MHL™
(Mobile High-definition Link)

Specification

Confidential information of:

Nokia Corporation,

Samsung Electronics Corporation,

Silicon Image, Inc.,

Sony Corporation,

Toshiba Corporation,

each, an "MHL Promoter" and collectively, the "MHL Promoters".

Version 2.1

MHL, LLC
March 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.

Document Revision History

2010-06	Distributed Revision 1.0.
2011-03	Distributed Revision 1.1.
2011-12	Distributed Revision 1.2.
2012-02	Distributed Revision 2.0.
2013-03	Distributed Revision 2.1.

Please refer also to the Revision Highlights section, beginning on page 14.

Table of Contents

1 INTRODUCTION 13

2 DEFINITIONS 16

3 OVERVIEW 23

3.1 MHL 2.0 ENHANCEMENTS 24

4 SIGNALING AND ENCODING 24

4.1 OVERVIEW 24

4.1.1 Link Architecture 24
4.1.2 Physical and Logical Channels 26
4.1.3 Operating Modes Overview 28
4.1.4 Character Synchronization 30
4.1.5 Data Island Error Correction 30
4.1.6 Data Island Placement and Duration 31

4.2 OPERATING MODES 31

4.2.1 24-Bit-per-Pixel Mode 31
4.2.2 PackedPixel Mode 36

4.3 DATA ISLAND PACKET DEFINITIONS 40

4.3.1 Packet Header 40
4.3.2 Null Packet 41
4.3.3 Audio Clock Regeneration Packet 42
4.3.4 Audio Sample Packet 43
4.3.5 Content Mute Packet 45
4.3.6 InfoFrame Packet 47

4.4 ENCODING 48

4.4.1 Lane Multiplexing and De-multiplexing 48
4.4.2 Serialization 48
4.4.3 Control Period Coding 48
4.4.4 TERC4 Coding 49
4.4.5 Video Data Coding 50

4.5 DVI COMPATIBILITY 54

5 VIDEO 54

5.1 OVERVIEW 54

5.2 VIDEO FORMATS 54

5.2.1 Format Support Requirements 54
5.2.2 Video Control Signals: HSYNC, VSYNC 56
5.2.3 24-Bit Pixel Encoding Requirement 57
5.2.4 PackedPixel Encoding Requirement 57
5.2.5 Vendor-Defined DTV Format 57

5.3 VIDEO FORMAT SPECIFICATIONS 58

5.4 PIXEL-REPETITION 58

5.5 PIXEL ENCODINGS 58

5.5.1 RGB 4:4:4 Pixel Encoding 58
5.5.2 YCbCr 4:2:2 Pixel Encoding 59
5.5.3 YCbCr 4:4:4 Pixel Encoding 59
5.5.4 Pixel-Doubling Pixel Encoding 60
5.5.5 PackedPixel Encoding 61

5.6 VIDEO QUANTIZATION RANGES 61

5.7 COLORIMETRY 61

5.7.1 Standard-definition Formats 61

5.7.2 High-definition Formats 62
5.7.3 Additional Color Space 62
5.8 REQUIRED VIDEO PATTERNS 62
5.9 3D VIDEO 63
5.9.1 3D Video Support Detection 63
5.9.2 Active 3D Video Mode Indicator 67
5.9.3 3D Video Timings 68
5.9.4 3D Video Support Requirements 72

6 AUDIO 74
6.1 RELATIONSHIP WITH IEC 60958 / IEC 61937 74
6.2 AUDIO SAMPLE CLOCK CAPTURE AND REGENERATION 74
6.2.1 N parameter 75
6.2.2 CTS parameter 76
6.2.3 Recommended N and Expected CTS Values 76
6.3 AUDIO SAMPLE RATES AND SUPPORT REQUIREMENTS 78
6.3.1 Video Dependency 79
6.4 CHARACTERISTICS INDICATION 79
6.5 CHANNEL / SPEAKER ASSIGNMENT 79
6.6 AUDIO, VIDEO SYNCHRONIZATION 79
6.7 AUDIO DATA PACKETIZATION 80
6.7.1 Sample Present Bits in Layout 0 Audio Sample Packets 80
6.7.2 Sample Present Bits in Layout 1 Audio Sample Packets 80
6.8 PACKET DELIVERY RULES 81
6.8.1 Audio Sample Packets 81
6.8.2 Audio Clock Regeneration Packets 82
6.9 REQUIRED AUDIO PATTERNS 82

7 LINK CONTROL BUS 83
7.1 LINK CONTROL BUS OVERVIEW 83
7.2 LINK LAYER 84
7.2.1 Bi-phase mark encoding 84
7.2.2 Bus Driver States 84
7.2.3 Packet structure 85
7.2.4 Arbitration 87
7.2.5 Flow Control 91
7.2.6 Link Control Bus Timing 91
7.3 TRANSLATION LAYER 91
7.3.1 Translation-Layer Flow Control 92
7.3.2 Command Description Methodology 94
7.4 MSC COMMANDS 96
7.4.1 MSC Command Reactions 97
7.4.2 Support Requirements for MSC Commands 98
7.4.3 MHL Sideband Channel (MSC) Commands 98
7.5 DDC COMMANDS 110
7.5.1 Support Requirements for DDC Commands 111
7.5.2 DDC Command Reactions 111
7.5.3 DDC Channel Commands 112
7.6 REQUEST ACTION PROTOCOL SUB-COMMANDS 119
7.6.1 RAP Sub-Command 119
7.6.2 RAPK Sub-Command 120
7.6.3 MSC Request Action Protocol Support Requirements 120
7.7 REMOTE CONTROL PROTOCOL SUB-COMMANDS 120

7.7.1 RCP Sub-Command ... 121
7.7.2 RCPK Sub-Command ... 122
7.7.3 RCPE Sub-Command ... 122
7.7.4 MSC Remote Control Protocol Key Codes ... 123
7.7.5 RCP Key Code Behavior Definitions ... 124
7.7.6 RCP Press and Hold ... 128
7.7.7 Media Selection Mode ... 130
7.7.8 MSC Remote Control Protocol Support Requirements ... 131
7.8 UTF-8 CHARACTER PROTOCOL SUB-COMMANDS ... 137
7.8.1 UCP Sub-Command ... 137
7.8.2 UCPK Sub-Command ... 138
7.8.3 UCPE Sub-Command ... 138
7.8.4 MSC UTF-8 Character Protocol Support Requirements ... 139
7.8.5 MSC UTF-8 Character Sequence Example ... 139
7.9 DEVICE REGISTER SPACE ... 140
7.9.1 Device Capability Registers ... 141
7.9.2 Device Interrupt Registers ... 151
7.9.3 Device Status Registers ... 155
7.9.4 Device Scratchpad Registers ... 158
7.10 BURST_ID ... 158
7.10.1 BURST_ID Codes ... 158
7.10.2 MHL Support Requirements for BURST_ID ... 159

8 DEVICE DISCOVERY ... 159
8.1 DISCOVERY AND DISCONNECTION MECHANISMS ... 160
8.1.1 CBUS Impedance ... 160
8.1.2 Receiver Sense (RxSense) ... 160
8.1.3 Hot Plug Detect (HPD) ... 161
8.1.4 MHL Cable Discovery ... 161
8.2 DISCOVERY AND DISCONNECT STATE DIAGRAMS ... 161
8.2.1 Source Viewpoint ... 162
8.2.2 Sink Viewpoint ... 169
8.2.3 Dongle Viewpoint ... 178
8.3 DISCOVERY TIMELINES ... 182
8.3.1 Source Successfully Discovers Powered Sink ... 182
8.3.2 Source Successfully Discovers Unpowered Dongle ... 183
8.3.3 Source Plugs into Standby Sink ... 184
8.4 ELECTRICAL DISCOVERY SEQUENCE ... 185
8.4.1 CBUS Impedance Recognition ... 185
8.4.2 Wake Pulse Sequence ... 186
8.4.3 CBUS Drive-and-Float "Discovery Pulse" Counting ... 187
8.4.4 MHL+ and MHL- Termination ... 189
8.5 DATA EXCHANGE DURING DISCOVERY SEQUENCE ... 190
8.5.1 CBUS First Packet Arbitration ... 190
8.5.2 Exchange Device Capability Registers ... 190
8.5.3 Exchange Status Registers ... 191
8.5.4 Read Sink EDID ... 191
8.6 DISCONNECTION SEQUENCE ... 191
8.6.1 CBUS Goes Low ... 191
8.6.2 RxSense Goes Low ... 191
8.6.3 Informative: Coordinating Content Flow to the Screen ... 192
8.7 SHARING MHL CONNECTOR PINS ... 198

9 POWER PROVISIONING ... 198
9.1 SINK AND DONGLE POWER OUTPUT ... 198

9.2 SOURCE POWER OUTPUT 198
9.3 EXTERNAL POWER TO MHL SOURCE VIA DEDICATED CHARGING DEVICE 199
9.4 POWER CONFIGURATIONS 200

10 CONTROL AND CONFIGURATION 201
10.1 OVERVIEW 201
10.2 EIA/CEA-861E INFOFRAMES 201
10.2.1 AVI InfoFrame 201
10.2.2 Audio InfoFrame 204
10.3 EDID ACCESS 205
10.3.1 Data Transfer Protocol 205
10.3.2 Segment Pointer 205
10.3.3 Enhanced EDID Compliant Devices 206
10.4 EDID DATA STRUCTURES 207
10.4.1 Overview 207
10.4.2 EDID Timing Extension 207

11 CONTENT PROTECTION 208
11.1 RECOMMENDATION 208
11.2 HDCP IMPLEMENTATIONS 208
11.3 HDCP COMPLIANCE 208
11.3.1 Data Encryption 208
11.3.2 Stream Mapping 208
11.3.3 Hot Plug Detect 209

12 PHYSICAL LAYER 210
12.1 PURPOSE AND SCOPE 210
12.2 HDMI TYPE A CONNECTOR 210
12.2.1 Normative References 210
12.2.2 Contact Assignments 210
12.2.3 Use of the HDMI Type A Connector 211
12.2.4 HDMI Type A Cable Detect Mechanism 212
12.3 MICRO USB CONNECTOR 214
12.3.1 Normative References 214
12.3.2 Contact Assignments 214
12.3.3 Use of the Micro USB Connector 214

13 ELECTRICAL SPECIFICATION 216
13.1 MHL OVERVIEW 216
13.2 OPERATING CONDITIONS 218
13.3 ELECTRICAL SPECIFICATION METHODOLOGY 218
13.3.1 MHL Measurement Methods 218
13.4 TMDS PROTOCOL TIMING SPECIFICATIONS 225
13.5 SOURCE DEVICE ELECTRICAL AND TIMING SPECS 226
13.5.1 Source Device DC Characteristics at TP1 226
13.5.2 Source Device AC Characteristics at TP1 227
13.5.3 Source Device Eye Diagram Characteristics at TP1 229
13.5.4 Source Device TP2 Clock Jitter and Eye Diagram 230
13.6 SINK DEVICE ELECTRICAL AND TIMING SPECS 232
13.6.1 Sink Device DC Characteristics at TP2 232
13.6.2 Sink Device AC Characteristics at TP2 232
13.6.3 Sink Device Impedance Characteristics at TP2 233
13.6.4 Sink Device Pixel Error Rate Characteristics 234

