.
12. FAIL if DUT does not eventually become idle within 10 seconds of Discovery, leaving a constant CBUS HIGH.
13. FAIL if DUT detects the Glitch as a Disconnect event. Dongle DUT disconnect is inferred by the Source detecting the MHL+ and MHL- wires changing to a LOW voltage level, under the influence of the Source-side pull down resistors.
14. If all preceding steps pass, then PASS; else FAIL.

5.3.23.2 CBT-Dongle: CBUS LOW for Greater than Maximum $T_{\text{SINK_CBUS_DISCONN}}$ Reject Time

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.23.2.1 Test Objective

Verify that Dongle DUT detects wide LOW-asserted pulses on the CBUS as a disconnect event.

Verify that Dongle DUT quickly disables its MHL+/- pull-ups when it sees a CBUS disconnect.

Verify that Dongle DUT quickly changes its pulldown to $Z_{\text{CBUS_DONGLE_DISCOVER}}$ when it sees a CBUS disconnect.

5.3.23.2.2 References

[MHL] Section 13.10.1.2; Table 13-32; $T_{\text{SINK_CBUS_DISCONN}}$

[MHL] Section 13.10.1.2; Table 13-32; $T_{\text{SINK_RXSENSE_DIS}}$

[MHL] Section 13.10.1.2; Table 13-32; $T_{\text{SINK:CBUS_SINK_CONN}}$

5.3.23.2.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.23.2.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Set Tester Source port to disconnected state.
4. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
5. Set Tester Source to use its Source Packet engine to ACK everything.
6. Execute the Tester_Source_Discovery procedure.
7. When the Tester Source receives a SET_HPD, reply with an ACK packet.
8. Service other packets from the Dongle DUT if they arrive.
9. Set Tester Source to drive CBUS LOW for $T_{\text{SINK_CBUS_DISCONN}}\{\text{max}\} + 1 \text{ uSec}$.
10. Dongle DUT disconnect is inferred by seeing the MHL+/- wires go LOW under the influence of the Source-side pulldowns.
11. FAIL if DUT does not eventually send SET_HPD.
12. FAIL if DUT does not detect the Pulse LOW as a Disconnect event.
13. FAIL if DUT does not remove its pull-ups on MHL+/- within $T_{\text{SINK_RXSENSE_DIS}}$ (5 mSec).
14. FAIL if DUT does not switch to $Z_{\text{CBUS_DONGLE_DISCOVER}}$ within $T_{\text{SINK:CBUS_SINK_CONN}}$ (5 mSec).
15. If all preceding steps pass, then PASS; else FAIL.

5.3.23.3 CBT-Dongle: Dongle Disconnect on Non-MHL Side Causes CLR_HPD

| |
|----------------------------|
| Application: Dongle |
|----------------------------|

5.3.23.3.1 Test Objective

Verify that Dongle DUT detects that it has been disconnected on its non-MHL side, and that it makes an MHL CLR_HPD.

5.3.23.3.2 References

[MHL] Section 8.1.3

[MHL] Section 7.8.2.2

[MHL] Section 8.5.4

5.3.23.3.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.23.3.4 Required Methodology

1. If necessary, connect Dongle DUT to Tester Source port.
2. Apply power to the Dongle DUT.
3. Connect Dongle DUT to downstream AV display device and select Dongle DUT as input source.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Set Tester Source to use its Source Packet engine to ACK everything.
7. Execute the Tester_Source_Discovery procedure.
8. When the Tester Source receives a SET_HPD, reply with an ACK packet.
9. Service other packets from the Dongle DUT if they arrive.
10. Disconnect Dongle DUT from the downstream AV display device.
11. FAIL if DUT does not eventually send SET_HPD in response to the Discovery procedure.
12. FAIL if DUT does not send a CLR_HPD within 10 seconds of being disconnected from the AV display device.
13. If all preceding steps pass, then PASS; else FAIL.

5.3.24 Link Layer Electrical – Dongle DUT VBUS Output

Verify that a powered Dongle drives the VBUS as required.

5.3.24.1 CBE-Dongle: Dongle DUT VBUS Output

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.24.1.1 Test Objective

Verify that a powered Dongle supplies power over the VBUS as required.

5.3.24.1.2 References

[MHL] Section 9.1

[MHL] Section 13.9.1.1; Table 13-21; $V_{VBUS_TP1_DNGL_DRV}$

[MHL] Section 13.9.1.2; Table 13-30; $I_{VBUS_DONGLE_TO_SOURCE}$

5.3.24.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.24.1.4 Required Methodology

1. If the CDF value **CDF_D_POWERED** indicates that DUT is a Powered Dongle, then continue, else end test with PASS (SKIP).
2. If necessary, connect Dongle DUT to Tester Source port.
3. Connect the Dongle DUT to the downstream device, and connect power to the Dongle DUT.
4. Set Tester Source port to disconnected state.
5. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
6. Tester does not drive VBUS.
7. Set Tester Source to apply $I_{VBUS_PREDISCOVERY\{max\}}$ load (100 mA) to the VBUS.
8. Tester Source measures the VBUS driven by the Dongle DUT and loaded by the Tester Source.
9. FAIL if VBUS Voltage is not within the range $V_{VBUS_TP1_DNGL_DRV\{min\}}$ to $V_{VBUS_TP1_DNGL_DRV\{max\}}$.
10. Execute the Tester_Source_Discovery procedure.
11. Execute the Tester_Wait_Capability_Registers_Valid procedure.
12. FAIL if the procedure indicates TIMEOUT.
13. Tester Source reads Dongle's DEV_CAT register.
14. FAIL if DEV_CAT register not read as DONGLE, and immediately terminate the test.

15. FAIL if DEV_CAT POW bit is not '1'.
16. If {PLIM1, PLIM0} are {0, 0}, Set Tester Source to apply $I_{VBUS_DONGLE_TO_SOURCE}$ load (500mA) to the VBUS.
17. FAIL if VBUS Voltage is not within the range $V_{VBUS_TP1_DNGL_DRV}\{min\}$ to $V_{VBUS_TP1_DNGL_DRV}\{max\}$.
18. If {PLIM1, PLIM0} are {0, 1}, Set Tester Source to apply $I_{VBUS_DONGLE_TO_SOURCE}$ load (900mA) to the VBUS.
19. FAIL if VBUS Voltage is not within the range $V_{VBUS_TP1_DNGL_DRV}\{min\}$ to $V_{VBUS_TP1_DNGL_DRV}\{max\}$.
20. If {PLIM1, PLIM0} are {1,0}, set Tester Source to apply $I_{VBUS_DONGLE_TO_SOURCE}$ load (1500mA) to the VBUS.
21. FAIL if VBUS Voltage is not within the range $V_{VBUS_TP1_DNGL_DRV}\{min\}$ to $V_{VBUS_TP1_DNGL_DRV}\{max\}$.
22. If {PLIM1, PLIM0} are {1, 1}, then set Tester Source to apply $I_{VBUS_DONGLE_TO_SOURCE}$ load (100mA) to the VBUS.
 - a. If VBUS Voltage is within the range $V_{VBUS_TP1_DNGL_DRV}\{min\}$ to $V_{VBUS_TP1_DNGL_DRV}\{max\}$ then continue; else end test with FAIL.
23. If all preceding steps pass, then PASS; else FAIL.

5.3.25 Link Layer Electrical – Dongle DUT VBUS Input

Verify that a Dongle DUT without an attached Power Adapter uses a limited amount of Source VBUS current .

5.3.25.1 CBE-Dongle: Dongle DUT VBUS Input

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.25.1.1 Test Objective

Verify that a Dongle DUT without an attached Power Adapter uses a limited amount of Source VBUS current .

5.3.25.1.2 References

[MHL] Section 9.1

[MHL] Section 13.9.1.1; Table 13-21; $V_{VBUS_TP1_SRC_DRV}$

[MHL] Section 13.9.1.2; Table 13-22: $I_{VBUS_PREDISCOVERY}$, $I_{VBUS_SOURCE_TO_DONGLE}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{CBUS_VBUS_OUT_TO_STABLE}$

5.3.25.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.25.1.4 Required Methodology

1. If the CDF value **CDF_D_ACCEPTS_POWER_FROM_SOURCE** indicates that DUT accepts power, then proceed with this test. Else PASS (SKIP).
2. Disconnect power from the Dongle DUT.
3. If necessary, connect Dongle DUT to Tester Source port.
4. Connect the Dongle DUT to the downstream device.
5. Set Tester Source port to disconnected state.
6. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
7. Tester measures CBUS Discovery Resistor.
8. FAIL if Discovery Resistor is not a valid Dongle Discovery Resistor.
9. Tester drives the VBUS with a current limited 200mA , then waits $T_{SRC_VBUS_OUT_TO_STABLE}\{min\}$.
10. Tester Source measures pre-discovery VBUS Current.
11. FAIL if VBUS current is not less than or equal to $I_{VBUS_PREDISCOVERY}\{max\}$.
12. Set Tester Source to use its Source Packet engine to ACK everything.
13. Execute the Tester_Source_Discovery procedure.
14. After Discovery completes, Tester Source measures post-discovery VBUS Current.
15. FAIL if VBUS current is not less than or equal to $I_{VBUS_SOURCE_TO_DONGLE}\{min\}$.
16. If all preceding steps pass, then PASS; else FAIL.

5.3.26 Link Layer Timing – Dongle DUT VBUS Transition

Verify that a connected Dongle DUT behaves correctly when its local power supply is applied.

5.3.26.1 CBT-Dongle: Dongle DUT VBUS Transition

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

5.3.26.1.1 Test Objective

Verify that a connected Dongle DUT behaves correctly when its local power supply is applied.

5.3.26.1.2 References

[MHL] Section 8.2.3; Table 8-5; DONGLE1, DONGLE2, DONGLE3

[MHL] Section 8.2.3; Table 8-5; DONGLE5, DONGLE6, DONGLE7

[MHL] Section 13.10.1; Table 13-27; $Z_{CBUS_DONGLE_DISCOVER}$

[MHL] Section 13.10.1.1; Table 13-29: $T_{SRC_VBUS_OUT_TO_STABLE}$

[MHL] Section 13.10.1.1; Table 13-30: $T_{SRC_CBUS_FLOAT}$

5.3.26.1.3 Required Test Equipment

Use one of the CBUS Source DUT Common Test Equipment Setups.

5.3.26.1.4 Required Methodology

1. If the CDF value **CDF_D_ACCEPTS_POWER_FROM_SOURCE** indicates that the DUT accepts power, then proceed with this test. Else PASS (SKIP).
2. If the CDF value **CDF_D_POWERED** indicates that DUT is not a Powered Dongle, then PASS(SKIP).
3. Disconnect power from the Dongle DUT.
4. If necessary, connect Dongle DUT to Tester Source port.
5. Connect the Dongle DUT to the downstream device.
6. Set Tester Source port to disconnected state.
7. Set Tester Source port to use typical timings and to exhibit typical resistances and capacitances.
8. Tester drives the VBUS, then waits $T_{SRC_VBUS_OUT_TO_STABLE}\{min\}$.
9. Execute the Tester_Source_Discovery procedure.
10. After discovery completes, connect Dongle Power Supply to the Dongle DUT.
11. FAIL if Dongle DUT does not post a DCAP_CHG interrupt within the Dongle Power-up time specified in the CDF as **CDF_D_MAX_POWER_UP**.
12. Execute the Tester_Wait_Capability_Registers_Valid procedure.
13. FAIL if the procedure indicates TIMEOUT.
14. Tester Source does READ_DEVCAP of register DEV_CAT.
15. FAIL if READ_DEVCAP fails.
16. FAIL if POW bit is not 1.
17. Source Tester removes VBUS based on reading POW==1.
18. FAIL if VBUS does not stay HIGH, driven by the Powered Dongle.
19. If all preceding steps pass, then PASS; else FAIL.

6 Tests Common to Sources, Sinks and Dongles

The following additional tests are required for Sources, Sinks and Dongles unless noted otherwise in this section.

6.1 Electrical Tests

There are no electrical tests common to sources, sinks and dongles in this release.

6.2 System Tests

There are no system tests common to Sources, Sinks and Dongles in this release.

6.3 CBUS Tests

If not otherwise indicated, the following tests must be run on all MHL outputs of source DUTs, and on all MHL inputs of Sink DUTs.

6.3.1 MSC – Source and Sink DUT Input: Device Register Space Contents; Reads

Issue READ_DEVCAP commands to the DUT with good arguments.

6.3.1.1 CBM: Capability Regs; READ_DEVCAP of Capability Register Contents

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.1.1.1 Test Objective

Verify that DUT returns valid data when it receives a READ_DEVCAP of Capability register.

6.3.1.1.2 References

[MHL] Section 7.8

[MHL] Section 13.10.1.1; Table 13-31: T_{SRC_CONN}

[MHL] Section 13.10.1.2; Table 13-32; T_{SINK:CBUS_SINK_CONN}

6.3.1.1.3 Required Test Equipment

MSC Test Tool

6.3.1.1.4 Required Methodology

Execute the following test procedure once for every row in Table 6-1. TEST_DEVCAP_OFFSET will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. After Discovery completes, wait T_{SRC_CONN} or T_{SINK_CONN} from the last discovery pulse.
10. Execute the Tester_Wait_Capability_Registers_Valid procedure.
11. FAIL if the procedure indicates TIMEOUT.
12. Send READ_DEVCAP command (0x2_1_61).
13. Send an Offset packet of TEST_DEVCAP_OFFSET.
14. FAIL if DUT does not reply with an ACK packet followed by a Data packet within 2 * T_{PKT_SENDER_TIMEOUT}{max} from the time of the READ_DEVCAP.

15. FAIL if read data is not as documented in the CDF. The **CDF_CR_POW** bit should be ignored for this comparison for devices which can change this bit. When the DUT returns **CDF_CR_POW** == '0', the two other bits {PLIM1, PLIM0} should also be ignored.
16. For table entry 2, FAIL if DUT does not set the upper 4 bits of MHL_VERSION to 0x2.
17. For table entry 3, FAIL if DUT is a Source device and DEV_CAP bits {PLIM1, PLIM0} are not {0, 0}.
18. For table entry 3, FAIL if DUT is a Sink or a Dongle device and DEV_CAP bits {PLIM1, PLIM0} are {1, 1}.
19. For table entry 11, FAIL if DUT is a Source or Sink and RCP_SUPPORT and RAP_SUPPORT are not both 1.
20. For table entry 11, FAIL if DUT is a Source or Sink and SP_SUPPORT is not 1.
21. For table entry 14, FAIL if SP_SUPPORT is 1 and SCRATCHPAD_SIZE is less than 16 or greater than 64.
22. For table entry 15:
 - a. FAIL if the value in bits [7:4] is less than 0x3, indicating the minimum 4 bytes of status.
 - b. FAIL if the value in bits [3:0] is less than 0x3, indicating the minimum 4 bytes of interrupts.
23. If all preceding steps pass, then PASS; else FAIL.

Table 6-1. Read Device Capability Registers

| | TEST_DEVCAP_OFFSET | Reference | Purpose |
|----|--------------------|------------------------|-------------------------------------|
| 1 | 0x00 | [MHL] Section 7.8.1.1 | Capability Register DEV_STATE |
| 2 | 0x01 | [MHL] Section 7.8.1.2 | Capability Register MHL_VERSION |
| 3 | 0x02 | [MHL] Section 7.8.1.3 | Capability Register DEV_CAT |
| 4 | 0x03 | [MHL] Section 7.8.1.4 | Capability Register ADOPTER_ID_H |
| 5 | 0x04 | [MHL] Section 7.8.1.4 | Capability Register ADOPTER_ID_L |
| 6 | 0x05 | [MHL] Section 7.8.1.5 | Capability Register VID_LINK_MODE |
| 7 | 0x06 | [MHL] Section 7.8.1.6 | Capability Register AUD_LINK_MODE |
| 8 | 0x07 | [MHL] Section 7.8.1.7 | Capability Register VIDEO_TYPE |
| 9 | 0x08 | [MHL] Section 7.8.1.8 | Capability Register LOG_DEV_MAP |
| 10 | 0x09 | [MHL] Section 7.8.1.9 | Capability Register BANDWIDTH |
| 11 | 0x0A | [MHL] Section 7.8.1.10 | Capability Register FEATURE_FLAG |
| 12 | 0x0B | [MHL] Section 7.8.1.11 | Capability Register DEVICE_ID_H |
| 13 | 0x0C | [MHL] Section 7.8.1.11 | Capability Register DEVICE_ID_L |
| 14 | 0x0D | [MHL] Section 7.8.1.12 | Capability Register SCRATCHPAD_SIZE |
| 15 | 0x0E | [MHL] Section 7.8.1.13 | Capability Register INT_STAT_SIZE |

6.3.2 MSC – Source and Sink DUT Output: Vendor-specific and Reserved Header Values

6.3.2.1 CBM: DUT Sends Vendor-Specific and Reserved Header Values

Application: Source, Sink, Dongle

6.3.2.1.1 Test Objective

Continuously monitor the CBUS to observe that the DUT never sends MHL Sideband Channel Commands with Vendor-specific or Reserved header values.

6.3.2.1.2 References

[MHL] Section 7.2.3.2; Table 7-1

6.3.2.1.3 Required Test Equipment

MSC Test Tool

6.3.2.1.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. This continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester to continuously monitor the Header bits of packets on the CBUS.
3. If DUT does not send any command with a header value of 2'b01 or 2'b1, end test with PASS (SKIP).

6.3.3 MSC – Source and Sink DUT Output: Normal Commands

6.3.3.1 CBM: DUT sends (0x62) GET_STATE command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.3.1.1 Test Objective

Observe that the DUT sends valid MHL Sideband Channel Commands. Respond with legal results, and observe the DUT responses. Observe that the DUT sends valid GET_STATE commands, and waits for a response before sending another MSC command.

6.3.3.1.2 References

[MHL] Section 7.4.3.1

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; T_{PKT_RECEIVER_TIMEOUT}

[MHL] Section 13.10.3; Table 13-34; T_{PKT_SENDER_TIMEOUT}

6.3.3.1.3 Required Test Equipment

MSC Test Tool

6.3.3.1.4 Required Methodology

Execute the procedure once for each row in Table 6-2. TEST_WATCH_COMMAND, TEST_WAIT_TIME, and TEST_RESPONSE will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, Tester sends a SET_HPD to the DUT.
10. Watch CBUS activity for 10 seconds, and if tester receives a TEST_WATCH_COMMAND, proceed with this test.
11. Wait TEST_WAIT_TIME.

12. Respond with TEST_RESPONSE. TEST_RESPONSE is defined in the MHL Spec as the value stored in the DEV_STATE Capability Register, which is always 0.
13. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
14. FAIL if DUT sends an unexpected MSC command or unexpected Data during the time it is waiting for Data from the Tester.
15. FAIL if DUT makes a NACK packet or ABORT packet in response to the Data returned by the Tester.
16. If all preceding steps pass, then PASS; else FAIL.

Table 6-2. GET_STATE Command

| | TEST_WATCH_COMMAND | TEST_WAIT_TIME | TEST_RESPONSE |
|---|----------------------|---|---------------|
| 1 | (0x2_1_62) GET_STATE | 0 | (0x2_0_00) |
| 2 | (0x2_1_62) GET_STATE | (0.99) *T _{PKT_RECEIVER_TIMEOUT} {min} | (0x2_0_00) |

6.3.3.2 CBM: DUT sends (0x63) GET_VENDOR_ID Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.3.2.1 Test Objective

Observe that the DUT sends valid MHL Sideband Channel Commands. Respond with legal results, and observe the DUT responses. Observe that the DUT sends valid GET_VENDOR_ID commands, and waits for a response before sending another MSC command.

6.3.3.2.2 References

[MHL] Section 7.4.3.2

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; T_{PKT_RECEIVER_TIMEOUT}

[MHL] Section 13.10.3; Table 13-34; T_{PKT_SENDR_TIMEOUT}

6.3.3.2.3 Required Test Equipment

MSC Test Tool

6.3.3.2.4 Required Methodology

Execute the procedure once for each row in Table 6-3. TEST_WATCH_COMMAND, TEST_WAIT_TIME, and TEST_RESPONSE will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, Tester sends a SET_HPD to the DUT.
10. Watch CBUS activity for 10 seconds, and if tester receives a TEST_WATCH_COMMAND, proceed with this test.
11. Wait TEST_WAIT_TIME.

12. Respond with TEST_RESPONSE.
13. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
14. FAIL if DUT sends an unexpected MSC command or unexpected Data during the time it is waiting for Data from the Tester.
15. FAIL if DUT makes a NACK packet or ABORT packet in response to the Data returned by the Tester.
16. If all preceding steps pass, then PASS; else FAIL.

Table 6-3. GET_VENDOR_ID Command

| | TEST_WATCH_COMMAND | TEST_WAIT_TIME | TEST_RESPONSE |
|---|--------------------------|--|---------------|
| 1 | (0x2_1_63) GET_VENDOR_ID | 0 | (0x2_0_00) |
| 2 | (0x2_1_63) GET_VENDOR_ID | (0.99) * T _{PKT_RECEIVER_TIMEOUT} {min} | (0x2_0_00) |

6.3.3.3 CBM: DUT sends (0x6B) GET_MSC_ERRORCODE Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.3.3.1 Test Objective

Observe that the DUT sends valid MHL Sideband Channel Commands. Respond with legal results, and observe the DUT responses. Observe that the DUT sends valid GET_MSC_ERRORCODE commands, and waits for a response before sending another MSC command.

6.3.3.3.2 References

[MHL] Section 7.4.3.6

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; T_{PKT_RECEIVER_TIMEOUT}

[MHL] Section 13.10.3; Table 13-34; T_{PKT_SENDR_TIMEOUT}

6.3.3.3.3 Required Test Equipment

MSC Test Tool

6.3.3.3.4 Required Methodology

Execute the procedure once for each row in Table 6-4. TEST_WATCH_COMMAND, TEST_WAIT_TIME, and TEST_RESPONSE will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, Tester sends a SET_HPDP to the DUT.
10. Watch CBUS activity for 10 seconds, and if tester receives a TEST_WATCH_COMMAND, proceed with this test.
11. Wait TEST_WAIT_TIME.

12. Respond with TEST_RESPONSE.
13. At powerup, the TEST_RESPONSE to a GET_MSC_ERRORCODE will return (0x2_0_00).
14. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
15. FAIL if DUT sends unexpected MSC command or unexpected Data during the time it is waiting for Data from the Tester.
16. FAIL if DUT makes a NACK packet or ABORT packet in response to the Data returned by the Tester.
17. If all preceding steps pass, then PASS; else FAIL.

Table 6-4. GET_MSC_ERRORCODE Command

| | TEST_WATCH_COMMAND | TEST_WAIT_TIME | TEST_RESPONSE |
|---|------------------------------|---------------------------------|---------------|
| 1 | (0x2_1_6B) GET_MSC_ERRORCODE | 0 | (0x2_0_00) |
| 2 | (0x2_1_6B) GET_MSC_ERRORCODE | (0.99) * T_PKT_RECEIVER_TIMEOUT | (0x2_0_00) |

6.3.3.4 CBM: DUT sends (0x60) SET_INT/WRITE_STAT Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.3.4.1 Test Objective

Observe that the DUT sends valid MHL Sideband Channel Commands. Respond with legal results, and observe the DUT responses. Observe that the DUT sends valid SET_INT or WRITE_STAT commands, and waits for a response before sending another MSC command.

6.3.3.4.2 References

[MHL] Section 7.4.3.8

[MHL] Section 7.4.3.9

[MHL] Section 7.8; Table 7-20; Interrupt Registers, Status Registers

[MHL] Section 7.8.1.13; Table 7-36; INT_SIZE, STAT_SIZE

[MHL] Section 13.10.3; Table 13-34; T_PKT_RECEIVER_TIMEOUT

[MHL] Section 13.10.3; Table 13-34; T_PKT_SENDR_TIMEOUT

6.3.3.4.3 Required Test Equipment

MSC Test Tool

6.3.3.4.4 Required Methodology

Execute the procedure once for each row in Table 6-5. TEST_WATCH_COMMAND, TEST_WAIT_TIME, and TEST_RESPONSE will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.

9. If Source DUT, Tester sends a SET_HPDP to the DUT.
10. Watch CBUS activity for 10 seconds, and if tester receives a TEST_WATCH_COMMAND, proceed with this test.
11. Watch for the DUT to supply the Offset (0x2_0_??).
12. Watch for the DUT to supply the Value (0x2_0_??).
13. Wait TEST_WAIT_TIME.
14. Respond with TEST_RESPONSE.
15. The Tester implements 4 Interrupt Registers and 4 Status Registers. This defines the valid range for offsets included in SET_INT and WRITE_STAT commands received from the DUT.
16. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
17. FAIL if DUT sends another MSC command before it sends the Offset data.
18. FAIL if DUT does not send the Offset within $T_{PKT_SENDER_TIMEOUT}\{max\}$ of the command.
19. FAIL if Offset is not in the valid range for an Interrupt Register or for a Status Register.
20. FAIL if DUT sends another MSC command before it sends the Value data.
21. FAIL if DUT does not send the Value data within $T_{PKT_SENDER_TIMEOUT}\{max\}$ of the Offset Byte.
22. FAIL if DUT sends an unexpected MSC command or unexpected Data during the time it is waiting for ACK packet from the Tester.
23. FAIL if DUT makes a NACK packet or ABORT packet in response to the ACK packet returned by the Tester.
24. If all preceding steps pass, then PASS; else FAIL.

Table 6-5. SET_INT and WRITE_STAT Commands

| | TEST_WATCH_COMMAND | TEST_WAIT_TIME | TEST_RESPONSE |
|---|--------------------------------|---|-----------------------|
| 1 | (0x2_1_60) SET_INT –WRITE_STAT | 0 | ACK packet (0x2_1_33) |
| 2 | (0x2_1_60) SET_INT –WRITE_STAT | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | ACK packet (0x2_1_33) |

6.3.3.5 CBM: DUT sends (0x6C) WRITE_BURST Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.3.5.1 Test Objective

Observe that the DUT sends valid MHL Sideband Channel Commands. Respond with legal results, and observe the DUT responses. Observe that the DUT sends valid WRITE_BURST commands, and waits for a response before sending another MSC command.

6.3.3.5.2 References

[MHL] Section 7.4.3.7

[MHL] Section 7.4.3.10

[MHL] Section 7.8; Table 7-20; Scratchpad Registers

[MHL] Section 7.8.1; Table 7-21; SCRATCHPAD_SIZE

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}$

6.3.3.5.3 Required Test Equipment

MSC Test Tool

6.3.3.5.4 Required Methodology

Execute the procedure once for each row in Table 6-6. TEST_WATCH_COMMAND, TEST_WAIT_TIME, and TEST_RESPONSE will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to indicate that it supports scratchpad registers by forcing SP_SUPPORT in its own Feature_Flag Register to 1 and its own SCRATCHPAD_SIZE register to 0x10.
8. Set Tester to use its Packet engine to respond normally.
9. Direct Tester to proceed with Discovery.
10. If Source DUT, Tester sends a SET_HPD to the DUT.
11. Tester sends DCAP_RDY using WRITE_STAT and then DCAP_CHG using SET_INT to DUT.
12. Tester allows DUT to execute WRITE_BURST commands by responding to REQ_WRT interrupts with GRT_WRT interrupts.
13. Watch CBUS activity for 10 seconds, and if tester receives a TEST_WATCH_COMMAND, proceed with this test. Else end the test with PASS (SKIP).
14. Watch for the DUT to supply the Offset (0x2_0_??).
15. Watch for the DUT to supply Data 0 byte (0x2_0_??).
16. Watch for the DUT to supply Data 1 byte (0x2_0_??).
17. Watch for the DUT to supply up to an additional 14 Data bytes (0x2_0_??).
18. Watch for the DUT to supply an EOF control character (0x2_1_32).
19. Wait TEST_WAIT_TIME.
20. Respond with TEST_RESPONSE.
21. Valid addresses are 0x40 to (0x40 + SCRATCHPAD_SIZE – 1).
22. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
23. FAIL if DUT sends another MSC command before it sends the Offset data.
24. FAIL if DUT does not send the Offset within T_{PKT_SENDER_TIMEOUT}{max} of the command.
25. FAIL if Offset is not in the valid range for a Scratchpad Register.
26. FAIL if DUT sends another MSC command before it sends the first Data Byte.
27. FAIL if DUT does not send the first Data within T_{PKT_SENDER_TIMEOUT}{max} of the Offset byte.
28. FAIL if DUT sends another MSC command before it sends following Data Bytes or the EOF byte.
29. FAIL if DUT does not send every subsequent Data within T_{PKT_SENDER_TIMEOUT}{max} of the previous Data byte.
30. FAIL if DUT does not send the EOF control character within T_{PKT_SENDER_TIMEOUT}{max} of the last Data byte.
31. FAIL if DUT sends an unexpected MSC command or unexpected Data during the time it is waiting for ACK packet from the Tester.
32. FAIL if DUT makes a NACK packet or ABORT packet in response to the ACK packet returned by the Tester.
33. FAIL if DUT sends more than 16 Data bytes (including the ADOPTER bytes).
34. FAIL if DUT sends enough data bytes that the Offset + Byte Num increments beyond 0x40 + SCRATCHPAD_SIZE – 1.
35. FAIL if DUT does not send EOF packet within T_{CMD_REQUESTER_TIMEOUT}{max} from the time it sends the WRITE_BURST Command Byte.
36. If all preceding steps pass, then PASS; else FAIL.

Table 6-6. WRITE_BURST Command

| | TEST_WATCH_COMMAND | TEST_WAIT_TIME | TEST_RESPONSE |
|---|------------------------|--|-----------------------|
| 1 | (0x2_1_6C) WRITE_BURST | 0 | ACK packet (0x2_1_33) |
| 2 | (0x2_1_6C) WRITE_BURST | (0.99) * T _{PKT_RECEIVER_TIMEOUT} {min} | ACK packet (0x2_1_33) |

6.3.3.6 CBM: DUT sends (0x68) MSC_MSG Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.3.6.1 Test Objective

Observe that the DUT sends valid MHL Sideband Channel Commands. Respond with legal results, and observe the DUT responses. Observe that the DUT sends valid MSC_MSG commands, and waits for a response before sending another MSC command.

6.3.3.6.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; T_{PKT_RECEIVER_TIMEOUT}

[MHL] Section 13.10.3; Table 13-34; T_{PKT_SENDER_TIMEOUT}

[MHL] Section 7.4.3.11

6.3.3.6.3 Required Test Equipment

MSC Test Tool

6.3.3.6.4 Required Methodology

Execute the procedure once for each row in Table 6-7. TEST_WATCH_COMMAND, TEST_WAIT_TIME, and TEST_RESPONSE will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Set Tester to indicate that it supports the MSC_MSG command by forcing both RCP_SUPPORT and RAP_SUPPORT in the Feature_Flag Register to 1.
9. Direct Tester to proceed with Discovery.
10. If Source DUT, Tester sends a SET_HPD to the DUT.
11. Watch CBUS activity for 10 seconds, and if tester receives a TEST_WATCH_COMMAND, proceed with this test.
12. Watch for the DUT to supply first Data byte (0x2_0_??).
13. Watch for the DUT to supply second Data byte (0x2_0_??).
14. Wait TEST_WAIT_TIME.
15. Respond with TEST_RESPONSE.
16. Valid first Data items are:
 - a. RCP (0x10)
 - b. RCPK (0x11)
 - c. RCPE (0x12)

- d. RAP (0x20)
 - e. RAPK (0x21)
 - f. UCP (0x30)
 - g. UCPK (0x31)
 - h. UCPE (0x32)
17. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
 18. FAIL if DUT sends another MSC command before it sends the first Data byte.
 19. FAIL if DUT does not send the first Data byte within $T_{PKT_SENDER_TIMEOUT\{max\}}$ of the command.
 20. FAIL if the first Data byte value is not an allowed value.
 21. FAIL if DUT sends another MSC command before it sends second Data Byte.
 22. FAIL if DUT does not send the second Data byte within $T_{PKT_SENDER_TIMEOUT\{max\}}$ of the first Data byte.
 23. FAIL if DUT sends an unexpected MSC command or unexpected Data during the time it is waiting for ACK packet from the Tester.
 24. FAIL if DUT makes a NACK packet or ABORT packet in response to the ACK packet returned by the Tester.
 25. If all preceding steps pass, then PASS; else FAIL.

Table 6-7. MSC_MSG Command

| | TEST_WATCH_COMMAND | TEST_WAIT_TIME | TEST_RESPONSE |
|---|--------------------|--|-----------------------|
| 1 | (0x2_1_68) MSC_MSG | 0 | ACK packet (0x2_1_33) |
| 2 | (0x2_1_68) MSC_MSG | $(0.99) * T_{PKT_RECEIVER_TIMEOUT\{min\}}$ | ACK packet (0x2_1_33) |

6.3.3.7 CBM: DUT sends (0x6A) GET_DDC_ERRORCODE Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.3.7.1 Test Objective

Observe that the DUT sends valid MHL Sideband Channel Commands. Respond with legal results, and observe the DUT responses. Observe that the DUT sends valid GET_MSC_ERRORCODE commands, and waits for a response before sending another MSC command.

6.3.3.7.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}$

[MHL] Section 7.4.3.6

6.3.3.7.3 Required Test Equipment

MSC Test Tool

6.3.3.7.4 Required Methodology

Execute the procedure once for each row in Table 6-8. TEST_WATCH_COMMAND, TEST_WAIT_TIME, and TEST_RESPONSE will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.

6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, Tester sends a SET_HPD to the DUT.
10. Watch CBUS activity for 10 seconds, and if tester receives a TEST_WATCH_COMMAND, proceed with this test.
11. Wait TEST_WAIT_TIME.
12. Respond with TEST_RESPONSE.
13. At powerup, the TEST_RESPONSE to a GET_DDC_ERRORCODE will return (0x2_0_00).
14. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
15. FAIL if DUT sends unexpected MSC command or unexpected Data during the time it is waiting for Data from the Tester.
16. FAIL if DUT makes a NACK packet or ABORT packet in response to the Data returned by the Tester.
17. If all preceding steps pass, then PASS; else FAIL.

Table 6-8. GET_DDC_ERRORCODE Command

| | TEST_WATCH_COMMAND | TEST_WAIT_TIME | TEST_RESPONSE |
|---|------------------------------|--|---------------|
| 1 | (0x2_1_6A) GET_DDC_ERRORCODE | 0 | (0x2_0_00) |
| 2 | (0x2_1_6A) GET_DDC_ERRORCODE | (0.99) * T _{PKT_RECEIVER_TIMEOUT} {min} | (0x2_0_00) |

6.3.4 MSC – Source and Sink DUT Output: NACK Packet Response to MSC_MSG

6.3.4.1 CBM: DUT Receives NACK to MSC_MSG

Application: Source

6.3.4.1.1 Test Objective

Verify that DUT responds appropriately when it receives a NACK packet in response to a command.

6.3.4.1.2 References

[MHL] Section 7.4.3.6; Table 7-6; Peer is Busy

[MHL] Section 7.8.1.1

6.3.4.1.3 Required Test Equipment

MSC Test Tool

6.3.4.1.4 Required Methodology

Execute the procedure once for each row in Table 6-9. TEST_BYTE_TO_NACK will take on values defined in the following table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Set Tester to be a Dongle.
4. Tester Dongle drives the VBUS.
5. Set Tester to indicate that MSC_MSG commands are not allowed, by setting bits 0 and 1 to “0” in the Feature Support Flag Capability Register (0x0A).

6. Select typical timings, resistances, capacitances.
7. Set Tester to use its Discovery engine to enable normal MHL discovery.
8. Set Tester to use its Packet engine to respond normally.
9. Direct Tester to proceed with Discovery.
10. If Source DUT, Tester sends a SET_HPD to the DUT.
11. Watch for MSC_MSG (0x2_1_68) to occur on the CBUS.
12. Send a NACK packet (0x2_1_34) to the DUT after TEST_BYTE_TO_NACK (or as soon as arbitration makes it possible).
13. If DUT does not send the command the Tester is sensitized to within 10 seconds, end test with PASS (SKIP).
14. FAIL if DUT sends an ABORT (0x2_1_35) in response to the NACK packet.
15. If all preceding steps pass, then PASS; else FAIL.

Table 6-9. DUT Receives NACK to MSC_MSG

| | TEST_BYTE_TO_NACK | Purpose |
|---|-------------------|---|
| 1 | 1 | NACK packet to MSC_MSG; after opcode |
| 2 | 2 | NACK packet to MSC_MSG; after sub-command |
| 3 | 3 | NACK packet to MSC_MSG; after value |

6.3.5 MSC – Source and Sink DUT Output: Never Initiates Bad Commands

Continuously monitor the CBUS to observe that the DUT sends only valid MHL Sideband Channel Commands.

6.3.5.1 CBM: DUT Never Sends Reserved Commands

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.5.1.1 Test Objective

Observe that the DUT never sends reserved MSC commands.