13.7 DONGLE DEVICE ELECTRICAL AND TIMING SPECS 235
13.7.1 Dongle Device DC Characteristics at TP3 ..... *235*
13.7.2 Dongle Device AC Characteristics at TP3 ..... *236*
13.7.3 Dongle Device Impedance Characteristics at TP3 ..... *237*
13.7.4 Dongle Device Pixel Error Rate Characteristics *237*
13.8 CABLE ELECTRICAL AND TIMING SPECS 238
13.8.1 Cable DC Electrical Specifications *239*
13.8.2 Cable AC Electrical Specification *241*
13.8.3 Cable Power Signal (VBUS) *249*
13.9 +5V POWER SIGNAL (VBUS) 249
13.10 CBUS ELECTRICAL AND TIMING SPECIFICATIONS 251
13.10.1 CBUS Signaling *251*
13.10.2 CBUS Link Layer Timing Specifications *260*
13.10.3 CBUS Translation Layer Timing Specifications *261*
13.11 DATA ISLAND TIMING SPECIFICATIONS 263
13.11.1 Audio InfoFrame Timing Specifications *263*
13.12 ROBUSTNESS REQUIREMENTS 264

14 INFORMATIVE APPENDICES 264
14.1 PHYSICAL DEVICES AND LOGICAL DEVICES 264
14.1.1 DVD Player with One Disk *264*

List of Figures

FIGURE 3-1. MHL SOURCE AND SINK BLOCK DIAGRAM 23
FIGURE 4-1. MHL TRANSMITTER AND RECEIVER BLOCK DIAGRAM 25
FIGURE 4-2. MHL ENCODER/DECODER OVERVIEW – 24-BIT-PER-PIXEL MODE 26
FIGURE 4-3. MHL ENCODER/DECODER OVERVIEW – PACKEDPIXEL MODE 27
FIGURE 4-4. INFORMATIVE EXAMPLE: TMDS PERIODS IN 480P VIDEO FRAME 28
FIGURE 4-5. ERROR CORRECTION CODE GENERATOR 30
FIGURE 4-6. 24-BIT-PER-PIXEL MODE – PREAMBLE AND DATA ISLAND CHANNEL BIT ALLOCATION 32
FIGURE 4-7. 24-BIT-PER-PIXEL MODE – DATA ISLAND PACKET AND ECC STRUCTURE 35
FIGURE 4-8. PACKEDPIXEL MODE – PREAMBLE AND DATA ISLAND CHANNEL BIT ALLOCATION 36
FIGURE 4-9. PACKEDPIXEL MODE – DATA ISLAND PACKET AND ECC STRUCTURE 39
FIGURE 4-10. 24-BIT-PER-PIXEL MODE MHL LANE MULTIPLEXING 48
FIGURE 4-11. PACKEDPIXEL MODE MHL LANE MULTIPLEXING 48
FIGURE 4-12. TMDS VIDEO DATA ENCODE ALGORITHM 52
FIGURE 4-13. TMDS VIDEO DECODE ALGORITHM 53
FIGURE 5-1. DEFAULT PIXEL ENCODING: RGB 4:4:4, 8 BITS/COMPONENT 58
FIGURE 5-2. YCBCR 4:2:2 MAPPING 59
FIGURE 5-3. 8-BIT YCBCR 4:4:4 MAPPING 59
FIGURE 5-4. RGB WITH PIXEL-DOUBLING 60
FIGURE 5-5. YCBCR 4:2:2 WITH PIXEL-DOUBLING 60
FIGURE 5-6. YCBCR 4:4:4 WITH PIXEL-DOUBLING 60
FIGURE 5-7. PACKEDPIXEL ENCODING 61
FIGURE 5-8. EXAMPLE OF WRITE_BURST TO EDID MAPPING FOR 3D VIDEO 65
FIGURE 5-9. TOP-BOTTOM TIMINGS 68
FIGURE 5-10. LEFT-RIGHT TIMINGS 69
FIGURE 5-11. FRAME SEQUENTIAL TIMINGS 70
FIGURE 5-12. EXAMPLE FRAME SEQUENTIAL TIMINGS 72
FIGURE 6-1. AUDIO CLOCK REGENERATION MODEL 75
FIGURE 6-2. EXAMPLE AUDIO SAMPLE TIMING 81
FIGURE 7-1. BI-PHASE MARK ENCODING AND CBUS DRIVER STATES 84
FIGURE 7-2. LINK LAYER PACKET STRUCTURE 85
FIGURE 7-3. LINK LAYER PACKET ON THE CONTROL BUS 85
FIGURE 7-4. ACK BIT TIMING ON THE CONTROL BUS 87
FIGURE 7-5. LINK LAYER ARBITRATION TIMING – SOURCE VERSUS SINK ASSERTION TIMES 87
FIGURE 7-6. LINK LAYER ARBITRATION TIMING – FOUR CASES 89
FIGURE 7-7. LINK LAYER ARBITRATION TIMING – INITIATOR MAINTAINS CONTROL OF CBUS 89
FIGURE 7-8. LINK LAYER ARBITRATION TIMING – INITIATOR HOLD-OFF AFTER NACK BIT 89
FIGURE 7-9. LINK LAYER ARBITRATION TIMING – INITIATOR HOLD-OFF WITH NO PENDING PACKET 90
FIGURE 7-10. LINK LAYER ARBITRATION TIMING – FOLLOWER WINS ARBITRATION 90
FIGURE 7-11. LINK LAYER ARBITRATION TIMING – FOLLOWER PREEMPTS INITIATOR'S POST-NACK HOLD-OFF 91
FIGURE 7-12. PACKET FLOW DIAGRAM LEGEND 94
FIGURE 7-13. EXAMPLE STATE DIAGRAM LEGEND 95
FIGURE 7-14. EXAMPLE OF MSC "BAD OPCODE" HANDLING 98
FIGURE 7-15. MSC GET_STATE COMMAND 99
FIGURE 7-16. MSC GET_STATE STATE DIAGRAM 99
FIGURE 7-17. MSC GET_VENDOR_ID COMMAND 99
FIGURE 7-18. MSC GET_VENDOR_ID STATE DIAGRAM 99
FIGURE 7-19. MSC SET_HPD COMMAND 100
FIGURE 7-20. MSC SET_HPD STATE DIAGRAM 100
FIGURE 7-21. MSC CLR_HPD COMMAND 100
FIGURE 7-22. MSC CLR_HPD STATE DIAGRAM 100
FIGURE 7-23. MSC GET_DDC_ERRORCODE COMMAND 101
FIGURE 7-24. MSC GET_DDC_ERRORCODE STATE DIAGRAM 101
FIGURE 7-25. MSC GET_MSC_ERRORCODE COMMAND 102
FIGURE 7-26. MSC GET_MSC_ERRORCODE STATE DIAGRAM 102
FIGURE 7-27. MSC READ DEVICE CAPABILITY COMMAND 103
FIGURE 7-28. MSC READ_DEVCAP STATE DIAGRAM 103

FIGURE 7-29. MSC WRITE STATUS COMMAND 104
FIGURE 7-30. MSC WRITE_STAT STATE DIAGRAM 104
FIGURE 7-31. MSC SET INTERRUPT COMMAND 105
FIGURE 7-32. MSC SET_INT STATE DIAGRAM 105
FIGURE 7-33. MSC BURST WRITE COMMAND 106
FIGURE 7-34. MSC WRITE_BURST STATE DIAGRAM 106
FIGURE 7-35. MSC MSG COMMAND 108
FIGURE 7-36. MSC MSC_MSG STATE DIAGRAM 108
FIGURE 7-37. PROPER MSC MSC_MSG SUB-COMMAND SEQUENCING 109
FIGURE 7-38. IMPROPER MSC MSC_MSG SUB-COMMAND SEQUENCING 109
FIGURE 7-39. MSC MSGE SUB-COMMAND 110
FIGURE 7-40. DDC READ COMMAND 112
FIGURE 7-41. MHL DDC READ STATE DIAGRAM 113
FIGURE 7-42. DDC SEGMENT READ COMMAND TO SINK SUPPORTING SEGMENT POINTER 114
FIGURE 7-43. DDC SEGMENT READ COMMAND TO SINK NOT SUPPORTING SEGMENT POINTER 115
FIGURE 7-44. MHL DDC SEGMENT READ STATE DIAGRAM 116
FIGURE 7-45. DDC SHORT READ AND CURRENT READ COMMANDS 117
FIGURE 7-46. MHL DDC SHORT READ AND CURRENT READ STATE DIAGRAM 117
FIGURE 7-47. DDC WRITE COMMAND 118
FIGURE 7-48. MHL DDC WRITE STATE DIAGRAM 118
FIGURE 7-49. MSC RAP SUB-COMMAND 119
FIGURE 7-50. MSC RAPK SUB-COMMAND 120
FIGURE 7-51. MSC_MSG RCP/RCPK/RCPE STATE DIAGRAM 121
FIGURE 7-52. MSC RCP SUB-COMMAND 121
FIGURE 7-53. MSC RCPK SUB-COMMAND 122
FIGURE 7-54. MSC RCPE SUB-COMMAND 122
FIGURE 7-55. MSC_MSG RCP PRESS-AND-HOLD BEHAVIOR STATE DIAGRAM 128
FIGURE 7-56. ENTERING AND EXITING MEDIA SELECTION MODE 131
FIGURE 7-57. MSC UCP SUB-COMMAND 138
FIGURE 7-58. MSC UCPK SUB-COMMAND 138
FIGURE 7-59. MSC UCPE SUB-COMMAND 138
FIGURE 7-60. MSC_MSG CHARACTER CODE SEQUENCE "AB12" EXAMPLE 140
FIGURE 7-61. MSC WRITE_BURST COORDINATED WITH RCHANGE_INT INTERRUPT BITS 154
FIGURE 8-1. CONNECTION OF AN MHL SOURCE TO AN MHL SINK 159
FIGURE 8-2. SOURCE DISCOVERY AND DISCONNECT STATE DIAGRAM 163
FIGURE 8-3. SOURCE WITHOUT VBUS OUTPUT – DISCOVERY AND DISCONNECT STATE DIAGRAM 167
FIGURE 8-4 SINK DISCOVERY AND DISCONNECT STATE DIAGRAM #1 170
FIGURE 8-5. SINK DISCOVERY AND DISCONNECT STATE DIAGRAM #2 171
FIGURE 8-6. SINK DISCOVERY AND DISCONNECT STATE DIAGRAM #3 172
FIGURE 8-7. DONGLE DISCOVERY AND DISCONNECT STATE DIAGRAM 178
FIGURE 8-8. SOURCE SUCCESSFULLY CONNECTS TO ACTIVE SINK – TIMING DIAGRAM 182
FIGURE 8-9. SOURCE SUCCESSFULLY DISCOVERS UNPOWERED DONGLE – TIMING DIAGRAM 183
FIGURE 8-10. SOURCE PLUGS INTO SINK THAT RECOGNIZES WAKE PULSES FROM STANDBY – TIMING DIAGRAM 184
FIGURE 8-11. CONNECTION –CBUS PULL-DOWN 185
FIGURE 8-12. CONNECTION –WAKE PULSE SEQUENCE 186
FIGURE 8-13. CONNECTION –DRIVE CBUS TO HIGH 188
FIGURE 8-14. CONNECTION –FLOAT CBUS AND COUNT DISCOVERY PULSE IF LOW 188
FIGURE 8-15. CONNECTION –FLOAT CBUS AND COUNT DISCOVERY PULSE 189
FIGURE 8-16. CONNECTION –TERMINATE MHL+/MHL- 190
FIGURE 8-17. "DISCONNECTION ON DETACH" TIMING DIAGRAM 192
FIGURE 8-18. ONE-TOUCH-ON – CASE #1 193
FIGURE 8-19. ONE-TOUCH-ON – CASE #2 194
FIGURE 8-20. ONE-TOUCH-ON – CASE #3 195
FIGURE 8-21. ONE-TOUCH-PLAY 196
FIGURE 8-22. ONE-TOUCH-PLAY 197
FIGURE 10-1. ENHANCED EDID SEGMENT POINTER AND BLOCK LAYOUT 206
FIGURE 12-1. CABLE DETECT CIRCUIT FOR MHL SINK WITH HDMI TYPE A RECEPTACLE 212
FIGURE 12-2. CABLE DETECT CIRCUIT FOR MHL CABLE WITH HDMI TYPE A PLUG 213
FIGURE 13-1. CONCEPTUAL SCHEMATIC FOR AN MHL DIFFERENTIAL PAIR 216

FIGURE 13-2. SINGLE-ENDED, DIFFERENTIAL AND COMMON-MODE SIGNALS 217
FIGURE 13-3. MHL LINK TEST POINTS 218
FIGURE 13-4. MHL TYPE M DONGLE TEST POINTS 219
FIGURE 13-5. MHL TYPE F DONGLE TEST POINTS 219
FIGURE 13-6. SOURCE DEVICE TEST POINTS 220
FIGURE 13-7. SOURCE DEVICE TEST LOAD 220
FIGURE 13-8. TEST SIGNAL GENERATORTEST LOAD 221
FIGURE 13-9. SINK DEVICE TEST POINTS 221
FIGURE 13-10. DONGLE DEVICE TEST POINTS 222
FIGURE 13-11. REFERENCE CABLE EQUALIZER GAIN CURVE 223
FIGURE 13-12. CLOCK RECOVERY UNIT IN EYE DIAGRAM MEASUREMENT 224
FIGURE 13-13. MHL CLOCK JITTER AT TP1 228
FIGURE 13-14. DIFFERENTIAL EYE DIAGRAM MASK AT TP1 FOR SOURCE DEVICE REQUIREMENTS 229
FIGURE 13-15. MHL DATA JITTER AT TP1 229
FIGURE 13-16. CABLE EMULATION MODEL –DIFFERENTIAL INSERTION LOSS 230
FIGURE 13-17. CABLE EMULATION MODEL –COMMON MODE INSERTION LOSS 230
FIGURE 13-18. CABLE EMULATION MODEL –DIFFERENTIAL INTRA-PAIR SKEW 231
FIGURE 13-19. CABLE EMULATION MODEL –COMMON MODE INTRA-PAIR SKEW 231
FIGURE 13-20. MHL CLOCK JITTER AT TP2 233
FIGURE 13-21. MHL DATA EYE MASK AT TP2 234
FIGURE 13-22. MHL DATA JITTER AT TP2 235
FIGURE 13-23. CABLE ASSEMBLY TEST POINTS 238
FIGURE 13-24. MHL CABLE DIFFERENTIAL INSERTION LOSS FOR MHL DATA SIGNALS 241
FIGURE 13-25. MHL CABLE COMMON-MODE INSERTION LOSS FOR MHL CLOCK SIGNALS 242
FIGURE 13-26. MHL CABLE MODE CONVERSION BETWEEN DIFFERENTIAL AND COMMON-MODE SIGNALS 243
FIGURE 13-27. MHL CBUS SIGNAL INSERTION LOSS 244
FIGURE 13-28. MHL CABLE ASSEMBLY FAR-END CROSSTALK BETWEEN MHL AND CBUS 245
FIGURE 13-29. MHL CABLE ASSEMBLY FAR-END CROSSTALK BETWEEN MHL AND VBUS 246
FIGURE 13-30. MHL CABLE ASSEMBLY FAR-END CROSSTALK BETWEEN CBUS AND VBUS 247
FIGURE 13-31. CABLE ASSEMBLY VOLTAGE DROP 249
FIGURE 13-32. CBUS LOADING SCHEMATIC VIEW 251
FIGURE 13-33. CBUS WAKE AND DISCOVERY PULSE WIDTH MEASUREMENT 255

List of Tables

TABLE 4-1. ENCODING TYPE AND DATA TO BE TRANSMITTED 29
TABLE 4-2. TRANSITION COUNT PER TMDS CHARACTER 30
TABLE 4-3. GUARD BAND VALUES 33
TABLE 4-4. PREAMBLE INDICATOR OF DATA PERIOD TYPE 33
TABLE 4-5. PACKEDPIXEL MODE GUARD BAND VALUES 36
TABLE 4-6. PREAMBLE INDICATOR OF DATA PERIOD TYPE 37
TABLE 4-7. PACKET HEADER 40
TABLE 4-8. PACKET TYPES 40
TABLE 4-9. NULL PACKET 41
TABLE 4-10. AUDIO CLOCK REGENERATION PACKET 42
TABLE 4-11. AUDIO SAMPLE PACKET 44
TABLE 4-12. CONTENT MUTE PACKET 46
TABLE 4-13. INFOFRAME PACKET HEADER 47
TABLE 4-14. INFOFRAME PACKET CONTENTS 47
TABLE 4-15. CONTROL-SIGNAL ASSIGNMENT 49
TABLE 4-16. ENCODING ALGORITHM DEFINITIONS 51
TABLE 5-1. MHL SOURCE VIDEO MODE SUPPORT REQUIREMENTS 55
TABLE 5-2. MHL SINK VIDEO MODE SUPPORT REQUIREMENTS 56
TABLE 5-3. 3D VIDEO FORMAT DATA STRUCTURE IN WRITE_BURST 63
TABLE 5-4. 3D VIDEO DESCRIPTOR DATA STRUCTURE 66
TABLE 5-5. VENDOR-SPECIFIC INFOFRAME PACKET 67
TABLE 5-6. EXAMPLE FRAME SEQUENTIAL TIMINGS 71
TABLE 5-7. MHL SOURCE 3D VIDEO MODE SUPPORT REQUIREMENTS 73

TABLE 5-8. MHL SINK 3D VIDEO MODE SUPPORT REQUIREMENTS 73
TABLE 6-1. RECOMMENDED N AND EXPECTED CTS FOR 32KHZ AUDIO 76
TABLE 6-2. RECOMMENDED N AND EXPECTED CTS FOR 44.1KHZ AND MULTIPLES 76
TABLE 6-3. RECOMMENDED N AND EXPECTED CTS FOR 48KHZ AND MULTIPLES 77
TABLE 6-4. CHANNEL STATUS VALUES FOR AUDIO SAMPLE FREQUENCIES 78
TABLE 6-5. AUDIO SAMPLE PACKET LAYOUT AND LAYOUT VALUE 80
TABLE 6-6. VALID PRESENT BIT CONFIGURATIONS 80
TABLE 7-1. HEADER BITS ENCODING 86
TABLE 7-2. CONTROL BIT ENCODING 86
TABLE 7-3. MSC TRANSLATION LAYER CONTROL PACKETS 96
TABLE 7-4. SUMMARY OF MSC TRANSLATION LAYER FLOW CONTROL 97
TABLE 7-5. GET_DDC_ERRORCODE RETURN CODES 101
TABLE 7-6. GET_MSC_ERRORCODE RETURN CODES 102
TABLE 7-7. MSC MESSAGE SUB-COMMANDS 108
TABLE 7-8. DDC CHANNEL TRANSLATION LAYER CONTROL PACKETS 110
TABLE 7-9. DDC COMMAND SUPPORT IN MHL 111
TABLE 7-10. SUMMARY OF DDC TRANSLATION LAYER FLOW CONTROL 111
TABLE 7-11. MSC REQUEST ACTION PROTOCOL SUB-COMMAND ACTION CODES 119
TABLE 7-12. MSC REQUEST ACTION PROTOCOL STATUS CODES 120
TABLE 7-13. MSC REMOTE CONTROL PROTOCOL STATUS CODES 122
TABLE 7-14. MSC REMOTE CONTROL PROTOCOL SUB-COMMAND 7-BIT KEY CODE 123
TABLE 7-15. KEY CODE BEHAVIORS 124
TABLE 7-16. MSC_MSG RCP PRESS-AND-HOLD STATES AND TRANSITIONS 128
TABLE 7-17. KEY CODE RECEIVING SUPPORT MAP 132
TABLE 7-18. KEY ID SENDING SUPPORT MAP FOR SINK 135
TABLE 7-19. KEY ID SENDING SUPPORT MAP FOR SOURCE 136
TABLE 7-20. MSC UTF-8 CHARACTER PROTOCOL STATUS CODES 139
TABLE 7-21. DEVICE REGISTER SPACE 140
TABLE 7-22. MHL DEVICE CAPABILITY REGISTERS 142
TABLE 7-23. MHL CAPABILITY REGISTER REQUIRED FIELDS BY DEVICE TYPE 143
TABLE 7-24. DEVICE STATE REGISTER DETAILS 143
TABLE 7-25. MHL VERSION REGISTER DETAILS 144
TABLE 7-26. DEVICE CATEGORY REGISTER DETAILS 144
TABLE 7-27. DEV_TYPE SETTINGS 144
TABLE 7-28. POW AND PLIM SETTINGS 145
TABLE 7-29. ADOPTER ID REGISTER DETAILS 146
TABLE 7-30. VIDEO LINK MODE REGISTER DETAILS 146
TABLE 7-31. AUDIO LINK MODE REGISTER DETAILS 147
TABLE 7-32. VIDEO TYPE REGISTER DETAILS 147
TABLE 7-33. LOGICAL DEVICE MAP REGISTER DETAILS 148
TABLE 7-34. BANDWIDTH LIMIT REGISTER DETAILS 149
TABLE 7-35. FEATURE SUPPORT FLAG REGISTER DETAILS 149
TABLE 7-36. DEVICE ID REGISTER DETAILS 150
TABLE 7-37. SCRATCHPAD SIZE REGISTER DETAILS 150
TABLE 7-38. INTERRUPT AND STATUS SIZE REGISTER DETAILS 151
TABLE 7-39. MANDATORY DEVICE INTERRUPT REGISTERS 151
TABLE 7-40. MHL INTERRUPT BITS REQUIRED BY DEVICE TYPE 152
TABLE 7-41. REGISTER CHANGE INTERRUPT REGISTER DETAILS 153
TABLE 7-42. DEVICE STATE CHANGE INTERRUPT REGISTER DETAILS 155
TABLE 7-43. MANDATORY DEVICE STATUS REGISTERS 155
TABLE 7-44. MHL STATUS REGISTER FIELDS REQUIRED BY DEVICE TYPE 156
TABLE 7-45. CONNECTED DEVICE READY STATUS REGISTER DETAILS 156
TABLE 7-46. LINK MODE STATUS REGISTER DETAILS 157
TABLE 7-47. BURST_ID CODES 158
TABLE 8-1. MHL SOURCE – SIGNALS DURING DISCOVERY 163
TABLE 8-2. MHL STATES AND TRANSITIONS FROM SOURCE DURING DISCOVERY 164
TABLE 8-3. MHL SOURCE WITHOUT VBUS OUTPUT – SIGNALS DURING DISCOVERY 167
TABLE 8-4. MHL STATES AND TRANSITIONS FROM SOURCE WITHOUT VBUS OUTPUT - DURING DISCOVERY 168
TABLE 8-5. MHL SIGNALS FROM SINK DURING DISCOVERY 173