6.3.5.1.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}\{max\}$

6.3.5.1.3 Required Test Equipment

MSC Test Tool

6.3.5.1.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. The continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. FAIL if DUT sends a reserved command:
 - a. (0x69) GET_SC1_ERRORCODE or
 - b. (0x6D) GET_SC3_ERRORCODE
2. If all preceding steps pass, then PASS; else FAIL.

6.3.5.2 CBM: DUT Never Sends Illegal Commands

Application: Source, Sink, Dongle

6.3.5.2.1 Test Objective

Observe that the DUT never sends illegal MSC commands.

6.3.5.2.2 References

[MHL] Section 7.4; Table 7-3

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; T_{PKT_RECEIVER_TIMEOUT}

[MHL] Section 13.10.3; Table 13-34; T_{PKT_SENDER_TIMEOUT}{max}

6.3.5.2.3 Required Test Equipment

MSC Test Tool

6.3.5.2.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. The continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. If any illegal MSC command is received, respond with an ABORT packet (0x2_1_35).
2. If all preceding steps pass, then PASS; else FAIL.

Table 6-10. Legal MSC Commands

| Opcode | Legal Command | Legal Command for DUT: | | |
|--------|-------------------|------------------------|------|--------|
| | | Source | Sink | Dongle |
| 0x35 | ABORT | X | X | X |
| 0x62 | GET_STATE | X | X | X |
| 0x32 | EOF | X | X | X |
| 0x33 | ACK | X | X | X |
| 0x34 | NACK | X | X | X |
| 0x60 | SET_INT | X | X | X |
| 0x60 | WRITE_STAT | X | X | X |
| 0x61 | READ_DEVCAP | X | X | X |
| 0x63 | GET_VENDOR_ID | X | X | X |
| 0x64 | SET_HPD | | X | X |
| 0x65 | CLR_HPD | | X | X |
| 0x68 | MSC_MSG | X | X | X |
| 0x6A | GET_DDC_ERRORCODE | X | X | X |
| 0x6B | GET_MSC_ERRORCODE | X | X | X |
| 0x6C | WRITE_BURST | X | X | X |

6.3.5.3 CBM: DUT Never Sends Data While No Command is Outstanding

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.5.3.1 Test Objective

Observe that DUT never sends Data when none is expected.

6.3.5.3.2 References

[MHL] Section 7.3.1.1

[MHL] Section 7.4.1

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_RECEIVER_TIMEOUT}}$

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$

6.3.5.3.3 Required Test Equipment

MSC Test Tool

6.3.5.3.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. The continuous monitor procedure will remain in effect throughout all following CBUS tests.

If Data (0x2_0_??) is received when none expected, respond with an ABORT packet (0x2_1_35).

If all preceding steps pass, then PASS; else FAIL.

6.3.5.4 CBM: DUT Never Sends (0x33) ACK packet While No Command is Outstanding

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.5.4.1 Test Objective

Observe that DUT never sends ACK packet when none is expected.

6.3.5.4.2 References

[MHL] Section 7.3.1.1

[MHL] Section 7.4.1

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_RECEIVER_TIMEOUT}}$

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$

6.3.5.4.3 Required Test Equipment

MSC Test Tool

6.3.5.4.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. The continuous monitor procedure will remain in effect throughout all following CBUS tests.

If ACK packet (0x2_1_33) is received when none expected, respond with an ABORT packet (0x2_1_35).

If all preceding steps pass, then PASS; else FAIL.

6.3.5.5 CBM: DUT Never Sends (0x34) NACK Packet While No Command is Outstanding

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.5.5.1 Test Objective

Observe that DUT never sends NACK (0x2_1_34) packet when none is expected.

6.3.5.5.2 References

[MHL] Section 7.3.1.2

[MHL] Section 7.4.1

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_RECEIVER_TIMEOUT}}$

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$

6.3.5.5.3 Required Test Equipment

MSC Test Tool

6.3.5.5.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. The continuous monitor procedure will remain in effect throughout all following CBUS tests.

If NACK packet (0x2_1_34) is received when none expected, respond with an ABORT packet (0x2_1_35).

If all preceding steps pass, then PASS; else FAIL.

6.3.5.6 CBM: DUT Never Sends (0x35) ABORT While No Command is Outstanding

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.5.6.1 Test Objective

Observe that DUT never sends ABORT (0x2_1_35) when none is expected.

6.3.5.6.2 References

[MHL] Section 7.3.1.3

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_RECEIVER_TIMEOUT}}$

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$

6.3.5.6.3 Required Test Equipment

MSC Test Tool

6.3.5.6.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. The continuous monitor procedure will remain in effect throughout all following CBUS tests.

If ABORT (0x2_1_35) is received when none expected, do nothing.

If all preceding steps pass, then PASS; else FAIL.

6.3.5.7 CBM: DUT Never Sends (0x32) EOF While No Command is Outstanding

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.5.7.1 Test Objective

Observe that DUT never sends EOF (0x2_1_32) when none is expected.

6.3.5.7.2 References

[MHL] Section 7.3.1.2

[MHL] Section 7.4.1

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_RECEIVER_TIMEOUT}}$

[MHL] Section 13.10.3; Table 13-34; $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$

6.3.5.7.3 Required Test Equipment

MSC Test Tool

6.3.5.7.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. The continuous monitor procedure will remain in effect throughout all following CBUS tests.

If EOF (0x2_1_32) is received when none expected, respond with an ABORT packet (0x2_1_35).

If all preceding steps pass, then PASS; else FAIL.

6.3.5.8 CBM: DUT Never Sends WRITE_BURST Command Without First Arbitrating Using REQ_WRT/GRT_WRT Interrupt Procedure.

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.5.8.1 Test Objective

Observe that DUT never sends WRITE_BURST without first arbitrating using REQ_WRT/GR_WRT interrupt procedure.

6.3.5.8.2 References

[MHL] Section 7.9.2.1

6.3.5.8.3 Required Test Equipment

MSC Test Tool

6.3.5.8.4 Required Methodology

Define the following continuous monitor procedure. It will be executed before any other test procedure in the CBUS Test Suite. The continuous monitor procedure will remain in effect throughout all following CBUS tests.

1. FAIL if DUT sends a WRITE_BURST without first doing REQ_WRT/GRT_WRT arbitration.
2. FAIL if DUT does not do a DSCR_CHG interrupt soon after the WRITE_BURST.
3. If all preceding steps pass, then PASS; else FAIL.

6.3.6 MSC – Source and Sink DUT Output: Errors and Exceptions

6.3.6.1 CBM: DUT Receives Bad Reply; Control instead of Data

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.6.1.1 Test Objective

Respond to valid MSC commands with illegal results, and observe the DUT responses. Verify that DUT does something predictable when Tester replies to a Command with a Control character when Data is expected.

6.3.6.1.2 References

[MHL] Section 7.4.1

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}\{max\}$

6.3.6.1.3 Required Test Equipment

MSC Test Tool

6.3.6.1.4 Required Methodology

Execute the following test procedure. TEST_COMMAND_TO_CLOBBER will take on values defined in Table 6-11.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, wait T_{SRC_CONN} after Discovery complete and send a SET_HPD to the DUT.
10. Watch for TEST_COMMAND_TO_CLOBBER to occur on the CBUS.
11. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
12. Respond with an ACK packet (0x2_1_33) instead of a Data packet.
13. FAIL if DUT does not respond with an ABORT packet (0x2_1_35) within $T_{PKT_SENDER_TIMEOUT}\{max\}$ of the ACK packet.
14. If all preceding steps pass, then PASS; else FAIL.

Table 6-11. DUT Receives Bad Reply – Control instead of Data

| | TEST_COMMAND_TO_CLOBBER | Purpose |
|---|------------------------------|--|
| 1 | (0x2_1_62) GET_STATE | Control instead of Data; GET_STATE |
| 2 | (0x2_1_63) GET_VENDOR_ID | Control instead of Data; GET_VENDOR_ID |
| 3 | (0x2_1_6B) GET_MSC_ERRORCODE | Control instead of Data; GET_MSC_ERRORCODE |
| 4 | (0x2_1_6A) GET_DDC_ERRORCODE | Control instead of Data; GET_DDC_ERRORCODE |

6.3.6.2 CBM: DUT Receives Bad Reply; Data instead of Control

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.6.2.1 Test Objective

Respond to valid MSC commands with illegal results, and observe the DUT responses. Verify that DUT does something predictable when Tester replies to a Command with Data when a Control character is expected.

6.3.6.2.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}\{max\}$

[MHL] Section 7.4.1

6.3.6.2.3 Required Test Equipment

MSC Test Tool

6.3.6.2.4 Required Methodology

Execute the following test procedure. TEST_COMMAND_TO_CLOBBER will take on values defined in Table 6-12.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, wait T_{SRC_CONN} after Discovery complete and send a SET_HPD to the DUT.
10. Tester sends DCAP_RDY using WRITE_STAT and then DCAP_CHG using SET_INT to DUT.
11. Tester allows DUT to execute WRITE_BURST commands by responding to REQ_WRT interrupts with GRT_WRT interrupts.
12. Watch for TEST_COMMAND_TO_CLOBBER to occur on the CBUS.
13. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
14. Respond with a Data packet (0x2_0_0).
15. FAIL if DUT does not respond with an ABORT packet (0x2_1_35) within $T_{PKT_SENDER_TIMEOUT}\{max\}$ of the ACK packet.
16. If all preceding steps pass, then PASS; else FAIL.

Table 6-12. DUT Receives Bad Reply – Data instead of Control

| | TEST_COMMAND_TO_CLOBBER | Purpose |
|---|-------------------------------|---|
| 1 | (0x2_1_61) READ_DEVCAP | Data instead of Control; READ_DEVCAP |
| 2 | (0x2_1_60) WRITE_STAT/SET_INT | Data instead of Control; WRITE_STAT/SET_INT |
| 3 | (0x2_1_68) MSC_MSG | Data instead of Control; MSC_MSG |
| 4 | (0x2_1_6C) WRITE_BURST | Data instead of Control; WRITE_BURST |
| 5 | (0x2_1_64) SET_HPD | Data instead of Control; SET_HPD |
| 6 | (0x2_1_65) CLR_HPD | Data instead of Control; CLR_HPD |

6.3.6.3 CBM: DUT Receives Bad Reply; Control, Control instead of Control, Data

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.6.3.1 Test Objective

Respond to valid MSC commands with illegal results, and observe the DUT responses. Verify that DUT does something predictable when Tester replies to a Command with a Control character when Data is expected.

6.3.6.3.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}\{max\}$

[MHL] Section 7.4.1

6.3.6.3.3 Required Test Equipment

MSC Test Tool

6.3.6.3.4 Required Methodology

Execute the following test procedure. TEST_COMMAND_TO_BAD_REPLY will take on values defined in Table 6-13.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, wait T_{SRC_CONN} after Discovery complete and send a SET_HPD to the DUT.
10. Tester sends DCAP_RDY using WRITE_STAT and then DCAP_CHG using SET_INT to DUT.
11. Watch for TEST_COMMAND_TO_BAD_REPLY to occur on the CBUS.
12. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
13. Respond with an ACK packet (0x2_1_33).
14. Respond with a second ACK packet (0x2_1_33).
15. FAIL if DUT does not respond with an ABORT packet (0x2_1_35) within $T_{PKT_SENDER_TIMEOUT}\{max\}$ of the second ACK packet.
16. If all preceding steps pass, then PASS; else FAIL.

Table 6-13. DUT Receives Bad Reply – Control-then-Control instead of Control-then-Data

| | TEST_COMMAND_TO_BAD_REPLY | Purpose |
|---|---------------------------|--------------------------------------|
| 1 | (0x2_1_61) READ_DEVCAP | Control instead of Data; READ_DEVCAP |

6.3.6.4 CBM: DUT Receives Result Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.6.4.1 Test Objective

Respond to valid MSC commands with illegal results or timeout, and observe the DUT responses.

6.3.6.4.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}\{max\}$

6.3.6.4.3 Required Test Equipment

MSC Test Tool

6.3.6.4.4 Required Methodology

Execute the following test procedure. TEST_COMMAND_TO_TIMEOUT, TEST_TIMEOUT_ACK, and TEST_TIMEOUT_DATA will take on values defined in Table 6-14.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, wait T_{SRC_CONN} after Discovery complete and send a SET_HPD to the DUT.
10. Tester sends DCAP_RDY using WRITE_STAT and then DCAP_CHG using SET_INT to DUT.
11. Tester allows DUT to execute WRITE_BURST commands by responding to REQ_WRT interrupts with GRT_WRT interrupts.
12. Watch for TEST_COMMAND_TO_TIMEOUT to occur on the CBUS.
13. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
14. Watch for arguments to be received from DUT (if any).
15. If TEST_TIMEOUT_ACK==1, then Tester never sends the ACK packet, and does not send any other MSC packet for at least 2 seconds.
16. If TEST_TIMEOUT_DATA==1, then Tester never sends the Data packet, and does not send any other MSC packet for at least 2 seconds.
17. After the timeout expires, direct Tester to issue a GET_STATE command to the DUT.
18. If DUT sends any MSC packet earlier than $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ from its original command, then FAIL.
19. FAIL if DUT does not respond to the GET_STATE by returning either a Data packet or an ABORT packet.
20. If all preceding steps pass, then PASS; else FAIL.

Table 6-14. DUT Receives Result Timeout

| | TEST_COMMAND_TO_TIMEOUT | TEST_TIMEOUT_ACK | TEST_TIMEOUT_DATA | Purpose |
|---|-------------------------------|------------------|-------------------|---------------------------------|
| 1 | (0x2_1_62) GET_STATE | 0 | 1 | Timeout to GET_STATE |
| 2 | (0x2_1_63) GET_VENDOR_ID | 0 | 1 | Timeout to GET_VENDOR_ID |
| 3 | (0x2_1_61) READ_DEVCAP | 1 | 0 | READ_DEVCAP; timeout ACK packet |
| 4 | (0x2_1_61) READ_DEVCAP | 0 | 1 | READ_DEVCAP; timeout Data |
| 5 | (0x2_1_6B) GET_MSC_ERRORCODE | 0 | 1 | Timeout to GET_MSC_ERRORCODE |
| 6 | (0x2_1_60) WRITE_STAT/SET_INT | 1 | 0 | Timeout to WRITE_STAT/SET_INT |
| 7 | (0x2_1_68) MSC_MSG | 1 | 0 | Timeout to MSC_MSG |
| 8 | (0x2_1_6C) WRITE_BURST | 1 | 0 | Timeout to WRITE_BURST |

| | TEST_COMMAND_TO_TIMEOUT | TEST_TIMEOUT_ACK | TEST_TIMEOUT_DATA | Purpose |
|----|------------------------------|------------------|-------------------|------------------------------|
| 9 | (0x2_1_64) SET_HPD | 1 | 0 | Timeout to SET_HPD |
| 10 | (0x2_1_65) CLR_HPD | 1 | 0 | Timeout to CLR_HPD |
| 11 | (0x2_1_6A) GET_DDC_ERRORCODE | 0 | 1 | Timeout to GET_DDC_ERRORCODE |

6.3.6.5 CBM: Verify No Next Command Until Hold-Off after ABORT Seen

Application: Source, Sink, Dongle

6.3.6.5.1 Test Objective

Respond to valid MSC commands with ABORT packets, and observe the DUT responses. Verify that DUT does not start a new command after it receives an ABORT, until the timeout completes.

6.3.6.5.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}\{max\}$

[MHL] Section 13.10.3, Table 13-34, T_{ABORT_NEXT}

6.3.6.5.3 Required Test Equipment

MSC Test Tool

6.3.6.5.4 Required Methodology

Execute the following test procedure. TEST_COMMAND_TO_CLOBBER and TEST_BYTE_TO_ABORT will take on values defined in Table 6-15.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, wait T_{SRC_CONN} after Discovery complete and send a SET_HPD to the DUT.
10. Tester sends DCAP_RDY using WRITE_STAT and then DCAP_CHG using SET_INT to DUT.
11. Tester allows DUT to execute WRITE_BURST commands by responding to REQ_WRT interrupts with GRT_WRT interrupts.
12. Watch for TEST_COMMAND_TO_CLOBBER to occur on the CBUS.
13. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
14. Wait until CBUS is available to the Tester.
15. Send an ABORT packet (0x2_1_35) to the DUT after TEST_BYTE_TO_ABORT (or as soon as arbitration makes it possible).
16. FAIL if DUT sends any MSC command within T_{ABORT_NEXT} of the ABORT packet.
17. If all preceding steps pass, then PASS; else FAIL.

Table 6-15. Verify No Next Command until Hold-Off after ABORT Seen

| | TEST_COMMAND_TO_CLOBBER | TEST_BYTE_TO_ABORT | Purpose |
|----|-------------------------------|--------------------|---|
| 1 | (0x2_1_62) GET_STATE | 1 | ABORT to GET_STATE |
| 2 | (0x2_1_63) GET_VENDOR_ID | 1 | ABORT to GET_VENDOR_ID |
| 3 | (0x2_1_61) READ_DEVCAP | 1 | ABORT to READ_DEVCAP; after opcode |
| 4 | (0x2_1_61) READ_DEVCAP | 2 | ABORT to READ_DEVCAP; after offset |
| 5 | (0x2_1_6B) GET_MSC_ERRORCODE | 1 | ABORT to GET_MSC_ERRORCODE |
| 6 | (0x2_1_60) WRITE_STAT/SET_INT | 1 | ABORT to WRITE_STAT/SET_INT; after opcode |
| 7 | (0x2_1_60) WRITE_STAT/SET_INT | 2 | ABORT to WRITE_STAT/SET_INT; after offset |
| 8 | (0x2_1_60) WRITE_STAT/SET_INT | 3 | ABORT to WRITE_STAT/SET_INT; after value |
| 9 | (0x2_1_6C) WRITE_BURST | 1 | ABORT to WRITE_BURST; after opcode |
| 10 | (0x2_1_6C) WRITE_BURST | 2 | ABORT to WRITE_BURST; after offset |
| 11 | (0x2_1_6C) WRITE_BURST | 3 | ABORT to WRITE_BURST; after first data |
| 12 | (0x2_1_64) SET_HPD | 1 | ABORT to SET_HPD |
| 13 | (0x2_1_65) CLR_HPD | 1 | ABORT to CLR_HPD |
| 14 | (0x2_1_6A) GET_DDC_ERRORCODE | 1 | ABORT to GET_DDC_ERRORCODE |
| 15 | (0x2_1_68) MSC_MSG | 1 | ABORT to MSC_MSG; after opcode |
| 16 | (0x2_1_68) MSC_MSG | 2 | ABORT to MSC_MSG; after subcommand |
| 17 | (0x2_1_68) MSC_MSG | 3 | ABORT to MSC_MSG; after value |

6.3.7 MSC – Source and Sink DUT Output; Disconnect

Verify that a MHL Bus Disconnect does not leave a device hanging in an unknown state.

6.3.7.1 CBM: DUT Receives Disconnect during Various Commands

Application: Source, Sink, Dongle

6.3.7.1.1 Test Objective

Verify that DUT performs disconnect in the middle of executing various commands.

6.3.7.1.2 References

[MHL] Section 7.4.3.6; Table 7-6; Peer is Busy

[MHL] Section 7.8.1.1

6.3.7.1.3 Required Test Equipment

MSC Test Tool

6.3.7.1.4 Required Methodology

Execute the following procedure once for each row in Table 6-16. TEST_COMMAND_TO_DISCONNECT will take on values defined in the tables.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.

3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. If Source DUT, Tester sends a SET_HPDP to the DUT.
10. Tester sends DCAP_RDY using WRITE_STAT and then DCAP_CHG using SET_INT to DUT.
11. Tester allows DUT to execute WRITE_BURST commands by responding to REQ_WRT interrupts with GRT_WRT interrupts.
12. Watch for TEST_COMMAND_TO_DISCONNECT to occur on the CBUS.
13. If seen, set Tester port to disconnected state.
14. Wait for the Tester's $T_{\text{SRC_CBUS_FLOAT}}$ or $T_{\text{SINK_CBUS_FLOAT}}$.
15. Command Tester to proceed with Discovery.
16. Wait $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
17. Send a GET_STATE (0x2_1_62) command.
18. FAIL if DUT has problems doing the first Discovery.
19. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
20. FAIL if DUT has problems doing the second Discovery.
21. FAIL if DUT does not respond correctly to the GET_STATE command.
22. If all preceding steps pass, then PASS; else FAIL.

Table 6-16. DUT Receives Disconnect During Various Commands

| | TEST_COMMAND_TO_DISCONNECT | Purpose |
|----|-------------------------------|-------------------------------------|
| 1 | (0x2_1_62) GET_STATE | Disconnect during GET_STATE |
| 2 | (0x2_1_63) GET_VENDOR_ID | Disconnect during GET_VENDOR_ID |
| 3 | (0x2_1_61) READ_DEVCAP | Disconnect during READ_DEVCAP |
| 4 | (0x2_1_6B) GET_MSC_ERRORCODE | Disconnect during GET_MSC_ERRORCODE |
| 5 | (0x2_1_60) SET_INT/WRITE_STAT | Disconnect during SET_INT |
| 6 | (0x2_1_6C) WRITE_BURST | Disconnect during WRITE_BURST |
| 7 | (0x2_1_68) MSC_MSG | Disconnect during MSC_MSG |
| 8 | (0x2_1_64) SET_HPDP | Disconnect during SET_HPDP |
| 9 | (0x2_1_65) CLR_HPDP | Disconnect during CLR_HPDP |
| 10 | (0x2_1_6A) GET_DDC_ERRORCODE | Disconnect during GET_DDC_ERRORCODE |

6.3.8 MSC – Source and Sink DUT Input: Device Register Space Contents; Writes

Issue write commands to the DUT with good arguments.

6.3.8.1 CBM: Interrupt Regs; SET_INT (0x60); Valid Registers Respond

| | |
|---------------------|----------------------|
| Application: | Source, Sink, Dongle |
|---------------------|----------------------|

6.3.8.1.1 Test Objective

Verify that DUT responds appropriately when SET_INT is done to valid INT registers.

6.3.8.1.2 References

[MHL] Section 7.8.2

6.3.8.1.3 Required Test Equipment

MSC Test Tool

6.3.8.1.4 Required Methodology

INT_SIZE is the value of bits [3:0] in the INT_STAT_SIZE Device Capability Register. This test uses the value found in the CDF as **CDF_CR_INT_SIZE**.

Execute the following procedure once for each row in Table 6-17. TEST_OFFSET will be assigned a value for each procedure execution.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ as appropriate.
10. Send SET_INT command (0x2_1_60).
11. Send an Offset packet of TEST_OFFSET 0x2_0_##.
12. Send a Data packet of 0x2_0_00, indicating no interrupt.
13. Loop back to the SET_INT until all registers are tried.
14. FAIL if DUT does not respond to each address with an ACK packet (0x2_1_33).
15. If all preceding steps pass, then PASS; else FAIL.

Table 6-17. TEST_OFFSET for Interrupt Registers

| | TEST_OFFSET | Purpose |
|---|--|------------------------|
| 1 | 0x2_0_20 | INT1 register |
| 2 | 0x2_0_21 | INT2 register |
| 3 | 0x2_0_22 | INT3 register |
| 4 | 0x2_0_23 | INT4 register |
| 5 | Additional values if allowed by INT_SIZE | Optional INT registers |

6.3.8.2 CBM: Status Regs; WRITE_STAT (0x60); Valid Registers Respond

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.8.2.1 Test Objective

Verify that DUT responds appropriately when WRITE_STAT is done to valid STAT registers.

6.3.8.2.2 References

[MHL] Section 7.8.3

6.3.8.2.3 Required Test Equipment

MSC Test Tool

6.3.8.2.4 Required Methodology

STAT_SIZE is the value of bits [7:4] in the INT_STAT_SIZE Device Capability Register. This test uses the value found in the CDF as **CDF_CR_STAT_SIZE**.

Execute the following test procedure once for each row in Table 6-18. TEST_OFFSET will be assigned a value for each procedure execution.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ as appropriate.
10. Send WRITE_STAT command (0x2_1_60).
11. Send an Offset packet of TEST_OFFSET 0x2_0_##.
12. Send a Data packet of 0x2_0_00.
13. FAIL if DUT does not respond to each address with an ACK packet (0x2_1_33).
14. If all preceding steps pass, then PASS; else FAIL.

Table 6-18. TEST_OFFSET for Status Registers

| | TEST_OFFSET | Purpose |
|---|---|---------------------------|
| 1 | 0x2_0_30 | STAT1 register |
| 2 | 0x2_0_31 | STAT2 register |
| 3 | 0x2_0_32 | STAT3 register |
| 4 | 0x2_0_33 | STAT4 register |
| 5 | Additional values if allowed by STAT_SIZE | Optional Status registers |

6.3.9 MSC – Source and Sink DUT Input: Vendor-specific and Reserved Header Values

Verify that the DUT responds appropriately if it receives MHL Sideband Channel Commands with Vendor-specific or Reserved header values.

6.3.9.1 CBM: DUT Receives Vendor-specific and Reserved Header Values

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.9.1.1 Test Objective

Check the behavior of the DUT when it received packets on the vendor-specific and reserved sub-channels.

6.3.9.1.2 References

[MHL] Section 7.2.3.2; Table 7-1

6.3.9.1.3 Required Test Equipment

MSC Test Tool

6.3.9.1.4 Required Methodology

Execute the following procedure once for each row in Table 6-19. TEST_COMMAND will take on values defined in the tables.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Send the MHL command TEST_COMMAND.
10. FAIL if DUT responds to the command with an ACK packet (0x2_1_33), a NACK packet (0x2_1_34) or ABORT packet (0x2_1_35).
11. If all preceding steps pass, then PASS; else FAIL.

Table 6-19. DUT Receives Vendor-specific and Reserved Header Values

| | TEST_COMMAND | Purpose |
|---|--------------|--------------------------------|
| 1 | (0x1_1_00) | Vendor-Specific Control Packet |
| 2 | (0x1_0_00) | Vendor-Specific Data Packet |
| 3 | (0x3_1_00) | Reserved Control Packet |
| 4 | (0x3_0_00) | Reserved Data Packet |

6.3.10 MSC – Source and Sink DUT Input: Normal Commands

Verify that the DUT responds appropriately when it receives normal MHL Sideband Channel Commands.

6.3.10.1 CBM: DUT receives (0x62) GET_STATE Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.10.1.1 Test Objective

Verify that DUT responds appropriately when it receives a GET_STATE. It should return the value defined in the MHL Spec as the value stored in the DEV_STATE Capability Register, which is always 0.

6.3.10.1.2 References

[MHL] Section 7.4.3.1

6.3.10.1.3 Required Test Equipment

MSC Test Tool

6.3.10.1.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.

8. Direct Tester to proceed with Discovery.
9. Wait $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. If the Tester Source receives a SET_HPD command, send an ACK packet in response.
11. Execute the Tester_Wait_Capability_Registers_Valid procedure.
12. If the procedure indicates a TIMEOUT, then FAIL.
13. Send a GET_STATE (0x2_1_62) command.
14. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$.
15. FAIL if DUT responds by sending any control packet, specifically a NACK packet or an ABORT packet.
16. FAIL if DUT does not reply with a single data packet with the value 0x00.
17. If all preceding steps pass, then PASS; else FAIL.

6.3.10.2 CBM: DUT receives (0x63) GET_VENDOR_ID Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.10.2.1 Test Objective

Verify that DUT responds appropriately when it receives a GET_VENDOR_ID. This test does not check the return value.

6.3.10.2.2 References

[MHL] Section 7.4.3.2

6.3.10.2.3 Required Test Equipment

MSC Test Tool

6.3.10.2.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. If the Tester Source receives a SET_HPD command, send an ACK packet in response.
11. Send a GET_VENDOR_ID (0x2_1_63) command.
12. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$.
13. FAIL if DUT responds by sending any control packet, specifically a NACK packet or an ABORT packet.
14. FAIL if DUT does not reply with a single data packet.
15. If all preceding steps pass, then PASS; else FAIL.

6.3.10.3 CBM: DUT receives (0x61) READ_DEVCAP Command

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.10.3.1 Test Objective

Verify that DUT responds appropriately when it receives a READ_DEVCAP. This test returns the values found in the CDF in section 2.3 (MHL Capability Registers).

6.3.10.3.2 References

[MHL] Section 7.4.3.7

6.3.10.3.3 Required Test Equipment

MSC Test Tool

6.3.10.3.4 Required Methodology

Execute the following test procedure once for each row in Table 6-20. TEST_WAIT_TIME will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
10. If the Tester Source receives a SET_HPD command, send an ACK packet in response.
11. Send a READ_DEVCAP (0x2_1_61) command.
12. Wait TEST_WAIT_TIME.
13. Send Offset 0 as a Data item (0x2_0_00).
14. FAIL if DUT has problems doing Discovery.
15. FAIL if DUT does not reply with a first item within $T_{\text{PKT_SENDER_TIMEOUT}}$.
16. FAIL if DUT responds by first sending anything besides an ACK packet (0x2_1_33).
17. FAIL if DUT does not reply with a second item within $T_{\text{PKT_SENDER_TIMEOUT}}$.
18. FAIL if DUT does not reply with a single data packet following the ACK packet.
19. If all preceding steps pass, then PASS; else FAIL.

Table 6-20. DUT Receives (0x61) READ_DEVCAP Command

| | TEST_WAIT_TIME | Purpose |
|---|--|----------------------------------|
| 1 | 0 | READ_DEVCAP command; Fast offset |
| 2 | $(0.99) * T_{\text{PKT_RECEIVER_TIMEOUT}}\{\text{min}\}$ | READ_DEVCAP command; Slow offset |

6.3.10.4 CBM: DUT Receives (0x6B) GET_MSC_ERRORCODE Command (When No Error)

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.10.4.1 Test Objective

Verify that DUT responds appropriately when it receives a GET_MSC_ERRORCODE when no error has occurred.

6.3.10.4.2 References

[MHL] Section 7.4.3.6

6.3.10.4.3 Required Test Equipment

MSC Test Tool

6.3.10.4.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. If the Tester Source receives a SET_HPD command, send an ACK packet in response.
11. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
12. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT\{max\}}$.
13. FAIL if DUT responds by sending any control packet, specifically a NACK packet or an ABORT packet.
14. FAIL if DUT does not return a single Data byte in response to the GET_MSC_ERRORCODE command.
15. If all preceding steps pass, then PASS; else FAIL.

6.3.10.5 CBM: DUT Receives (0x6B) GET_MSC_ERRORCODE Command (When Error)

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.10.5.1 Test Objective

Verify that DUT responds appropriately when it receives a GET_MSC_ERRORCODE when no error has occurred.

6.3.10.5.2 References

[MHL] Section 7.4.3.6

6.3.10.5.3 Required Test Equipment

MSC Test Tool

6.3.10.5.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. If the Tester Source receives a SET_HPD command, send an ACK packet in response.
11. Send a READ_DEVCAP (0x2_1_61).
12. Send an invalid Offset byte of 0x10.
13. Receive an ABORT packet.
14. Wait T_{ABORT_NEXT} .
15. Send a GET_MSC_ERRORCODE (0x2_1_6B).
16. Receive Data.
17. Send a second GET_MSC_ERRORCODE (0x2_1_6B).
18. Receive Data.
19. Send a GET_STATE (0x2_1_62).
20. Receive Data.

21. Send a third GET_MSC_ERRORCODE (0x2_1_6B).
22. Receive Data.
23. FAIL if DUT does not reply with the ABORT within $T_{PKT_SENDER_TIMEOUT}\{max\}$.
24. FAIL if DUT does not respond with a first Error Code with the Invalid Offset bit set (0x10).
25. FAIL if DUT does not respond with a second Error Code with the Invalid Offset bit set (0x10).
26. FAIL if DUT does not respond with GET_STATE data.
27. FAIL if DUT does not respond with a third Error Code with the Invalid Offset bit set (0x10).
28. If all preceding steps pass, then PASS; else FAIL.

6.3.10.6CBM: DUT Receives (0x6C) WRITE_BURST Command – Supported

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.10.6.1 Test Objective

Verify that DUT responds appropriately when it receives a WRITE_BURST.

6.3.10.6.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.8.1.4

[MHL] Section 7.8.1.10

6.3.10.6.3 Required Test Equipment

MSC Test Tool

6.3.10.6.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

Execute the following test procedure once for each row in Table 6-21. TEST_WAIT_TIME, TEST_WAIT_TIME_1, TEST_WAIT_TIME_2, and TEST_WAIT_TIME_3 will take on values defined in the table.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is 0, return PASS (SKIP) and terminate the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Wait $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
12. If the Tester Source receives a SET_HPD command, send an ACK packet in response.
13. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
14. Send a WRITE_BURST (0x2_1_6C) command.
15. Wait TEST_WAIT_TIME.
16. Send Offset 0x40 as a Data item (0x2_0_00).
17. Wait TEST_WAIT_TIME_1.
18. Send ADOPTER_ID_H as first Data item (0x2_0_??). (Data from CDF as **CDF_CR_ADOPTER_ID_H**.)
19. Wait TEST_WAIT_TIME_2.

20. Send ADOPTER_ID_L as second Data item (0x2_0_??). (Data from CDF as **CDF_CR_ADOPTER_ID_L**.)
21. Wait TEST_WAIT_TIME_3.
22. Send ((SCRATCHPAD_SIZE-2) or 14, which ever is smaller) more data items. (SCRATCHPAD_SIZE from CDF as **CDF_CR_SCRATCHPAD_SIZE**)
23. Send EOF (0x2_1_32).
24. FAIL if DUT does not reply with an ACK packet within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the EOF is sent.
25. FAIL if DUT responds by first sending anything besides an ACK packet (0x2_1_33).
26. If all preceding steps pass, then PASS; else FAIL.

Table 6-21. DUT Receives (0x60) WRITE_STAT Command

| | TEST_WAIT_TIME | TEST_WAIT_TIME_1 | TEST_WAIT_TIME_2 | TEST_WAIT_TIME_3 |
|----|---|--|--|--|
| 1 | 0 | 0 | 0 | 0 |
| 2 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 | 0 | 0 |
| 3 | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 | 0 |
| 4 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 | 0 |
| 5 | 0 | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 |
| 6 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 |
| 7 | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 |
| 8 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 |
| 9 | 0 | 0 | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ |
| 10 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ |
| 11 | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ |
| 12 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ |
| 13 | 0 | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ |
| 14 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ |
| 15 | 0 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ |
| 16 | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | $T_{MSC_CMD_RECEIVER_TIMEOUT}\{min\} - (3 * T_{PKT_RECEIVER_TIMEOUT}\{min\})$ |

6.3.10.7 CBM: DUT Receives (0x68) MSC_MSG Command – RCP_Support == 1 and RAP_Support == 1

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.10.7.1 Test Objective

Verify that DUT responds appropriately when it receives a MSC_MSG.

This stimulus will result in different DUT behavior, depending on the RAP_SUPPORT and RCP_SUPPORT bits in the FEATURE_FLAG Device Register.

6.3.10.7.2 References

[MHL] Section 7.4.3.11

[MHL] Section 7.6.1

[MHL] Section 7.8.1.10

6.3.10.7.3 Required Test Equipment

MSC Test Tool

6.3.10.7.4 Required Methodology

Execute the following test procedure once for each row in Table 6-22. TEST_WAIT_TIME and TEST_WAIT_TIME_1 will take on values defined in the table.

1. FAIL if DUT is a Source or Sink and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1.
2. FAIL if DUT is a Dongle and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both the same.
3. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are both 0, return PASS (SKIP) and terminate the test.
4. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1, return FAIL and terminate the test.
5. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
6. Set Tester port to disconnected state.
7. If Tester Source port, do not drive the VBUS.
8. If Tester Sink port, drive the VBUS.
9. Select typical timings, resistances, capacitances.
10. Set Tester to use its Discovery engine to enable normal MHL discovery.
11. Set Tester to use its Packet engine to respond normally.
12. Direct Tester to proceed with Discovery.
13. Wait $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
14. If the Tester Source receives a SET_HPD command, send an ACK packet in response.
15. Send a MSC_MSG (0x2_1_68) command.
16. Wait TEST_WAIT_TIME.
17. Send Opcode RAP as a Data item (0x2_0_20).
18. Wait TEST_WAIT_TIME_1.
19. Send Poll as a Data item (0x2_0_00).
20. If DUT replies with no packet before $1.01 * T_{PKT_SENDER_TIMEOUT\{max\}}$, then:
 - a. Tester sends a GET_MSC_ERRORCODE (0x2_1_6B) command.
 - b. If returned error code is not Too Few Packets (0x2_0_04), then FAIL.
21. Else if DUT replies with NACK packet (0x2_1_34), then:
 - a. Tester sends a GET_MSC_ERRORCODE (0x2_1_6B) command.
 - b. If returned error code is not Device Busy (0x2_0_20), then FAIL.

22. Else if DUT replies with ACK packet (0x2_1_33), then PASS.
23. Else FAIL.
24. If all preceding steps pass, then PASS; else FAIL.