TABLE 8-6. MHL SINK STATES AND TRANSITIONS DURING DISCOVERY WITH VBUS FROM SINK 174
TABLE 8-7. MHL SIGNALS FROM DONGLE DURING DISCOVERY 179
TABLE 8-8. MHL DONGLE STATES AND TRANSITIONS DURING DISCOVERY WITH VBUS FROM DONGLE 179
TABLE 9-1. POWER CONFIGURATIONS 200
TABLE 10-1. AVI INFOFRAME PACKET 202
TABLE 10-2. MHL VALID PIXEL REPEAT VALUES FOR EACH FORMAT 203
TABLE 10-3. AUDIO INFOFRAME PACKET 204
TABLE 11-1. ENCRYPTION STREAM MAPPING 208
TABLE 11-2. ENCRYPTION STREAM MAPPING FOR TERC4 ENCODING 209
TABLE 12-1. HDMI TYPE A PIN ASSIGNMENT LIST 211
TABLE 12-2. ELECTRICAL SPECIFICATIONS FOR HDMI TYPE A RECEPTACLE FOR CABLES 212
TABLE 12-3. CABLE DETECT AC ELECTRICAL SPECIFICATIONS FOR MHL SINK WITH HDMI TYPE A RECEPTACLE 213
TABLE 12-4. CABLE DETECT AC ELECTRICAL SPECIFICATIONS FOR MHL CABLE WITH HDMI TYPE A PLUG 213
TABLE 12-5. PIN ASSIGNMENT LIST 214
TABLE 13-1. OPERATING CONDITIONS 218
TABLE 13-2 AMPLITUDE TRANSFER FUNCTION PARAMETERS FOR REFERENCE CABLE EQUALIZER 223
TABLE 13-3. TMDS LINK TIMING PARAMETERS – 24-BIT MODE 225
TABLE 13-4. TMDS LINK TIMING PARAMETERS – PACKEDPIXEL MODE 225
TABLE 13-5. SOURCE DEVICE DC CHARACTERISTICS AT TP1 226
TABLE 13-6. SOURCE DEVICE AC CHARACTERISTICS AT TP1 227
TABLE 13-7. RECEIVER DC CHARACTERISTICS AT TP2 232
TABLE 13-8. RECEIVER AC CHARACTERISTICS AT TP2 232
TABLE 13-9. SINK DEVICE IMPEDANCE CHARACTERISTICS AT TP2 233
TABLE 13-10. DONGLE DC CHARACTERISTICS AT TP3 235
TABLE 13-11. DONGLE AC CHARACTERISTICS AT TP3 236
TABLE 13-12. DONGLE DEVICE IMPEDANCE CHARACTERISTICS AT TP3 237
TABLE 13-13. CABLE ASSEMBLY MHL CHARACTERISTICS 239
TABLE 13-14. MINIMAL CABLE AC PARAMETRIC REQUIREMENTS 240
TABLE 13-15. MHL CABLE DIFFERENTIAL INSERTION LOSS FOR MHL DATA SIGNALS 241
TABLE 13-16. MHL CABLE COMMON-MODE INSERTION LOSS FOR MHL CLOCK SIGNALS 242
TABLE 13-17. MHL CABLE MODE CONVERSION BETWEEN DIFFERENTIAL AND COMMON-MODE SIGNALS 243
TABLE 13-18. MHL CBUS SIGNAL INSERTION LOSS 244
TABLE 13-19. MHL CABLE ASSEMBLY FAR-END CROSSTALK BETWEEN MHL AND CBUS 245
TABLE 13-20. MHL CABLE ASSEMBLY FAR-END CROSSTALK BETWEEN MHL AND VBUS 246
TABLE 13-21. MHL CABLE ASSEMBLY FAR-END CROSSTALK BETWEEN CBUS AND VBUS 247
TABLE 13-22. CABLE +5V POWER PIN (VBUS) VOLTAGE 249
TABLE 13-23. +5V POWER PIN (VBUS) VOLTAGE 250
TABLE 13-24. +5V POWER PIN (VBUS) CURRENT 251
TABLE 13-25. REQUIRED OUTPUT CHARACTERISTICS OF CBUS SIGNAL 252
TABLE 13-26. REQUIRED DETECTION LEVELS OF CBUS SIGNAL 253
TABLE 13-27. SOURCE SIDE CBUS LINE CHARACTERISTICS 254
TABLE 13-28. SINK SIDE CBUS LINE CHARACTERISTICS 254
TABLE 13-29. DONGLE SIDE CBUS LINE CHARACTERISTICS 254
TABLE 13-30. WAKE PULSE SEQUENCE TIMING PARAMETERS 255
TABLE 13-31. CONNECTION PARAMETERS (PART ONE) 256
TABLE 13-32. CONNECTION PARAMETERS (PART TWO) 257
TABLE 13-33. CONNECTION PARAMETERS (PART THREE) 258
TABLE 13-34. DISCONNECTION PARAMETERS 259
TABLE 13-35. CBUS LINK LAYER TIMING PARAMETERS 260
TABLE 13-36. CBUS TRANSLATION LAYER TIMING PARAMETERS 261
TABLE 13-37. AUDIO INFOFRAME TIMING PARAMETERS 263

1 Introduction

1.1 Purpose and Scope

This document constitutes the specification for the Mobile High-definition Link (MHL).

The Mobile High-definition Link is an interface for transmitting digital television audiovisual signals from mobile phones, mobile video players, video cameras, still image cameras and other audiovisual sources to television sets, projectors, monitors and other video displays. The MHL interface is a low-pin-count high definition (HD) video interface based on the same Transition Minimized Differential Signaling (TMDS) technology used in the Digital Video Interface (DVI).

MHL can carry high quality multi-channel audio data and can carry all standard and high-definition consumer electronics video formats. Content protection technology is available.

MHL also carries control and status information in both directions. In addition, MHL has a dedicated pin to supply power from one MHL system to another.

MHL 2.0 brings to the specification the detailed mechanisms on how to carry 3D video across the link; and the requirement of MHL 2.0 Sinks to provide more output power on VBUS.

This specification completely describes the interface such that one could implement a complete transmission and interconnect solution or any portion of the interface. The underlying TMDS-based protocol and associated electrical signaling is described in detail.

A device that is compliant with this specification is interoperable with other compliant devices through the configuration and implementation provided for in this specification.

Electrical, behavioral and protocol requirements necessary for compliance are described for source devices, sink devices and cables.

1.2 Organization of this document

This specification is organized as follows:

Section	Content
1	Introduces MHL.
2	Defines terms and acronyms used throughout this document.
3	Provides a high-level overview of the operation of MHL.
4	Describes the Signaling and Encoding on the TMDS lines.
5	Describes Video related issues, including 3D video transport.
6	Describes Audio related issues.
7	Describes the Link Control Bus including its protocol and timing, translation layer commands and the devices' internal register space addressable across MHL.
8	Describes device discovery and initialization.
9	Describes Power Provisioning and how 5 volts is delivered.
10	Describes packets, InfoFrames and EDID data used in MHL.
11	Describes how HDCP content protection is used on MHL.
12	Physical Layer, including Connection details.
13	Describes electrical and timing specifications of MHL.
14	Informative Appendices

1.3 Revision Highlights

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.

Rev.	Highlights
1.0	Release Specification
1.1	Clarify usage of various CBUS commands, sub-commands, and error handling. (Section 7) Allow a Sink in Standby to optionally output VBUS. (Section 8) Relax electrical specification for short Cables and for Dongles. (Section 13) Clarify various electrical timing, parameter usage. (Section 13) Correct the hex codes for Device Type. (Table 7-27) Specify output high voltage for CBUS prior to connected state. (Table 13-25) Move Connection Supplement into a separate document with no change in related requirements. Various clarifications and editorial corrections. (throughout)
1.2	Refine requirements for transmitting AVI InfoFrame. (Section 10.2.1) Replace CBUS power-off capacitance limits with limits for power-on. (13.8.1,13.10.1) Clarify options for Sink in Standby to recognize wake pulses, or send VBUS, or both. (Section 8.2.2) Assign RCP key codes for "red/yellow/green/blue" buttons. (Sections 7.7.4-7.7.5) Clarify Source shall initiate discovery when in standby or active mode. (Section 8.2.1) Clarify handling of CLK_MODE. (Section 7.9.3.2) Clarify that Sink shall read Source ADOPTER_ID before sending WRITE_BURST. (Section 7.4.3.7) Add $I_{CABLE\_VBUS\_MAX}$ absolute maximum current required of all cables. (Section 13.8.3) Add timeout parameters to limit total CBUS MSC command time. (Section 13.10.3) Corrections to TMDS coding syntax. (Table 4-3, Table 4-5, Section 4.4) Various clarifications and editorial corrections. (throughout)
2.0	Add support for 3D video. (Section 5.9) Increase minimum required charging power output from Sinks. (Table 13-24) Remove Source option to draw power while only monitoring VBUS. (Section 13.9.1.2) Add support for sending UTF-8 characters on MSC. (Section 7.8.4)
2.1	Incorporate Connection Supplement into main specification. (Section 12) Allow Sink and Source to use plug or receptacle, and definitions for “Direct Attach Source” and “Direct Attach Sink”. Clarify ‘Sink’ vs. ‘Dongle. (Section 2) Add note to allow PLIM[1:0]={1,1} for Dongles with 100mA. (Table 7-26) Add I_{VBUS_Ref} (500mA) for measuring voltage drops across cable. (Table 13-22) Clarify T_{ABORT_NEXT} usage in note #1. (Table 13-36) Fix typo to show proper definition of PATH_EN bit. (Table 7-46) Add recommendation $T_{NACK\_RETRY\_NEXT}$ in MSC_MSG. (Section 7.4.3.10 and Table 13-36) Add $V_{CABLE\_CM\_SWING}$ cable parameter. (Table 13-13) Optimize various cable electrical parameters. (Figure 13-23 through Table 13-21) Define DC-Powered Sink, and allow it to output no VBUS. (Sections 2.2, 8.2.2, 9.1-9.2) Limit current draw by Source pre- and post-discovery. (Section 13.9.1.2) Do not allow non-zero values in Audio InfoFrame’s CT field. (Section 10.2.2) Allow Source eye diagram to qualify intra-pair skew. (Sections 13.5.2-13.5.4) Clarify Audio Data Packet ‘sample_present’ bit usage with Layout=1. (Section 6.7) Various clarifications and editorial corrections. (throughout)

1.4 Informative References

The following documents contain information that is useful in understanding this standard. Some of these documents are drafts of standards that may become normative references in a future release of this standard.

ANSI/SMPTE, SMPTE Standard 170M-2004 (November 2004) for Television – Composite Analog Video Signal – NTSC for Studio Applications

ANSI/SMPTE, SMPTE Standard 274M-2005 (February 2005) for Television – 1920 x 1080 Image Sample Structure Digital Representation and Digital Timing Reference Sequences for Multiple Picture Rates

ANSI/SMPTE, SMPTE Standard 296M-2001 (January 2001) for Television – 1280 x 720 Progressive Image Sample Structure – Analog and Digital Representation and Analog Interface

1.5 Normative References

The following standards contain provisions that, through reference in this text, constitute normative provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. If the referenced standard is dated, the reader is advised to use the version specified.

1. EIA, EIA/CEA-861E, "A DTV Profile For Uncompressed High Speed Digital Interfaces"
2. CEA-931C, "Remote Control Command Pass-through for Home Networking"
3. VESA ENHANCED EXTENDED DISPLAY IDENTIFICATION DATA (E-EDID) STANDARD, Release A, Revision 2, September 25, 2006
4. VESA Enhanced Display Data Channel (E-DDC) Standard, Version 1.2, December 26, 2007
5. ITU-R BT.601-6 Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios (January 2007)
6. ITU-R BT.709-5 Parameter values for the HDTV standards for production and international programme exchange (April 2002)
7. IEC, IEC 60958-1, "Digital audio interface – Part 1: General", First edition 1999-12
8. IEC, IEC 60958-3, "Digital audio interface – Part 3: Consumer applications", Third edition 1999-12
9. IEC, IEC 61937, "Digital Audio – Interface for non-linear PCM encoded audio bitstreams applying IEC 60958", First edition 2000-04
10. DDWG, "Digital Visual Interface" (DVI), Revision 1.0, April 2, 1999 (DVI)
11. Digital Content Protection LLC, "High-bandwidth Digital Content Protection System", (HDCP 1.4)
12. The Unicode Consortium, "The Unicode Standard, Version 6.0.0", (The Unicode Consortium, 2011. ISBN 978-1-936213-01-6) (Unicode)

1.6 Usages and Conventions

Convention	Use
bit N	Bits are numbered in little-endian format, i.e. the least-significant bit of a byte or word is referred to as bit 0.
0xNN	Hexadecimal representation of base-16 numbers are represented using 'C' language notation, preceded by '0x'. The first 'N' after the '0x' represents the four most-significant bits in the value.
0bNN	Binary (base-2) numbers are represented using 'C' language notation, preceded by '0b'. The most-significant bit in the expression follows immediately after the '0b' prefix.
NN	Decimal (base-10) numbers are represented using no additional prefixes.
[m:n]	Denotes a bit range, from the most-significant bit 'm' to the least-significant bit 'n' within one register address. Therefore, 'm' is numerically always greater than or equal to 'n'. Example: [3:0] are the four least-significant bits in a register.

Within this specification, any descriptions of data structures, values or sequences that occur on the MHL interface should be interpreted only as data structures, values and sequences that are transmitted by the MHL source. Due to the possibility of errors during the transmission, these items should not be construed as data structures, values or sequences that are guaranteed to be detected by the MHL receiving device.

2 Definitions

2.1 Conformance Levels

Conformance	Description
Expected	A key word used to describe the behavior of the hardware or software in the design models *assumed* by this Specification. Other hardware and software design models may also be implemented.
May	A key word that indicates flexibility of choice with *no implied preferenc*e.
Shall	A key word indicating a mandatory requirement. Designers are *required* to implement all such mandatory requirements.
Should	A key word indicating flexibility of choice with a strongly preferred alternative. Equivalent to the phrase *is recommende*d.
Reserved fields	A set of bits within a data structure that are defined in this specification as reserved, and are not otherwise used. Implementations of this specification shall zero these fields. Future revisions of this specification, however, may define their usage.
Reserved values	A set of values for a field that are defined in this specification as reserved, and are not otherwise used. Implementations of this specification shall not generate these values for the field. Future revisions of this specification, however, may define their usage.

2.2 Glossary of Terms

Term	Definition
_L	A suffix for a symbol to indicate the low-order byte in a pair of 2 bytes.
_H	A suffix for a symbol to indicate the high-order byte in a pair of 2 bytes.
attached	The state of two MHL devices in which a plug on one device is mechanically inserted completely into a receptacle of the other device; the two are physically joined, such as a cable to a Source or Sink.
(Audio) Channel	Audio data meant to be delivered to a single audio speaker
Audio-Only Device	An MHL Source which outputs only audio content, although it is transported with a link clock and video control signals but no video pixel data: "blank video"; or an MHL Sink which takes in audio only and ignores any video content on the MHL stream.
Audio Sample Clock	Original clock related to the audio input samples at the Source device or the generated clock used to time the output of audio samples from the Sink.
Basic Audio	Stereo audio, a minimum requirement on an MHL interconnect.
BCH	Error correction technique named after the developers: Bose, Chauduri, and Hocquenghem.
Blank Video	A video stream having compliant sync timings and DE timings, but with no picture content, i.e., all pixels in a frame are the same color or otherwise have no content from the original stream.
Byte	Eight bits of data.
Category	A classification of MHL system type. MHL Sources and MHL Sinks are defined categories.
CBUS	MHL Link Control Bus. This is a single-wire bus used to carry auxiliary data between an MHL source and an MHL sink. *Not to be confused with the CBUS described in the I2C Specification.*
Chroma Sampling	Indication if a video signal has "full" ("4:4:4") or "half" ("4:2:2") chroma bandwidth. "Full chroma bandwidth" indicates that there is one Cb and one Cr sample for every Y sample (i.e. for every pixel in the video). "Half chroma bandwidth" indicates that there is only one Cb sample for every two Y samples and one Cr sample for every two Y samples.
Connector	A general term inclusive of plugs and receptacles. A connector has a specific signal list and pin assignment.
Command	An operation, indicated by a numerical value or opcode, carried in a control packet on CBUS, which is intended by one MHL device to be performed by another, connected MHL device.
Data Island	A sequence of TMDS characters, beginning with a leading guard band, ending with a trailing guard band, and containing one of more packets.
Data Stream Disparity	Integer indicating "DC-offset" level of link. A positive value represents the excess number of "1"s that have been transmitted. A negative value represents the excess number of "0"s that have been transmitted.
DC-Powered Sink	A type of MHL Sink which relies on a DC internal power source, when not provided power from an external source, and does not contain an internal capability to generate DC power. (Most DC-Powered Sinks are powered with a battery.)

Term	Definition
Direct Attach Sink	A type of MHL Sink device which uses a plug on its MHL input port to directly attach, without a cable, to an MHL Source's output receptacle.
Direct Attach Source	A type of MHL Source device which uses a plug on its MHL output port to directly attach, without a cable, to an MHL Sink's input receptacle.
Dongle	See "MHL Dongle", below.
Downstream	In the direction of the primary audio and video data flow, i.e. towards the Sink.
Frame Sequential	A type of Full-3D video in which all of the lines of the left sub-frame are transmitted first, followed by an Inter Sub-Frame Blanking time, then all the lines of the right sub-frame. See Section 5.9.3.3.
Full-3D	A type of 3D video formatting in which the line count and pixel count in each of the left sub-frame and right sub-frame are equal to the line count and pixel count in the 2D video mode with the same name. For example, "Frame sequential 1080p60" is a Full-3D mode which has 1920 pixels per line and 1080 lines per sub-frame.
Hot Plug Detect	A signal indicated in the Source on receipt of SET_HPD or CLR_HPD command from the Sink, which indicates the availability of the DDC sub-channel for EDID and HDCP.
Left-Right	A type of 3D video formatting in which half of each line's pixels are used for the left sub-frame, and the other half are used for the right sub-frame.
Link Clock	A clock used across the link from MHL Source to MHL Sink (or Dongle), proportional to the video mode being transported. See Section 4.1.1.
Logical Channel	MHL has three logical data channels composed of the three data streams that are time-domain multiplexed onto the physical channel – Channel 0, Channel 1 and Channel 2.
MHL Cable	A device with one MHL input and one MHL output, with a plug and not a receptacle at each end.
MHL Component	A device which implements a subset of the functionality required of an MHL Source or Sink device. The supplier of the Component indicates exactly which MHL functions are supported by the Component. It shall be possible to design an MHL-compliant system that uses the Component to perform the MHL functions implemented in the Component as indicated by the Component supplier.
MHL Dongle	A device with one MHL input and one non-MHL output, which does not provide a display or audible rendering of content, nor necessarily provide power to the Source attached at its input. A Dongle may be powered from its own power source, or may be powered from the upstream MHL Source. An MHL Dongle shall have a plug at its MHL input, and may have either a plug or receptacle at its non-MHL output. A Dongle is differentiated from a Sink in that a Dongle is designed to convert MHL protocol on the input to a different interconnect protocol on the output. NOTE: Dongles, as the termini of MHL Signals, are deemed to be Sinks for purposes of, and as such term is defined in, the Mobile High-Definition Link Specification Adopter Agreement.

Term	Definition
MHL Sink	A device with one or more MHL input port. An MHL Sink may use a receptacle or a plug at each MHL input port. A Sink is differentiated from a Dongle in that a Sink is designed to decode the MHL input protocol to video, audio and control data, usually in visible or audible form.
MHL Source	A device with an MHL output port. An MHL Source may use a receptacle or a plug at its MHL output port.
Multi-channel (Audio)	More than two audio channel audio. Typically this term is applied to 6 (5.1) channel streams. Also called surround formats.
PackedPixel	A method of encoding pixel data. Only 16 bits of data are encoded, and transmitted, rather than the 24 bits per pixel as in the other pixel encoding modes.
Packet	In the context of CBUS, a sequence of bits sent across CBUS, beginning with a SYNC and ending with link layer ACK, containing a payload of 8 bits. Packets may be control packets or data packets. In the context of TMDS signaling, a 32-byte sequence of payload data, as part of a data island.
Physical Channel	MHL has one physical data channel composed of the single differential pair carrying TMDS data.
Pixel	Picture Element. Refers to the actual element on the display and the data point in the digital video stream representing such an element. This term may also apply to the data that is carried across the MHL link during a single pixel clock cycle, even if that data does not actually represent a picture element.
Plug	A type of connector which, on its outermost dimension, is inserted into a receptacle. A plug is the normal termination on either end of a cable.
RGB	Digital representation of any video signal using a red/green/blue color spaces.
Receiver	An MHL Component that is responsible for receiving the single physical channel (three logical channels) at the input to an MHL Sink device and converting that signal into a digital output indicating a 24 bit, 12 bit, or 8 bit TMDS decoded word and indicating the TMDS coding mode used to decode those bits. This digital output may be contained within a semiconductor device or may be output from a semiconductor device.
Receptacle	A type of connector which receives a plug to complete an interconnection between two systems. In most cases a receptacle has an opening into which a plug is inserted. The pin assignment of the receptacle must match the pin assignment of the mating plug across the set of conducting surfaces.
Requester	An MHL device which uses the CBUS MSC to perform a command to another MHL device – the responder.
Responder	An MHL device which uses the CBUS MSC to respond to a requested command from a requester.
Receiver Sense	A signal sensed by an MHL Source which indicates that the 2 TMDS signals between the Source and the Sink (or Dongle) are terminated to an active voltage.
Sink	See "MHL Sink", above.