Table 6-22. DUT Receives (0x68) MSC_MSG Command – RCP_Support == 1 and RAP_Support == 1

| | TEST_WAIT_TIME | TEST_WAIT_TIME_1 | Purpose |
|---|--|--|---------------------------------|
| 1 | 0 | 0 | MSC_MSG; Fast Offset, Fast Data |
| 2 | (0.99) * T _{PKT_RECEIVER_TIMEOUT} {min} | 0 | MSC_MSG; Fast Offset, Slow Data |
| 3 | 0 | (0.99) * T _{PKT_RECEIVER_TIMEOUT} {min} | MSC_MSG; Slow Offset, Fast Data |
| 4 | (0.99) * T _{PKT_RECEIVER_TIMEOUT} {min} | (0.99) * T _{PKT_RECEIVER_TIMEOUT} {min} | MSC_MSG; Slow Offset, Slow Data |

6.3.10.8CBM: DUT Receives (0x6A) GET_DDC_ERRORCODE Command

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.10.8.1 Test Objective

Verify that a DUT responds appropriately when it receives a GET_DDC_ERRORCODE.

6.3.10.8.2 References

[MHL] Section 7.4.3.5

6.3.10.8.3 Required Test Equipment

MSC Test Tool

6.3.10.8.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait T_{SRC_ARBITRATE} or T_{SINK_ARBITRATE} from discovery completion.
10. If the Tester Source receives a SET_HPD command, send an ACK packet in response.
11. Send a GET_DDC_ERRORCODE (0x2_1_6A) command.
12. FAIL if DUT does not reply within T_{PKT_SENDER_TIMEOUT}{max}.
13. FAIL if DUT responds by sending any control packet, specifically a NACK packet or an ABORT packet.
14. FAIL if DUT does not return a single Data byte in response to the GET_DDC_ERRORCODE command.
15. If all preceding steps pass, then PASS; else FAIL.

6.3.11 MSC – Source and Sink DUT Input: Errors and Exceptions

Verify that the DUT responds predictably to illegal MHL Sideband Channel packet sequences.

6.3.11.1 CBM: DUT Receives Reserved Commands

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.1.1 Test Objective

Verify that DUT can continue when it receives any Reserved commands

6.3.11.1.2 References

[MHL] Section 7.4

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.1.3 Required Test Equipment

MSC Test Tool

6.3.11.1.4 Required Methodology

Execute the following procedure once for each row in Table 6-23. TEST_COMMAND will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
10. Send TEST_COMMAND.
11. Wait $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$.
12. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
13. FAIL if DUT does not respond to the GET_MSC_ERRORCODE by returning either a Data Packet or an ABORT packet.
14. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after receiving GET_MSC_ERRORCODE.
15. If all preceding steps pass, then PASS; else FAIL.

Table 6-23. DUT Receives Reserved Commands

| | TEST_COMMAND | Purpose |
|---|------------------------------|------------------------------------|
| 1 | (0x2_1_69) GET_SC1_ERRORCODE | Reserved Command GET_SC1_ERRORCODE |
| 2 | (0x2_1_6D) GET_SC3_ERRORCODE | Reserved Command GET_SC3_ERRORCODE |

6.3.11.2 CBM: DUT Receives Illegal Commands

Application: Source, Sink, Dongle

6.3.11.2.1 Test Objective

Verify that DUT responds with an ABORT when it receives illegal commands.

6.3.11.2.2 References

[MHL] Section 7.4

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.2.3 Required Test Equipment

MSC Test Tool

6.3.11.2.4 Required Methodology

Table 6-24. Valid Commands when Tester is Acting as Source

| Opcode | Command |
|----------|-------------------|
| 0x2_1_32 | EOF |
| 0x2_1_33 | ACK |
| 0x2_1_34 | NACK |
| 0x2_1_35 | ABORT |
| 0x2_1_60 | SET_INT |
| 0x2_1_60 | WRITE_STAT |
| 0x2_1_61 | READ_DEVCAP |
| 0x2_1_62 | GET_STATE |
| 0x2_1_63 | GET_VENDOR_ID |
| 0x2_1_68 | MSC_MSG |
| 0x2_1_69 | GET_SC1_ERRORCODE |
| 0x2_1_6A | GET_DDC_ERRORCODE |
| 0x2_1_6B | GET_MSC_ERRORCODE |
| 0x2_1_6C | WRITE_BURST |
| 0x2_1_6D | GET_SC3_ERRORCODE |

Table 6-25. Valid Commands when Tester is Acting as Sink

| Opcode | Command |
|----------|-------------------|
| 0x2_1_32 | EOF |
| 0x2_1_33 | ACK |
| 0x2_1_34 | NACK |
| 0x2_1_35 | ABORT |
| 0x2_1_60 | SET_INT |
| 0x2_1_60 | WRITE_STAT |
| 0x2_1_61 | READ_DEVCAP |
| 0x2_1_62 | GET_STATE |
| 0x2_1_63 | GET_VENDOR_ID |
| 0x2_1_64 | SET_HPD |
| 0x2_1_65 | CLR_HPD |
| 0x2_1_68 | MSC_MSG |
| 0x2_1_69 | GET_SC1_ERRORCODE |
| 0x2_1_6A | GET_DDC_ERRORCODE |
| 0x2_1_6B | GET_MSC_ERRORCODE |
| 0x2_1_6C | WRITE_BURST |
| 0x2_1_6D | GET_SC3_ERRORCODE |

Execute the following procedure, and set TEST_ILLEGAL_COMMAND sequentially from 0x2_1_00 to 0x2_1_FF, skipping any value in Table 6-24 if DUT is a Sink or Dongle, or in Table 6-25 if DUT is a Source.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
10. Send TEST_ILLEGAL_COMMAND.
11. Wait for the Abort response.

12. Wait T_{ABORT_NEXT} .
13. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
14. Wait for the error data to return.
15. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the illegal command is sent.
16. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
17. FAIL if DUT does not reply with data within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
18. FAIL if DUT does not return an error code of Invalid Opcode (0x2_0_08).
19. If all preceding steps pass, then PASS; else FAIL.

6.3.11.3 CBM: DUT Receives Data While No Command Outstanding

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.3.1 Test Objective

Verify that DUT emits an ABORT when it receives Data when it is not expected.

6.3.11.3.2 References

[MHL] Section 7.3.1.1

[MHL] Section 7.4.1

[MHL] Section 7.4.3.6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.3.3 Required Test Equipment

MSC Test Tool

6.3.11.3.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. Send an unexpected Data item (0x2_0_0).
11. Wait for the Abort response.
12. Wait T_{ABORT_NEXT} .
13. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
14. Wait for the error data to return.
15. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the Data is sent.
16. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
17. FAIL if DUT does not reply with data within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
18. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02) or Bad opcode (0x2_0_08) in command.
19. If all preceding steps pass, then PASS; else FAIL.

6.3.11.4 CBM: DUT Receives (0x33) ACK Packet While No Command Outstanding

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.4.1 Test Objective

Verify that DUT emits an ABORT when it receives an ACK packet when it is not expected.

6.3.11.4.2 References

[MHL] Section 7.3.1.1

[MHL] Section 7.4.1

[MHL] Section 7.4.3.6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.4.3 Required Test Equipment

MSC Test Tool

6.3.11.4.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. Send an unexpected ACK packet (0x2_1_33).
11. Wait for the Abort response.
12. Wait T_{ABORT_NEXT} .
13. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
14. Wait for the error data to return.
15. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the ACK Packet is sent.
16. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
17. FAIL if DUT does not reply with data within $T_{PKT_SENDER_TIMEOUT}\{max\}$ of the GET_MSC_ERRORCODE is sent.
18. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02) or Bad opcode (0x2_0_08) in command.
19. If all preceding steps pass, then PASS; else FAIL.

6.3.11.5 CBM: DUT Receives (0x34) a NACK Packet While No Command Outstanding

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.5.1 Test Objective

Verify that DUT emits an ABORT when it receives a NACK packet when it is not expected.

6.3.11.5.2 References

[MHL] Section 7.3.1.1

[MHL] Section 7.4.1

[MHL] Section 7.4.3.6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.5.3 Required Test Equipment

MSC Test Tool

6.3.11.5.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
10. Send an unexpected NACK packet (0x2_1_34).
11. Wait for the Abort response.
12. Wait $T_{\text{ABORT_NEXT}}$.
13. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
14. Wait for the error data to return.
15. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the NACK packet is sent.
16. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
17. FAIL if DUT does not reply with data within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the GET_MSC_ERRORCODE is sent.
18. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02) or Bad opcode (0x2_0_08) in command.
19. If all preceding steps pass, then PASS; else FAIL.

6.3.11.6CBM: DUT Receives (0x32) EOF While No Command Outstanding

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.11.6.1 Test Objective

Verify that DUT emits an ABORT when it receives an EOF when it is not expected.

6.3.11.6.2 References

[MHL] Section 7.3.1.1

[MHL] Section 7.4.1

[MHL] Section 7.4.3.6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.6.3 Required Test Equipment

MSC Test Tool

6.3.11.6.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.

4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
10. Send an unexpected EOF (0x2_1_32).
11. Wait for the Abort response.
12. Wait $T_{\text{ABORT_NEXT}}$.
13. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
14. Wait for the error data to return.
15. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the EOF is sent.
16. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
17. FAIL if DUT does not reply with data within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the GET_MSC_ERRORCODE is sent.
18. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02) or Bad opcode (0x2_0_08) in command.
19. If all preceding steps pass, then PASS; else FAIL.

6.3.11.7CBM: DUT Receives (0x35) ABORT While No Command Outstanding

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.11.7.1 Test Objective

Verify that DUT does NOT emit an ABORT when it receives an ABORT.

6.3.11.7.2 References

[MHL] Section 7.3.1.3

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.7.3 Required Test Equipment

MSC Test Tool

6.3.11.7.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
10. Send an unexpected ABORT (0x2_1_35).
11. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
12. Wait for the error data to return.
13. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the GET_MSC_ERRORCODE is sent.
14. If all preceding steps pass, then PASS; else FAIL.

6.3.11.8CBM: DUT Receives (0x61) READ_DEVCAP – Offset Control

| |
|---|
| Application: Source, Sink, Dongle |
|---|

6.3.11.8.1 Test Objective

Verify that DUT does something predictable when it receives a READ_DEVCAP followed by a Control character.

6.3.11.8.2 References

[MHL] Section 7.4.3.7

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.8.3 Required Test Equipment

MSC Test Tool

6.3.11.8.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
10. Send a READ_DEVCAP (0x2_1_61) command.
11. Send Offset as a Control item GET_STATE (0x2_1_62).
12. Wait for the Abort response.
13. Wait $T_{\text{ABORT_NEXT}}$.
14. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
15. Wait for the error data to return.
16. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the bad Offset is sent.
17. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
18. FAIL if DUT does not reply with data within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the GET_MSC_ERRORCODE is sent.
19. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).
20. If all preceding steps pass, then PASS; else FAIL.

6.3.11.9CBM: DUT Receives (0x61) READ_DEVCAP – Offset Invalid

| |
|---|
| Application: Source, Sink, Dongle |
|---|

6.3.11.9.1 Test Objective

Verify that DUT does something predictable when it receives a READ_DEVCAP followed by an invalid Offset value.

6.3.11.9.2 References

[MHL] Section 7.8.1.1

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.9.3 Required Test Equipment

MSC Test Tool

6.3.11.9.4 Required Methodology

Repeatedly execute the following procedure, setting TEST_OFFSET sequentially from (0x2_0_10) through (0x2_0_FF).

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. Send a READ_DEVCAP (0x2_1_61) command.
11. Send TEST_OFFSET as a Data item (0x2_0_??).
12. Wait for the Abort response.
13. Wait T_{ABORT_NEXT} .
14. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
15. Wait for the error data to return.
16. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT\{max\}}$ after the bad Offset is sent.
17. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
18. FAIL if DUT does not reply with data within $T_{PKT_SENDER_TIMEOUT\{max\}}$ after the GET_MSC_ERRORCODE is sent.
19. FAIL if DUT does not return an error code of Bad Offset (0x2_0_10).
20. If all preceding steps pass, then PASS; else FAIL.

6.3.11.10 CBM: DUT Receives (0x60) SET_INT – Offset Control

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.10.1 Test Objective

Verify that DUT does something predictable when it receives a SET_INT followed by a Control character.

6.3.11.10.2 References

[MHL] Section 7.4.3.9

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.10.3 Required Test Equipment

MSC Test Tool

6.3.11.10.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.

3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. Send a SET_INT (0x2_1_60) command.
11. Send Offset as a Control item GET_STATE (0x2_1_62).
12. Wait for the Abort response.
13. Wait T_{ABORT_NEXT} .
14. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
15. Wait for the error data to return.
16. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the bad Offset is sent.
17. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
18. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
19. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).
20. If all preceding steps pass, then PASS; else FAIL.

6.3.11.11 CBM: DUT Receives (0x60) SET_INT – Data Control

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.11.1 Test Objective

Verify that DUT does something predictable when it receives a SET_INT followed by an offset followed by a Control character.

6.3.11.11.2 References

[MHL] Section 7.4.3.9

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.11.3 Required Test Equipment

MSC Test Tool

6.3.11.11.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. Send a SET_INT (0x2_1_60) command.
11. Send Offset 0x20 as a Data Item (0x2_0_20).
12. Send Data as a Control item GET_STATE (0x2_1_62).

13. Wait for the Abort response.
14. Wait T_{ABORT_NEXT} .
15. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
16. Wait for the error data to return.
17. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the bad Data is sent.
18. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
19. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
20. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).
21. If all preceding steps pass, then PASS; else FAIL.

6.3.11.12 CBM: DUT Receives (0x60) WRITE_STAT - Data Control

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.12.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_STAT followed by an Offset followed by a Control character instead of data.

6.3.11.12.2 References

[MHL] Section 7.4.3.8

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.12.3 Required Test Equipment

MSC Test Tool

6.3.11.12.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
10. Send a WRITE_STAT (0x2_1_60) command.
11. Send Offset 0x30 as a Data Item (0x2_0_30).
12. Send Data as a Control item GET_STATE (0x2_1_62).
13. Wait for the Abort response.
14. Wait T_{ABORT_NEXT} .
15. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
16. Wait for the error data to return.
17. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the bad Data is sent.
18. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
19. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
20. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).

21. If all preceding steps pass, then PASS; else FAIL.

6.3.11.13 CBM: DUT Receives (0x6C) WRITE_BURST Command – Not Supported

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.13.1 Test Objective

Verify that DUT responds appropriately when it receives a WRITE_BURST which it does not support.

6.3.11.13.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.8.1.4

[MHL] Section 7.8.1.10

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.13.3 Required Test Equipment

MSC Test Tool

6.3.11.13.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is '1', return PASS (SKIP) and terminate the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
12. Send the following sequence:
 - a. Send a WRITE_BURST (0x2_1_6C) command.
 - b. Then send an offset of 0x40 as a data item (0x2_0_40).
 - c. Then send a data byte of 0x00 as a data item (0x2_0_00).
 - d. Then send an EOF packet (0x2_1_32).
13. FAIL if DUT responds with an ABORT packet (0x2_1_35).
14. If all preceding steps pass, then PASS; else FAIL.

6.3.11.14 CBM: DUT Receives (0x6C) WRITE_BURST – Offset Control

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.11.14.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by a Control character.

6.3.11.14.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.14.3 Required Test Equipment

MSC Test Tool

6.3.11.14.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is '0', return PASS (SKIP) and stop the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
12. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
13. Send a WRITE_BURST (0x2_1_6C) command.
14. Send Offset as a Control item GET_STATE (0x2_1_62).
15. Wait for the Abort response.
16. Wait $T_{\text{ABORT_NEXT}}$.
17. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
18. Wait for the error data to return.
19. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the bad Offset is sent.
20. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
21. FAIL if DUT does not reply with a first item within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the GET_MSC_ERRORCODE is sent.
22. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).
23. If all preceding steps pass, then PASS; else FAIL.

6.3.11.15 CBM: DUT Receives (0x6C) WRITE_BURST – Control Instead of First Data

| |
|--|
| Application: Source, Sink, Dongle |
|--|

6.3.11.15.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by an offset followed by a Control character instead of Data.

6.3.11.15.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.15.3 Required Test Equipment

MSC Test Tool

6.3.11.15.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is '0', return PASS (SKIP) and stop the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Wait the appropriate Tester $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
12. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
13. Send a WRITE_BURST (0x2_1_6C) command.
14. Send Offset 0x40 as a Data Item (0x2_0_40).
15. Send Data 0 as a Control item GET_STATE (0x2_1_62).
16. Wait for the Abort response.
17. Wait $T_{\text{ABORT_NEXT}}$.
18. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
19. Wait for the error data to return.
20. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the bad Data is sent.
21. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
22. FAIL if DUT does not reply with a first item within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the GET_MSC_ERRORCODE is sent.
23. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).
24. If all preceding steps pass, then PASS; else FAIL.

6.3.11.16 CBM: DUT Receives (0x6C) WRITE_BURST – Data 0 EOF

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.16.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by an offset followed by a Control character instead of Data.

6.3.11.16.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.16.3 Required Test Equipment

MSC Test Tool

6.3.11.16.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is '0', return PASS (SKIP) and stop the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
12. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
13. Send a WRITE_BURST (0x2_1_6C) command.
14. Send Offset 0x40 as a Data Item (0x2_0_40).
15. Send Data 0 as a Control item EOF (0x2_1_32).
16. Wait for the Abort response.
17. Wait T_{ABORT_NEXT} .
18. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
19. Wait for the error data to return.
20. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the bad Data is sent.
21. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
22. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
23. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).
24. If all preceding steps pass, then PASS; else FAIL.

6.3.11.17 CBM: DUT Receives (0x6C) WRITE_BURST – Data 1 Control

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.17.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by an offset followed by data followed by a Control character instead of Data.

6.3.11.17.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.17.3 Required Test Equipment

MSC Test Tool

6.3.11.17.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is '0', return PASS (SKIP) and stop the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
12. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
13. Send a WRITE_BURST (0x2_1_6C) command.
14. Send Offset 0x40 as a Data Item (0x2_0_40).
15. Send ADOPTER_ID_H as a Data item (0x2_0_??). (Data from CDF as **CDF_CR_ADOPTER_ID_H**.)
16. Send Data 1 as a Control item GET_STATE (0x2_1_62).
17. Wait for the Abort response.
18. Wait T_{ABORT_NEXT} .
19. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
20. Wait for the error data to return.
21. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT\{max\}}$ after the bad Data is sent.
22. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
23. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT\{max\}}$ after the GET_MSC_ERRORCODE is sent.
24. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).

25. If all preceding steps pass, then PASS; else FAIL.

6.3.11.18 *RESERVED*

6.3.11.19 *CBM: DUT Receives (0x6C) WRITE_BURST – Too Much Data*

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.19.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by more than 16 Data items.

6.3.11.19.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.19.3 Required Test Equipment

MSC Test Tool

6.3.11.19.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is '0', return PASS (SKIP) and stop the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
12. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
13. Send a WRITE_BURST (0x2_1_6C) command.
14. Send Offset 0x40 as a Data Item (0x2_0_40).
15. Send ADOPTER_ID_H as first Data item (0x2_0_??). (Data from CDF as **CDF_CR_ADOPTER_ID_H**.)
16. Send ADOPTER_ID_L as second Data item (0x2_0_??). (Data from CDF as **CDF_CR_ADOPTER_ID_L**.)
17. Send SCRATCHPAD_SIZE-1 (From the CDF as **CDF_CR_SCRATCHPAD_SIZE**) more Data items of (0x2_0_00).
18. Send EOF (0x2_1_32).
19. Wait for the Abort response.
20. Wait T_{ABORT_NEXT} .
21. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.

22. Wait for the error data to return.
23. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the extra Data is sent.
24. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
25. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
26. FAIL if DUT does not return an error code of either Protocol Error (0x2_0_02) or Bad Offset in Command (0x2_0_10).
27. If all preceding steps pass, then PASS; else FAIL.

6.3.11.20 CBM: DUT Receives (0x6C) WRITE_BURST – Offset Wrap ABORT

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.20.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by an invalid Offset value.

6.3.11.20.2 References

[MHL] Section 7.8.4
[MHL] Section 7.4.3.10
[MHL] Section 7.4.3.6; Table 7-6
[MHL] Section 7.8.1; Table 7-21; SCRATCHPAD_SIZE
[MHL] Section 7.4.3
[MHL] Section 7.8; Table 7-20

6.3.11.20.3 Required Test Equipment

MSC Test Tool

6.3.11.20.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is '0', return PASS (SKIP) and stop the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
12. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
13. Send a WRITE_BURST (0x2_1_6C) command.
14. Send (0x2_0_40) + SCRATCHPAD_SIZE – 1 as a Data item. (SCRATCHPAD_SIZE from the CDF as **CDF_CR_SCRATCHPAD_SIZE**).

15. Send ADOPTER_ID_H as first Data item (0x2_0_??). (Data from CDF as **CDF_CR_ADOPTER_ID_H**.)
16. Send ADOPTER_ID_L as second Data item (0x2_0_??). (Data from CDF as **CDF_CR_ADOPTER_ID_L**.)
17. Send EOF (0x2_1_32).
18. Wait for the Abort response.
19. Wait $T_{\text{ABORT_NEXT}}$.
20. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
21. Wait for the error data to return.
22. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the wrapping Data item is sent.
23. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
24. FAIL if DUT does not reply with a first item within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after the GET_MSC_ERRORCODE is sent.
25. FAIL if DUT does not return an error code of Bad Offset (0x2_0_10).
26. If all preceding steps pass, then PASS; else FAIL.

6.3.11.21 *CBM: DUT Receives (0x68) MSC_MSG Command – Both RCP_Support and RAP_Support == 0*

| | |
|---------------------|---------------|
| Application: | Dongle |
|---------------------|---------------|

6.3.11.21.1 Test Objective

Verify that DUT responds appropriately when it receives a MSC_MSG.

6.3.11.21.2 References

[MHL] Section 7.4.3.11
[MHL] Section 7.6.1
[MHL] Section 7.8.1.10
[MHL] Section 7.4.3
[MHL] Section 7.8; Table 7-20

6.3.11.21.3 Required Test Equipment

MSC Test Tool

6.3.11.21.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the RAP_SUPPORT and RCP_SUPPORT bits in the FEATURE_FLAG Device Register. The test is controlled by the CDF values **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT**.

1. FAIL if DUT is a Source or Sink and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1.
2. FAIL if DUT is a Dongle and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both the same.
3. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are both 1, then end test with PASS (SKIP).
4. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 0, return FAIL and stop the test.
5. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
6. Set Tester port to disconnected state.
7. If Tester Source port, do not drive the VBUS.
8. If Tester Sink port, drive the VBUS.

9. Select typical timings, resistances, capacitances.
10. Set Tester to use its Discovery engine to enable normal MHL discovery.
11. Set Tester to use its Packet engine to respond normally.
12. Direct Tester to proceed with Discovery.
13. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
14. Send a MSC_MSG (0x2_1_68) command.
15. If DUT replies with ABORT packet (0x2_1_35), then:
 - a. Tester sends a GET_MSC_ERRORCODE (0x2_1_6B) command.
 - b. If returned error code is not Bad Opcode (0x2_0_08), then FAIL.
16. Else if DUT replies with NACK packet (0x2_1_34), then:
 - c. Tester sends a GET_MSC_ERRORCODE (0x2_1_6B) command.
 - d. If returned error code is not Bad Opcode (0x2_0_08), then FAIL.
17. Else if DUT replies with no packet before $1.01 * T_{PKT_SENDER_TIMEOUT\{max\}}$, then:
 - e. Tester sends a GET_MSC_ERRORCODE (0x2_1_6B) command.
 - f. If returned error code is not Too Few Packets (0x2_0_04), then FAIL.
18. Else PASS.

6.3.11.22 CBM: DUT Receives (0x68) MSC_MSG – Data 0 Control

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.22.1 Test Objective

Verify that DUT does something predictable when it receives an MSC_MSG followed by a Control character.

6.3.11.22.2 References

[MHL] Section 7.4.3.11

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.22.3 Required Test Equipment

MSC Test Tool

6.3.11.22.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the RAP_SUPPORT and RCP_SUPPORT bits in the FEATURE_FLAG Device Register. The test is controlled by the CDF values **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT**.

1. FAIL if DUT is a Source or Sink and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1.
2. FAIL if DUT is a Dongle and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both the same.
3. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are both 0, return PASS (SKIP).
4. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1, return FAIL and stop the test.
5. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
6. Set Tester port to disconnected state.
7. If Tester Source port, do not drive the VBUS.

8. If Tester Sink port, drive the VBUS.
9. Select typical timings, resistances, capacitances.
10. Set Tester to use its Discovery engine to enable normal MHL discovery.
11. Set Tester to use its Packet engine to respond normally.
12. Direct Tester to proceed with Discovery.
13. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
14. Send a MSC_MSG (0x2_1_68) command.
15. Send a Control item GET_STATE (0x2_1_62) as Data 0.
16. Wait for the Abort response.
17. Wait T_{ABORT_NEXT} .
18. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
19. Wait for the error data to return.
20. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the bad Data is sent.
21. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
22. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
23. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).
24. If all preceding steps pass, then PASS; else FAIL.

6.3.11.23 CBM: DUT Receives (0x68) MSC_MSG – Sub-command Illegal

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.23.1 Test Objective

Verify that DUT does something predictable when it receives an MSC_MSG followed by an illegal Sub-command.

6.3.11.23.2 References

[MHL] Section 7.4.3.11

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.23.3 Required Test Equipment

MSC Test Tool

6.3.11.23.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the RAP_SUPPORT and RCP_SUPPORT bits in the FEATURE_FLAG Device Register. The test is controlled by the CDF values **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT**. Repeatedly execute the procedure, setting TEST_SUB_COMMAND sequentially from (0x2_0_00) through (0x2_0_0F). Repeatedly execute the procedure, setting TEST_SUB_COMMAND sequentially from (0x2_0_13) through (0x2_0_1F). Repeatedly execute the procedure, setting TEST_SUB_COMMAND sequentially from (0x2_0_22) through (0x2_0_FF).

1. FAIL if DUT is a Source or Sink and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1.
2. FAIL if DUT is a Dongle and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both the same.
3. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are both 0, return PASS (SKIP).

4. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1, return FAIL and stop the test.
5. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
6. Set Tester port to disconnected state.
7. If Tester Source port, do not drive the VBUS.
8. If Tester Sink port, drive the VBUS.
9. Select typical timings, resistances, capacitances.
10. Set Tester to use its Discovery engine to enable normal MHL discovery.
11. Set Tester to use its Packet engine to respond normally.
12. Direct Tester to proceed with Discovery.
13. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
14. Send a MSC_MSG (0x2_1_68) command.
15. Send TEST_SUB_COMMAND as a data item (0x2_0_??).
16. Send 0x2_0_00 as Data 1.
17. If DUT replies with no packet before $1.01 * T_{PKT_SENDER_TIMEOUT\{max\}}$, then:
 - a. Tester sends a GET_MSC_ERRORCODE (0x2_1_6B) command.
 - b. If returned error code is not Too Few Packets (0x2_0_04), then FAIL.
18. Else if DUT replies with NACK packet (0x2_1_34), then:
 - c. Tester sends a GET_MSC_ERRORCODE (0x2_1_6B) command.
 - d. If returned error code is not Device Busy (0x2_0_20), then FAIL.
19. Else if DUT replies with ACK packet (0x2_1_33), then PASS.
20. Else FAIL.
21. If all preceding steps pass, then PASS; else FAIL.

6.3.11.24 CBM: DUT Receives (0x68) MSC_MSG – Data 1 Control

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.11.24.1 Test Objective

Verify that DUT does something predictable when it receives an MSC_MSG followed by data followed by a Control character.

6.3.11.24.2 References

[MHL] Section 7.4.3.11

[MHL] Section 7.4.3.6; Table 7-6

[MHL] Section 7.4.3

[MHL] Section 7.8; Table 7-20

6.3.11.24.3 Required Test Equipment

MSC Test Tool

6.3.11.24.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the RAP_SUPPORT and RCP_SUPPORT bits in the FEATURE_FLAG Device Register. The test is controlled by the CDF values **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT**.

1. FAIL if DUT is a Source or Sink and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1.

2. FAIL if DUT is a Dongle and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both the same.
3. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are both 0, return PASS (SKIP) and stop the test.
4. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1, return FAIL and stop the test.
5. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
6. Set Tester port to disconnected state.
7. If Tester Source port, do not drive the VBUS.
8. If Tester Sink port, drive the VBUS.
9. Select typical timings, resistances, capacitances.
10. Set Tester to use its Discovery engine to enable normal MHL discovery.
11. Set Tester to use its Packet engine to respond normally.
12. Direct Tester to proceed with Discovery.
13. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
14. Send a MSC_MSG (0x2_1_68) command.
15. Send Opcode RAP as a Data item (0x2_0_20).
16. Send Value as a Control item GET_STATE (0x2_1_62).
17. Wait for the Abort response.
18. Wait T_{ABORT_NEXT} .
19. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
20. Wait for the error data to return.
21. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the bad Data is sent.
22. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
23. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
24. FAIL if DUT does not return an error code of Protocol Error (0x2_0_02).
25. If all preceding steps pass, then PASS; else FAIL.

6.3.12 MSC – Source and Sink DUT Input: Argument Timeouts

Verify that the DUT responds predictably to illegal MHL Sideband Channel packet sequences.

6.3.12.1 CBM: DUT Receives (0x61) READ_DEVCAP – Offset Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.12.1.1 Test Objective

Verify that DUT does something predictable when it receives a READ_DEVCAP followed by a timeout instead of an Offset.

6.3.12.1.2 References

[MHL] Section 7.4.3.7

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.1.3 Required Test Equipment

MSC Test Tool

6.3.12.1.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Send a READ_DEVCAP (0x2_1_61) command.
10. Do NOT send an Offset for 2 seconds.
11. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
12. Wait for either a Data packet or an ABORT packet to return.
13. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
14. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of "Peer Timed Out" (0x2_0_04).
15. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the READ_DEVCAP command being sent.
16. If all preceding steps pass, then PASS; else FAIL.

6.3.12.2 CBM: DUT Receives (0x60) SET_INT / WRITE_STAT - Offset Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.12.2.1 Test Objective

Verify that DUT does something predictable when it receives a SET_INT / WRITE_STAT followed timeout instead of an Offset.

6.3.12.2.2 References

[MHL] Section 7.4.3.9

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.2.3 Required Test Equipment

MSC Test Tool

6.3.12.2.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Send a SET_INT / WRITE_STAT (0x2_1_60) command.
10. Do NOT send an Offset for 2 seconds.
11. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
12. Wait for either a Data packet or an ABORT packet to return.
13. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
14. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of "Peer Timed Out" (0x2_0_04).

15. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the SET_INT command being sent.
16. If all preceding steps pass, then PASS; else FAIL.

6.3.12.3 CBM: DUT Receives (0x60) SET_INT – Data Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.12.3.1 Test Objective

Verify that DUT does something predictable when it receives a SET_INT followed by an offset followed by a timeout waiting for the Data.

6.3.12.3.2 References

[MHL] Section 7.4.3.9

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.3.3 Required Test Equipment

MSC Test Tool

6.3.12.3.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Send a SET_INT (0x2_1_60) command.
10. Send an Offset of 0x20 as Data (0x2_0_20).
11. Do NOT send a Value for 2 seconds.
12. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
13. Wait for either a Data packet or an ABORT packet to return.
14. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
15. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of “Peer Timed Out” (0x2_0_04).
16. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the SET_INT command being sent.
17. If all preceding steps pass, then PASS; else FAIL.

6.3.12.4 CBM: DUT Receives (0x60) WRITE_STAT – Data Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.12.4.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_STAT followed by an Offset followed by a timeout instead of data.

6.3.12.4.2 References

[MHL] Section 7.4.3.8

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.4.3 Required Test Equipment

MSC Test Tool

6.3.12.4.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. If Tester Source port, do not drive the VBUS.
4. If Tester Sink port, drive the VBUS.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Send a WRITE_STAT (0x2_1_60) command.
10. Send an Offset of 0x30 as Data (0x2_0_30).
11. Do NOT send a Value for 2 seconds.
12. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
13. Wait for either a Data packet or an ABORT packet to return.
14. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
15. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of "Peer Timed Out" (0x2_0_04).
16. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the WRITE_STAT command being sent.
17. If all preceding steps pass, then PASS; else FAIL.

6.3.12.5 CBM: DUT Receives (0x6C) WRITE_BURST – Offset Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.12.5.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by timeout instead of an Offset.

6.3.12.5.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.5.3 Required Test Equipment

MSC Test Tool

6.3.12.5.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is 0, return PASS (SKIP) and stop the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.

7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
12. Send a WRITE_BURST (0x2_1_6C) command.
13. Do NOT send an Offset for 2 seconds.
14. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
15. Wait for either a Data Packet or an ABORT packet to return.
16. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
17. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of "PeerTimed Out" (0x2_0_04).
18. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the WRITE_BURST command being sent.
19. If all preceding steps pass, then PASS; else FAIL.

6.3.12.6CBM: DUT Receives (0x6C) WRITE_BURST – Data 0 Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.12.6.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by an offset followed by a timeout instead of Data.

6.3.12.6.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.6.3 Required Test Equipment

MSC Test Tool

6.3.12.6.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is 0, return PASS (SKIP) and stop the test.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
12. Send a WRITE_BURST (0x2_1_6C) command.

13. Send an Offset of 0x40 as Data (0x2_0_40).
14. Do NOT send a Data 0 item for 2 seconds.
15. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
16. Wait for either a Data packet or an ABORT packet to return.
17. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
18. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of "Peer Timed Out" (0x2_0_04).
19. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the WRITE_BURST command being sent.
20. If all preceding steps pass, then PASS; else FAIL.

6.3.12.7 CBM: DUT Receives (0x6C) WRITE_BURST – EOF Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.12.7.1 Test Objective

Verify that DUT does something predictable when it receives a WRITE_BURST followed by an offset, followed by Data, followed by a timeout instead of EOF packet.

6.3.12.7.2 References

[MHL] Section 7.4.3.10

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.7.3 Required Test Equipment

MSC Test Tool

6.3.12.7.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the SP_SUPPORT bit in the FEATURE_FLAG Device Register.

1. If DUT is a Source or a Sink and **CDF_CR_SP_SUPPORT** is 0, then FAIL.
2. If the SP_SUPPORT bit defined in the CDF as **CDF_CR_SP_SUPPORT** is 0, return PASS (SKIP).
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. If Tester Source port, do not drive the VBUS.
6. If Tester Sink port, drive the VBUS.
7. Select typical timings, resistances, capacitances.
8. Set Tester to use its Discovery engine to enable normal MHL discovery.
9. Set Tester to use its Packet engine to respond normally.
10. Direct Tester to proceed with Discovery.
11. Tester arbitrates to execute WRITE_BURST commands by performing REQ_WRT/GRT_WRT procedure.
12. Send a WRITE_BURST (0x2_1_6C) command.
13. Send an Offset of 0x40 as Data (0x2_0_40).
14. Send ADOPTER_ID_H as first Data item (0x2_0_??). (Data from CDF SP_CR_ADOPTER_ID_H.)
15. Do NOT send an EOF for 2 seconds.
16. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
17. Wait for either a Data packet or an ABORT packet to return.
18. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.

19. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of “Peer Timed Out” (0x2_0_04).
20. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the WRITE_BURST command being sent.
21. If all preceding steps pass, then PASS; else FAIL.

6.3.12.8CBM: DUT Receives (0x68) MSC_MSG – Data 0 Timeout

| | |
|---------------------|-----------------------------|
| Application: | Source, Sink, Dongle |
|---------------------|-----------------------------|

6.3.12.8.1 Test Objective

Verify that DUT does something predictable when it receives an MSC_MSG followed by a timeout.

6.3.12.8.2 References

[MHL] Section 7.4.3.11

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.8.3 Required Test Equipment

MSC Test Tool

6.3.12.8.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the RAP_SUPPORT and RCP_SUPPORT bits in the FEATURE_FLAG Device Register. The test is controlled by the CDF values **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT**.

1. FAIL if DUT is a Source or Sink and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1.
2. FAIL if DUT is a Dongle and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both the same.
3. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are both 0, return PASS (SKIP).
4. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1, return FAIL and stop the test.
5. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
6. Set Tester port to disconnected state.
7. If Tester Source port, do not drive the VBUS.
8. If Tester Sink port, drive the VBUS.
9. Select typical timings, resistances, capacitances.
10. Set Tester to use its Discovery engine to enable normal MHL discovery.
11. Set Tester to use its Packet engine to respond normally.
12. Direct Tester to proceed with Discovery.
13. Send a MSC_MSG (0x2_1_68) command.
14. Do NOT send an Opcode for 2 seconds.
15. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
16. Wait for either a Data packet or an ABORT packet to return.
17. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
18. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of “Peer Timed Out” (0x2_0_04).
19. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the MSC_MSG command being sent.
20. If all preceding steps pass, then PASS; else FAIL.