Term	Definition
Sink action	A step taken autonomously by a Sink device, such as an expired timer or action at a different input port, which is meant to cause a change in state in the Sink device, or to send data from the device to another MHL device (such as an RCP sub-command).
Source	See "MHL Source", above.
Stereo	2 audio channel audio, one left and one right.
Stream	A time-ordered set of digital data originating from one Source and terminating at one Sink. A stream is characterized by bounded bandwidth requirements and by synchronization points, or time stamps, within the stream data.
Sub-command	An operation, indicated by a numerical value or opcode, carried in a data packet on CBUS, which is intended by one MHL device to be performed by a second, directly-connected MHL device.
Sub-frame	One portion of a 3D video mode which contains the pixel data for the left or the right eye. A 3D video frame contains two sub-frames.
T_{BIT}	Time duration of a single bit carried across the MHL link.
T_{CHAR}	Time duration of a single 10-bit TMDS character across the MHL link.
T_{MHL}	Period of the common-mode clock carried across the MHL link.
Top-Bottom	A type of 3D video formatting in which half of the frame's lines are used for the left sub-frame, and half for the right sub-frame.
T_{PIXEL}	Inverse of F_{PIXEL}, the pixel clock period, defined for each video mode in external specifications, such as 148.5MHz for 1080p/60.
Transmitter	An MHL Component that is responsible for driving the three logical TMDS channels, one TMDS clock, and one physical differential channel, into an MHL Source output port.
unattached	The state of two MHL devices in which a plug on one device is not mechanically inserted into a receptacle of the other device; the two are not physically joined, such as a cable to a Source or Sink.
Upstream	In the opposite direction from the primary audio and video data flow, i.e. towards the Source.
User action	A step taken by a user of an MHL device, such as pressing a button or operating a remote control, which is meant to cause a change in state in the MHL device, or to send data from one device to another (such as an RCP sub-command).
VBUS	MHL Voltage Bus, a one-wire interconnect between Source and Sink, to provide power to the Source device from the Sink, or power from the Source device to a Dongle.
YCbCr	Digital representation of any video signal using one of several luminance/color-difference color spaces.
Video Field	The transmission duration from one vertical sync (VSYNC) to the next VSYNC in the video stream.

2.3 Acronyms and Abbreviations

Term	Meaning
ANSI	American National Standards Institute
AVI	Auxiliary Video Information
CEA	Consumer Electronics Association
CP	Content Protection as a general term.
CTS	Cycle Time Stamp
DDC	Display Data Channel
DTV	Digital Television
DVD	Digital Versatile Disk
DVI	Digital Visual Interface
ECC	Error Correction Code
EDC	Error Detection Code
E-DDC	Enhanced Display Data Channel
EDID	Extended Display Identification Data
E-EDID	Enhanced Extended Display Identification Data
EIA	Electronic Industries Alliance
HDCP	High-bandwidth Digital Content Protection
HPD	Hot Plug Detect
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISFB	Inter Sub-Frame Blanking
ITU	International Telecommunications Union
MHL	Mobile High-definition Link
MPEG	Moving Picture Experts Group
MSC	MHL Sideband Channel
OTG	On-the-Go, a type of USB system.
OUI	Organizationally Unique Identifier (IEEE). See Section 5.9.2.
PCB	Printed Circuit Board
RAP	Request Action Protocol
RCP	Remote Control Protocol
Rx	Receiver
RxSense	Receiver Sense
SMPTE	Society of Motion Picture & Television Engineers
STB	Set-Top Box
TERC4	TMDS Error Reduction Coding – 4 bit
TMDS	Transition Minimized Differential Signaling, or more broadly, both transition-minimized and transition-maximized differential signaling, as in, for example, “logical channel” and “physical channel”.
Tx	Transmitter

Term	Meaning
USB	Universal Serial Bus
VBUS	Voltage Bus
VSIF	Vendor Specific InfoFrame

3 Overview

MHL carries video, audio and auxiliary data from an MHL Source to an MHL Sink across a single, high-speed, differential signal pair – the TMDS Channel. Multiple logical channels are multiplexed onto one physical channel, where the bit stream is then modulated by a clock signal running at a fixed ratio to the video pixel rate, and then transmitted from the Source. The receiver uses the clock modulation as a frequency reference for data recovery from the physical channel. TMDS encoding in the Source converts the 8 bits of data into a 10 bit DC-balanced, transition minimized sequence which is then transmitted serially across the signal pair.

In addition, MHL carries a single-wire Control Bus (CBUS). The CBUS is used for configuration and status exchange between a single Source and a single Sink and provides high-level control functions between all of the various audiovisual products in a user's environment. The CBUS, for example, effectively replaces the DDC bus used in a standard DVI connection. It also carries the MHL Sideband Channel (MSC) which provides high-level control functions between the Source and the Sink.

Finally, MHL has a dedicated VBUS pin and associated ground pin to provide power between the Sink and the Source.

This set of signals is shown in Figure 3-1.

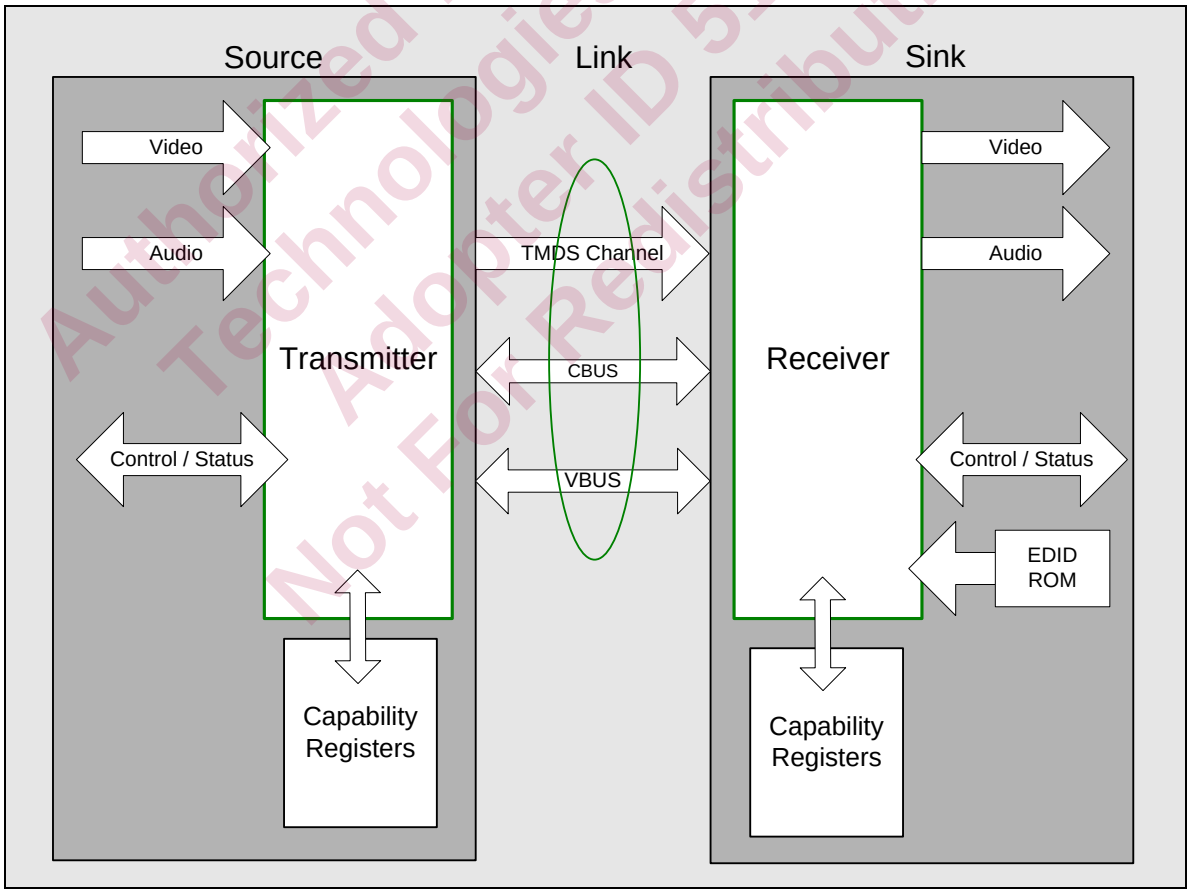


Figure 3-1. MHL Source and Sink Block Diagram

The Link frequency is up to 75 MHz. In the 24 bits per pixel modes, this allows for pixel clock rates of up to 75 MHz, e.g. 720p/1080i. Video pixel rates may range from 25 MHz to 150 MHz. Video formats with rates below 25 MHz (e.g. 13.5 MHz for 480i/NTSC) may be transmitted using

a pixel-repetition scheme. The video pixels may be encoded in either RGB, YCbCr 4:4:4 or YCbCr 4:2:2 formats. In all three cases, 24 bits per pixel are transferred.

In MHL's PackedPixel mode, only 16 bits per pixel are transferred. The incoming pixel clock rate may be higher than 75 MHz in this mode, which thereby allows MHL to support 1080p/60 video.

In order to transmit audio and auxiliary data across the TMDS channels, MHL uses a packet structure. In order to attain the higher reliability required of audio and control data, this data is protected with a BCH error correction code and is encoded using a special TERC4 error reduction coding to produce the 10-bit word that is transmitted.

Basic audio functionality consists of a single IEC 60958 audio stream at sample rates of 32kHz, 44.1kHz or 48kHz. This accommodates any normal stereo stream. Optionally, MHL may carry a single such stream at sample rates up to 192kHz. MHL may also carry IEC 61937 compressed (e.g. surround-sound) stream at sample rates up to 192kHz.

The MHL system architecture is defined to consist of Source devices and Sink devices. All MHL output ports on these devices shall follow the rules for an MHL Source. All MHL input ports shall follow the rules for an MHL Sink.

The CBUS is used as a point-to-point single-wire bus carrying VESA defined DDC commands and other protocols to improve control of various consumer electronics devices. The CBUS is used by the Source device to read the Sink's EDID ROM in order to discover the Sink's configuration and/or capabilities. The CBUS may also be used by a content protection scheme for handshaking between the Source and Sink. The CBUS is defined to enable easy implementation of an MHL bridge to standard E-DDC commands.

The CBUS is also used to read and write the Device Capability Registers, Status Registers, Interrupt Registers and Scratchpad Registers between two devices on the link.

Note regarding Dongle implementation: An MHL Dongle shall, in general, meet all the requirements of an MHL Sink, since both have an MHL input port. Exceptions to this rule are stated explicitly where necessary in this specification.

3.1 MHL 2.0 Enhancements

MHL 2.0 provides support for 3D video. This allows an MHL Source to output 3D video in any of a set of 3D video modes, to an MHL Sink which also supports multiple 3D video modes. An MHL 2.0 Dongle may be designed to transfer 3D content into a different downstream protocol. The Specification provides mechanisms for Source, Sink and Dongle to become aware of each other's capabilities, and to signal the transmitted video mode when in 3D.

MHL 2.0 increases the minimum amount of power on VBUS required from an MHL Sink, thereby providing more wattage to MHL Sources which want either to re-charge more quickly, or to operate at HD speeds while still charging.

MHL 2.0 provides support for sending character codes across CBUS using added opcodes within the existing MSC_MSG command.

4 Signaling And Encoding

4.1 Overview

4.1.1 Link Architecture

As shown in Figure 4-1, an MHL link consists of a single differential pair for a TMDS Data channel. The same pair carries a MHL Clock signal using common-mode signaling. The MHL Clock signal constantly runs at a rate one times the pixel rate of the transmitted video. This clock is the link clock. Every cycle of the MHL Clock signal corresponds to three TMDS 10-bit

characters in 24-bit mode, and to four TMDS characters for PackedPixel mode. Each 10-bit word is encoded using one of several different coding techniques.