6.3.12.9CBM: DUT Receives (0x68) MSC_MSG – Data 1 Timeout

| |
|---|
| Application: Source, Sink, Dongle |
|---|

6.3.12.9.1 Test Objective

Verify that DUT does something predictable when it receives an MSC_MSG followed by data followed by a timeout.

6.3.12.9.2 References

[MHL] Section 7.4.3.11

[MHL] Section 7.4.3.6; Table 7-6

6.3.12.9.3 Required Test Equipment

MSC Test Tool

6.3.12.9.4 Required Methodology

This stimulus will result in different DUT behavior, depending on the RAP_SUPPORT and RCP_SUPPORT bits in the FEATURE_FLAG Device Register. The test is controlled by the CDF values **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT**.

1. FAIL if DUT is a Source or Sink and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1.
2. FAIL if DUT is a Dongle and **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both the same.
3. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are both 0, return PASS (SKIP).
4. If the RAP_SUPPORT and RCP_SUPPORT bits defined in the CDF as **CDF_CR_RCP_SUPPORT** and **CDF_CR_RAP_SUPPORT** are not both 1, return FAIL and stop the test.
5. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
6. Set Tester port to disconnected state.
7. If Tester Source port, do not drive the VBUS.
8. If Tester Sink port, drive the VBUS.
9. Select typical timings, resistances, capacitances.
10. Set Tester to use its Discovery engine to enable normal MHL discovery.
11. Set Tester to use its Packet engine to respond normally.
12. Direct Tester to proceed with Discovery.
13. Send a MSC_MSG (0x2_1_68) command.
14. Send Opcode RAP as a Data item (0x2_0_20).
15. Do NOT send a Value for 2 seconds.
16. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
17. Wait for either a Data packet or an ABORT packet to return.
18. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the GET_MSC_ERRORCODE is sent.
19. FAIL if DUT does not return either an ABORT packet or a Data packet with an error code of "Peer Timed Out" (0x2_0_04).
20. FAIL if DUT sends any command within $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ of the MSC_MSG command being sent.
21. If all preceding steps pass, then PASS; else FAIL.

6.3.13 MSC – Source DUT Output: Never Initiates Bad Commands

Continuously monitor the CBUS to observe that the DUT sends only valid MHL Sideband Channel Commands.

6.3.13.1 CBM-Source: Source DUT Never Sends (0x64) SET_HPD Command

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.13.1.1 Test Objective

Observe that a Source DUT never sends SET_HPD commands.

6.3.13.1.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}$

[MHL] Section 7.4.1; Table 7-4; Bad Opcode.

[MHL] Section 7.4.3.3

6.3.13.1.3 Required Test Equipment

MSC Test Tool

6.3.13.1.4 Required Methodology

Test only applies when Tester is acting like a Sink.

FAIL if Source DUT sends a SET_HPD (0x2_1_64) command.

If all preceding steps pass, then PASS; else FAIL.

6.3.13.2 CBM-Source: Source DUT Never Sends (0x65) CLR_HPD Command

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.13.2.1 Test Objective

Observe that a Source DUT never sends CLR_HPD commands.

6.3.13.2.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_RECEIVER_TIMEOUT}$

[MHL] Section 13.10.3; Table 13-34; $T_{PKT_SENDER_TIMEOUT}$

[MHL] Section 7.4.1; Table 7-4; Bad Opcode.

[MHL] Section 7.4.3.4

6.3.13.2.3 Required Test Equipment

MSC Test Tool

6.3.13.2.4 Required Methodology

Test only applies when Tester is acting like a Sink.

FAIL if Source DUT sends a CLR_HPD (0x2_1_65) command.

If all preceding steps pass, then PASS; else FAIL.

6.3.14 MSC – Source DUT Input: Normal Commands

Verify that the DUT responds appropriately when it receives normal MHL Sideband Channel Commands.

6.3.14.1 CBM-Source: Source DUT Receives (0x64) SET_HPD Command

| |
|-----------------------------------|
| Application: Source |
|-----------------------------------|

6.3.14.1.1 Test Objective

Verify that Source DUT responds appropriately when it receives a SET_HPD.

6.3.14.1.2 References

[MHL] Section 7.4.3.3

6.3.14.1.3 Required Test Equipment

MSC Test Tool

6.3.14.1.4 Required Methodology

Test only applies when Tester is acting as a Sink.

1. If necessary, connect DUT to Tester Sink port.
2. Set Tester port to disconnected state.
3. Tester Sink drives the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Wait $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
9. Send a SET_HPD (0x2_1_64) command.
10. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT\{max\}}$ after receiving the command.
11. FAIL if DUT responds by first sending anything besides an ACK packet (0x2_1_33).
12. If all preceding steps pass, then PASS; else FAIL.

6.3.14.2 CBM-Source: Source DUT Receives (0x65) CLR_HPD Command

| |
|-----------------------------------|
| Application: Source |
|-----------------------------------|

6.3.14.2.1 Test Objective

Verify that Source DUT responds appropriately when it receives a CLR_HPD.

6.3.14.2.2 References

[MHL] Section 7.4.3.4

[MHL] Section 13.10.3; Table 13-34

6.3.14.2.3 Required Test Equipment

MSC Test Tool

6.3.14.2.4 Required Methodology

Test only applies when Tester is acting as a Sink.

1. If necessary, connect DUT to Tester Sink port.
2. Set Tester port to disconnected state.
3. Tester Sink drives the VBUS.
4. Select typical timings, resistances, capacitances.

5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Wait $T_{\text{SRC_ARBITRATE}}$ or $T_{\text{SINK_ARBITRATE}}$ from discovery completion.
9. Send a CLR_HPD (0x2_1_65) command.
10. FAIL if DUT does not reply within $T_{\text{PKT_SENDER_TIMEOUT}}\{\text{max}\}$ after receiving the command.
11. FAIL if DUT responds by first sending anything besides an ACK Packet (0x2_1_33).
12. If all preceding steps pass, then PASS; else FAIL.

6.3.15 MSC – Sink DUT Output: Normal Commands

Observe that the DUT sends valid MHL Sideband Channel Commands.

Respond with legal results, and observe the DUT responses.

6.3.15.1 CBM-Sink: Sink DUT Sends (0x64) SET_HPD Command

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.15.1.1 Test Objective

Observe that Sink DUT sends valid SET_HPD command, and waits for response before sending another command.

6.3.15.1.2 References

[MHL] Section 7.4.3.7

[MHL] Section 13.10.3; Table 13-34

6.3.15.1.3 Required Test Equipment

MSC Test Tool

6.3.15.1.4 Required Methodology

Test only applies when Tester is acting as a Source.

1. If necessary, connect DUT to Tester Source port.
2. Set Tester port to disconnected state.
3. Tester Source port does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use built-in Discovery RTL to enable normal MHL discovery.
6. Set Tester to use built-in Packet RTL to respond normally.
7. Direct Tester to proceed with Discovery.
8. Watch CBUS activity for 10 seconds, and if tester receives a (0x2_1_64) SET_HPD, proceed with this test.
9. Respond with ACK packet (0x2_1_33).
10. FAIL if DUT has problems doing Discovery.
11. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
12. FAIL if DUT sends an unexpected MSC command or unexpected Data during the time it is waiting for ACK packet from the Tester.
13. FAIL if DUT makes a NACK or ABORT packet in response to the ACK packet (0x2_1_33) Packet returned by the Tester.
14. If all preceding steps pass, then PASS; else FAIL.

6.3.15.2 CBM-Sink: Sink DUT Sends (0x65) CLR_HPD Command

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.15.2.1 Test Objective

Observe that Sink DUT sends valid CLR_HPD command, and waits for response before sending another command.

6.3.15.2.2 References

[MHL] Section 7.4.3.3

6.3.15.2.3 Required Test Equipment

MSC Test Tool

6.3.15.2.4 Required Methodology

Test only applies when Tester is acting as a Source.

1. If necessary, connect DUT to Tester Source port.
2. Set Tester port to disconnected state.
3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use built-in Discovery RTL to enable normal MHL discovery.
6. Set Tester to use built-in Packet RTL to respond normally.
7. Direct Tester to proceed with Discovery.
8. Watch CBUS activity for 10 seconds, and if tester receives a (0x2_1_65) CLR_HPD, proceed with this test.
9. Respond with ACK packet (0x2_1_33).
10. FAIL if DUT has problems doing Discovery.
11. If DUT does not send the command being watched for within 10 seconds, end test with PASS (SKIP).
12. FAIL if DUT sends an unexpected MSC command or unexpected Data during the time it is waiting for ACK packet from the Tester.
13. FAIL if DUT makes a NACK or ABORT packet in response to the ACK packet (0x2_1_33) Packet returned by the Tester.
14. If all preceding steps pass, then PASS; else FAIL.

6.3.16 MSC – Sink DUT Input: Errors and Exceptions

6.3.16.1 CBM-Sink: Sink DUT Receives (0x64) SET_HPD Command

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.16.1.1 Test Objective

Verify that a Sink DUT responds appropriately when it receives a SET_HPD command.

6.3.16.1.2 References

[MHL] Section 7.8; Table 7-20

[MHL] Section 7.4.3.3

[MHL] Section 7.4.3

6.3.16.1.3 Required Test Equipment

MSC Test Tool

6.3.16.1.4 Required Methodology

Test only applies when Tester is acting as a Source.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Sink drives the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
9. Send a SET_HPD (0x2_1_64) command.
10. Wait for the Abort response.
11. Wait T_{ABORT_NEXT} .
12. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
13. Wait for the error data to return.
14. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after receiving the command.
15. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
16. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after after the GET_MSC_ERRORCODE is sent.
17. FAIL if DUT does not return an error code of Invalid Opcode (0x2_0_08).
18. If all preceding steps pass, then PASS; else FAIL.

6.3.16.2 CBM-Sink: Sink DUT Receives (0x65) CLR_HPD Command

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.16.2.1 Test Objective

Verify that a Sink DUT responds appropriately when it receives a CLR_HPD command.

6.3.16.2.2 References

[MHL] Section 7.8; Table 7-20

[MHL] Section 7.4.3.4

[MHL] Section 7.4.3.

6.3.16.2.3 Required Test Equipment

MSC Test Tool

6.3.16.2.4 Required Methodology

Test only applies when Tester is acting as a Source.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Sink drives the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Wait the appropriate Tester $T_{SRC_ARBITRATE}$ or $T_{SINK_ARBITRATE}$ from discovery completion.
9. Send a CLR_HPD (0x2_1_65) command.
10. Wait for the Abort response.

11. Wait T_{ABORT_NEXT} .
12. Send a GET_MSC_ERRORCODE (0x2_1_6B) command.
13. Wait for the error data to return.
14. FAIL if DUT does not reply within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after the command is sent.
15. FAIL if DUT responds by first sending anything besides an ABORT packet (0x2_1_35).
16. FAIL if DUT does not reply with a first item within $T_{PKT_SENDER_TIMEOUT}\{max\}$ after after the GET_MSC_ERRORCODE is sent.
17. FAIL if DUT does not return an error code of Invalid Opcode (0x2_0_08).
18. If all preceding steps pass, then PASS; else FAIL.

6.3.17 DDC – Source DUT Output; DUT Never Sends Illegal DDC Command

6.3.17.1 Common Test Environment

The combined Source DDC state machine including Reads and Writes is as follows:

| | | |
|------|---------|--|
| T0: | Idle | DUT sends SOF -> T1 |
| T1: | SOF1 | DUT sends Address+W -> T2 or Address+R -> T7 or SegAddr 0x60+W -> T2S |
| T2S: | AddrS | Tester replies ACK packet -> T3S; NACK packet -> T91; Timeout -> T92 |
| T3S: | ACK1S | DUT sends Segment Value -> T4S or STOP -> T11 |
| T4S: | SegS | Tester replies ACK packet -> T0S; NACK packet -> T91; Timeout -> T92 |
| T0S: | ACK2S | DUT sends SOF -> T1S or STOP -> T11 |
| T1S: | SOF1S | DUT sends Address+W -> T2 (0xA0 or 0xA1) |
| T2: | AddrW | Tester replies ACK packet -> T3; NACK packet -> T91; Timeout -> T92 |
| T3: | ACK1 | DUT sends Offset -> T4 or STOP -> T11 |
| T4: | Offset | Tester replies ACK packet -> T5; NACK packet -> T91; Timeout -> T92 |
| T5: | ACK2 | DUT sends SOF -> T6 or W_Data -> T45 or STOP -> T11 |
| T6: | SOF2 | DUT sends Address+R -> T7 |
| T7: | AddrR | Tester replies ACK packet -> T8; NACK packet -> T91; Timeout -> T92 |
| T8: | ACK3 | DUT sends CONT -> T9 or STOP -> T11 |
| T9: | Cont | Tester replies R_Data -> T8; Timeout -> T92 |
| T45: | DataW | Tester replies ACK packet -> T46; NACK packet -> T91; Timeout -> T92 |
| T46: | ACKW | DUT sends W_Data -> T45 or STOP -> T11 (or SOF -> T6) |
| T11: | Stop | DUT receives ABORT -> T0 (deprecated); otherwise DUT sends EOF -> T0 |
| T91: | Error | DUT sends EOF -> T0 (Note: Tester may respond with ABORT -> T0 (deprecated)) |
| T92: | Timeout | DUT sends ABORT -> T0 |

NOTE: Terms such as ‘SOF’ and ‘CONT’ are defined in the MHL Specification.

5 named transactions reside within the state machine, and can be identified as follows:

- DDC Short Read/Current Read: Address+R seen in T1
- DDC Read: Address+W seen in T1, SOF seen in T5
- DDC Segment Read: 0x60+W seen in T1, SOF seen in T5
- DDC Write: Address+W seen in T1, W_Data seen in T5 (no SOF seen in T46)

One other possible path through the state machine is:

- DDC Segment Write: 0x60+W seen in T1, W_Data seen in T5
- DDC Large Read: Address+W seen in T1, W_Data seen in T5, SOF seen in T46

6.3.17.2 CBM-Source: DUT Never Sends Illegal DDC Command

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.17.2.1 Test Objective

Continuously watch the CBUS to make sure the Source never sends illegal DDC command codes.

6.3.17.2.2 References

[MHL] Section 7.5.3

6.3.17.2.3 Required Test Equipment

MSC Test Tool

6.3.17.2.4 Required Methodology

FAIL if DUT Source sends DDC Control packets with anything besides Source sendable Op-codes.
If all preceding steps pass, then PASS; else FAIL.

Source sendable DDC Op-codes:

- DDC SOF (0x0_1_30)
- DDC EOF (0x0_1_32)
- DDC ABORT (0x0_1_35)
- DDC CONT (0x0_1_50)
- DDC STOP (0x0_1_51)
- DDC Data (0x0_0_??)

6.3.17.3 CBM-Source: DUT Never Sends Illegal DDC Command Sequence

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.17.3.1 Test Objective

Continuously watch the CBUS to make sure the Source never sends illegal DDC command sequences.
If all preceding steps pass, then PASS; else FAIL.

6.3.17.3.2 References

[MHL] Section 7.5.3

6.3.17.3.3 Required Test Equipment

MSC Test Tool

6.3.17.3.4 Required Methodology

FAIL if DUT does not strictly follow the Source DDC state machine, augmented by transitions forced by the Tester code which is exercising the Source DUT.

6.3.18 DDC – Source DUT Output; Normal Operation

Verify that Source responds predictably if it interacts with a Sink doing normal DDC activity.

6.3.18.1 CBM-Source: DUT Issues DDC Short Read and Current Read

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.18.1.1 Test Objective

Verify that Source DUT adheres to the MHL Standard State Machine for DDC Short Read and Current Read transactions.

6.3.18.1.2 References

[MHL] Section 7.5.3.3

6.3.18.1.3 Required Test Equipment

MSC Test Tool

6.3.18.1.4 Required Methodology

Execute the following procedure once for each row in Table 6-26 TEST_RESPONSE_0 and TEST_RESPONSE_1 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Set Tester to issue SET_HPD.
9. T0 (Idle): snoop the CBUS looking for the DUT Source to send a SOF.
10. Wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
11. If not Address+R, wait for all other test state machines to become Idle, go back to T0.
12. T7: Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_0.
13. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
14. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
15. T8 (ACK): wait for DUT Source to send either a STOP, or a CONT (or time it out, send ABORT, go to T0).
16. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
17. If CONT, send Read_Data or a Timeout, based on TEST_RESPONSE_1.
18. If Read_Data sent, loop to T8.
19. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
20. In all "time it out, send ABORT" cases, Tester remembers DDC Errorcode "Too few bytes in command", 0x04.
21. If test 1, FAIL if DUT does not send STOP, then EOF, after state T8.
22. If test 2, FAIL if DUT does not send STOP, then EOF, after state T8.
23. If test 3, FAIL if DUT does not send EOF after state T2.
24. If test 4, FAIL if DUT does not send ABORT after state T2.
25. If test 5, FAIL if DUT does not send ABORT after state T9.
26. FAIL if Tester issues an ABORT Packet due to a timeout waiting for a packet from the DUT.
27. If no FAIL issued within 10 seconds after completion of discovery, then end test with PASS (SKIP).
28. If DUT does not send a DDC Short Read or Current Read within 10 seconds, end test with PASS (SKIP).

Table 6-26. DUT Issues DDC Short Read and Current Read

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | Purpose |
|---|--|------------------|---|
| 1 | ImmediateDDC ACK (0x0_1_33) | appropriate Data | Short Read and Current Read; Fast ACKs |
| 2 | DDC ACK (0x0_1_33) after (0.99) * T _{PKT_RECEIVER_TIMEOUT} {min} | appropriate Data | Short Read and Current Read; Slow ACKs |
| 3 | DDC NACK (0x0_1_34) | Do not Send | NACK the initial hardware address |
| 4 | Timeout | Do not Send | Timeout the ACK of the initial hardware address |
| 5 | DDC ACK (0x0_1_33) | Timeout | Timeout the Read Data |

6.3.18.2 CBM-Source: DUT Issues Regular DDC Read

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.18.2.1 Test Objective

Verify that Source DUT adheres to the MHL Standard State Machine for DDC Read transactions.

6.3.18.2.2 References

[MHL] Section 7.5.3.1

6.3.18.2.3 Required Test Equipment

MSC Test Tool

6.3.18.2.4 Required Methodology

Execute the following procedure once for each row in Table 6-27. TEST_RESPONSE_0, TEST_RESPONSE_1, TEST_RESPONSE_2, and TEST_RESPONSE_3 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Set Tester to issue SET_HPD.
9. T0 (Idle): snoop the CBUS looking for the DUT Source to send a SOF.
10. In T1, wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
11. If not Address+W, wait for all other test state machines to become Idle, go back to T0.
12. In T2, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_0.
13. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
14. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
15. If ACK packet, in T3, wait for the DUT Source to send an Offset data item.
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
16. In T4, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_1.

17. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
18. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out waiting for it); go to T0.
19. If ACK packet, in T5, wait for the DUT Source to send a SOF or W_Data or STOP (or time it out, send ABORT, go to T0).
20. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
21. If W_Data, in T45, DDC Write. Wait for all other test state machines to become Idle, go back to T0.
22. If SOF, in T6, wait for DUT Source to send an Address+R (or time it out, send ABORT, go to T0).
23. In T7, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_2.
24. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
25. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
26. T8 (ACK): wait for DUT Source to send either a STOP, or a CONT (or time it out, send ABORT, go to T0).
27. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
28. If CONT, in T9, send Read_Data or a Timeout, based on TEST_RESPONSE_3.
29. If Read_Data sent, loop to T8.
30. If Timeout, in T92, wait for DUT Source to send an ABORT (or timeout waiting for it); go to T0.
31. In all "time it out, send ABORT" cases, Tester remembers DDC Errorcode "Too few bytes in command", 0x04.
32. If test 1, FAIL if DUT does not send STOP, then EOF, after state T8.
33. If test 2, FAIL if DUT does not send STOP, then EOF, after state T8.
34. If test 3, FAIL if DUT does not send EOF after state T2.
35. If test 4, FAIL if DUT does not send EOF after state T4.
36. If test 5, FAIL if DUT does not send EOF after state T7.
37. If test 6, FAIL if DUT does not send ABORT after state T2.
38. If test 7, FAIL if DUT does not send ABORT after state T4.
39. If test 8, FAIL if DUT does not send ABORT after state T9.
40. If test 9, FAIL if DUT does not send ABORT after state T9.
41. FAIL if Tester issues an ABORT Packet due to a timeout waiting for a packet from the DUT.
42. If no FAIL is issued within 10 seconds after completion of discovery, then end test with PASS (SKIP).
43. If DUT does not send a regular DDC Read within 10 seconds, end test with PASS (SKIP).

Table 6-27. DUT Issues Regular DDC Read

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | TEST_RESPONSE_2 | TEST_RESPONSE_3 | Purpose |
|---|---|---|---|------------------|------------------------------|
| 1 | Immediate DDC ACK (0x0_1_33) | ImmediateDDC ACK (0x0_1_33) | Immediate DDC ACK (0x0_1_33) | Appropriate Data | Read; fast ACKs |
| 2 | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{\min\}$ | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{\min\}$ | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{\min\}$ | Appropriate Data | Read; slow ACKs |
| 3 | DDC NACK (0x0_1_34) | Do not Send | Do not Send | Do not Send | NACK initial hardware addr |
| 4 | DDC ACK (0x0_1_33) | DDC NACK (0x0_1_34) | Do not Send | Do not Send | NACK initial hardware offset |
| 5 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC NACK (0x0_1_34) | Do not send | NACK second hardware addr |

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | TEST_RESPONSE_2 | TEST_RESPONSE_3 | Purpose |
|---|--------------------|--------------------|--------------------|-----------------|--|
| 6 | Timeout | Do not Send | Do not Send | Do not Send | Timeout ACK of initial hardware addr |
| 7 | DDC ACK (0x0_1_33) | Timeout | Do not Send | Do not Send | Timeout ACK of initial hardware offset |
| 8 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Timeout | Do not Send | Timeout ACK of second hardware addr |
| 9 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Timeout | Timeout the Read Data |

6.3.18.3 CBM-Source: DUT Issues DDC Segment Read

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.18.3.1 Test Objective

Verify that Source DUT adheres to the MHL Standard State Machine for DDC Segment Read transactions.

6.3.18.3.2 References

[MHL] Section 7.5.3.2

6.3.18.3.3 Required Test Equipment

MSC Test Tool

6.3.18.3.4 Required Methodology

Execute the following procedure once for each row in Table 6-28. TEST_RESPONSE_0, TEST_RESPONSE_1, TEST_RESPONSE_2, TEST_RESPONSE_3, TEST_RESPONSE_4, and TEST_RESPONSE_5 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Set Tester to issue SET_HPDP.
9. T0 (Idle): snoop the CBUS looking for the DUT Source to send a SOF.
10. In T1, wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
11. If not 0x60, wait for all other test state machines to become Idle, go back to T0.
12. In T2S, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_0.
13. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
14. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
15. If ACK packet, in T3S, wait for the DUT Source to send a Segment data item
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.

16. In T4S, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_1.
17. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
18. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
19. If ACK packet, in T0S, wait for the DUT Source to send an SOF
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
20. In T1S, wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
21. In T2, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_2.
22. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
23. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
24. If ACK, in T3, wait for the DUT Source to send an Offset data item
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
25. In T4, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_3.
26. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
27. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out waiting for it); go to T0.
28. If ACK packet, in T5, wait for the DUT Source to send a SOF or W_Data or STOP (or time it out, send ABORT, go to T0).
29. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
30. If W_Data, in T4S, DDC Write. Wait for all other test state machines to become Idle, go back to T0.
31. If SOF, in T6, wait for DUT Source to send an Address+R (or time it out, send ABORT, go to T0).
32. In T7, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_4.
33. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
34. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
35. T8 (ACK): wait for DUT Source to send either a STOP, or a CONT (or time it out, send ABORT, go to T0).
36. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
37. If CONT, in T9, send Read_Data or a Timeout, based on TEST_RESPONSE_5.
38. If Read_Data sent, loop to T8.
39. If Timeout, in T92, wait for DUT Source to send an ABORT (or timeout waiting for it); go to T0.
40. In all "time it out, send ABORT" cases, Tester remembers DDC Errorcode "Too few bytes in command", 0x04.
41. If test 1, FAIL if DUT does not send STOP, then EOF, after state T2.
42. If test 2, FAIL if DUT does not send STOP, then EOF, after state T2.
43. If test 3, FAIL if DUT does not send EOF after state T2S.
44. If test 4, FAIL if DUT does not send EOF after state T4S.
45. If test 5, FAIL if DUT does not send EOF after state T2.
46. If test 6, FAIL if DUT does not send EOF after state T4.
47. If test 7, FAIL if DUT does not send EOF after state T7.
48. If test 8, FAIL if DUT does not send ABORT after state T2S.
49. If test 9, FAIL if DUT does not send ABORT after state T4S.
50. If test 10, FAIL if DUT does not send ABORT after state T2.
51. If test 11, FAIL if DUT does not send ABORT after state T4.
52. If test 12, FAIL if DUT does not send ABORT after state T7.
53. If test 13, FAIL if DUT does not send ABORT after state T9.
54. FAIL if Tester issues an ABORT Packet due to a timeout waiting for a packet from the DUT.
55. If no FAIL is issued within 10 seconds of completion of discovery, then end test with PASS (SKIP).

56. If DUT does not send a regular DDC Segment Read within 10 seconds, end test with PASS (SKIP).

Table 6-28. DUT Issues DDC Segment Read

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | TEST_RESPONSE_2 | TEST_RESPONSE_3 | TEST_RESPONSE_4 | TEST_RESPONSE_5 | Purpose |
|----|---|---|---|---|---|------------------|------------------------------|
| 1 | Immediate DDC ACK (0x0_1_33) | Immediate DDC ACK (0x0_1_33) | Immediate DDC ACK (0x0_1_33) | Immediate DDC ACK (0x0_1_33) | Immediate DDC ACK (0x0_1_33) | Appropriate Data | Fast ACK's |
| 2 | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | appropriate Data | Slow ACK's |
| 3 | DDC NACK (0x0_1_34) | Do not Send | Do not Send | Do not Send | Do not Send | Do not Send | NACK Seg Reg |
| 4 | DDC ACK (0x0_1_33) | DDC NACK (0x0_1_34) | Do not Send | Do not Send | Do not Send | Do not Send | NACK Seg Addr |
| 5 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC NACK (0x0_1_34) | Do not Send | Do not Send | Do not Send | Nack 1 st Hardw A |
| 6 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC NACK (0x0_1_34) | Do not Send | Do not Send | Nack 1 st Hardw O |
| 7 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC NACK (0x0_1_34) | Do not Send | Nack 2 nd Hardw A |
| 8 | Timeout | Do not Send | Do not Send | Do not Send | Do not Send | Do not Send | Timeout Seg Reg |
| 9 | DDC ACK (0x0_1_33) | Timeout | Do not Send | Do not Send | Do not Send | Do not Send | Timeout Seg Addr |
| 10 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Timeout | Do not Send | Do not Send | Do not Send | Timeout Hardw A |
| 11 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Timeout | Do not Send | Do not Send | Timeout Hardw O |
| 12 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Timeout | Do not Send | Timeout Hardw A |
| 13 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Timeout | Timeout Data |

6.3.18.4 CBM-Source: DUT Issues DDC Write

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.18.4.1 Test Objective

Verify that Source DUT adheres to the MHL Standard State Machine for DDC Write transactions.

6.3.18.4.2 References

[MHL] Section 7.5.3.4

6.3.18.4.3 Required Test Equipment

MSC Test Tool

6.3.18.4.4 Required Methodology

Execute the following procedure once for each row in Table 6-29. TEST_RESPONSE_0, TEST_RESPONSE_1, and TEST_RESPONSE_2 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Set Tester to issue SET_HPD.
9. T0 (Idle): snoop the CBUS looking for the DUT Source to send a SOF.
10. In T1, wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
11. If not Address+W, wait for all other test state machines to become Idle, go back to T0.
12. In T2, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_0.
13. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
14. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
15. If ACK packet, in T3, wait for the DUT Source to send an Offset data item
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
16. In T4, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_1.
17. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
18. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out waiting for it); go to T0.
19. If ACK packet, in T5, wait for the DUT Source to send a SOF or Write_Data or STOP (or time it out, send ABORT, go to T0).
20. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
21. If SOF, in T6, DDC Read. Wait for all other test state machines to become Idle, go back to T0.
22. T45 (Write Data): Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_2.
23. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
24. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
25. If ACK packet, in T46, wait for DUT Source to send either a STOP or Write_Data or SOF (or time it out, send ABORT, go to T0).
26. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
27. If SOF, DDC Long Read. Wait for all other test state machines to become Idle, go back to T0.
28. If Write_Data, loop to T45.
29. In all "time it out, send ABORT" cases, Tester remembers DDC Errorcode "Too few bytes in command", 0x04.
30. If test 1, FAIL if DUT does not send STOP, then EOF, after state T45.
31. If test 2, FAIL if DUT does not send STOP, then EOF, after state T45.
32. If test 3, FAIL if DUT does not send EOF after state T2.
33. If test 4, FAIL if DUT does not send EOF after state T4.
34. If test 5, FAIL if DUT does not send EOF after state T46.
35. If test 6, FAIL if DUT does not send ABORT after state T2.

36. If test 7, FAIL if DUT does not send ABORT after state T4.
37. If test 8, FAIL if DUT does not send ABORT after state T46.
38. FAIL if Tester issues an ABORT Packet due to a timeout waiting for a packet from the DUT.
39. If no FAIL is issued within 10 seconds of completion of discovery, then end test with PASS (SKIP).
40. If DUT does not issue a DDC Write within 10 seconds, end test with PASS (SKIP).

Table 6-29. DUT Issues DDC Write

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | TEST_RESPONSE_2 | Purpose |
|---|--|--|--|--------------------------------|
| 1 | ImmediateDDC ACK (0x0_1_33) | ImmediateDDC ACK (0x0_1_33) | ImmediateDDC ACK (0x0_1_33) | Read; fast ACKs |
| 2 | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | DDC ACK (0x0_1_33) after (0.99) * $T_{PKT_RECEIVER_TIMEOUT}\{min\}$ | Read; slow ACKs |
| 3 | DDC NACK (0x0_1_34) | Do not Send | Do not Send | NACK initial hardware addr |
| 4 | DDC ACK (0x0_1_33) | DDC NACK (0x0_1_34) | Do not Send | NACK initial hardware offset |
| 5 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC NACK (0x0_1_34) | NACK write data |
| 6 | Timeout | Do not Send | Do not Send | Timeout ACK of hardware addr |
| 7 | DDC ACK (0x0_1_33) | Timeout | Do not Send | Timeout ACK of hardware offset |
| 8 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Timeout | Timeout ACK of Write Data |

6.3.19 DDC – Source DUT Output; Illegal Responses

Make illegal or unexpected responses to the Source DUT when it issues DDC commands.

6.3.19.1 CBM-Source: DUT Issues Short Read and Current

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.19.1.1 Read Test Objective

Verify that Source DUT adheres to the MHL Standard State Machine for DDC Short Read and Current Read transactions.

6.3.19.1.2 References

[MHL] Section 7.5.3.3

6.3.19.1.3 Required Test Equipment

MSC Test Tool

6.3.19.1.4 Required Methodology

Execute the following procedure once for each row in Table 6-30. TEST_RESPONSE_0 and TEST_RESPONSE_1 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.

3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Set Tester to issue SET_HPD.
9. T0 (Idle): snoop the CBUS looking for the DUT Source to send a SOF.
10. Wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
11. If not Address+R, wait for all other test state machines to become Idle, go back to T0.
12. In T7, Tester responds with either a Data packet, an ACK packet, or an ABORT packet, based on TEST_RESPONSE_0.
13. If Data packet, wait for DUT Source to send an ABORT (when DUT source does not send an ABORT in time, which causes time out condition, then send ABORT); go to T0.
14. If ABORT; go to T0.
15. T8 (ACK): wait for DUT Source to send either a STOP, or a CONT (or time it out, send ABORT, go to T0).
16. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
17. If CONT, send ACK packet or ABORT packet, based on TEST_RESPONSE_1.
18. If ACK packet sent, wait for the DUT Source to send an ABORT (or time it out, send ABORT, go to T0).
19. If ABORT packet sent, go to T0.
20. In all "time it out, send ABORT" cases, Tester remembers DDC Errorcode "Too few bytes in command", 0x04.
21. If test 1, FAIL if DUT does not send ABORT after state T2.
22. If test 2, FAIL if DUT does not send ABORT after state T9.
23. If test 3, FAIL if DUT does not silently stop the command after state T2.
24. If test 4, FAIL if DUT does not silently stop the command after state T9.
25. If no FAIL is issued within 10 seconds of completion of discovery, then end test with PASS (SKIP).
26. If DUT does not send a DDC Short Read or Current Read within 10 seconds, end test with PASS (SKIP).

Table 6-30. DUT Issues DDC Short Read and Current Read

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | Purpose |
|---|----------------------|----------------------|---|
| 1 | Data (0x0_0_00) | Do not Send | Data instead of ACK initial hardware addr |
| 2 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | ACK instead of Read Data |
| 3 | DDC ABORT (0x0_1_35) | Do not Send | ABORT instead of ACK initial hardware add |
| 4 | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | ABORT instead of Read Data |

6.3.19.2 CBM-Source: DUT Issues Regular Read

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.19.2.1 Test Objective

Verify that Source DUT adheres to the MHL Standard State Machine for DDC Read transactions.

6.3.19.2.2 References

[MHL] Section 7.5.3.1

6.3.19.2.3 Required Test Equipment

MSC Test Tool

6.3.19.2.4 Required Methodology

Execute the following procedure once for each row in Table 6-31. TEST_RESPONSE_0, TEST_RESPONSE_1, TEST_RESPONSE_2, and TEST_RESPONSE_3 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Set Tester to issue SET_HPD.
9. T0 (Idle): snoop the CBUS looking for the DUT Source to send a SOF.
10. In T1, wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
11. If not Address+W, wait for all other test state machines to become Idle, go back to T0.
12. In T2, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_0.
13. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
14. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
15. If ACK packet, in T3, wait for the DUT Source to send an Offset data item
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
16. In T4, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_1.
17. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
18. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out waiting for it); go to T0.
19. If ACK packet, in T5, wait for the DUT Source to send a SOF or W_Data or STOP (or time it out, send ABORT, go to T0).
20. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
21. If W_Data, in T45, DDC Write. Wait for all other test state machines to become Idle, go back to T0.
22. If SOF, in T6, wait for DUT Source to send an Address+R (or time it out, send ABORT, go to T0).
23. In T7, Tester responds with either an ACK packet, a NACK packet, or a Timeout, based on TEST_RESPONSE_2.
24. If NACK packet, in T91, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
25. If Timeout, in T92, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
26. T8 (ACK): wait for DUT Source to send either a STOP, or a CONT (or time it out, send ABORT, go to T0).
27. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
28. If CONT, in T9, send Read_Data or a Timeout, based on TEST_RESPONSE_3.
29. If Read_Datasent, loop to T8.
30. If Timeout, in T92, wait for DUT Source to send an ABORT (or timeout waiting for it); go to T0.
31. In all "time it out, send ABORT" cases, Tester remembers DDC Errorcode "Too few bytes in command", 0x04.
32. If test 1, FAIL if DUT does not send ABORT after state T2.
33. If test 2, FAIL if DUT does not send ABORT after state T4.
34. If test 3, FAIL if DUT does not send ABORT after state T7.
35. If test 4, FAIL if DUT does not send ABORT after state T9.
36. If test 5, FAIL if DUT does not silently stop the command after state T2.
37. If test 6, FAIL if DUT does not silently stop the command after state T4.
38. If test 7, FAIL if DUT does not silently stop the command after state T7.
39. If test 8, FAIL if DUT does not silently stop the command after state T9.

40. If no FAIL is issued within 10 seconds of completion of discovery, then end test with PASS (SKIP).
41. If DUT does not send a regular DDC Read within 10 seconds, end test with PASS (SKIP).

Table 6-31. DUT Issues Regular DDC Read

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | TEST_RESPONSE_2 | TEST_RESPONSE_3 | Purpose |
|---|----------------------|----------------------|----------------------|----------------------|---|
| 1 | Data (0x0_0_00) | Do not Send | Do not Send | Do not Send | Data not ACK 1 st hardware addr |
| 2 | DDC ACK (0x0_1_33) | Data (0x0_0_00) | Do not Send | Do not Send | Data not ACK the hardware offset |
| 3 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Data (0x0_0_00) | Do not Send | Data not ACK 2 nd hardware address |
| 4 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | ACK instead of Read Data |
| 5 | DDC ABORT (0x0_1_35) | Do not Send | Do not Send | Do not Send | ABORT not ACK 1 st hardware addr |
| 6 | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | Do not Send | Do not Send | ABORT not ACK hardware offset |
| 7 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | Do not Send | ABORT not ACK 2 nd hardware addr |
| 8 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | ABORT instead of read Data |

6.3.19.3 CBM-Source: DUT Issues Segment Read

| | |
|---------------------|---------------|
| Application: | Source |
|---------------------|---------------|

6.3.19.3.1 Test Objective

Verify that Source DUT adheres to the MHL Standard State Machine for DDC Segment Read transactions.