The input stream to the transmitter's encoding logic contains video pixel, packet and control data. The packet data consists of audio and auxiliary data and associated error correction codes.

These data items are processed in a variety of ways and are presented to the TMDS encoder as either 2 bits of control data, 4 bits of packet data or 8 bits of video data on the TMDS channel. The Source device encodes one of these data types on any given clock cycle.

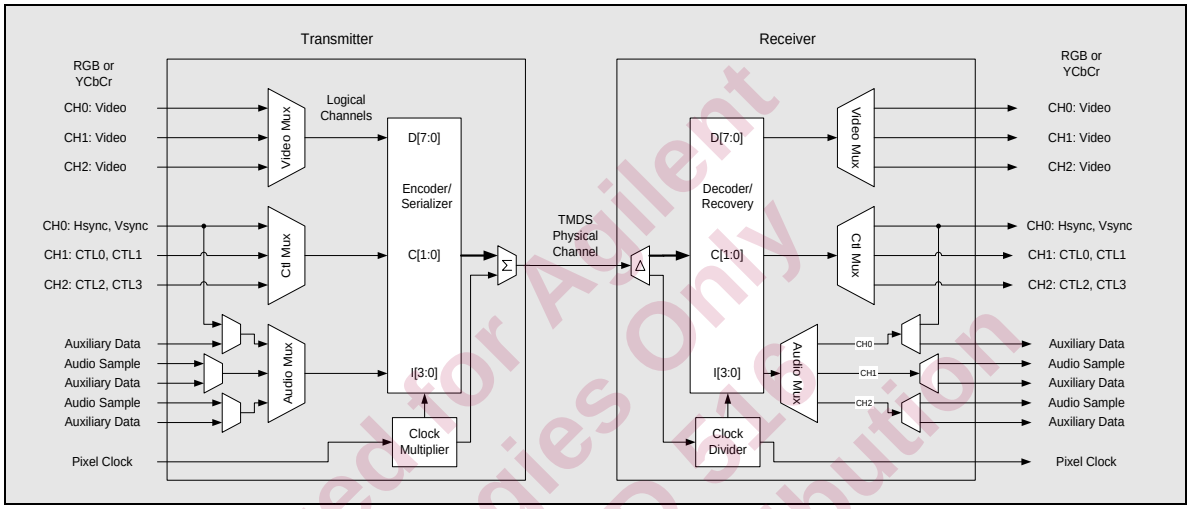


Figure 4-1. MHL Transmitter and Receiver Block Diagram

All MHL devices shall comply with the requirements specified in EIA/CEA-861E except where specifically noted in this document.

4.1.2 Physical and Logical Channels

Channel multiplexing and de-multiplexing are shown in the following two sections. PackedPixel mode is shown separately, to make clear its use of Logical Channels.

The encryption and decryption blocks in Figure 4-2 and Figure 4-3 are optional, and included here to show that the encryption and decryption occur on the Logical Channels.

The Period Control Signal is shown conceptually to select between encoding in Video Data periods, Data Island periods or Control periods, according to the type of data being transmitted. A similar Period Control Signal may be used in decoding.

4.1.2.1 24-Bit Mode Channel Multiplexing

In 24-bit mode, MHL uses a single differential channel to transmit TMDS data that, for example, is carried over three differential channels, in DVI. The incoming 3-channel data needs to be packed into one TMDS channel on the TX side, and unpacked into 3 channels on the RX side. There is one TMDS encoder on the TX side, and one TMDS decoder on the RX side. Conceptually, TX data is multiplexed into a single lane prior to being encoded. RX data is decoded prior to being de-multiplexed into three lanes.

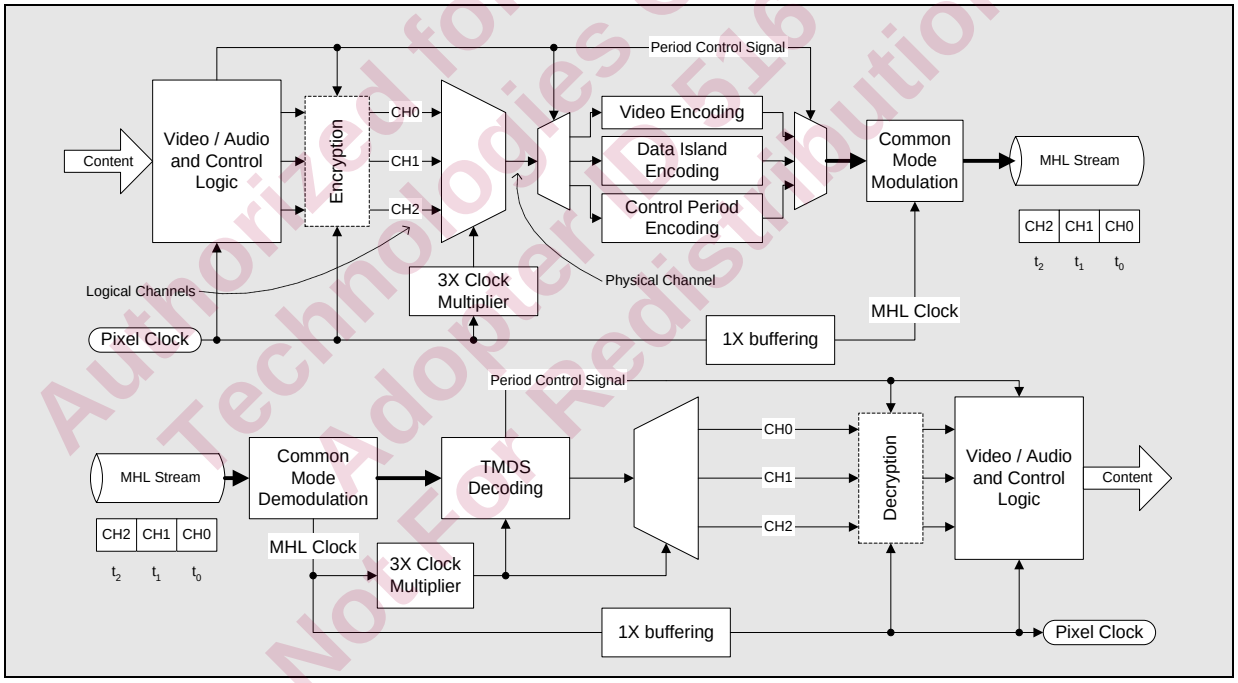


Figure 4-2. MHL Encoder/Decoder Overview – 24-Bit-per-Pixel Mode

Due to lane multiplexing, MHL has a concept of Physical and Logical Channels. There is one Physical Channel, composed of the single differential pair carrying TMDS data. There are three Logical Channels, composed of the three data streams that are time-domain multiplexed onto the Physical Channel. In the remainder of this document, a reference to Channel 0, Channel 1 or Channel 2 shall be construed to refer to these Logical Channels. The data on the link is sent in the following sequence: Channel 0, Channel 1, Channel 2, Channel 0, Channel 1, Channel 2… One clock period of the pixel clock equals one period of the MHL clock, a period in which three TMDS characters are transmitted.

Note that the 'Pixel Clock' shown in Figure 4-2 may be a multiple of the pixel rate, if pixel repetition is used. See Section 5.4.

4.1.2.2 PackedPixel Mode Channel Multiplexing

In PackedPixel mode, only two channels are multiplexed: Channel 0 and Channel 1. One clock period of the MHL clock equals two periods of the pixel clock. Each period of the MHL clock transmits four TMDS characters.

Note that the MHL link clock frequency must be in the range shown in Table 13-6.

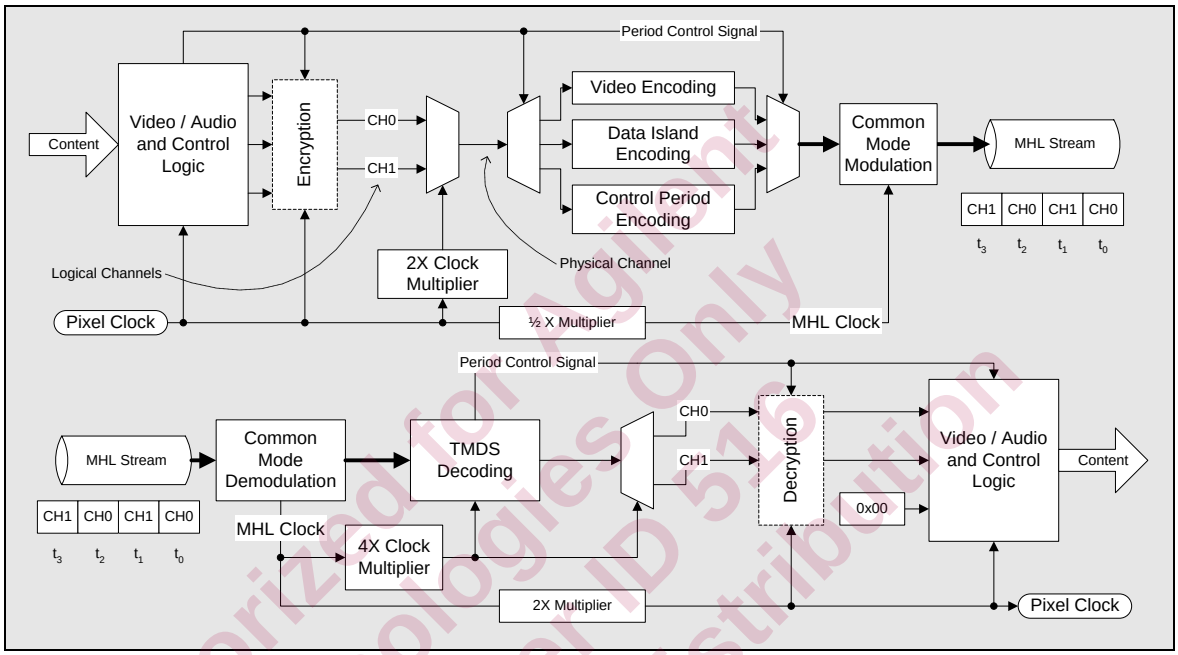


Figure 4-3. MHL Encoder/Decoder Overview – PackedPixel Mode

Note that the 'Pixel Clock' shown in Figure 4-3 may be a multiple of the pixel rate, if pixel repetition is used. See Section 5.4.

4.1.3 Operating Modes Overview

The MHL link operates in one of three modes: Video Data Period, Data Island Period, and Control Period. During the Video Data Period, the active pixels of an active video line are transmitted. During the Data Island Period, audio and auxiliary data are transmitted using a series of packets. The Control Period is used when no video, audio, or auxiliary data needs to be transmitted. A short Control Period is required between any other two periods.

An example of each period placement is shown in the following figure.