6.3.19.3.2 References

[MHL] Section 7.5.3.2

6.3.19.3.3 Required Test Equipment

MSC Test Tool

6.3.19.3.4 Required Methodology

Execute the following procedure once for each row in Table 6-32. TEST_RESPONSE_0, TEST_RESPONSE_1, TEST_RESPONSE_2, TEST_RESPONSE_3, TEST_RESPONSE_4, and TEST_RESPONSE_5 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Set Tester to issue SET_HPDP.
9. TO (Idle): snoop the CBUS looking for the DUT Source to send a SOF.

10. In T1, wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
11. If not 0x60, wait for all other test state machines to become Idle, go back to T0.
12. In T2, Tester responds with either an ACK packet, an ABORT packet, or a Data packet, based on TEST_RESPONSE_0.
13. If Data packet, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
14. If ABORT, go to T0.
15. If ACK packet, in T3S, wait for the DUT Source to send a Segment data item.
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
16. In T4S, Tester responds with either an ACK packet, an ABORT packet, or a Data packet, based on TEST_RESPONSE_1.
17. If Data packet, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
18. If ABORT, go to T0.
19. If ACK packet, in T0S, wait for the DUT Source to send an SOF.
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
20. In T1S, wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
21. In T2, Tester responds with either an ACK packet, an ABORT packet, or a Data packet, based on TEST_RESPONSE_2.
22. If Data packet, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
23. If ABORT, go to T0.
24. If ACK packet, in T3, wait for the DUT Source to send an Offset data item.
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
25. In T4, Tester responds with either an ACK packet, an ABORT packet, or a Data packet, based on TEST_RESPONSE_3.
26. If Data packet, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
27. If ABORT, go to T0.
28. If ACK packet, in T5, wait for the DUT Source to send a SOF or W_Data or STOP (or time it out, send ABORT, go to T0).
29. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
30. If W_Data, in T45, DDC Write. Wait for all other test state machines to become Idle, go back to T0.
31. If SOF, in T6, wait for DUT Source to send an Address+R (or time it out, send ABORT, go to T0).
32. In T7, Tester responds with either an ACK packet, an ABORT packet, or a Data packet, based on TEST_RESPONSE_4.
33. If Data packet, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
34. If ABORT, go to T0.
35. T8 (ACK): wait for DUT Source to send either a STOP, or a CONT (or time it out, send ABORT, go to T0).
36. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
37. If CONT, in T9, send an ACK packet or an ABORT packet, based on TEST_RESPONSE_5.
38. If ACK packet sent, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
39. If ABORT, go to T0.
40. In all "time it out, send ABORT" cases, Tester remembers DDC Errorcode "Too few bytes in command", 0x04.
41. If test 1, FAIL if DUT does not send ABORT after state T2S.
42. If test 2, FAIL if DUT does not send ABORT after state T4S.
43. If test 3, FAIL if DUT does not send ABORT after state T2.
44. If test 4, FAIL if DUT does not send ABORT after state T4.
45. If test 5, FAIL if DUT does not send ABORT after state T7.

46. If test 6, FAIL if DUT does not send ABORT after state T9.
47. If test 7, FAIL if DUT does not silently stop the command after state T2S.
48. If test 8, FAIL if DUT does not silently stop the command after state T4S.
49. If test 9, FAIL if DUT does not silently stop the command after state T2.
50. If test 10, FAIL if DUT does not silently stop the command after state T4.
51. If test 11, FAIL if DUT does not silently stop the command after state T7.
52. If test 12, FAIL if DUT does not silently stop the command after state T9.
53. If no FAIL is issued within 10 seconds of completion of discovery, then end test with PASS (SKIP).
54. If DUT does not send a DDC Segment Read within 10 seconds, end test with PASS (SKIP).

Table 6-32. DUT Issues DDC Segment Read

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | TEST_RESPONSE_2 | TEST_RESPONSE_3 | TEST_RESPONSE_4 | TEST_RESPONSE_5 | Purpose |
|----|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|---|
| 1 | Data (0x0_0_00) | Do not Send | Do not Send | Do not Send | Do not Send | Do not Send | Data not ACK Seg Reg Addr |
| 2 | DDC ACK (0x0_1_33) | Data (0x0_0_00) | Do not Send | Do not Send | Do not Send | Do not Send | Data not ACK Seg Reg Val |
| 3 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Data (0x0_0_00) | Do not Send | Do not Send | Do not Send | Data not ACK 1 st hardware addr |
| 4 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Data (0x0_0_00) | Do not Send | Do not Send | Data not ACK hardware offset |
| 5 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Data (0x0_0_00) | Do not Send | Data not ACK 2 nd hardware address |
| 6 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | ACK instead of Read Data |
| 7 | DDC ABORT (0x0_1_35) | Do not Send | Do not Send | Do not Send | Do not Send | Do not Send | ABORT not ACK Seg Reg Addr |
| 8 | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | Do not Send | Do not Send | Do not Send | Do not Send | ABORT not ACK Seg Reg Val |
| 9 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | Do not Send | Do not Send | Do not Send | ABORT not ACK 1 st hardware addr |
| 10 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | Do not Send | Do not Send | ABORT not ACK hardware offset |
| 11 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | Do not Send | ABORT not ACK 2 nd hardware addr |
| 12 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | ABORT instead of Read Data |

6.3.19.4 CBM-Source: DUT Issues Write

Application: Source

6.3.19.4.1 Test Objective

Verify that Source DUT adheres to the MHL Standard State Machine for DDC Write transactions.

6.3.19.4.2 References

[MHL] Section 7.5.3.4

6.3.19.4.3 Required Test Equipment

MSC Test Tool

6.3.19.4.4 Required Methodology

Execute the following procedure once for each row in Table 6-33. TEST_RESPONSE_0, TEST_RESPONSE_1, and TEST_RESPONSE_2 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Tester Source does not drive the VBUS.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Set Tester to issue SET_HPD.
9. T0 (Idle): snoop the CBUS looking for the DUT Source to send a SOF.
10. In T1, wait for the DUT Source to send an Address (or time it out, send ABORT, go to T0).
11. If not Address+W, wait for all other test state machines to become Idle, go back to T0.
12. In T2, Tester responds with either an ACK packet, a Data packet, or an ABORT packet, based on TEST_RESPONSE_0.
13. If Data packet, in T91, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
14. If ABORT packet; go to T0.
15. If ACK packet, in T3, wait for the DUT Source to send an Offset data item.
 - a. If Tester receives DDC STOP (0x0_1_51) instead of Offset, then wait for DUT Source to send an EOF (or time out and send ABORT); go to T0.
 - b. If Tester times out (no packet from DUT), then Tester sends DDC ABORT (0x0_1_35), go to T0.
16. In T4, Tester responds with either an ACK packet, a Data packet, or an ABORT packet, based on TEST_RESPONSE_1.
17. If Data packet, in T91, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
18. If ABORT packet; go to T0.
19. If ACK packet, in T5, wait for the DUT Source to send a SOF or Write_Data or STOP (or time it out, send ABORT, go to T0).
20. If STOP, in T11, wait for DUT Source to send an EOF (or time it out, send ABORT); go to T0.
21. If SOF, in T6, DDC Read. Wait for all other test state machines to become Idle, go back to T0.
22. T45 (Write Data): Tester responds with either an ACK packet, a Data packet, or an ABORT packet, based on TEST_RESPONSE_2.
23. If Data packet, in T91, wait for DUT Source to send an ABORT (or time it out, send ABORT); go to T0.
24. If ABORT packet; go to T0.
25. In all "time it out, send ABORT" cases, Tester remembers DDC Errorcode "Too few bytes in command", 0x04.
26. If test 1, FAIL if DUT does not send ABORT after state T2.
27. If test 2, FAIL if DUT does not send ABORT after state T4.
28. If test 3, FAIL if DUT does not send ABORT after state T46.
29. If test 4, FAIL if DUT does not silently stop the command after state T2.
30. If test 5, FAIL if DUT does not silently stop the command after state T4.
31. If test 6, FAIL if DUT does not silently stop the command after state T46.
32. If no FAIL is issued within 10 seconds of completion of discovery, then end test with PASS (SKIP).
33. If DUT does not send a DDC Write within 10 seconds, end test with PASS (SKIP).

Table 6-33. DUT Issues DDC Write

| | TEST_RESPONSE_0 | TEST_RESPONSE_1 | TEST_RESPONSE_2 | Purpose |
|---|----------------------|----------------------|----------------------|---|
| 1 | Data (0x0_0_00) | Do not Send | Do not Send | Data not ACK 1 st hardware addr |
| 2 | DDC ACK (0x0_1_33) | Data (0x0_0_00) | Do not Send | Data not ACK hardware offset |
| 3 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | Data (0x0_0_00) | Data instead of ACK the Write Data |
| 4 | DDC ABORT (0x0_1_35) | Do not Send | Do not Send | ABORT not ACK 1 st hardware addr |
| 5 | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | Do not Send | ABORT not ACK hardware offset |
| 6 | DDC ACK (0x0_1_33) | DDC ACK (0x0_1_33) | DDC ABORT (0x0_1_35) | ABORT instead of ACK the Write Data |

6.3.20 DDC – Sink DUT Input; Continuous Monitors and Normal Operation

Continuously monitor the CBUS to observe that the Sink DUT sends only valid DDC Commands.

6.3.20.1 Common Test Environment

The combined Sink DDC state machine including Reads and Writes is as follows:

| | | |
|------|---------|--|
| R0: | Idle | Tester sends SOF -> R1 |
| R1: | SOF1 | Tester sends Address+W -> R2 or Address+R -> R7 or SegAddr 0x60+W -> R2S |
| R2S: | AddrS | DUT replies ACK packet -> R3S; NACK packet -> R91; Timeout -> R92 |
| R3S: | ACK1S | Tester sends Segment Value -> R4S or STOP -> R11 |
| R4S: | SegS | DUT replies ACK packet -> R0S; NACK packet -> R91; Timeout -> R92 |
| R0S: | ACK2S | Tester sends SOF -> R1S or STOP -> R11 |
| R1S: | SOF1S | Tester sends Address+W -> R2 (0xA0 or 0xA1) |
| R2: | AddrW | DUT replies ACK packet -> R3; NACK packet -> R91; Timeout -> R92 |
| R3: | ACK1 | Tester sends Offset -> R4 or STOP -> R11 |
| R4: | Offset | DUT replies ACK packet -> R5; NACK packet -> R91; Timeout -> R92 |
| R5: | ACK2 | Tester sends SOF -> R6 or W_Data -> R45 or STOP -> R11 |
| R6: | SOF2 | Tester sends Address+R -> R7 |
| R7: | AddrR | DUT replies ACK packet -> R8; NACK packet -> R91; Timeout -> R92 |
| R8: | ACK3 | Tester sends CONT -> R9 or STOP -> R11 |
| R9: | Cont | DUT replies R_Data -> R8; Timeout -> R92 |
| R45: | DataW | DUT replies ACK packet -> R46; NACK packet -> R91; Timeout -> R92 |
| R46: | ACKW | Tester sends W_Data -> R45 or STOP -> R11 (or SOF -> R6) |
| R11: | Stop | DUT replies ABORT -> R0 (deprecated), or Tester sends EOF -> R0 |
| R91: | Error | Tester sends EOF -> R0, or DUT responds to EOF by sending ABORT -> R0 (deprecated) |
| R92: | Timeout | Tester sends ABORT -> R0 |

NOTE: Terms such as ‘SOF’ and ‘CONT’ are defined in the MHL Specification.

The 5 named transactions reside within the state machine, and can be identified as follows:

- DDC Short Read/Current Read: Address+R seen in R1

- DDC Read: Address+W seen in R1, SOF seen in R5
- DDC Segment Read: 0x60+W seen in R1, SOF seen in R5
- DDC Write: Address+W seen in R1, W_Data seen in R5 (no SOF seen in R46)

One other possible path through the state machine is:

- DDC Segment Write: 0x60+W seen in R1, W_Data seen in R5

(There might also be a case which allows the Offset to be 2 bytes to address bigger EEPROMs.)

- DDC Large Read: Address+W seen in R1, W_Data seen in R5, SOF seen in R46

6.3.20.2 CBM-Sink: DUT Never Sends Illegal DDC Command

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.20.2.1 Test Objective

Continuously watch the CBUS to make sure the Sink never sends illegal DDC command codes.

6.3.20.2.2 References

[MHL] Section 7.5.3

6.3.20.2.3 Required Test Equipment

MSC Test Tool

6.3.20.2.4 Required Methodology

Sink sendable DDC Op-codes:

- DDC ACK (0x0_1_33)
- DDC NACK (0x0_1_34)
- DDC ABORT (0x0_1_35)
- DDC Data (0x0_0_??)

FAIL if DUT Sink sends DDC Control packets with anything besides Sink sendable Op-codes.

If all preceding steps pass, then PASS; else FAIL.

6.3.20.3 CBM-Sink: DUT Never Sends Illegal DDC Command Sequence

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.20.3.1 Test Objective

Continuously watch the CBUS to make sure the Sink never sends illegal DDC command sequences.

If all preceding steps pass, then PASS; else FAIL.

6.3.20.3.2 References

[MHL] Section 7.5.3

6.3.20.3.3 Required Test Equipment

MSC Test Tool

6.3.20.3.4 Required Methodology

FAIL if DUT does not strictly follow the Sink DDC state machine, augmented by transitions forced by the Tester code which is exercising the Sink DUT.

If all preceding steps pass, then PASS; else FAIL.

6.3.21 DDC – Sink DUT Input; Normal Operation

Verify that Sink responds predictably if it interacts with a Source doing normal DDC activity.

6.3.21.1 CBM-Sink: DUT Receives DDC Write

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.21.1.1 Test Objective

Verify that Sink DUT adheres to the DDC Standard State Machine for DDC Write transactions.

6.3.21.1.2 References

[MHL] Section 7.5.3.4

6.3.21.1.3 Required Test Equipment

MSC Test Tool

6.3.21.1.4 Required Methodology

Execute the following procedure once for each row in Table 6-34. TEST_ADDRESS, TEST_DATA_0 and TEST_DATA_1 will take on values defined in the table.

1. If CDF entry **CDF_HDCP_SUPPORT** is not YES, then return PASS (SKIP), and terminate the test.
2. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
3. Set Tester port to disconnected state.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Wait until Tester receives SET_HPD command.
9. FAIL if DUT does not eventually send SET_HPD.
10. Tester sends DDC SOF (0x0_1_30).
11. Tester sends TEST_ADDRESS.
12. Tester receives and checks DDC ACK packet (0x0_1_33).
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
13. Tester sends Offset offset 16 (0x0_0_10) HDCP Aksv.
14. Tester receives and checks DDC ACK packet (0x0_1_33).
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
15. If TEST_DATA_0 indicated:
 - a. Tester sends TEST_DATA_0.
 - b. Tester receives and checks DDC ACK packet (0x0_1_33).

- c. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
- d. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
- e. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
16. If TEST_DATA_1 indicated:
 - a. Tester sends TEST_DATA_1.
 - b. Tester receives and checks DDC ACK packet (0x0_1_33).
 - c. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - d. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - e. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
17. Tester sends DDC STOP (0x0_1_51).
18. If 0 data bytes are being sent and the Tester receives an ABORT packet, do not signal FAIL. Go to End.
19. Tester sends DDC EOF (0x0_1_32). Go to END.
20. **NACK_TERM:** DUT might respond to an EOF sent after a NACK packet with an ABORT packet. This deprecated packet must be tolerated, and does not result in a test FAIL.
21. **END:** Do POST_DDC_TEST_READ_PROCEDURE:
 - a. Tester sends DDC SOF (0x0_1_30).
 - b. Tester sends EDID ADDRESS (0x0_0_A1).
 - c. Tester receives and checks DDC ACK packet (0x0_1_33).
 - d. Tester sends DDC CONT (0x0_1_50).
 - e. Tester receives and remembers DDC Data (0x0_0_??).
 - f. Tester sends DDC STOP (0x0_1_51).
 - g. Tester sends DDC EOF (0x0_1_32).
22. FAIL if DUT has problems doing Discovery.
23. If not test 4, FAIL if DUT causes any early transition to END.
24. If not test 4, FAIL if DUT sends any DDC Data after it receives the EOF.
25. If test 4, FAIL if DUT does not respond with a NACK packet after the Address is sent.
26. FAIL if POST_DDC_TEST_READ_PROCEDURE does not succeed.
27. If all preceding steps pass, then PASS; else FAIL.

Table 6-34. DUT Receives DDC Write

| | TEST_ADDRESS | TEST_DATA_0 | TEST_DATA_1 | Purpose |
|---|------------------------------------|---------------------|---------------------|-------------------------------------|
| 1 | HDCP ADDRESS (0x0_0_74) (Write) | Do not Send | Do not Send | DDC Write; 0 data items |
| 2 | HDCP ADDRESS (0x0_0_74) (Write) | DDC Data (0x0_0_55) | Do not Send | DDC Write; 1 data items |
| 3 | HDCP ADDRESS (0x0_0_74) (Write) | DDC Data (0x0_0_CC) | DDC Data (0x0_0_55) | DDC Write; 2 data items |
| 4 | Invalid ADDRESS (0x0_0_00) (Write) | Do not Send | Do not Send | DDC Write; Invalid IIC Address 0x00 |

6.3.21.2 CBM-Sink: DUT Receives DDC Current Read

Application: Sink, Dongle

6.3.21.2.1 Test Objective

Verify that Sink DUT adheres to the DDC Standard State Machine for DDC Current Read transactions.

6.3.21.2.2 References

[MHL] Section 7.5.3.3

6.3.21.2.3 Required Test Equipment

MSC Test Tool

6.3.21.2.4 Required Methodology

Execute the following procedure once for each row in Table 6-35. TEST_ADDRESS, TEST_DATA_0 and TEST_DATA_1 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Select typical timings, resistances, capacitances.
4. Set Tester to use its Discovery engine to enable normal MHL discovery.
5. Set Tester to use its Packet engine to respond normally.
6. Direct Tester to proceed with Discovery.
7. Wait until Tester receives SET_HPD command.
8. FAIL if DUT does not eventually send SET_HPD.
9. Tester sends DDC SOF (0x0_1_30).
10. Tester sends TEST_ADDRESS.
11. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
12. If TEST_DATA_0 indicated:
 - a. Tester sends CONT (0x0_1_50).
 - b. Tester receives and remembers DDC DATA (0x0_0_??).
 - c. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - d. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
13. If TEST_DATA_1 indicated:
 - a. Tester sends CONT (0x0_1_50).
 - b. Tester receives and remembers DDC DATA (0x0_0_??).
 - c. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - d. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
14. Tester sends DDC STOP (0x0_1_51).
15. Tester sends DDC EOF (0x0_1_32). Go to END.
16. **NACK_TERM:** DUT might respond to an EOF sent after a NACK packet with an ABORT packet. This deprecated packet must be tolerated, and does not result in a test FAIL.
17. **END:** Do POST_DDC_TEST_READ_PROCEDURE:
 - a. Tester sends DDC SOF (0x0_1_30)
 - b. Tester sends EDID ADDRESS (0x0_0_A1)
 - c. Tester receives and checks DDC ACK packet (0x0_1_33)
 - d. Tester sends DDC CONT (0x0_1_50)
 - e. Tester receives and remembers DDC Data (0x0_0_??)
 - f. Tester sends DDC STOP (0x0_1_51)
 - g. Tester sends DDC EOF (0x0_1_32)
18. FAIL if DUT has problems doing Discovery.
19. If not test 4, FAIL if DUT causes any early transition to END.
20. If test 4, FAIL if DUT does not respond with a NACK packet after the Address is sent.
21. FAIL if DUT sends any DDC Data after it receives the EOF.
22. FAIL if POST_DDC_TEST_READ_PROCEDURE does not succeed.
23. If all preceding steps pass, then PASS; else FAIL.

Table 6-35. DUT Receives DDC Current Read

| | TEST_ADDRESS | TEST_DATA_0 | TEST_DATA_1 | Purpose |
|---|-----------------------------------|---------------------|---------------------|--|
| 1 | EDID ADDRESS (0x0_0_A1) (Read) | Receive Data | Do not Receive Data | DDC Current Read 1 data items |
| 2 | EDID ADDRESS (0x0_0_A1) (Read) | Receive Data | Receive Data | DDC Current Read 2 data items |
| 3 | Invalid ADDRESS (0x0_0_01) (Read) | Do not Receive Data | Do not Receive Data | DDC Current Read Invalid IIC Address 0x00 |

6.3.21.3 CBM-Sink: DUT Receives regular DDC Read

Application: Sink, Dongle

6.3.21.3.1 Test Objective

Verify that Sink DUT adheres to the MHL Standard State Machine for regular DDC Read transactions.

6.3.21.3.2 References

[MHL] Section 7.5.3.1

6.3.21.3.3 Required Test Equipment

MSC Test Tool

6.3.21.3.4 Required Methodology

Execute the following procedure once for each row in Table 6-36. TEST_ADDRESS_2, TEST_DATA_0 and TEST_DATA_1 will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Select typical timings, resistances, capacitances.
4. Set Tester to use its Discovery engine to enable normal MHL discovery.
5. Set Tester to use its Packet engine to respond normally.
6. Direct Tester to proceed with Discovery.
7. Wait until Tester receives SET_HPD command.
8. FAIL if DUT does not eventually send SET_HPD.
9. Tester sends DDC SOF (0x0_1_30):
10. Tester sends EDID ADDRESS (0x0_0_A0) (Write).
11. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go END.
12. Tester sends Offset 0 (0x0_0_00).
13. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go END.
14. Tester sends DDC SOF (0x0_1_30).

15. Tester sends TEST_ADDRESS_2.
16. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
17. If TEST_DATA_0 indicated:
 - a. Tester sends CONT (0x0_1_50).
 - b. Tester receives and remembers DDC DATA (0x0_0_??).
 - c. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - d. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
18. If TEST_DATA_1 indicated:
 - a. Tester sends CONT (0x0_1_50).
 - b. Tester receives and remembers DDC DATA (0x0_0_??).
 - c. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - d. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
19. Tester sends DDC STOP (0x0_1_51).
20. Tester sends DDC EOF (0x0_1_32). Go to END.
21. **NACK_TERM:** DUT might respond to an EOF sent after a NACK packet with an ABORT packet. This deprecated packet must be tolerated, and does not result in a test FAIL.
22. **END:** Do POST_DDC_TEST_READ_PROCEDURE:
 - a. Tester sends DDC SOF (0x0_1_30).
 - b. Tester sends EDID ADDRESS (0x0_0_A1).
 - c. Tester receives and checks DDC ACK packet (0x0_1_33).
 - d. Tester sends DDC CONT (0x0_1_50).
 - e. Tester receives and remembers DDC Data (0x0_0_??).
 - f. Tester sends DDC STOP (0x0_1_51).
 - g. Tester sends DDC EOF (0x0_1_32).
23. FAIL if DUT has problems doing Discovery.
24. If not test 4, FAIL if DUT causes any early transition to END.
25. If test 4, FAIL if DUT does not respond with a NACK packet after the second Address is sent.
26. FAIL if DUT sends any DDC Data after it receives the EOF.
27. FAIL if POST_DDC_TEST_READ_PROCEDURE does not succeed.
28. If all preceding steps pass, then PASS; else FAIL.

Table 6-36. DUT Receives Regular DDC Read

| | TEST_ADDRESS_2 | TEST_DATA_0 | TEST_DATA_1 | Purpose |
|---|-----------------------------------|---------------------|---------------------|--|
| 1 | EDID ADDRESS (0x0_0_A1) (Read) | Receive Data | Do not Receive Data | Regular DDC Read 1 data items |
| 2 | EDID ADDRESS (0x0_0_A1) (Read) | Receive Data | Receive Data | Regular DDC Read 2 data items |
| 3 | Invalid ADDRESS (0x0_0_01) (Read) | Do not Receive Data | Do not Receive Data | Regular DDC Read Invalid IIC Address 0x00 |

6.3.21.4 CBM-Sink: DUT Receives DDC Segment Read; Segment Register Implemented

Application: Sink, Dongle

6.3.21.4.1 Test Objective

Verify that Sink DUT adheres to the MHL Standard State Machine for DDC Segment Read transactions.

6.3.21.4.2 References

[MHL] Section 7.5.3.2

6.3.21.4.3 Required Test Equipment

MSC Test Tool

6.3.21.4.4 Required Methodology

Execute the following procedure once for each row in Table 6-37. TEST_ADDRESS_1, TEST_ADDRESS_2, TEST_DATA_0 and TEST_DATA_1 will take on values defined in the table.

1. If the CDF variable **CDF_DDC_SEGMENT_READ_SUPPORT** indicates that the DUT cannot do Segment Reads, then end the test with PASS (SKIP).
2. If the DUT is a Dongle, connect it to test equipment which supports multi-segment EDID reads.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait until Tester receives SET_HPD command.
10. FAIL if DUT does not eventually send SET_HPD.
11. Tester sends DDC SOF (0x0_1_30).
12. Tester sends SEGMENT_REG_ADDR (0x0_0_60) (Write).
13. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go END.
14. Tester sends SEGMENT_REG_DATA (0x0_0_00).
15. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go END.
16. Tester sends DDC SOF (0x0_1_30).
17. Tester sends TEST_ADDRESS_1.
18. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go END.
19. Tester sends Offset 0 (0x0_0_00).
20. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go END.
21. Tester sends DDC SOF (0x0_1_30).
22. Tester sends TEST_ADDRESS_2.
23. Tester receives and checks DDC ACK packet (0x0_1_33):

- a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
24. If TEST_DATA_0 indicated:
 - a. Tester sends CONT (0x0_1_50).
 - b. Tester receives and remembers DDC DATA (0x0_0_??).
 - c. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - d. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
25. If TEST_DATA_1 indicated:
 - a. Tester sends CONT (0x0_1_50).
 - b. Tester receives and remembers DDC DATA (0x0_0_??).
 - c. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - d. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
26. Tester sends DDC STOP (0x0_1_51).
27. Tester sends DDC EOF (0x0_1_32). Go to END.
28. **NACK_TERM:** DUT might respond to an EOF sent after a NACK packet with an ABORT packet. This deprecated packet must be tolerated, and does not result in a test FAIL.
29. **END:** Do POST_DDC_TEST_READ_PROCEDURE:
 - a. Tester sends DDC SOF (0x0_1_30).
 - b. Tester sends EDID ADDRESS (0x0_0_A1).
 - c. Tester receives and checks DDC ACK packet (0x0_1_33).
 - d. Tester sends DDC CONT (0x0_1_50).
 - e. Tester receives and remembers DDC Data (0x0_0_??).
 - f. Tester sends DDC STOP (0x0_1_51).
 - g. Tester sends DDC EOF (0x0_1_32).
30. FAIL if DUT has problems doing Discovery.
31. If not test 4 or 5, FAIL if DUT causes any early transition to END.
32. If test 4, FAIL if DUT does not respond with a NACK packet after the first Address is sent.
33. If test 5, FAIL if DUT does not respond with a NACK packet after the second Address is sent.
34. FAIL if DUT sends any DDC Data after it receives the EOF.
35. FAIL if POST_DDC_TEST_READ_PROCEDURE does not succeed.
36. If all preceding steps pass, then PASS; else FAIL.

Table 6-37. DUT Receives DDC Segment Read

| | TEST_ADDRESS_1 | TEST_ADDRESS_2 | TEST_DATA_0 | TEST_DATA_1 | Purpose |
|---|------------------------------------|-----------------------------------|---------------------|---------------------|--|
| 1 | EDID ADDRESS (0x0_0_A0) (Write) | EDID ADDRESS (0x0_0_A1) (Read) | Receive Data | Do not Receive Data | Segment Read 1 data items |
| 2 | EDID ADDRESS (0x0_0_A0) (Write) | EDID ADDRESS (0x0_0_A1) (Read) | Receive Data | Receive Data | Segment Read 2 data items |
| 3 | Invalid ADDRESS (0x0_0_00) (Write) | EDID ADDRESS (0x0_0_A1) (Read) | Do not Receive Data | Do not Receive Data | Segment Read; Invalid first IIC Address 0x00 |
| 4 | EDID ADDRESS (0x0_0_A0) (Write) | Invalid ADDRESS (0x0_0_01) (Read) | Do not Receive Data | Do not Receive Data | Segment Read; Invalid first IIC Address 0x00 |

6.3.21.5 CBM-Sink: DUT NACKs DDC Segment Read; Segment Register Not Implemented

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.21.5.1 Test Objective

Verify that Sink DUT adheres to the MHL Standard State Machine for DDC Segment Read transactions.

6.3.21.5.2 References

[MHL] Section 7.5.3.2

6.3.21.5.3 Required Test Equipment

MSC Test Tool

6.3.21.5.4 Required Methodology

1. If the CDF variable **CDF_DDC_SEGMENT_READ_SUPPORT** indicates that the DUT can do Segment Reads and the DUT is a Sink, then end the test with PASS (SKIP).
2. If the DUT is a Dongle, then connect it to test equipment which does not support multi-segment EDID reads.
3. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
4. Set Tester port to disconnected state.
5. Select typical timings, resistances, capacitances.
6. Set Tester to use its Discovery engine to enable normal MHL discovery.
7. Set Tester to use its Packet engine to respond normally.
8. Direct Tester to proceed with Discovery.
9. Wait until Tester receives SET_HPD command.
10. FAIL if DUT does not eventually send SET_HPD.
11. Tester sends DDC SOF (0x0_1_30).
12. Tester sends SEGMENT_REG_ADDR (0x0_0_60) (Write).
13. Tester receives and checks DDC NACK packet (0x0_1_34):
 - a. If Tester receives DDC ACK packet (0x0_1_33) instead:
 - i. Tester sends DDC DATA (0x0_0_00).
 - ii. Tester receives and checks DDC ACK packet (0x0_1_33).
 - iii. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to NACK_TERM.
 - iv. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - v. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go END.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead:
 - i. send DDC ABORT (0x0_1_35) and go END.
14. Tester sends DDC EOF (0x0_1_32). Go to END.
15. **NACK_TERM:** DUT might respond to an EOF sent after a NACK packet with an ABORT packet. This deprecated packet must be tolerated, and does not result in a test FAIL.
16. **END:** Do POST_DDC_TEST_READ_PROCEDURE:
 - a. Tester sends DDC SOF (0x0_1_30).
 - b. Tester sends EDID ADDRESS (0x0_0_A1).
 - c. Tester receives and checks DDC ACK packet (0x0_1_33).
 - d. Tester sends DDC CONT (0x0_1_50).
 - e. Tester receives and remembers DDC Data (0x0_0_??).
 - f. Tester sends DDC STOP (0x0_1_51).
 - g. Tester sends DDC EOF (0x0_1_32).
17. FAIL if DUT has problems doing Discovery.

18. FAIL if DUT does not respond with a NACK packet after the Segment Register Address is sent.
19. FAIL if DUT sends any DDC Data after it receives the EOF.
20. FAIL if POST_DDC_TEST_READ_PROCEDURE does not succeed.
21. If all preceding steps pass, then PASS; else FAIL.

6.3.22 DDC – Sink DUT Input; Illegal Responses

Make illegal or unexpected responses to the Sink DUT when it responds to DDC commands.

6.3.22.1 CBM-Sink: DUT Receives SOF, CONT

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.22.1.1 Test Objective

Verify that Sink DUT adheres to the MHL Standard State Machine for DDC Write transactions.

6.3.22.1.2 References

[MHL] Section 7.5.3.1

6.3.22.1.3 Required Test Equipment

MSC Test Tool

6.3.22.1.4 Required Methodology

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Select typical timings, resistances, capacitances.
4. Set Tester to use its Discovery engine to enable normal MHL discovery.
5. Set Tester to use its Packet engine to respond normally.
6. Direct Tester to proceed with Discovery.
7. Wait until Tester receives SET_HPD command.
8. FAIL if DUT does not eventually send SET_HPD.
9. Tester sends DDC SOF (0x0_1_30).
10. Tester sends CONT (0x0_1_50).
11. **END:** Do POST_DDC_TEST_READ_PROCEDURE:
 - a. Tester sends DDC SOF (0x0_1_30).
 - b. Tester sends EDID ADDRESS (0x0_0_A1).
 - c. Tester receives and checks DDC ACK packet (0x0_1_33).
 - d. Tester sends DDC CONT (0x0_1_50).
 - e. Tester receives and remembers DDC Data (0x0_0_??).
 - f. Tester sends DDC STOP (0x0_1_51).
 - g. Tester sends DDC EOF (0x0_1_32).
12. FAIL if DUT does not respond with a DDC ABORT (0x0_1_35) after the protocol error is seen.
13. FAIL if POST_DDC_TEST_READ_PROCEDURE does not succeed.
14. If all preceding steps pass, then PASS; else FAIL.

6.3.22.2 CBM-Sink: DUT Receives DDC Write; Various Errors

| |
|----------------------------------|
| Application: Sink, Dongle |
|----------------------------------|

6.3.22.2.1 Test Objective

Verify that Sink DUT adheres to the MHL Standard State Machine for DDC Write transactions.

6.3.22.2.2 References

[MHL] Section 7.5.3.4

6.3.22.2.3 Required Test Equipment

MSC Test Tool

6.3.22.2.4 Required Methodology

Execute the following procedure once for each row in Table 6-38. TEST_OFFSET, TEST_DATA, TEST_STOP and TEST_EOF will take on values defined in the table.

1. If CDF entry **CDF_HDCP_SUPPORT** is not YES, then return PASS (SKIP) and terminate the test.
2. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
3. Set Tester port to disconnected state.
4. Select typical timings, resistances, capacitances.
5. Set Tester to use its Discovery engine to enable normal MHL discovery.
6. Set Tester to use its Packet engine to respond normally.
7. Direct Tester to proceed with Discovery.
8. Wait until Tester receives SET_HPD command.
9. FAIL if DUT does not eventually send SET_HPD.
10. Tester sends DDC SOF (0x0_1_30).
11. Tester sends HDCP ADDRESS (0x0_0_74) (Write).
12. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to END.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
13. Tester sends TEST_OFFSET.
14. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to END.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
15. Tester sends TEST_DATA.
16. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to END.
 - b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
17. Tester sends TEST_STOP.
18. Tester sends TEST_EOF.
19. END: Do POST_DDC_TEST_READ_PROCEDURE:
 - a. Tester sends DDC SOF (0x0_1_30).
 - b. Tester sends EDID ADDRESS (0x0_0_A1).
 - c. Tester receives and checks DDC ACK packet (0x0_1_33).
 - d. Tester sends DDC CONT (0x0_1_50).
 - e. Tester receives and remembers DDC Data (0x0_0_??).
 - f. Tester sends DDC STOP (0x0_1_51).
 - g. Tester sends DDC EOF (0x0_1_32).
20. FAIL if DUT does not respond with a DDC ABORT (0x0_1_35) after a timeout is detected.
21. FAIL if DUT does not respond with a DDC ABORT (0x0_1_35) after a protocol error is detected.
22. FAIL if POST_DDC_TEST_READ_PROCEDURE does not succeed.

23. If all preceding steps pass, then PASS; else FAIL.

Table 6-38. DUT Receives DDC Write

| | TEST_OFFSET | TEST_DATA | TEST_STOP | TEST_EOF | Purpose |
|---|------------------------|------------------------|------------------------|------------------------|-----------------------------------|
| 1 | Timeout | Do not Send | Do not Send | Do not Send | DDC Write; Timeout on Offset |
| 2 | (0x0_0_10) | Timeout | Do not Send | Do not Send | DDC Write; Timeout on Data |
| 3 | (0x0_0_10) | (0x0_0_00) | Timeout | Do not Send | DDC Write; Timeout on STOP |
| 4 | (0x0_0_10) | (0x0_0_00) | DDC STOP (0x0_1_51) | Timeout | DDC Write; Timeout on EOF |
| 5 | DDC CONT (0x0_1_50) | Do not Send | Do not Send | Do not Send | DDC Write; CONT instead of Offset |
| 6 | (0x0_0_10) | DDC CONT (0x0_1_50) | Do not Send | Do not Send | DDC Write; CONT instead of Data |
| 7 | (0x0_0_10) | (0x0_0_00) | DDC EOF (0x0_1_32) | Do not Send | DDC Write; EOF instead of STOP |
| 8 | (0x0_0_10) | (0x0_0_10) | DDC STOP (0x0_1_51) | DDC STOP (0x0_1_51) | DDC Write; STOP instead of EOF |
| 9 | (0x0_0_10) | (0x0_0_10) | DDC STOP (0x0_1_51) | (0x0_0_00) | DDC Write; Data instead of EOF |

6.3.22.3 CBM-Sink: DUT Receives DDC Short Read and Current Read; Various Errors

Application: Sink, Dongle

6.3.22.3.1 Test Objective

Verify that Sink DUT adheres to the MHL Standard State Machine for DDC Short Read and Current Read transactions.

6.3.22.3.2 References

[MHL] Section 7.5.3.3

6.3.22.3.3 Required Test Equipment

MSC Test Tool

6.3.22.3.4 Required Methodology

Execute the following procedure once for each row in Table 6-39. TEST_STOP and TEST_EOF will take on values defined in the table.

1. If necessary, connect DUT to Tester Source or Sink port, as appropriate.
2. Set Tester port to disconnected state.
3. Select typical timings, resistances, capacitances.
4. Set Tester to use its Discovery engine to enable normal MHL discovery.
5. Set Tester to use its Packet engine to respond normally.
6. Direct Tester to proceed with Discovery.
7. Tester sends DDC SOF (0x0_1_30).
8. Tester sends EDID ADDRESS (0x0_0_A1) (Read).
9. Tester receives and checks DDC ACK packet (0x0_1_33):
 - a. If Tester receives DDC NACK packet (0x0_1_34) instead, send DDC EOF (0x0_1_32) and go to END.

- b. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - c. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
10. Tester sends DDC_CONT (0x0_1_50).
11. Tester receives and remembers DDC_DATA (0x0_0_??):
 - a. If Tester receives DDC ABORT (0x0_1_35) instead, go to END.
 - b. If Tester receives timeout instead, send DDC ABORT (0x0_1_35) and go to END.
12. Tester sends TEST_STOP.
13. Tester sends TEST_EOF.
14. **END:** Do POST_DDC_TEST_READ_PROCEDURE:
 - a. Tester sends DDC SOF (0x0_1_30).
 - b. Tester sends EDID ADDRESS (0x0_0_A1).
 - c. Tester receives and checks DDC ACK packet (0x0_1_33).
 - d. Tester sends DDC CONT (0x0_1_50).
 - e. Tester receives and remembers DDC Data (0x0_0_??).
 - f. Tester sends DDC STOP (0x0_1_51).
 - g. Tester sends DDC EOF (0x0_1_32).
15. FAIL if DUT does not respond with a DDC ABORT (0x0_1_35) after a timeout is detected.
16. FAIL if DUT does not respond with a DDC ABORT (0x0_1_35) after a protocol error is detected.
17. FAIL if POST_DDC_TEST_READ_PROCEDURE does not succeed.
18. If all preceding steps pass, then PASS; else FAIL.

Table 6-39. DUT Receives Short Read and Current Read – Various Errors

| | TEST_STOP | TEST_EOF | Purpose |
|---|---------------------|---------------------|---|
| 1 | Timeout | Do not Send | Short Read and Current Read; Timeout on CONT/STOP |
| 2 | DDC STOP (0x0_1_51) | Timeout | Short Read and Current Read; Timeout on EOF |
| 3 | DDC Data (0x0_0_00) | Do not Send | Short Read and Current Read; Data instead of STOP |
| 4 | DDC EOF (0x0_1_32) | Do not Send | Short Read and Current Read; EOF instead of STOP |
| 5 | DDC STOP (0x0_1_51) | DDC Data (0x0_0_00) | Short Read and Current Read; Data instead of EOF |

7 Cable Test

7.1 Mechanical tests

7.1.1 Cable Assembly Mechanical Tests

7.1.1.1 Connector Maximum Envelope

| | |
|---------------------|--------------|
| Application: | Cable |
|---------------------|--------------|

7.1.1.1.1 Test Objective

Verify that DUT's connector shell and cable fit inside minimum allowable receptacle envelope.

7.1.1.1.2 Required Test Equipment

Ruler, caliper, micrometer or similar.

7.1.1.1.3 Required Methodology

1. Measure all overmold dimensions.
2. Verify that all dimensions fall within maximum permitted value in each connector specification.
3. Measure the maximum extension from the shell in the rigid portion of the connector.

4. Verify that all dimensions fall within maximum permitted value in each connector specification.
5. If all measurements fall within limits, then PASS, else FAIL.

7.1.1.1.4 Recommended Methodology

Perform steps in Required Methodology above using a ruler, caliper, micrometer or similar.

7.2 Electrical Tests

7.2.1 Cable Assembly Electrical Tests

7.2.1.1 Wire Assignment

| |
|---------------------------|
| Application: Cable |
|---------------------------|

7.2.1.1.1 Test Objective

Verify that all specified connections are present in cable.

7.2.1.1.2 Required Test Equipment

Digital Multimeter

Source TPA-R-WO

Sink TPA-R-WO

SMA Coaxial Cables

7.2.1.1.3 Required Methodology

1. Refer to one connector as "Connector 1" and the other as "Connector 2".
2. Check if TMDS channel (D+ and D-), CBUS, VBUS, and GND pins are assigned in Connector 1 and Connector 2.
3. Check connection for five pins (D+, D-, CBUS, VBUS, and GND) between Connector 1 and Connector 2.
4. If no connection is specified between Connector 1 and Connector 2, then FAIL.
5. If connection is specified between Connector 1 and Connector 2 but no valid connection, then FAIL.

7.2.1.1.4 Recommended Methodology

A valid connection is defined as less than 100Ω. For all signal types, a valid no-connection is defined as greater than 1MΩ. Perform the "Required Method" using a standard Digital Multi-meter set for measurement of resistance using the valid connection criteria above.

7.2.1.2 Differential Intra-Pair Skew

| |
|---------------------------|
| Application: Cable |
|---------------------------|

7.2.1.2.1 Test Objective

This test confirms that the MHL cable Differential Intra-Pair Skew does not exceed the maximum value allowed in the spec.

7.2.1.2.2 References

[MHL] Section 13.8.1 Table 13-13 Cable Assembly MHL Characteristics: Differential Intra-Pair Skew ≤ 43 ps

7.2.1.2.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR or
4-port VNA
Source TPA-R-WO
Sink TPA-R-WO
SMA Coaxial Cables

7.2.1.2.4 Required Methodology

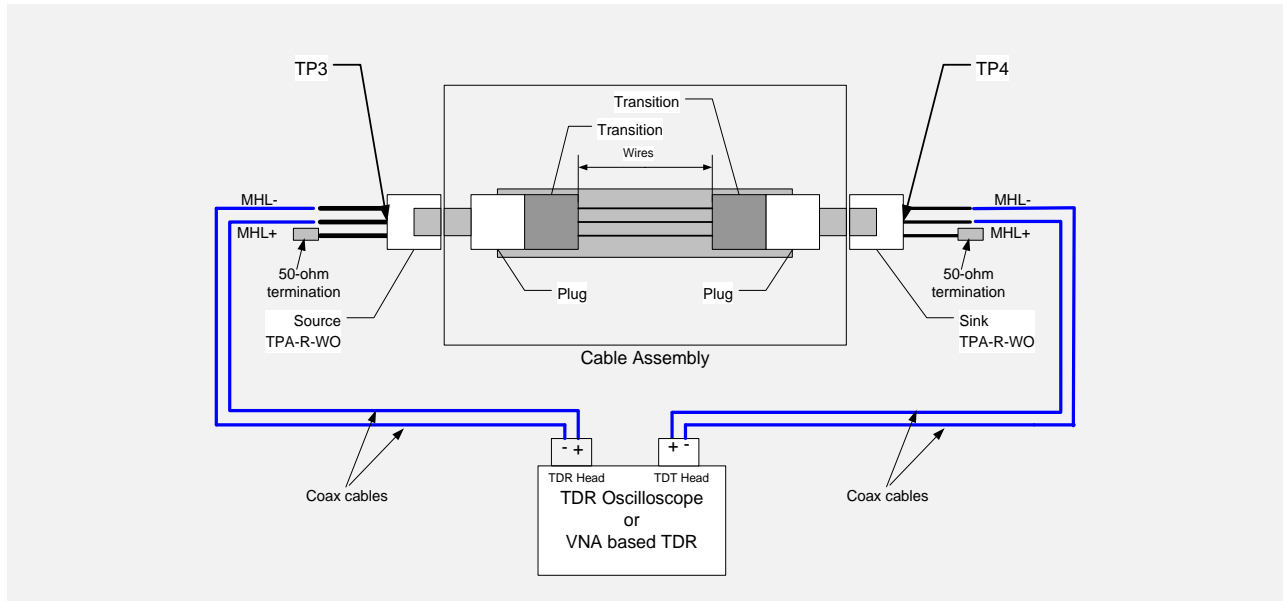


Figure 55. Differential Intra-Pair Skew Test Setup

1. Set the TDR oscilloscope to output differential.
2. Connect the equipment as shown in Figure 55.
3. Set vertical scale 100 mV/div and horizontal scale 400 ps/div.
4. Measure the differential skew at the 50% crossing times of the two differential edges on the TDT head side.*
 - a) If the differential skew is less than or equal to 43ps, then PASS.
 - b) Else if the MHL cable passes tests 7.2.1.16 and 0, then PASS.
 - c) Otherwise FAIL.

*Need to compensate the differential intra-pair skew caused by the test fixtures (TPA boards and co-ax cables).

7.2.1.3 Common-Mode Intra-Pair Skew

| | |
|---------------------|--------------|
| Application: | Cable |
|---------------------|--------------|

7.2.1.3.1 Test Objective

This test confirms that the MHL cable Common-Mode Intra-Pair Skew does not exceed the maximum value allowed in the spec.

7.2.1.3.2 References

[MHL] Section 13.8.1 Table 13-13 Cable Assembly MHL Characteristics: Common-Mode Intra-Pair Skew ≤ 43 ps

7.2.1.3.3 Required Test Equipment

TDR/TDT Oscilloscope or

VNA-based TDR

Source TPA-R-WO

Sink TPA-R-WO

SMA Coaxial Cables

7.2.1.3.4 Required Methodology

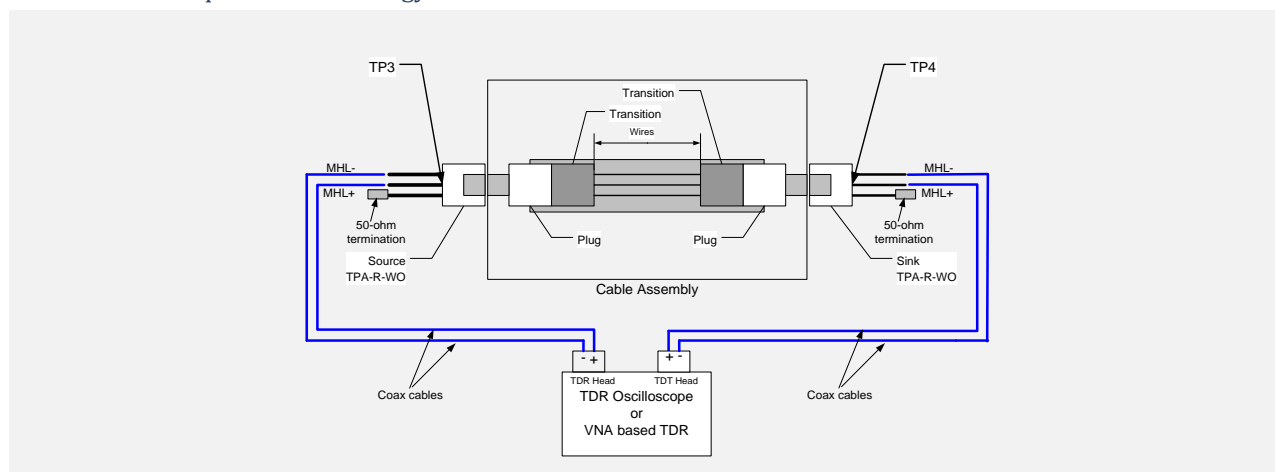


Figure 56. Common-mode Intra-Pair Skew Test Setup

1. Set the TDR oscilloscope to output common-mode.
2. Connect the equipment as shown in Figure 56.
3. Set vertical scale 100mV/div and horizontal scale 400ps/div.
4. Measure the common-mode skew at the 50% crossing times of the two common-mode edges on the TDT head side.*
 - a) If the common-mode skew is less than or equal to 43ps, then PASS.
 - b) Else if the MHL cable passes tests 7.2.1.16 and 0, then PASS.
 - c) Otherwise FAIL.

*Need to compensate the common-mode intra-pair skew caused by the test fixtures (TPA boards and co-ax cables)

7.2.1.4 Differential Characteristic Impedance

Application: Cable

7.2.1.4.1 Test Objective

This test confirms that the MHL cable Differential Characteristic Impedance is within the range allowed in the spec.

7.2.1.4.2 References

[MHL] Section 13.8.1 Table 13-13 Cable Assembly MHL Characteristics:

Cable longer than 300mm length:

Connection point and transition area (1 ns distance): 100 ohms +/- 15 ohms*

* A single excursion is permitted out to 65 – 125 ohms, no wider than 250ps

Cable area (from 1 ns to 2 ns distance): 100 ohms +/- 10 ohms

Cables up to 300mm length:

Connection point and transition area (1 ns distance): 100 ohms +/- 15 ohms**

Cable area (from 1 ns to 1.5 ns distance): 100 ohms +/- 15 ohms***

** A single excursion is permitted out to 65 – 125 ohms, no wider than 250ps

*** Arithmetic mean of impedance: 100 ohms +/- 10 ohms

7.2.1.4.3 Required Test Equipment

TDR/TDT Oscilloscope or

VNA-based TDR

Source TPA-R-WO

Sink TPA-R-WO

SMA Coaxial Cables

7.2.1.4.4 Required Methodology

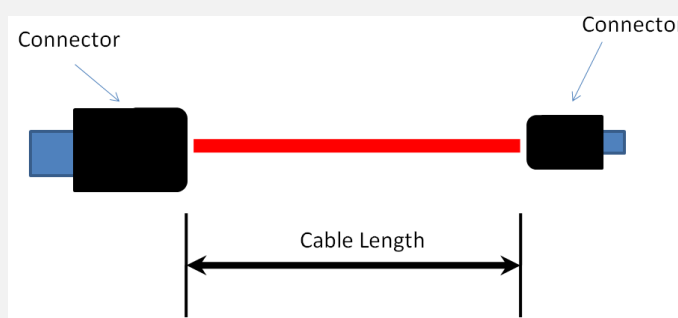


Figure 57. Illustration of Cable Length for the Impedance Test

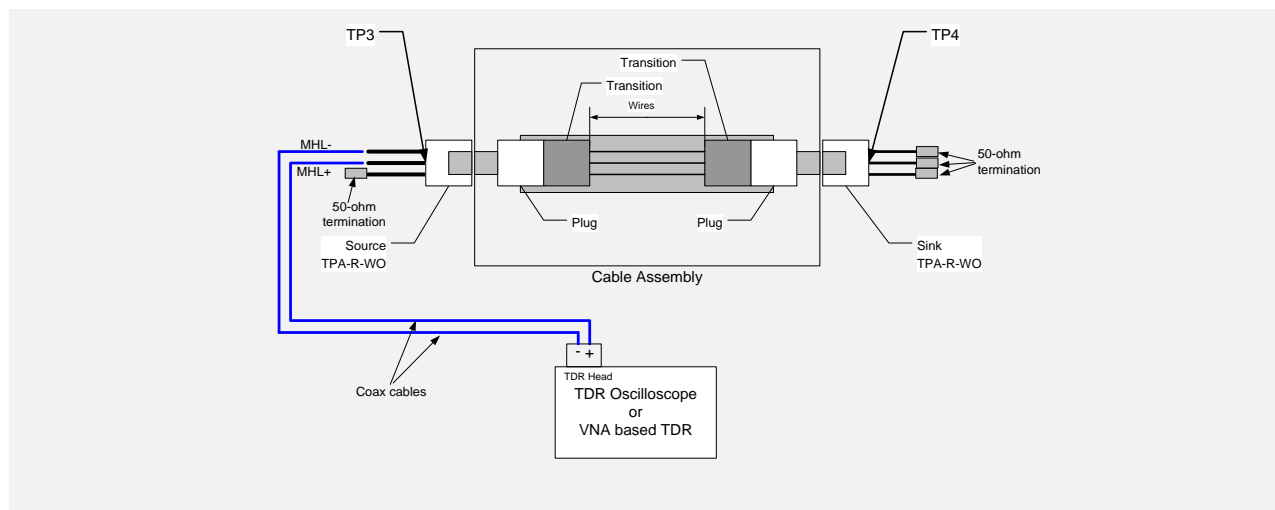


Figure 58. Differential Characteristic Impedance Test Setup

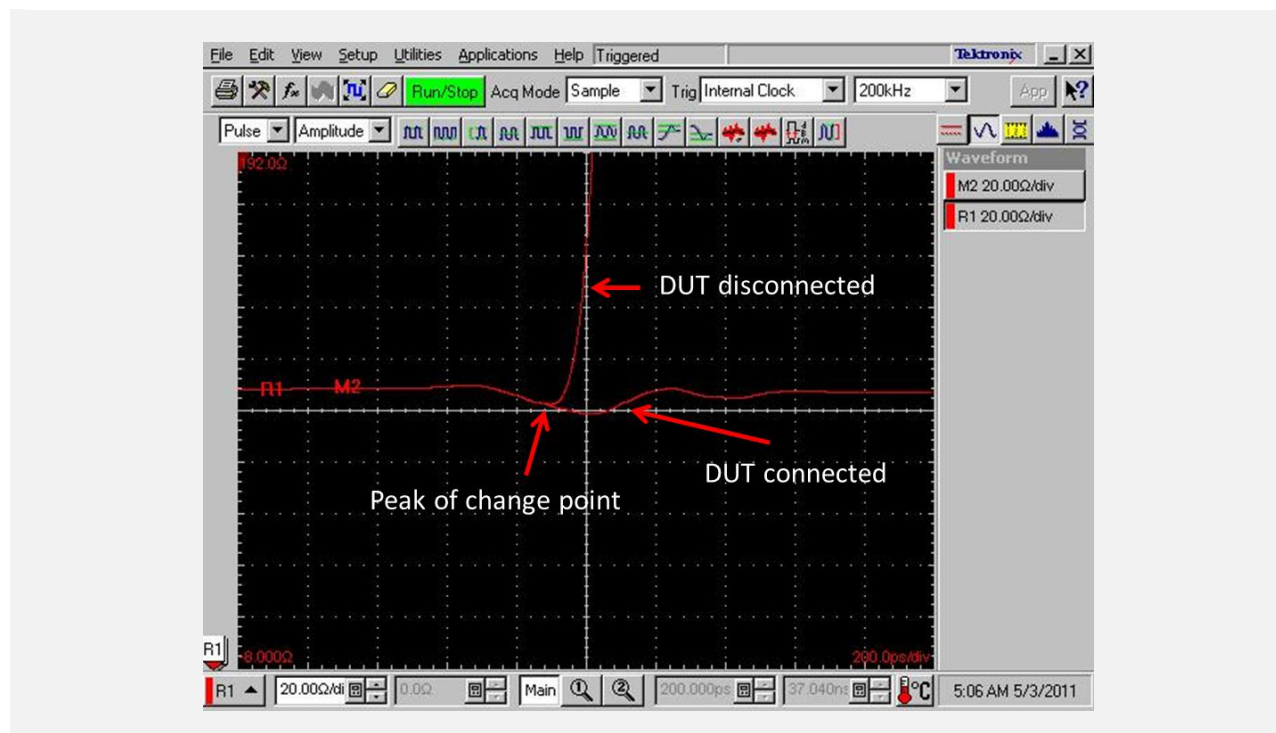


Figure 59. Set reference point for differential mode

Before testing, compensate the differential intra-pair skew caused by the test fixtures (TPA boards and co-ax cables) and measure cable length as shown in Figure 57.

If cable length is greater than 300mm, then:

1. Connect the Source TPA-R-WO to the TDR oscilloscope.
2. Set the effective rise time of the differential TDR pulse to 200ps (20 – 80%). Record the time at the peak of the change point in Figure 59. This is the connection point (T_{CONN}).
3. Connect the Source TPA-R-WO to the source end of the Cable DUT and measure the differential impedance from T_{CONN} to $T_{\text{CONN}} + 2$ ns.
4. The differential impedance from T_{CONN} to $T_{\text{CONN}} + 1$ ns is $Z_{\text{OD_TRANSITION}}$ and the differential impedance from $T_{\text{CONN}} + 1$ ns to $T_{\text{CONN}} + 2$ ns is $Z_{\text{OD_CABLE}}$.
 - a. If (85 ohms $\leq Z_{\text{OD_TRANSITION}} \leq 115$ ohms) AND (90 ohms $\leq Z_{\text{OD_CABLE}} \leq 110$ ohms), it is PASS
 - b. If (65 ohms $\leq Z_{\text{OD_TRANSITION}} < 85$ ohms) OR (115 ohms $< Z_{\text{OD_TRANSITION}} \leq 125$ ohms), the impedance is in the excursion range. If the excursion occurs at a single time AND the duration of the single excursion is less than 250ps AND (90 ohms $\leq Z_{\text{OD_CABLE}} \leq 110$ ohms), it is PASS.
 - c. All other cases are FAIL.
5. Repeat Step 4 for the sink end of the Cable DUT.

If cable length is less than or equal to 300mm, then:

1. Connect the Source TPA-R-WO to the TDR oscilloscope.
2. Set the effective rise time of the differential TDR pulse to 200ps (20 – 80%). Record the time at the peak of the change point in Figure 59. This is the connection point (T_{CONN200}).

3. Connect the Source TPA-R-WO to the source end of the Cable DUT and measure the differential impedance from T_{CONN200} to $T_{\text{CONN200}} + 1 \text{ ns}$.
4. The Differential impedance from T_{CONN200} to $T_{\text{CONN200}} + 1 \text{ ns}$ is $Z_{\text{OD_TRANSITION}}$
5. Set the effective rise time of the differential TDR pulse to 50ps (20 – 80%). Record the time at the peak of the change point in Figure 59. This is the connection point (T_{CONN50}).
6. Connect the Source TPA-R-WO to the source end of the Cable DUT and measure the differential impedance from $T_{\text{CONN50}} + 1 \text{ ns}$ to $T_{\text{CONN50}} + 1.5 \text{ ns}$.
7. The Differential impedance from $T_{\text{CONN50}} + 1 \text{ ns}$ to $T_{\text{CONN50}} + 1.5 \text{ ns}$ is $Z_{\text{OD_CABLE_SHORT}}$ and the arithmetic mean of $Z_{\text{OD_CABLE_SHORT}}$ is $Z_{\text{OD_CABLE_AVE}}$.
 - a. If $(85 \text{ ohms} \leq Z_{\text{OD_TRANSITION}} \leq 115 \text{ ohms})$ AND $(85 \text{ ohms} \leq Z_{\text{OD_CABLE_SHORT}} \leq 115 \text{ ohms})$ AND $(90 \text{ ohms} \leq Z_{\text{OD_CABLE_AVE}} \leq 110 \text{ ohms})$, it is PASS.
 - b. If $(65 \text{ ohms} \leq Z_{\text{OD_TRANSITION}} < 85 \text{ ohms})$ OR $(115 \text{ ohms} < Z_{\text{OD_TRANSITION}} \leq 125 \text{ ohms})$, the impedance is in the excursion range. If the excursion occurs at a single time AND the duration of the single excursion is less than 250ps AND $(85 \text{ ohms} \leq Z_{\text{OD_CABLE_SHORT}} \leq 115 \text{ ohms})$ AND $(90 \text{ ohms} \leq Z_{\text{OD_CABLE_AVE}} \leq 110 \text{ ohms})$, it is PASS.
 - c. All other cases are FAIL.
8. Repeat Step 4-7 for the sink end of the Cable DUT.

7.2.1.5 Common-Mode Characteristic Impedance

| |
|---------------------------|
| Application: Cable |
|---------------------------|

7.2.1.5.1 Test Objective

This test confirms that the MHL cable Common-Mode Characteristic Impedance is within the range allowed in the spec.

7.2.1.5.2 References

[MHL] Section 13.8.1 Table 13-13 Cable Assembly MHL Characteristics:

Cables longer than 600mm length:

Connection point and transition area (3 ns distance): 30 ohms +/- 6 ohms*

* A single excursion is permitted out to 20 – 40 ohms, no wider than 500ps

Cable area (from 3 ns to 4 ns distance): 30 ohms +/- 5 ohms

Cables up to 600mm length:

Connection point and transition area (3 ns distance): 30 ohms +/- 6 ohms**

Cable area (from 1 ns to 1.5 ns distance): 30 ohms +/- 10 ohms***

** A single excursion is permitted out to 20 – 40 ohms, no wider than 500ps

*** Arithmetic mean of impedance: 30 ohms +/- 6 ohms

7.2.1.5.3 Required Test Equipment

TDR/TDT Oscilloscope or

VNA-based TDR

Source TPA-R-WO

Sink TPA-R-WO

SMA Coaxial Cables

7.2.1.5.4 Required Methodology

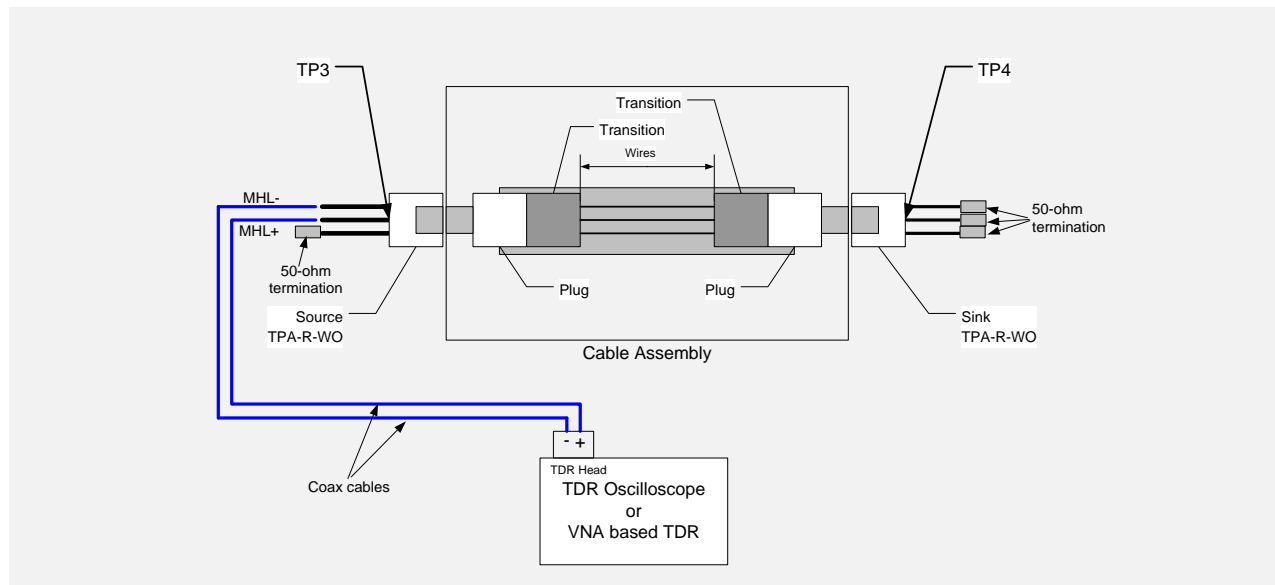


Figure 60. Common-Mode Characteristic Impedance Test Setup

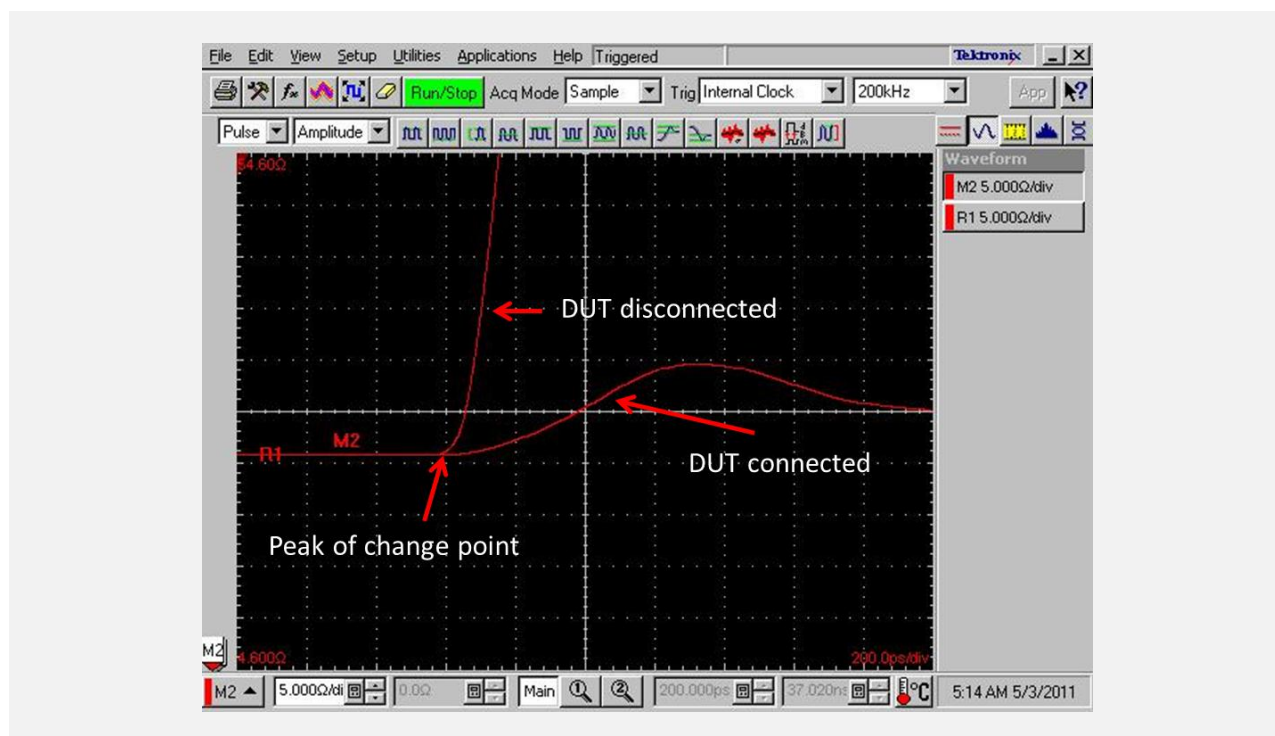


Figure 61. Set reference point for common mode

Before testing, compensate the common-mode intra-pair skew caused by the test fixtures (TPA boards and co-ax cables) and measure cable length as shown in Figure 57.

If cable length is greater than 600mm, then:

1. Connect the Source TPA-R-WO to the TDR oscilloscope.

2. Set the effective rise time of the common-mode TDR pulse to 600ps (20 – 80%).
3. Record the time at the peak of the change point in Figure 61. This is the connection point (T_{CONN}).
4. Connect the Source TPA-R-WO to the source end of the Cable DUT and measure the common-mode impedance from T_{CONN} to $T_{CONN} + 4$ ns.
5. The common-mode impedance from T_{CONN} to $T_{CONN} + 3$ ns is $Z_{OC_TRANSITION}$ and the common-mode impedance from $T_{CONN} + 3$ ns to $T_{CONN} + 4$ ns is Z_{OC_CABLE} .
 - a. If (24 ohms $\leq Z_{OC_TRANSITION} \leq 36$ ohms) AND (25 ohms $\leq Z_{OC_CABLE} \leq 35$ ohms), it is PASS
 - b. If (20 ohms $\leq Z_{OC_TRANSITION} < 24$ ohms) OR (36 ohms $< Z_{OC_TRANSITION} \leq 40$ ohms), the impedance is in the excursion range. If the excursion occurs at a single time AND the duration of the single excursion is less than 500ps AND (25 ohms $\leq Z_{OC_CABLE} \leq 35$ ohms), it is PASS.
 - c. All other cases are FAIL.
6. Repeat Step 5 for the sink end of the Cable DUT.

If cable length is less than or equal to 600 mm:

1. Connect the Source TPA-R-WO to the TDR oscilloscope.
2. Set the effective rise time of the common-mode TDR pulse to 600ps (20 – 80%).
3. Record the time at the peak of the change point in Figure 61. This is the connection point ($T_{CONN600}$).
4. Connect the Source TPA-R-WO to the source end of the Cable DUT and measure the common-mode impedance from $T_{CONN600}$ to $T_{CONN600} + 3$ ns.
5. The common-mode impedance from $T_{CONN600}$ to $T_{CONN600} + 3$ ns is $Z_{OC_TRANSITION}$
6. Set the effective rise time of the common-mode TDR pulse to 50ps (20 – 80%).
7. Record the time at the peak of the change point in Figure 61. This is the connection point (T_{CONN50}).
8. Connect the Source TPA-R-WO to the source end of the Cable DUT and measure the common-mode impedance from $T_{CONN50} + 1$ ns to $T_{CONN50} + 1.5$ ns.
9. The common-mode impedance from $T_{CONN50} + 1$ ns to $T_{CONN50} + 1.5$ ns is $Z_{OC_CABLE_SHORT}$ and the arithmetic mean of $Z_{OC_CABLE_SHORT}$ is $Z_{OC_CABLE_AVE}$.
 - a. If (24 ohms $\leq Z_{OC_TRANSITION} \leq 36$ ohms) AND (20 ohms $\leq Z_{OC_CABLE_SHORT} \leq 40$ ohms) AND (24 ohms $< Z_{OC_CABLE_AVE} \leq 36$ ohms), it is PASS
 - b. If (20 ohms $\leq Z_{OC_TRANSITION} < 24$ ohms) OR (36 ohms $< Z_{OC_TRANSITION} \leq 40$ ohms), the impedance is in the excursion range. If the excursion occurs at a single time AND the duration of the single excursion is less than 500ps AND (20 ohms $\leq Z_{OC_CABLE_SHORT} \leq 40$ ohms) AND (24 ohms $< Z_{OC_CABLE_AVE} \leq 36$ ohms), it is PASS
 - c. All other cases are FAIL.
10. Repeat Step 5-9 for the sink end of the Cable DUT.

7.2.1.6 CBUS Line Capacitance

| |
|---------------------------|
| Application: Cable |
|---------------------------|

7.2.1.6.1 Test Objective

This test confirms that the Capacitance value of the CBUS Line in a MHL cable is within the range allowed in the spec.

7.2.1.6.2 References

[MHL] Section 13.8.1 Table 13-13 Cable Assembly MHL Characteristics

7.2.1.6.3 Required Test Equipment

TDR/TDT Oscilloscope or

VNA-based TDR

Source TPA-R-WO

SMA Coaxial Cables

7.2.1.6.4 Required Methodology

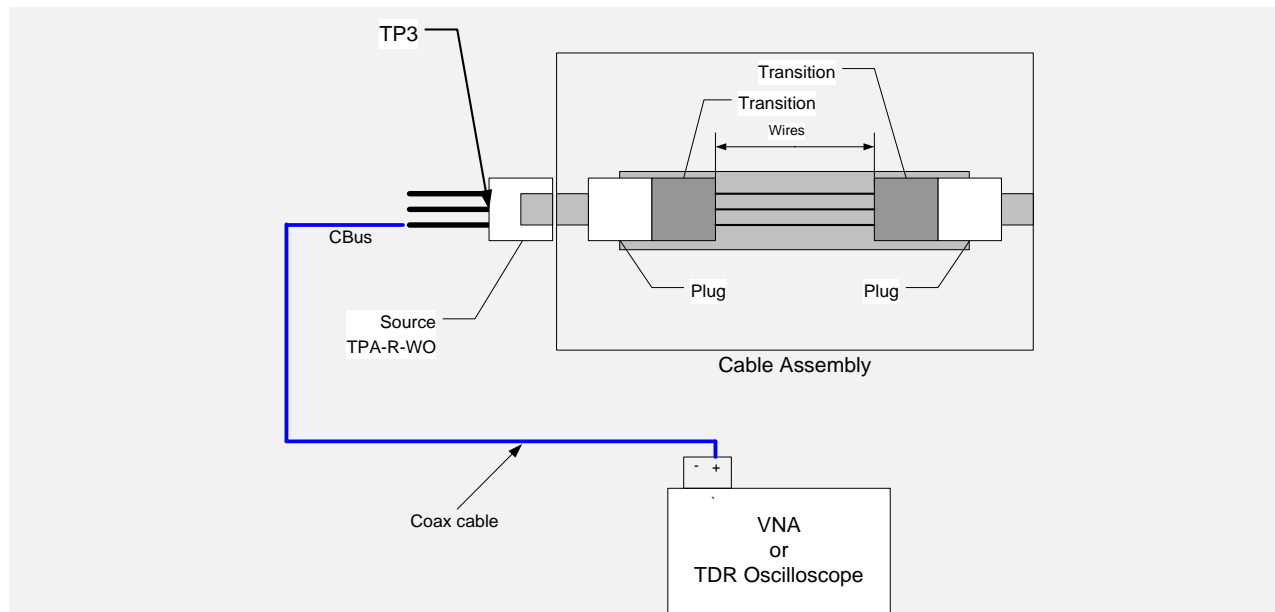


Figure 62. CBUS Line Capacitance Test Setup

1. Connect the co-ax cable as shown in Figure 62.
2. Calibrate the co-ax cable.
3. Connect the Source TPA-R-WO to the co-ax cable.
4. Measure and calculate C_{FIXTURE} .
5. Connect the Cable DUT to the Source TPA-R-WO.
6. Measure and calculate C_{TOTAL} .
7. C_{CBUS} is $C_{\text{TOTAL}} - C_{\text{FIXTURE}}$.
 - a. If $C_{\text{CBUS}}\{\text{min}\} \leq C_{\text{CBUS}} \leq C_{\text{CBUS}}\{\text{max}\}$, it is PASS.
 - b. Otherwise it is FAIL.

7.2.1.7 CBUS Cable Delay

Application: Cable

7.2.1.7.1 Test Objective

This test confirms that the MHL CBUS cable delay does not exceed the maximum value allowed in the spec.

7.2.1.7.2 References

[MHL] Section 13.8.1 Table 13-13 Cable Assembly MHLCharacteristics: $T_{\text{CBUS_CBL_DLY}} \leq 35 \text{ ns}$

7.2.1.7.3 Required Test Equipment

TDR/TDT Oscilloscope or
VNA-based TDR
Source TPA-R-WO
Sink TPA-R-WO
SMA Coaxial Cables

7.2.1.7.4 Required Methodology

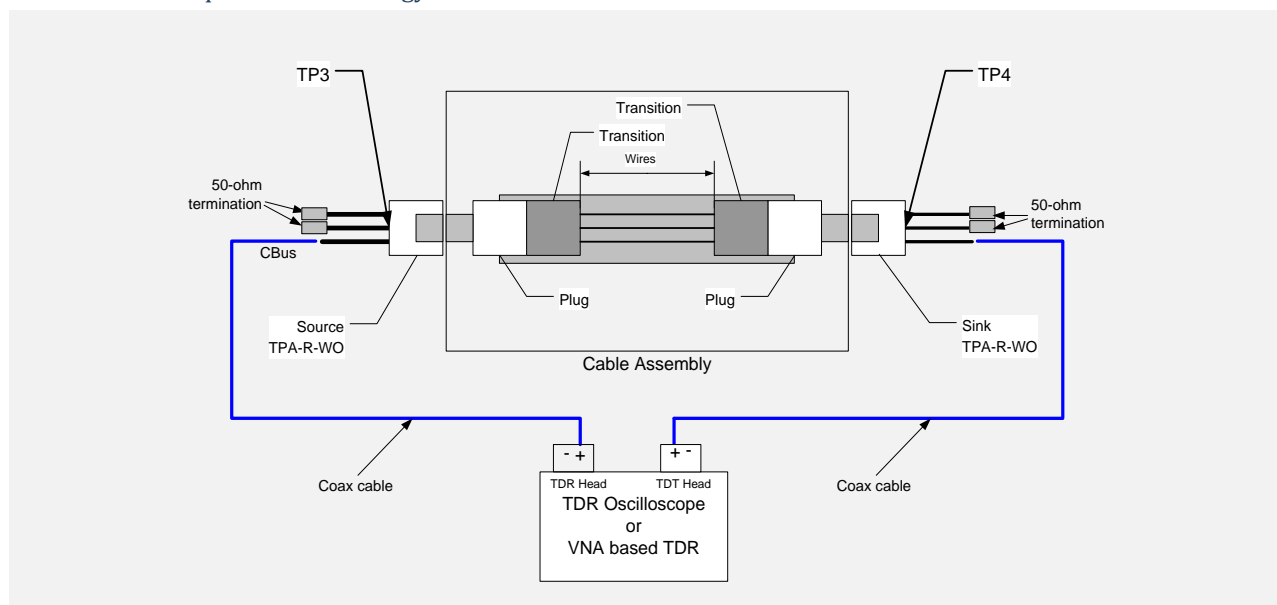


Figure 63. CBUS Cable Delay Test Setup

1. Connect the CBUS pin of the Source TPA-R-WO on the TDR head and the CBUS pin of the Sink TPA-R-WO on the TDT head.
2. Connect the Cable DUT to the TPA boards.
3. Measure the cable propagation delay by measuring 50% cross times of the pulse on the TDR and TDT sides. This is $T_{CBUSDelay}$. *
 - If $T_{CBUSDelay} \leq 35$ ns, it is PASS.
 - Otherwise it is FAIL.

*Need to compensate the propagation delay caused by the test fixtures (TPA boards, co-ax cables, and TDR Head)

7.2.1.8 Differential Insertion Loss

| | |
|---------------------|--------------|
| Application: | Cable |
|---------------------|--------------|

7.2.1.8.1 Test Objective

This test confirms that MHL cable Differential Insertion Loss meets the required spec.

7.2.1.8.2 References

[MHL] Section 13.8.2 Figure 13-20 MHL Cable Differential Insertion Loss for MHL Data Signals

[MHL] Section 13.8.2 Table 13-14 MHL Cable Differential Insertion Loss for MHL Data Signals

7.2.1.8.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR or
4-port VNA
Source TPA-R-WO
Sink TPA-R-WO
SMA Coaxial Cables

7.2.1.8.4 Required Methodology

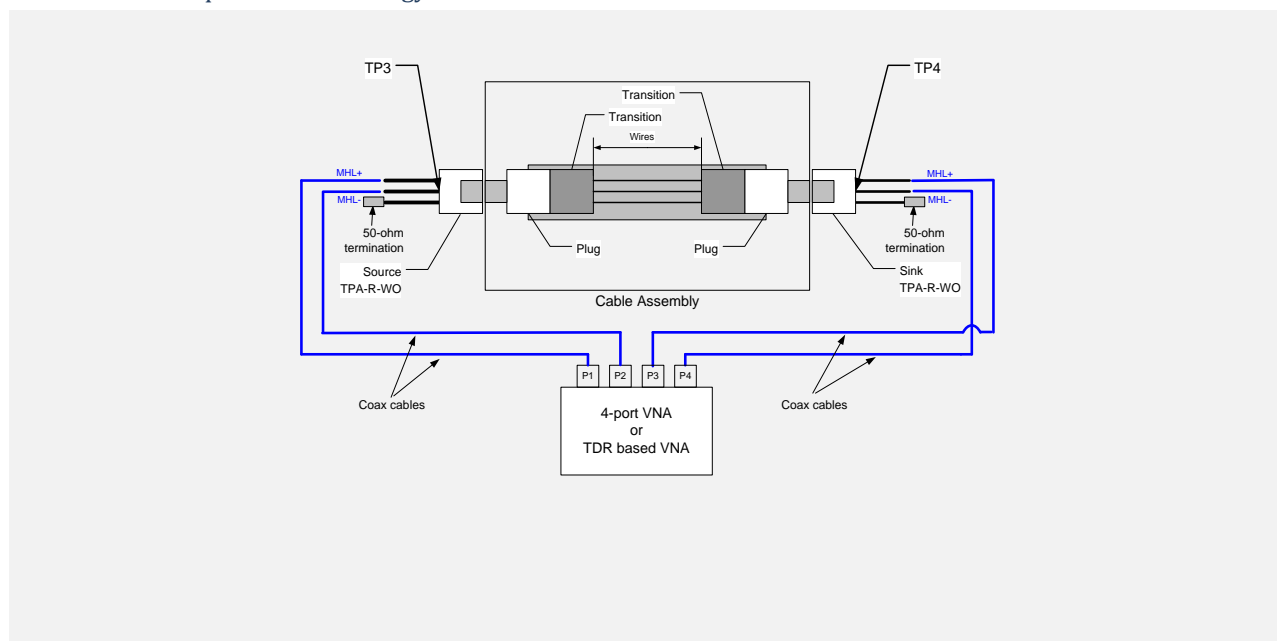


Figure 64. Differential Insertion Loss Test Setup

1. Connect the co-ax cables as shown in Figure 64.
2. Calibrate the co-ax cables using the calibration kit.
3. Connect the TPA-R-WO boards and Cable DUT as shown in Figure 64.
4. Measure the 4-port S parameters for MHL+ and MHL- lines.
5. Calculate S_{dd21} for Differential Insertion Loss:
 - a. If S_{dd21} is better than the limit in Section 7.2.1.8.2, it is PASS.
 - b. Else if the MHL cable passes tests 7.2.1.16 and 0, then PASS.
 - c. Otherwise it is FAIL

7.2.1.9 Common-Mode Insertion Loss

| | |
|---------------------|--------------|
| Application: | Cable |
|---------------------|--------------|

7.2.1.9.1 Test Objective

This test confirms that MHL cable Common-mode Insertion Loss meets the required spec.

7.2.1.9.2 References

[MHL] Section 13.8.2 Figure 13-21 MHL Cable Common-mode Insertion Loss for MHL Clock Signals

[MHL] Section 13.8.2 Table 13-15 MHL Cable Common-mode Insertion Loss for MHL Clock Signals

7.2.1.9.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR or
4-port VNA
Source TPA-R-WO
Sink TPA-R-WO
SMA Coaxial Cables

7.2.1.9.4 Required Methodology

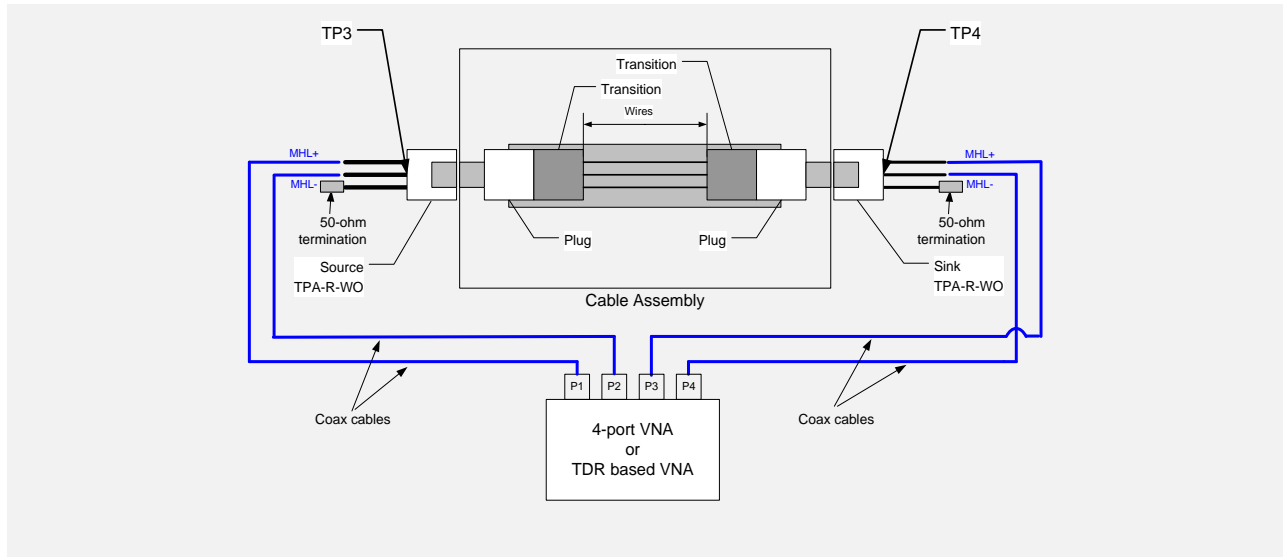


Figure 65. Common-mode Insertion Loss Test Setup

1. Connect the co-ax cables as shown in Figure 65.
2. Calibrate the co-ax cables using the calibration kit.
3. Connect the TPA-R-WO boards and Cable DUT as shown in Figure 65.
4. Measure the 4-port S parameters for MHL+ and MHL- lines.
5. Calculate S_{cc21} for Common-mode Insertion Loss:
 - a. If S_{cc21} is better than the limit in Section 7.2.1.9.2, it is PASS.
 - b. Else if the MHL cable passes tests 7.2.1.16 and 0, then PASS.
 - c. Otherwise it is FAIL.

7.2.1.10 Differential and Common-Mode Conversion

Application: Cable

7.2.1.10.1 Test Objective

This test confirms that MHL cable Mode Conversion between Differential and Common-mode signals meets the required spec.

7.2.1.10.2 References

[MHL] Section 13.8.2 Figure 13-22 MHL Cable Mode Conversion between Differential and Common-mode Signals

[MHL] Section 13.8.2 Table 13-16 MHL Cable Mode Conversion between Differential and Common-mode

7.2.1.10.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR or
4-port VNA
Source TPA-R-WO
Sink TPA-R-WO
SMA Coaxial Cables

7.2.1.10.4 Required Methodology

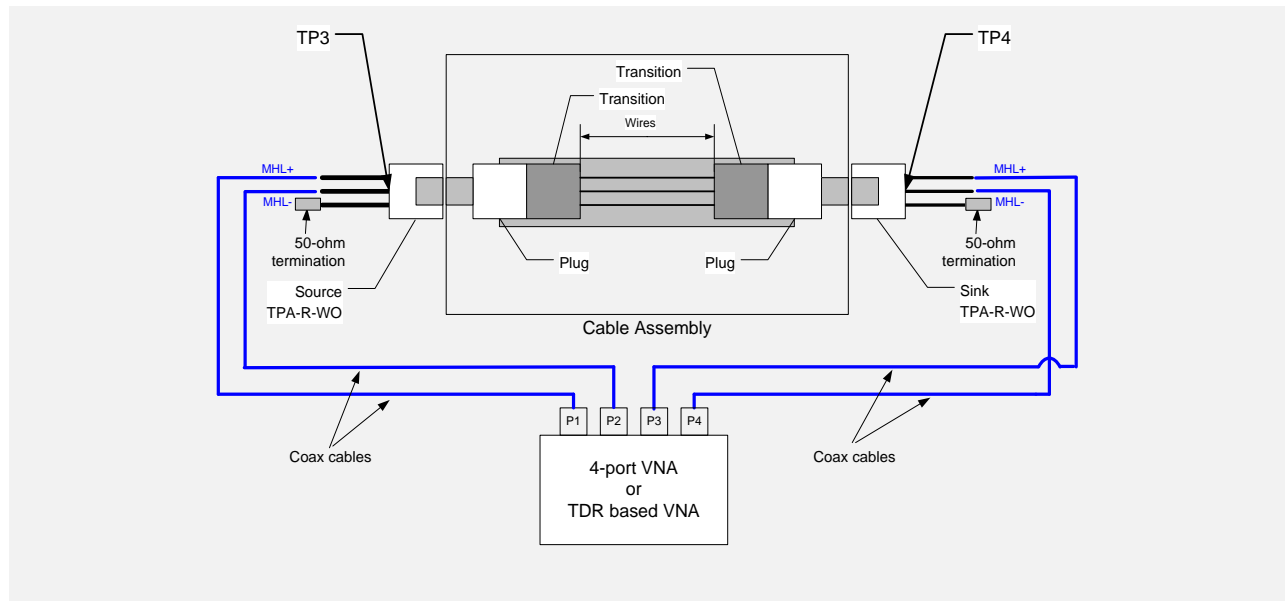


Figure 66. Differential and Common-mode Conversion Test Setup

1. Connect the co-ax cables as shown in Figure 66.
2. Calibrate the co-ax cables using the calibration kit.
3. Connect the TPA-R-WO boards and Cable DUT as shown in Figure 66.
4. Measure the 4-port S parameters for MHL+ and MHL- lines.
5. Calculate S_{dc21} and S_{cd21} for Mode Conversion:
 - a. If both S_{dc21} and S_{cd21} are better than the limit in Section 7.2.1.10.2, it is PASS.
 - b. Else if the MHL cable passes tests 7.2.1.16 and 0, then PASS.
 - c. Otherwise it is FAIL.

7.2.1.11 CBUS Insertion Loss

| |
|---------------------------|
| Application: Cable |
|---------------------------|

7.2.1.11.1 Test Objective

This test confirms that MHL cable CBUS Insertion Loss meets the required spec.

7.2.1.11.2 References

[MHL] Section 13.8.2 Figure 13-23 MHL CBUS Signal Insertion Loss

[MHL] Section 13.8.2 Table 13-17 MHL CBUS Signal Insertion Loss

7.2.1.11.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR or
4-port VNA
Source TPA-R-WO
Sink TPA-R-WO
SMA Coaxial Cables

7.2.1.11.4 Required Methodology

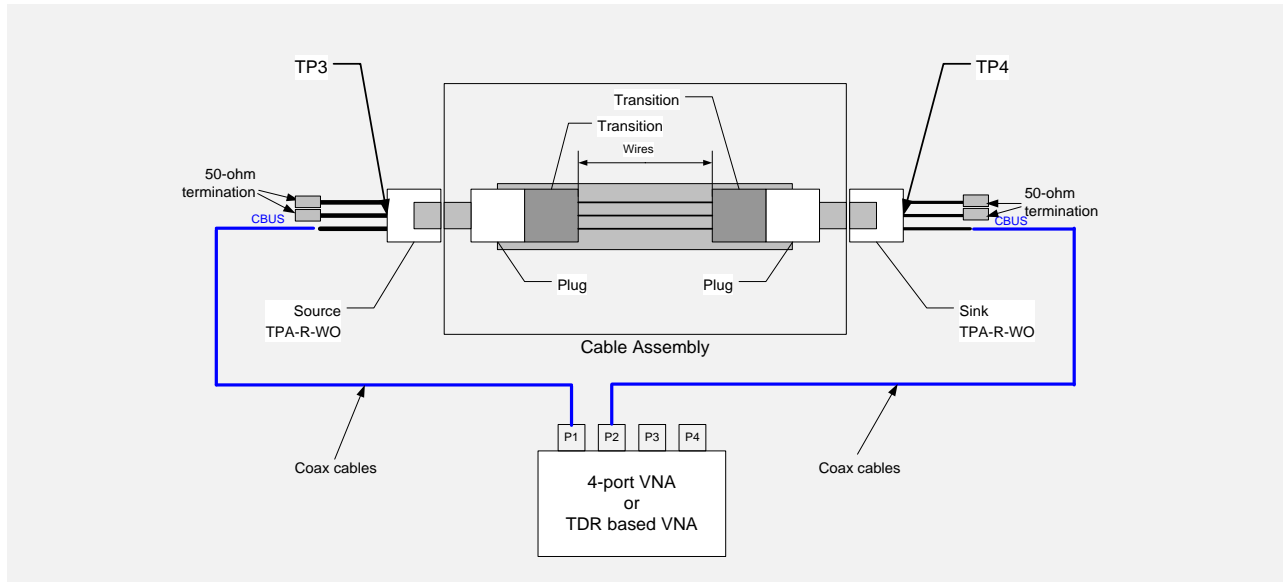


Figure 67. CBUS Insertion Loss Test Setup

1. Connect the co-ax cables as shown in Figure 67.
2. Calibrate the co-ax cables using the calibration kit.
3. Connect the TPA-R-WO boards and Cable DUT as shown in Figure 67.
4. Measure the 2-port S parameters for CBUS line.
5. Calculate S_{21} for CBUS Insertion Loss:
 - a. If S_{21} is better than the limit in Section 7.2.1.11.2, it is PASS.
 - b. Otherwise it is FAIL.

7.2.1.12 Far-End Crosstalk

| |
|---------------------------|
| Application: Cable |
|---------------------------|

7.2.1.12.1 Test Objective

This test confirms that MHL cable Far-End Crosstalk meets the required spec.

7.2.1.12.2 References

[MHL] Section 13.8.2 Figure 13-24 MHL Cable Assembly Far-End Crosstalk between MHL and CBUS
 [MHL] Section 13.8.2 Table 13-18 MHL Cable Assembly Far-End Crosstalk between MHL and CBUS
 [MHL] Section 13.8.2 Figure 13-29 MHL Cable Assembly Far-End Crosstalk between MHL and VBUS
 [MHL] Section 13.8.2 Table 13-20 MHL Cable Assembly Far-End Crosstalk between MHL and VBUS
 [MHL] Section 13.8.2 Figure 13-25 MHL Cable Assembly Far-End Crosstalk between CBUS and VBUS

[MHL] Section 13.8.2 Table 13-20 MHL Cable Assembly Far-End Crosstalk between CBUS and VBUS

7.2.1.12.3 Required Test Equipment

TDR Oscilloscope or VNA-based TDR or

4-port VNA

Source TPA-R-WO

Sink TPA-R-WO

SMA Coaxial Cables

7.2.1.12.4 Required Methodology

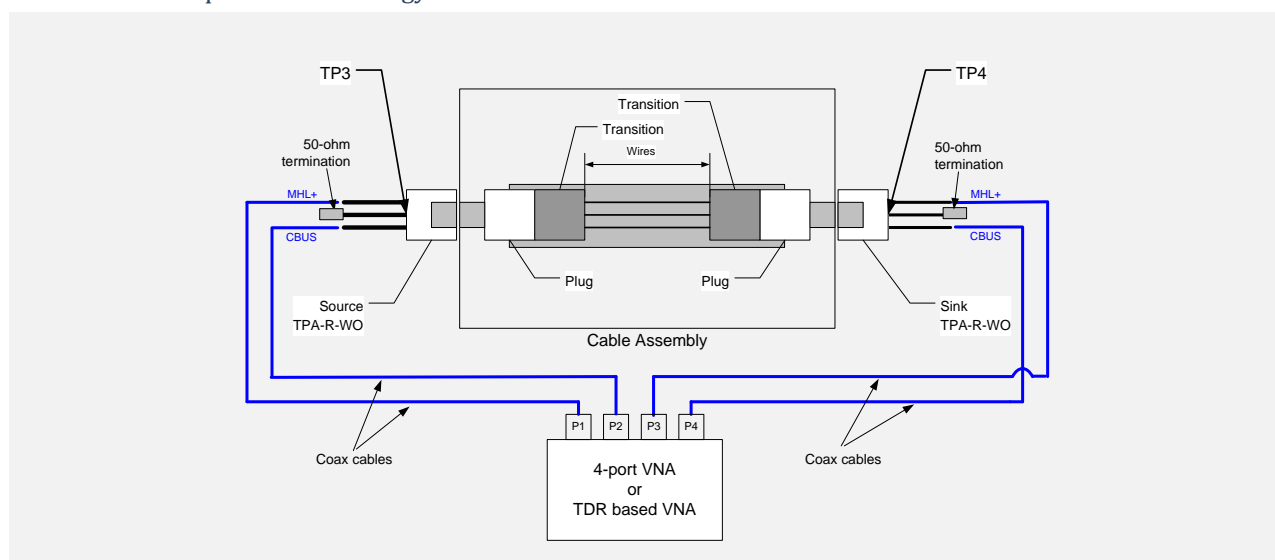


Figure 68. Far-End Crosstalk Test Setup

1. Connect the co-ax cables as shown in Figure 68.
2. Calibrate the co-ax cables using the calibration kit.
3. Connect the TPA-R-WO boards and Cable DUT as shown in Figure 68.
4. Measure the 4-port S parameters for MHL+ and CBUS lines.
5. Calculate S_{41} and S_{32} for Far-End Crosstalk:
 - a. If both S_{41} and S_{32} are better than the limit in Section 7.2.1.12.2, it is PASS.
 - b. Otherwise FAIL.
6. Repeat steps 3-5 for (MHL+, VBUS), (MHL-, CBUS), (MHL-, VBUS) and (CBUS, VBUS) pairs.

7.2.1.13 VBUS Drop

| | |
|---------------------|--------------|
| Application: | Cable |
|---------------------|--------------|

7.2.1.13.1 Test Objective

This test confirms that the MHL cable VBUS Voltage Drop does not exceed the maximum value allowed in the spec.

7.2.1.13.2 References

[MHL] Section 13.8.3 Table 13-20 Cable +5V Power Pin (VBUS) Voltage: $V_{VBUS_DROP} \leq 160mV$

7.2.1.13.3 Required Test Equipment

DC Source-meter

Source TPA-R-WO

Sink TPA-R-WO

SMA Coaxial Cables

7.2.1.13.4 Required Methodology

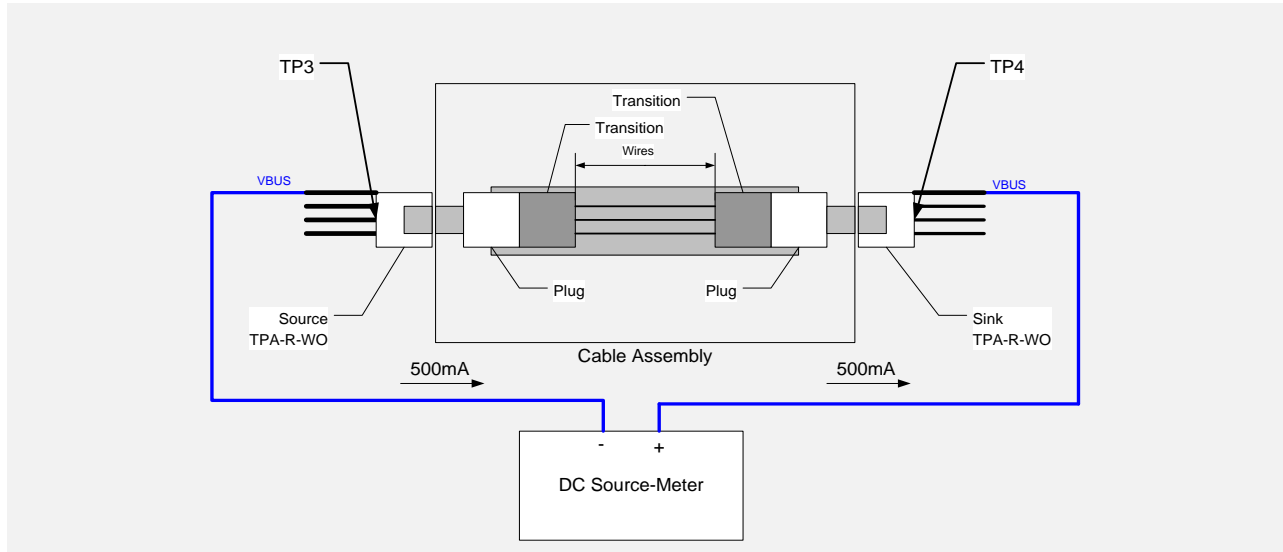


Figure 69. Cable VBUS Drop Test Setup

1. Connect the VBUS pin of the Sink TPA-R-WO to the positive port of the DC Source-Meter and the VBUS pin of the Source TPA-R-WO to the negative port.
2. Connect the Cable DUT to the TPA-R-WO boards.
3. Set the DC Source-Meter current amount to 500mA flowing from positive port to negative port and read the voltage between the two ports of the DC Source Meter. The read voltage is $V_{VbusDrop}$.*
4. If $V_{VbusDrop} \leq 160mV$, it is PASS. Otherwise FAIL.

* Need to compensate the voltage drop caused by the test fixtures (TPA boards and lead cables)

7.2.1.14 Ground Resistance and Ground Shift

Application: Cable

7.2.1.14.1 Test Objective

This test confirms that the MHL cable Ground Resistance and Voltage Shift value do not exceed the maximum value allowed in the spec.

7.2.1.14.2 References

[MHL] Section 13.8.1 Table 13-13 Cable Assembly MHL Characteristics: $R_{GND} \leq 0.08$ ohms

[MHL] Section 13.8.3 Table 13-20 Cable +5V Power Pin (VBUS) Voltage: $V_{GND_SHIFT} \leq 40mV$

7.2.1.14.3 Required Test Equipment

DC Source-meter

Source TPA-R-WO

Sink TPA-R-WO
SMA Coaxial Cables

7.2.1.14.4 Required Methodology

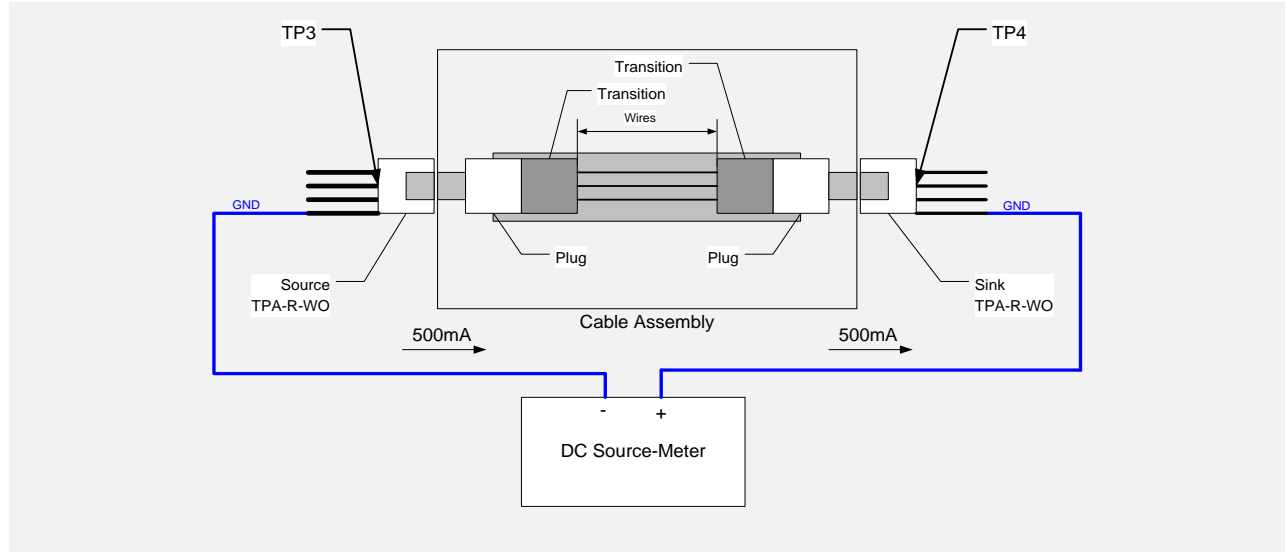


Figure 70. Cable Ground Resistance and Ground Shift Test Setup

1. Connect the GND pin of the Sink TPA-R-WO to the positive port of the DC Source-Meter and the GND pin of the Source TPA-R-WO to the negative port.
2. Connect the Cable DUT to the TPA-R-WO boards.
3. Set the DC Source-Meter current amount to 500mA flowing from positive port to negative port and read the voltage between the two ports of the DC Source Meter. The read voltage is V_{GNDShift} .*
4. If $V_{\text{GNDShift}} \leq 40\text{mV}$, it is PASS. Otherwise FAIL.

* Need to compensate the voltage drop caused by the test fixtures (TPA boards and lead cables)

7.2.1.15 MHL Cable Detect (MHL cable with HDMI Type A plug)

Application: Cable

7.2.1.15.1 Test Objective

This test confirms that the HDMI Type A plug of a MHL cable incorporates a detection circuit specified in the specification.

7.2.1.15.2 References

[MHL] CS-Figure 2-2 Cable Detect Circuit for MHL cable with HDMI Type A Plug

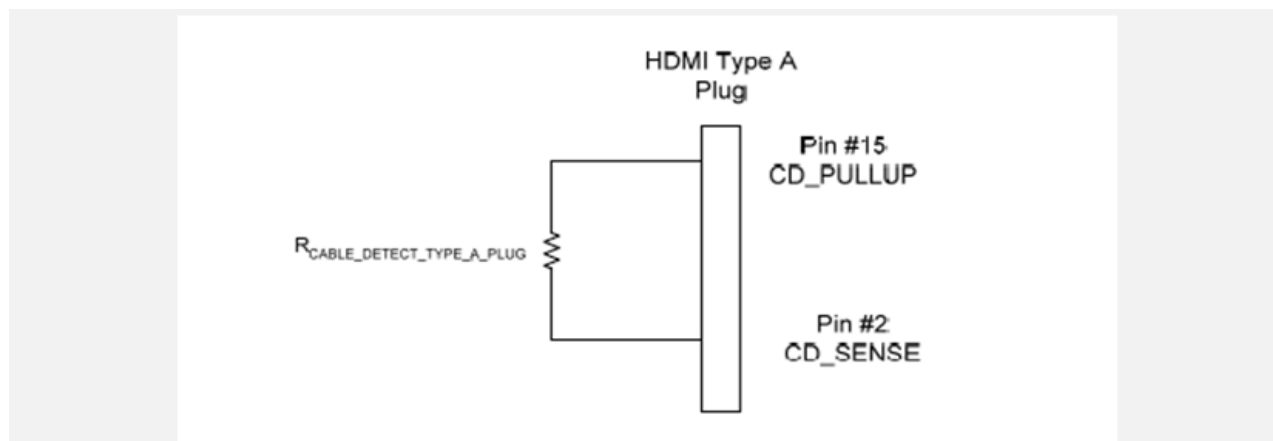


Figure 71. Cable Detect Circuit for MHL Cable with HDMI Type A Plug

[MHL] CS-Table 2-3 Cable Detect AC Electrical Specification for MHL Cable with HDMI Type A Plug:

$$2.97K \text{ ohms} \leq R_{\text{CABLE_DETECT_TYPE_A_PLUG}} \leq 3.63K \text{ ohms}$$

7.2.1.15.3 Required Test Equipment

Digital Multi-meter

HDMI Type A Receptacle TPA Fixture

SMA Coaxial Cables

7.2.1.15.4 Required Methodology

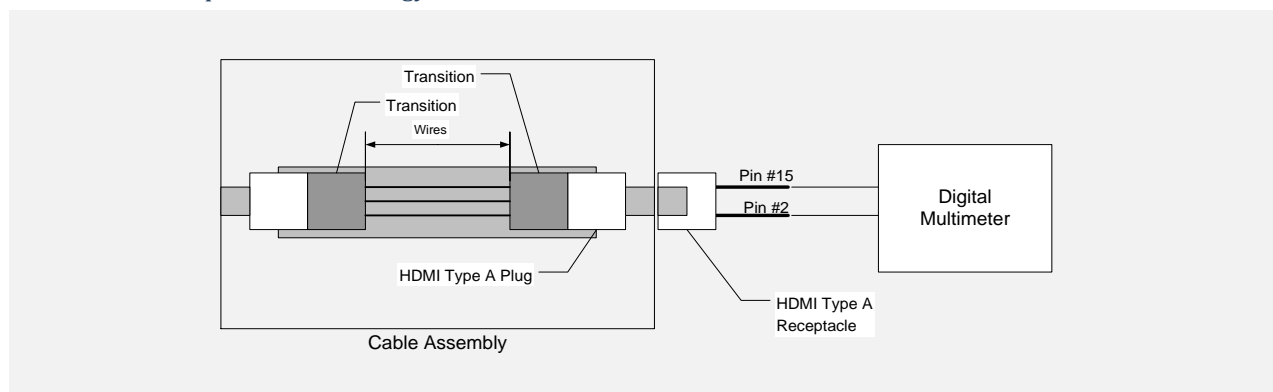


Figure 72. Cable Detect for MHL Cable with HDMI Type A Plug Test Setup

1. Connect Pin #2 and Pin #15 to the Digital Multimeter as shown in Figure 72.
2. Measure the resistance. This is $R_{\text{CABLE_DETECT}}$.
3. If $2.97K \text{ ohms} \leq R_{\text{CABLE_DETECT}} \leq 3.63K \text{ ohms}$, it is PASS. Otherwise FAIL.

7.2.1.16 Minimum CLK Swing Test

Application: Cable

7.2.1.16.1 Test Objective

This test confirms that the Min clock swing at the MHL cable output meets minimum common-mode input swing voltage requirement for Sink. If the MHL cable passes all the tests of 7.2.1.2, 7.2.1.3, 7.2.1.8, 7.2.1.9, and 7.2.1.10,

7.2.1.16.2 Reference

[MHL] Section 13.6.1 Table 13-7 : Min of Common-mode input swing voltage $V_{ICM} = 170 \text{ mV}$

7.2.1.16.3 Required Test Equipment

Source TPA-R-WO

7.2.1.16.4 Required Methodology

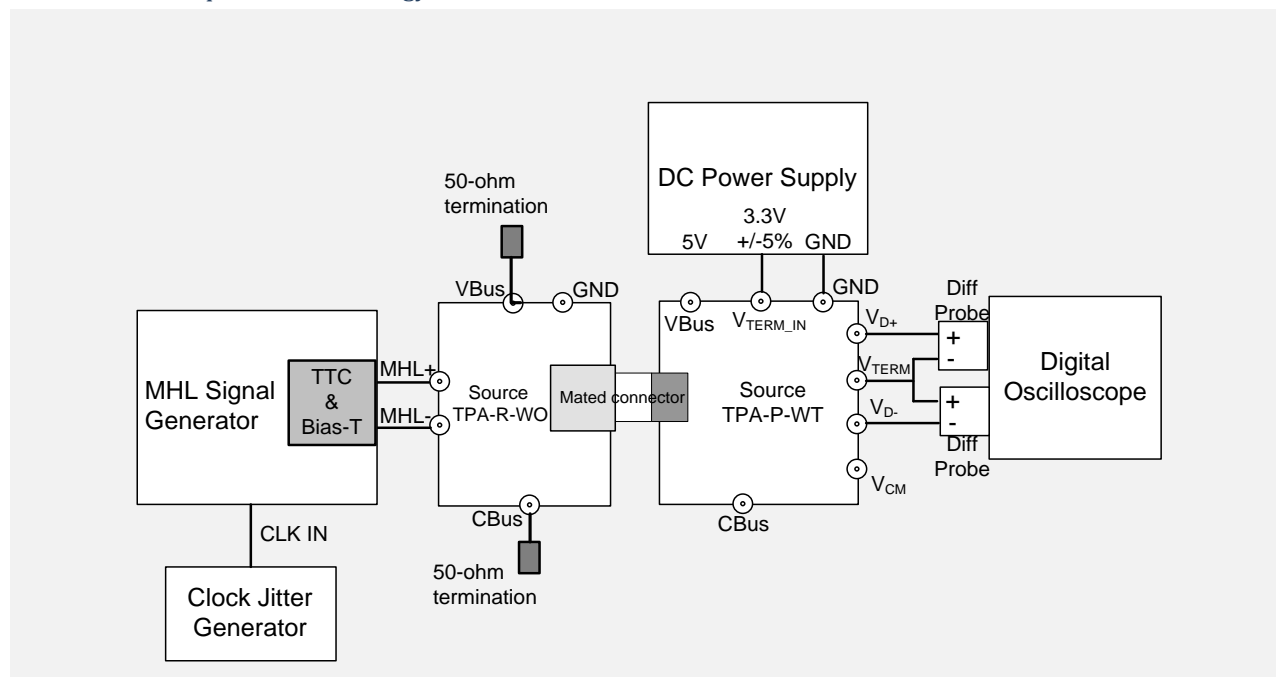


Figure 73 Single-Ended Signals Calibration Setup for Cable Tests

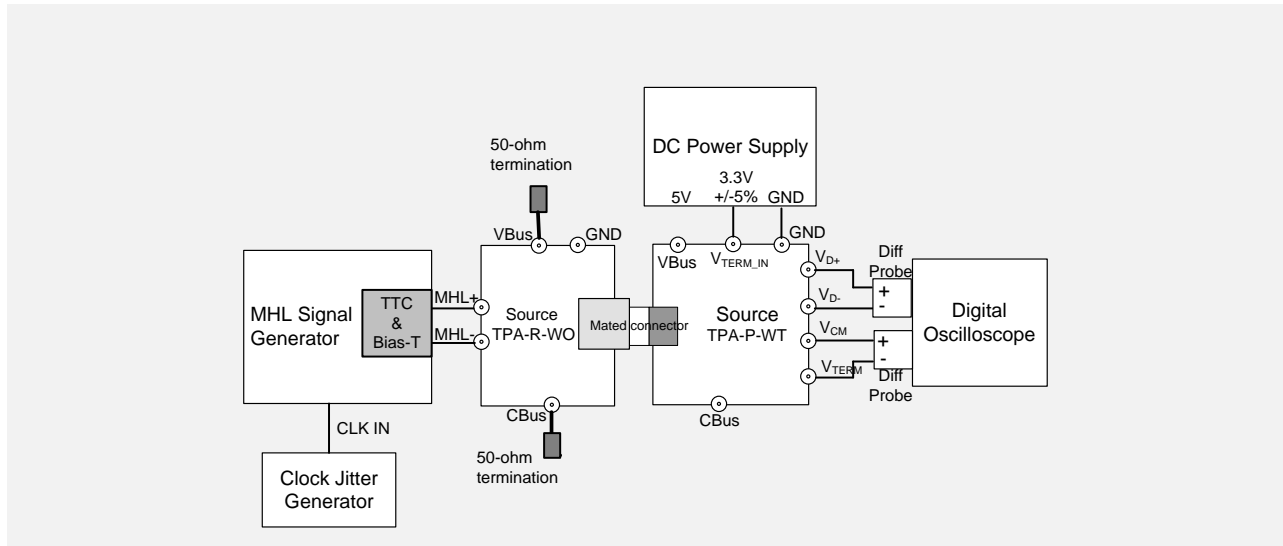


Figure 74 Differential & Common-Mode Signals Calibration Setup for Cable Tests

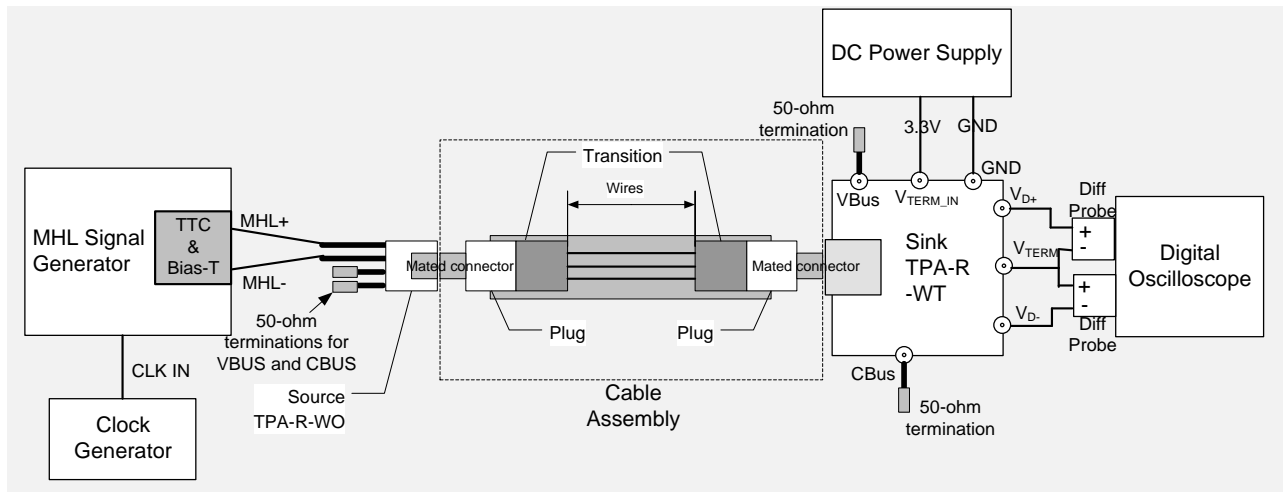


Figure 75. Cable – Minimum Clock Swing Test Setup

1. Connect the equipment as shown in Figure 73 for single-ended signals calibration and Figure 74 for differential and common-mode signals calibration. Set $V_{TERM} = 3.3V$.
2. Calibrate the Signal Generator output for the following settings:
 - a. Frequency: 2.97Gbps packed pixel data, 74.25MHz clock
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times: Common-mode signal 600ps (20% - 80%), Differential signal 100ps (20% - 80%)
 - d. Swing voltages: Common-mode signal 360mV, Differential signal 800mV
 - e. DC level of the output single-ended signal: $V_{TERM} - 540mV$
3. Set intra-pair skew: Differential skew +40ps, Common-mode skew +40ps
(Make the positive channel lead the negative channel in time, which is called positive skew direction.)
4. Apply the same amount of jitter for clock and data, and set the jitter frequency to 7MHz.
5. Calibrate the Signal Generator and Jitter Generator to create a data eye diagram which is marginal to TP1 specification.
 - a. Adjust the jitter to the TP1 maximum amount specified in MHL Spec Figures 13-15.

- b. Adjust the differential signal swing to create the marginal eye diagram specified in MHL Spec Figure 13-14.
6. Connect the equipment as shown in Figure 75.
7. Set the Digital Oscilloscope horizontal setting: sampling rate $\geq 20\text{Gsa/sec}$ and approximately 3 MHL clock cycles displayed on the screen.
8. Measure $V_{\text{CM_SWING}}$.
 - a. Display voltage histogram on the middle 20% of MHL Clock high period for the common-mode high level.
 - b. Capture 10,000 repetitions.
 - c. Record the histogram mode value for the common-mode high level.
 - d. Display voltage histogram on the middle 20% of MHL Clock low period for the common-mode low level.
 - e. Capture 10,000 repetitions.
 - f. Record the histogram mode value for the common-mode low level.
 - g. Calculate and record $V_{\text{CM_SWING}}$.
9. Connect the equipment as shown in Figure 74.
10. Set intra-pair skew: Differential skew -40ps, Common-mode skew -40ps.
11. Repeat steps 4 to 8 for negative skew direction.
12. If $170\text{mV} \leq V_{\text{CM_SWING}}$ for all the recorded values, then PASS; otherwise FAIL.

7.2.1.17 Eye Diagram Test

| |
|---------------------------|
| Application: Cable |
|---------------------------|

7.2.1.17.1 Test Objective

This test confirms that the MHL data output has signal quality that meets the eye opening required by the specification. If the MHL cable passes all the tests of 7.2.1.2, 7.2.1.3, 7.2.1.8, 7.2.1.9, and 7.2.1.10, then this test is optional and informative only. If the MHL cable fails one or more of 7.2.1.2, 7.2.1.3, 7.2.1.8, 7.2.1.9, and 7.2.1.10, then this test is mandatory.

7.2.1.17.2 Reference

[MHL] Figure 13-14 Differential Eye Diagram Mask at TP1

[MHL] Figure 13-15 MHL Data Jitter at TP1

[MHL] Figure 13-21 MHL Data Eye Mask at TP2

[MHL] Figure 13-22 MHL Data Jitter at TP2

7.2.1.17.3 Required Test Equipment

High-bandwidth Digital Oscilloscope

High-Bandwidth Probe

Transition Time Converter (TTC)

Bias Tee

MHL Signal Generator

CLK Jitter Generator

Jitter and Eye Analyzer

DC Power Supply

Sink TPA-R-WT

Source TPA-P-WT

Source TPA-R-WO

7.2.1.17.4 Required Methodology

1. Connect the equipment as shown in Figure 73 for single-ended signals calibration and Figure 74 for differential and common-mode signals calibration. Set $V_{\text{TERM}} = 3.3\text{V}$.
2. Calibrate the Signal Generator output for the following settings:
 - a. Frequency: 2.97Gbps PackedPixel data, 74.25MHz clock
 - b. Pattern: MHL Gray Ramp data streams
 - c. Rise/fall times: Common-mode signal 600ps (20% - 80%), Differential signal 100ps (20% - 80%)
 - d. Swing voltages: Common-mode signal 360mV, Differential signal 800mV
 - e. DC level of the output single-ended signal: $V_{\text{TERM}} - 540\text{mV}$
3. Set intra-pair skew: Differential skew +40ps, Common-mode skew +40ps.
(Make the positive channel lead the negative channel in time, which is called positive skew direction.)
4. Apply the same amount of jitter for clock and data and set the jitter frequency to 7MHz.
5. Calibrate the Signal Generator and Jitter Generator to create data eye diagram which is marginal to TP1 specification.
 - a. Adjust the jitter to the TP1 maximum amount specified in MHL Spec Figures 13-15.
 - b. Adjust the differential signal swing to create the marginal eye diagram specified in MHL Spec Figure 13-14.
6. Connect the equipment as shown in Figure 75.
7. Set the oscilloscope to the following condition:
 - a. Sampling rate $\geq 20\text{Gs/sec}$
 - b. Sampling length $\geq 0.5\text{msec}$
 - c. Low pass filtering for the MHL Clock signal: $f_{3\text{dB}} = 500\text{MHz}$
 - d. Software CRU for the MHL Clock signal: 1st order, 3dB Bandwidth = 4MHz
 - e. Histogram Window Vertical Width = 5mV
8. Measure and record the eye diagram with reference equalizer (equalized eye diagram).
9. Measure and record the eye diagram without reference equalizer (non-equalized eye diagram).
10. Connect the equipment as shown in Figure 74.
11. Set intra-pair skew: Differential skew -40ps, Common-mode skew -40ps.
12. Repeat steps 4 to 7 for negative skew direction.
13. If either the equalized or non-equalized eye diagram meets the MHL Data Eye diagram spec for each of positive and negative skew cases, then PASS; otherwise FAIL.

8 Appendices

8.1 Capability Declaration Form (CDF)

Each Adopter submits their system for compliance testing accompanied by a CDF. The CDF specifies the capabilities of the system, such as the list of supported video and audio modes.

The CDF uses fields defined in this CTS. Fields are listed in Table 8-1 with definitions. In all cases, definitions for these fields or features as written in the MHL Specification take precedence over the simplified definitions listed here.

Table 8-1. CDF Field Definitions

| CDF Field Name | CDF Field Definition |
|-------------------------|--|
| CDF_AUDIO_2CH_32kHz | Device supports 2-channel audio at 32KHz sample rate. |
| CDF_AUDIO_2CH_44kHz | Device supports 2-channel audio at 44.1KHz sample rate. |
| CDF_AUDIO_2CH_48kHz | Device supports 2-channel audio at 48KHz sample rate. |
| CDF_AUDIO_SAMPLE_SIZE | Define the sample size (in bits) of audio data. |
| CDF_AVI_SUPPORT | Device supports AVI InfoFrames. |
| CDF_CR_ADOPTER_ID_H | High-order byte of 16-bit MHL ADOPTER ID register. |
| CDF_CR_ADOPTER_ID_L | Low-order byte of 16-bit MHL ADOPTER ID register. |
| CDF_CR_AUD_2CH | Device supports 2-channel audio. |
| CDF_CR_AUD_8CH | Device supports 8-channel audio. |
| CDF_CR_BANDWIDTH | Define maximum TMDS bandwidth (refer to Specification). |
| CDF_CR_DEVICE_ID_H | High-order byte of 16-bit MHL DEVICE ID register. |
| CDF_CR_DEVICE_ID_L | Low-order byte of 16-bit MHL DEVICE ID register. |
| CDF_CR_INT_SIZE | Number of Interrupt Registers. |
| CDF_CR_LD_AUDIO | Logical Device Map: device supports audio handling. |
| CDF_CR_LD_DISPLAY | Logical Device Map: device supports display output. |
| CDF_CR_LD_GUI | Logical Device Map: device support graphical user interface. |
| CDF_CR_LD_MEDIA | Logical Device Map: device supports media handling. |
| CDF_CR_LD_RECORD | Logical Device Map: device supports recording. |
| CDF_CR_LD_SPEAKER | Logical Device Map: device supports audio output. |
| CDF_CR_LD_TUNER | Logical Device Map: device includes tuner control. |
| CDF_CR_LD_VIDEO | Logical Device Map: device supports video handling. |
| CDF_CR_MHL_VER_MAJOR | Major MHL Specification revision supported by device. |
| CDF_CR_MHL_VER_MINOR | Minor MHL Specification revision supported by device. |
| CDF_CR_MHL_VERSION | Combination of MHL_VER_MAJOR and MHL_VER_MINOR. |
| CDF_CR_POW | Define the supported power mode (refer to Specification). |
| CDF_CR_RAP_SUPPORT | Device supports RAP sub-commands. |
| CDF_CR_RCP_SUPPORT | Device supports RCP sub-commands. |
| CDF_CR_SCRATCHPAD_SIZE | Number of Scratchpad Registers for WRITE_BURST. |
| CDF_CR_SP_SUPPORT | Device supports WRITE_BURST (Scratchpad). |
| CDF_CR_STAT_SIZE | Number of Status Registers. |
| CDF_CR_SUPP_ISLANDS | Device supports data islands. |
| CDF_CR_SUPP_PPIXEL | Device supports PackedPixel mode. |
| CDF_CR_SUPP_RGB444 | Device supports RGB 4:4:4 color space. |
| CDF_CR_SUPP_VGA | Device supports VGA (640x480) video mode. |
| CDF_CR_SUPP_YCBCR422 | Device supports YcbCr 4:2:2 color space. |
| CDF_CR_SUPP_YCBCR444 | Device supports YcbCr 4:4:4 color space. |
| CDF_CR_UCP_RECV_SUPPORT | Device supports receiving UCP sub-commands. |

| CDF Field Name | CDF Field Definition |
|---------------------------------------|---|
| CDF_CR_UCP_SEND_SUPPORT | Device supports sending UCP sub-commands. |
| CDF_CR_VT_CINEMA | Device supports Cinema video type (refer to Specification). |
| CDF_CR_VT_GAME | Device supports Game video type (refer to Specification). |
| CDF_CR_VT_GRAPHICS | Device supports Graphics video type (refer to Specification). |
| CDF_CR_VT_PHOTO | Device supports Photo video type (refer to Specification). |
| CDF_D_ACCEPTS_POWER_FROM_SOURCE | Dongle device accepts VBUS power input from Source. |
| CDF_D_CM_CABLE_END_DISTANCE | Define common-mode length of Dongle's captive cable. |
| CDF_D_CM_CABLE_START_DISTANCE | Define common-mode length of Dongle's captive cable. |
| CDF_D_CM_TERM_DISTANCE | Define common-mode length of Dongle's termination. |
| CDF_D_DF_CABLE_END_DISTANCE | Define differential length of Dongle's captive cable. |
| CDF_D_DF_CABLE_START_DISTANCE | Define differential length of Dongle's captive cable. |
| CDF_D_DF_TERM_DISTANCE | Define differential length of Dongle's termination. |
| CDF_D_MAX_CBUS_CAP | Define maximum CBUS capacitance of Dongle. |
| CDF_D_MAX_POWER_DOWN | Define maximum wait time when powering down Dongle. |
| CDF_D_MAX_POWER_UP | Define maximum wait time when powering up Dongle. |
| CDF_D_MAX_STANDBY_TO_ACTIVE | Define maximum wait time from Standby to Active. |
| CDF_D_POWERED | Define if Dongle is powered or unpowered. |
| CDF_DDC_SEGMENT_READ_SUPPORT | Device supports DDC Segment Read commands. |
| CDF_HDCP_SUPPORT | Device supports HDCP. |
| CDF_HDCP_REPEATER | Device is an HDCP Repeater, internally decrypting and re-encrypting the content stream, as defined in the HDCP Specification. |
| CDF_LOG_DEV_MAP_CHANGE | Device allows Logical Device Map to change in real-time. |
| CDF_PROC_SET_ACTIVE | Define procedure to set Device into Active mode. |
| CDF_PROC_SET_STANDBY | Define procedure to set Device into Standby mode. |
| CDF_PROC_SOURCE_POWER_STANDBY | Define if Source Device can output VBUS in Standby. |
| CDF_RAP_SUPPORT | Device supports RAP sub-commands. |
| CDF_RCP_RECEIVE | Device supports receiving RCP sub-commands. |
| CDF_RCP_SEND | Device supports sending RCP sub-commands. |
| CDF_SINK_CABLE_DETECT_TO_R_DISCOVER | Define max wait time from Cable Detect to Z _{CBUS} check. |
| CDF_SINK_CM_TERM_DISTANCE | Define common-mode length of Sink termination. |
| CDF_SINK_DF_TERM_DISTANCE | Define differential length of Sink termination. |
| CDF_SINK_MAX_STANDBY_TO_ACTIVE | Define max wait time from Standby to Active. |
| CDF_SINK_WAKE_FROM_STANDBY | Define max wait time from power-off to Standby. |
| CDF_SOURCE_YQ_FULL | Device support full-range colors with YQ field. |
| CDF_SRC_POWERED | Device is powered. |
| CDF_SRC_PROC_DF_SIGNAL_ONLY | Define procedure to output differential signal only. |
| CDF_SRC_PROC_STANDBY_DISCOVERY | Define procedure to change from Standby to do discovery. |
| CDF_UCP_RECV_APPLICATION | Define application which supports UCP receiving. |
| CDF_UCP_RECV_SUPPORT | Device supports receiving UCP sub-commands. |
| CDF_UCP_SEND_APPLICATION | Define application which supports UCP sending. |
| CDF_UCP_SEND_SUPPORT | Device supports sending UCP sub-commands. |
| CDF_VIDEO_1080i_50 | Device supports 1080i/50Hz video mode. |
| CDF_VIDEO_1080i_60 | Device supports 1080i/60Hz video mode. |
| CDF_VIDEO_1280x720P_50_3D_Frame | Device supports 720p 3D / 50Hz Frame Sequential mode. |
| CDF_VIDEO_1280x720P_50_3D_Top_Bottom | Device supports 720p 3D / 50Hz Top-Bottom mode. |
| CDF_VIDEO_1280x720P_60_3D_Frame | Device supports 720p 3D / 60Hz Frame Sequential mode. |
| CDF_VIDEO_1280x720P_60_3D_Top_Bottom | Device supports 720p 3D / 60Hz Top-Bottom mode. |
| CDF_VIDEO_1920x1080i_50_3D_Left_Right | Device supports 1080i 3D / 50Hz Left-Right mode. |

| CDF Field Name | CDF Field Definition |
|---------------------------------------|--|
| CDF_VIDEO_1920x1080i_60_3D_Left_Right | Device supports 1080i 3D / 60Hz Left-Right mode. |
| CDF_VIDEO_1920x1080p_24_3D_Frame | Device supports 1080p 3D / 24Hz Frame Sequential mode. |
| CDF_VIDEO_1920x1080p_24_Top_Bottom | Device supports 1080p 3D / 24Hz Top-Bottom mode. |
| CDF_VIDEO_3D | Device supports 3D video. |
| CDF_VIDEO_480i_60_2X | Device supports 480i / 60Hz (2x replication) video. |
| CDF_VIDEO_480i_60_4X | Device supports 480i / 60Hz (4x replication) video. |
| CDF_VIDEO_480p_60 | Device supports 480p / 60Hz video. |
| CDF_VIDEO_480p_60_2X | Device supports 480p / 60Hz (2x replication) video. |
| CDF_VIDEO_576i_50_2X | Device supports 576i / 50Hz (2x replication) video. |
| CDF_VIDEO_576i_50_4X | Device supports 576i / 50Hz (4x replication) video. |
| CDF_VIDEO_576p_50 | Device supports 576p / 50Hz video. |
| CDF_VIDEO_576p_50_2X | Device supports 576p / 50Hz (2x replication) video. |
| CDF_VIDEO_576p_50_4X | Device supports 576p / 50Hz (4x replication) video. |
| CDF_VIDEO_720p_50 | Device supports 720p / 50Hz video. |
| CDF_VIDEO_720p_60 | Device supports 720p / 60Hz video. |
| CDF_VIDEO_PACKEDPIXEL | Device supports PackedPixel mode. |
| CDF_VIDEO_RGB | Device supports RGB color space. |
| CDF_VIDEO_VGA | Device supports VGA video mode. |
| CDF_VIDEO_YCBCR_422 | Device supports YCbCr 4:2:2 color space. |
| CDF_VIDEO_YCBCR_444 | Device supports YCbCr 4:4:4 color space. |
| CDF_VIDEO_YCC_FULL | Device supports full-range YCbCr. |

8.2 Test Results Form (TRF)

The Test Results Form is created by an authorized test center (ATC) while performing the tests from this CTS on the device under test (DUT).

END_OF_DOCUMENT

DOCUMENT REVISION HISTORY

Information in the following change history is intended for use by the author and developers of this document, and not for general distribution to MHL adopters.

Do not use any header formats in this section, so that no entries from these pages appear in the document's Table of Contents.

Entries with multiple pages indicate multiple edited points in the document. When a specific page number appears more than once in an entry, it indicates multiple edits on that page.

If a bookmark cross-reference appears with the words **Error! Bookmark not defined.**, or a heading cross-reference appears with the words **Error! Reference source not found.**, it means a link is unresolved in Word. Change the link to a proper target, or re-code it as plain text. Note that page numbers which are not actual cross-reference links to targets may be out of step with revisions to the document, and may not point to the exact page of the change.

| Rev | Date | Author | Page(s) | Change |
|-------------|------------|--------|---------|--|
| 2.1 fc01 | 2012-10-22 | WA | 2, 30 | Change version to 2.1; delete prior change history; add entries to change summary. |
| | | | | Replace all uses of CDF_SRC_CBUS_THRESHOLD_V, CDF_SINK_CBUS_THRESHOLD_V, and CDF_D_CBUS_THRESHOLD_V with "measure at 50%...". Per 5CTT, 10/19. |
| | | | | Remove CDF_SRC_CBUS_THRESHOLD_V, CDF_SINK_CBUS_THRESHOLD_V, and CDF_D_CBUS_THRESHOLD_V from CDF field listing. Per 5CTT, 10/19. |
| | 2012-10-23 | WA | 15 | Insert warning note "no changes to configuration" allowed. Per 5CTT, 10/15. |
| | | | 15 | Insert warning note "care taken to clear nearby devices..." Per 5CTT, 10/15. |
| | 2012-10-24 | WA | 390 | Remove CDF_VIDEO_1920x1080i_24_Top_Bottom from appendix listing. Per Sony//Tao, 10/24. |
| | 2012-11-30 | WA | 180 | Fix typo in refs in 4.3.17.1. Per SIMG/LB, 11/21.{BGZ#24340} |
| | | | 181 | Fix typo in step #10 in 4.3.17.1. Per SIMG/LB, 11/21.{BGZ#24340} |

| Rev | Date | Author | Page(s) | Change |
|-------------|---|--------|--------------------------------|--|
| 2.1 fc01 | 2012-12-18 | WA | 91 | Replace VOH steps #11-13 in 3.3.7.5. Per SIMG/LB, 12/17.{BGZ#25319} |
| | | | 155 | Replace VOH step #11 in 4.3.6.4. Per SIMG/LB, 12/17.{BGZ#25319} |
| | | | 232 | Replace VOH step #11 in 5.3.6.4. Per SIMG/LB, 12/17.{BGZ#25319} |
| | | | 83 | Replace Req'd Meth in 3.3.5.3. Per SIMG/LB, 11/21.{BGZ#25906} |
| | 2012-12-19 | WA | 230 | Remove unnecessary step #20. Per SIMG/LB, 12/19.{BGZ#25884} |
| | | | 92 | Add step to make DUT drive HIGH time observable in 3.3.8.1. Per SIMG/LB, 12/19.{BGZ#26407} |
| | | | 98 | Add step to make DUT drive HIGH time observable in 3.3.11.2. Per SIMG/LB, 12/19.{BGZ#26407} |
| | | | 156 | Add step to make DUT drive HIGH time observable in 4.3.7.1. Per SIMG/LB, 12/19.{BGZ#26407} |
| | | | 163 | Add step to make DUT drive HIGH time observable in 4.3.10.2. Per SIMG/LB, 12/19.{BGZ#26407} |
| | | | 233 | Add step to make DUT drive HIGH time observable in 5.3.7.1. Per SIMG/LB, 12/19.{BGZ#26407} |
| | | | 239 | Add step to make DUT drive HIGH time observable in 5.3.10.2. Per SIMG/LB, 12/19.{BGZ#26407} |
| | 2012-12-20 | WA | 70 | Reverse the order of steps 3 & 4 in 3.2.8.1. Per SIMG/LB, 12/18.{BGZ#26520} |
| 2.1 fc02 | 2013-01-28 | WA | 23 | Add new TPA boards for Direct Attach to Table 2-1, with note. Per SIMG/BS, 1/15. |
| | | | | Replace "Source TPA-P-WT" with "Source TPA-P-WT or DirSource TPA-R-WT" throughout Section 3. |
| | | | 31, 33, 34, 35, 36, 38, 39, 41 | Change figures 17, 18, 19, 20, 21, 22, 23, 25 to add DirSource TPA-R-WT. |
| | | | 40, 40 | Obsolete tests 3.1.1.8, 3.1.1.9, replace with 3.1.1.17 thru 20. |
| | Above changes from " CTS-2_1_Main-based_on_2_0_fc06-2012-09-11-1700-w_bgz_rev08_BSung_Review.docm", SIMG/B. Sung, 1/15. | | | |

| Rev | Date | Author | Page(s) | Change |
|-------------|---|--------|------------------------------|---|
| 2.1 fc02 | 2013-01-28 | WA | 42 | Test 3.1.1.11 is replaced by 3.1.1.17. |
| | | | 42 | Test 3.1.1.12 is replaced by test 3.1.1.18. |
| | | | 45 | Test 3.1.1.15 is replaced by test 3.1.1.19.. |
| | | | 45 | Test 3.1.1.16 is replaced by test 3.1.1.20. |
| | | | 46-50 | Insert new tests 3.1.1.17-3.1.1.20. |
| | | | 118, 118, 119, 121, 122, 124 | Change figures 30, 31, 32, 33, 34, 35 to add DirSink TPA-R-WO. |
| | | | | Replace "Sink TPA-P-WO" with " Sink TPA-P-WO or DirSink TPA-R-WO" throughout Section 4. |
| | | | 123, 131 | Add Source TPA-P-WT to equipment in 4.1.1.4, 4.1.1.8. |
| | 2013-02-05 | WA | 125 | Add 2 steps to 4.1.1.4. |
| | | | 125, 128 | Add DirSink TPA to Figs 34, 36. |
| | | | 126, 126 | Add step #7, 9-16 in 4.1.1.5. |
| | | | 129, 129 | Add step #7, 9-16 in 4.1.1.5. |
| | | | 132 | Add 2 steps to 4.1.1.8. |
| | | | 201 | Add step to 5.1.1.6. |
| | | | 369, 372 | Distinguish References in 7.2.1.4, 7.2.1.5 by length. |
| | | | 370 | Add fig to 7.2.1.4 for cable length. |
| | | | 371, 373 | Divide 7.2.1.4, 7.2.1.5 to 2 methods by cable length. |
| | | | 380 | Update References section in 7.2.1.12. |
| | | | 384 | Add 7.2.1.16 Minimum CLK Swing Test for cables. |
| | | | 388 | Add 7.2.1.17 Eye Diagram test for cables. |
| | Above changes from " CTS-2_1_Main-based_on_2_0_fc06-2012-09-11-1700-w_bgz_rev08_BSung_Review.docm", SIMG/B. Sung, 1/15. | | | |

| Rev | Date | Author | Page(s) | Change |
|--|------------|--------|---------------|--|
| 2.1 fc02 | 2013-02-05 | WA | 19 | Add figure with photo of MHL Test Cable (20cm). |
| | | | 20 | Add 2 figures, with Dir. Att. Source and Sink Cables. |
| | | | 75 | Add note #4 to CBUS Test intro, re: Dir. Att. devices. |
| | | | 75 | Add note #5 to CBUS Test intro re: slow-booting devices. |
| | | | 84, 84, 85 | Fix step #5 to allow new devices, with CDF_ field, in 3.3.5.3, 3.3.5.4, 3.3.5.5. Per SIMG/ES, 1/10.{BGZ#26423} |
| | | | 88 | Clarify step in 3.3.6.4. |
| | | | 116 | Add 3.3.23.4 for Prediscovery Current test. |
| | | | 116 | Add 3.3.23.5 for Postdiscovery Current test. |
| | | | 148 | Add clarifying note in 4.3 intro for DC-pow Sinks. |
| | | | 148 | Add clarifying note in 4.3 intro for Dir. Att Sinks. |
| | | | 189, 190 | Add step #1 to skip for unpowered Sink in 4.3.24.1, 4.3.25.1. |
| | | | 189 | Change test steps in 4.3.24.1 for DC-pow Sink. |
| | | | 268 | Add steps to 5.3.24.1 for DC-pow Dongle. |
| <i>Above changes from file "CTS-2_1_Main-fc07-2012-10-04-w_bgz_ESEIM.docx", SIMG/E.Seim, 1/10.</i> | | | | |
| | 2013-02-05 | WA | 155, 232 | Add note re: overshoot measurement in 4.3.6.4, 5.3.6.4. Per SIMG/ES, 2/05.{BGZ#26867} |
| | | | 229 | Clarify step #8 in 5.3.5.1. Per SIMG/ES, 2/05. |
| <i>Above changes from file "CTS-2_1_Main-fc01-2012-12-19-w_bgz_ESEIM.docx", SIMG/BSung, 2/05.</i> | | | | |

| Rev | Date | Author | Page(s) | Change |
|--|------------|--------|---------------|---|
| 2.1 fc02 | 2013-02-05 | WA | 51, 136 | Change 'or' to 'and' in step #1 of 3.2.2.1, 4.2.2.1. |
| | | | 68 | Mark step in 3.2.6.4 for continuous monitor. |
| | | | 68 | Move PATH_EN=1 step in 3.2.6.4. |
| | | | 69 | Add step #2 to 3.2.8.1.{BGZ#25904} |
| | | | 72 | Iterate 3.2.9.3 on video modes,{BGZ#26541} |
| | | | 133 | Clarify video support needed in 4.2.1.1.{BGZ#25011}{BGZ#26643} |
| | | | 133 | Clarify video modes in steps in 4.2.1.1. |
| | | | 134, 208 | Allow skip of 60Hz mode in steps in 4.2.1.2, 5.2.1.2. |
| | | | 135, 210 | Add 1080p50 to valid video modes for 4.2.1.4, 5.2.1.4. |
| | | | 135 | Clarify test title "in Normal Mode" in 4.2.2.1. |
| | | | 137 | Add 4.2.2.4 "Video Formats in PackedPixel Mode".{BGZ#25868} |
| | | | 143, 143 | Enhance 4.2.6.1, 4.2.6.2 to include RCP with and without PRESS. |
| | | | 145, 219 | Clarify list of video modes checked in 4.2.8.1, 5.2.8.1.{BGZ#26377} |
| | | | 207, 208, 214 | Add 'downstream display' to eqpt for 5.2.1.1, 5.2.1.2, 5.2.5.1. |
| | | | 215 | Clarify step #1 in 5.2.5.1. |
| | | | 220 | Add 3D Video Format in PP Mode test 5.2.8.3.{BGZ#26875} |
| Above changes from file "CTS-2_1_Main-fc01-2012-12-19-w_bgz_VJF Edits_01_11_13.docx", SIMG/V.F., 1/27. | | | | |

| Rev | Date | Author | Page(s) | Change |
|---|------------|--------|---|---|
| 2.1 fc02 | 2013-02-05 | WA | 297, 298 | Fix typo in Objective in 6.3.10.4, 6.3.10.5. Per SIMG/LB, 1/10.{BGZ#26712} |
| | 2013-02-06 | WA | 114 | Make 3.3.23.1 obsolete and point to 3.3.23.5. Per SIMG/ES, 2/06.{BGZ#25019} |
| | | | 91 | Add note about measuring with overshoots to 3.3.7.5. Per SIMG/ES, 2/06.{BGZ#26867} |
| | | | 114 | Add 3.3.23.4 reference to note in 3.3.23.1. Per SIMG/ES, 2/06.{BGZ#25019} |
| | | | 277, 287, 288, 289, 290, 292 | Clarify step in 6.3.3.5, 6.3.6.2, 6.3.6.3, 6.3.6.4, 6.3.6.5, 6.3.7.1 for DCAP_RDY and DCAP_CHG. Per SIMG/ES, 2/06.{BGZ#26707} |
| | | | 325, 325, 326, 327, 328, 329, 329, 330, 331 | Change wait time in step in 6.3.12.1, 6.3.12.2, 6.3.12.3, 6.3.12.4, 6.3.12.5, 6.3.12.6, 6.3.12.7, 6.3.12.8, 6.3.12.9. Per SIMG/ES, 2/04.{BGZ#26936} |
| | | | 227, 228 | Change 5.3.4.2 to work with Dongle which cannot accept Source power. Per SIMG/ES, 2/04.{BGZ#26939} |
| | | | 117 | Re-word step #10 in 3.3.23.5. Per SIMG/ES, 2/07.{BGZ#26707} |
| | | | 27 | Add note in Section 2 re: 3D patterns. Per SIMG/MS, 1/28.{BGZ#24922} |
| | | | 72 | Clarify step to use 3D PackedPixel pattern in 3.2.9.3. Per SIMG/MS, 1/28.{BGZ#24922} |
| | 2013-02-08 | WA | 46, 46, 48, 49, 49, 50 | Clarify eqpt step in 3.1.1.17, 3.1.1.18, 3.1.1.19, 3.1.1.20. Per SIMG/BSu, 2/08. |
| | | | 2 | Change DOCPROPERTY{Release Date} to "April 2013". |
| | | | all | Change copyright date from 2012 to 2013. |
| | | | 189 | Modify many steps in Req'd Meth in 4.3.24.1 to allow for DC-powered Sinks. Per SIMG/ES, 2/08. |
| | | | 144, 218 | Rearrange steps in 4.2.7.1, 5.2.7.1. Per SIMG/VF, 2/08.{BGZ#26520} |
| | | | | Modify step #1 in 4.3.24.1 to include CDF_SINK_DC field. Per SIMG/ES, 2/11.{BGZ#26988} |
| This document was delivered to SIMG Engineering for review on 2013-02-08. | | | | |

| Rev | Date | Author | Page(s) | Change |
|--|------------|--------|-------------------------------|--|
| 2.1 fc02 | 2013-02-12 | WA | 189 | Modify step #1 in 4.3.24.1 to include CDF_SINK_DC field. Per SIMG/ES, 2/11.{BGZ#26988} |
| | | | 40, 40 | Re-word objective in 3.1.1.8, 3.1.1.9 to point to new tests. Per SIMG/BSu, 2/12. |
| | | | 368, 369, 377, 378, 379 | Insert missing step in 7.2.1.2, 7.2.1.3, 7.2.1.8, 7.2.1.9, 7.2.1.10. Per SIMG/BSu, 2/12. |
| | | | 384 | Replace tests 7.2.1.16 and 7.2.1.17 with new write-ups and figures. Per SIMG/BSu, 2/08. |
| This document was delivered to 5CTT for review on _____. | | | | |

__END_OF_CHANGE_HISTORY__