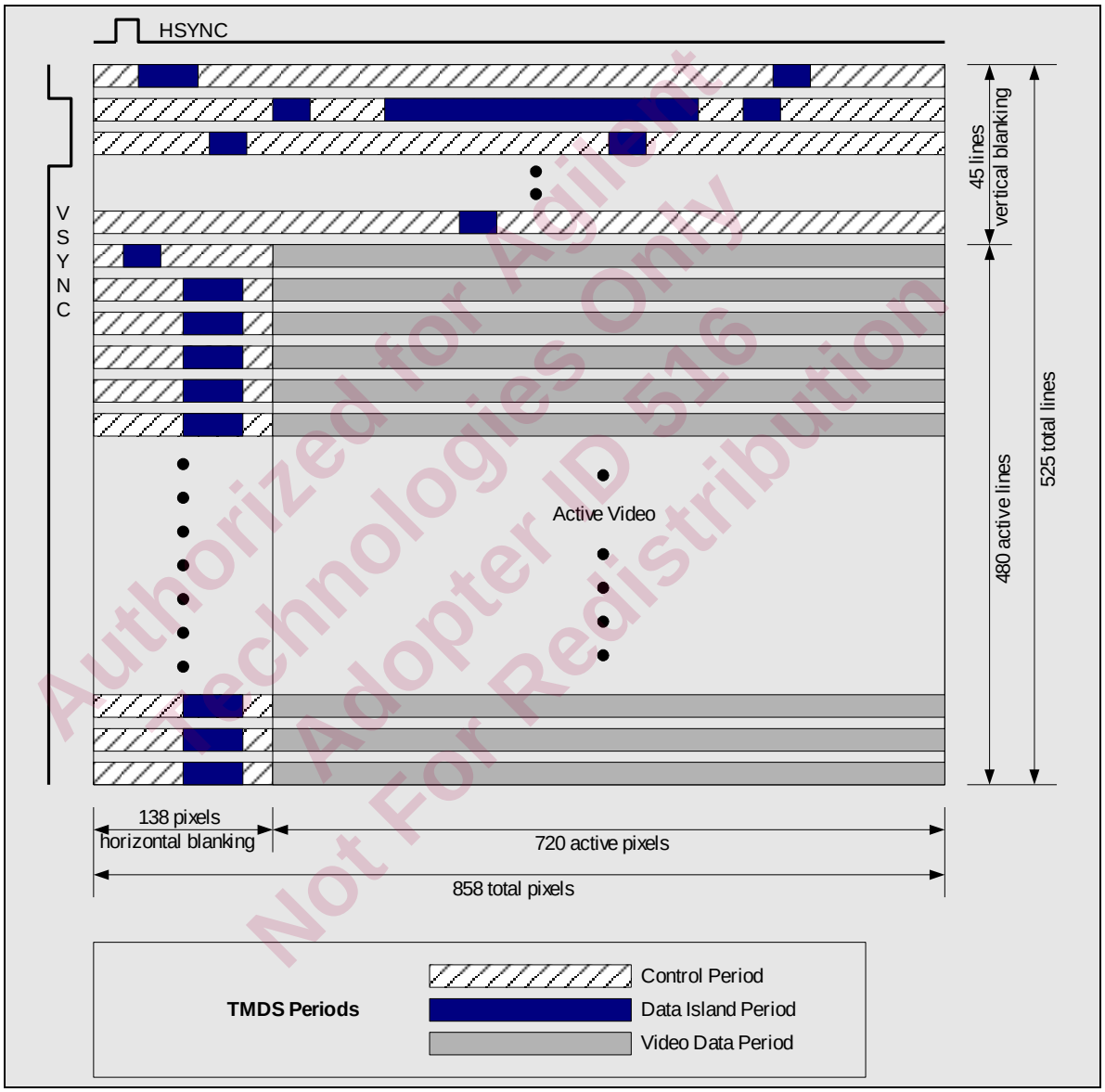


Figure 4-4. Informative Example: TMDS periods in 480p video frame

Video Data Periods use transition minimized coding to encode 8 bits per logical channel, or 16 or 24 bits total per pixel. Each Video Data Period includes its leading guard band characters as well. These are not shown in Figure 4-4, but are shown in Figure 4-6.

Data Island Periods are encoded using a similar transition minimized coding, TMDS Error Reduction Coding (TERC4), which transmits 4 bits per logical channel, or 12 bits total per pixel clock period in 24-bit mode. In PackedPixel mode only one channel uses TERC4; the other uses 8-bit-to-10-bit TMDS encoding.

Control Periods use transition maximized coding to encode 2 bits per logical channel, or 6 bits total per clock period. These 6 bits are HSYNC, VSYNC, CTL0, CTL1, CTL2 and CTL3.

Table 4-1 shows encoding type used and data transmitted during each operation mode.

Table 4-1. Encoding type and data to be transmitted

Structure	Duration	TMDS Class	Fixed/Variable[1]
HSYNC	Continuous	Transition-maximized[2,4,10]	Variable
VSYNC	Continuous	Transition-maximized[2,4,10]	Variable
Data Island Leading Guard Band	2 pixel clocks	Transition-minimized[8,11]	Fixed/Variable[4]
Preamble	8 pixel clocks	Transition-maximized[10]	Fixed
Packet Data6	32 pixel clocks	TERC4 Transition-minimized[9]	Variable
Data Island Trailing Guard Band	2 pixel clocks	Transition-minimized[8,11]	Fixed/Variable[4]
Video Period Leading Guard Band	2 pixel clocks	Transition-minimized[8,11]	Fixed[5]
Control Data	4 pixel clocks min.	Transition-maximized[10]	Fixed[1]
Video Data	Video mode line	Transition-minimized[7]	Variable

Notes on Table 4-1:

1. 'Fixed' denotes a constant value used to indicate the structure, such as guard bands. 'Variable' denotes a set of values which indicate the structure but also encode additional signal information, such as the HIGH or LOW value of HSYNC.
2. HSYNC and VSYNC are coded as Transition-maximized during control periods, and as TERC4 Transition-minimized during data islands.
3. 'Pixel clock' may be a multiple of the pixel rate, if pixel repetition is used. See Section 5.4.
4. Within a data island period (from the leading guard band through the trailing guard band), the HSYNC state and the VSYNC state are encoded in TERC4 characters on logical channel 0. Logical channels 1 and 2 carry fixed values during guard bands.
5. In video data periods, the leading guard band is a fixed value character.
6. Packets consist of packet header data, packet data and packet ECC data. A Data Island Period may contain one or more packets. See Section 4.2.1.4.
7. Transition-minimized data in these cases may be encoded from 8 bits to 9 bits before multiplexing, with the tenth bit added to encoding to control DC disparity. Refer to Figure 4-12 for more details.
8. Transition-minimized data in these cases may be assigned from fixed 10-bit values.
9. Transition-minimized data in these cases may be encoded from 4 bits to 10 bits.
10. Transition-maximized data is created from a fixed list of values, only in part depending on the value of HSYNC, VSYNC, or CTL0-3. Refer to Section 4.4 for more details.
11. Leading and trailing guard band characters are not encrypted by HDCP, even when the packets themselves (32 clocks each) are encrypted. Refer to HDCP specification.

4.1.4 Character Synchronization

The TMDS receiver needs to determine the location of character boundaries in the serial data streams. Once character boundaries are established on all logical data channels, the receiver is defined to be synchronized to the serial streams, and may recover TMDS characters from the logical data channels for decode. The TMDS data stream provides periodic cues for decoder synchronization.

The TMDS characters used during the Video Data Period and Data Island Period contain five or fewer transitions, while the TMDS characters used during the Control Period contain seven or more transitions. A transition is a change from a '1' to a '0' bit, or from a '0' to a '1' bit, at any point from the trailing edge of q_out[0] to and including the leading edge of q_out[9]. (See Section 4.4.2.) The high-transition content of the characters transmitted during the Control Period form the basis for character boundary synchronization at the decoder. While these characters are not individually unique in the serial data stream, they are sufficiently alike that the decoder may uniquely detect the presence of a succession of them during transmitted synchronization intervals. The exact algorithm for this detection is an implementation detail beyond the scope of this document, but minimum conditions for receiver synchronization are defined.

Table 4-2. Transition Count per TMDS Character

Minimum Transitions	Maximum Transitions	Data Period Type
0	5	Transition-minimized Characters
7	9	Transition-maximized Characters

The receiver is required to establish synchronization with the data stream during any Control Period greater than or equal to T_{SMIN} characters in length.

Prior to synchronization detection, and during periods of lost synchronization, the receiver shall not update the signals of the recovered stream. This includes the states of VSYNC, HSYNC, CTL0, CTL1, CTL2 and CTL3.

The transmitter is also required to transmit an occasional long Control Period of at least T_{CTL_LONG} pixels, at least once every $T_{CTL\_LONG\_PER}$ seconds.

The TMDS protocol timing specifications are listed in Table 13-3 (page 225).

4.1.5 Data Island Error Correction

To improve the reliability of the data and to improve the detection of bad data, Error Correction Code (ECC) is added to each packet. BCH(64,56) and BCH(32,24) are generated by the polynomial G(x) shown in Figure 4-5 and Equation 1. The sequence repeats on a 127 count cycle.

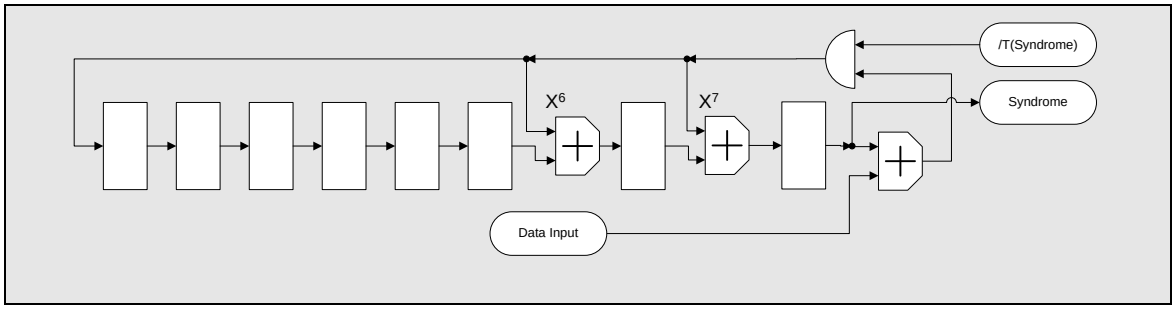


Figure 4-5. Error Correction Code generator

Equation 1. BCH Polynomial

$$G(x) = 1 + x^6 + x^7 + x^8$$

4.1.6 Data Island Placement and Duration

The Source device is required to determine the temporal placement and duration of the Data Island with respect to the video signal's horizontal and vertical blanking periods and synchronization signals. It shall do so following the rules stated below.

1) All Control Periods shall be at least T_{SMIN} pixels long. See Table 13-3.
2) The Data Island shall contain at least one packet, limiting its minimum size to 36 pixels.
3) Data Islands shall contain an integer number of packets. In order to assure the reliability of the data within the data island, they shall be limited to 18 packets or fewer.
4) Zero, one or more data islands may occur between subsequent video data periods.
5) An MHL Source device shall transmit at least one Data Island during every two Video Fields, unless the Source detects, through a Dongle, that the downstream non-MHL Sink device cannot handle data islands, in which case the Source may send a stream which meets the limitations of that downstream non-MHL Sink. See Section 4.5 for an example of this.

4.2 Operating Modes

MHL supports two types of channel encoding across the single data pair on the cable. Data may be transported using all three logical channels, or data may be transported using two logical channels (PackedPixel mode). The encoding and decoding are different for these two modes, and are described in Section 4.2.1 and Section 4.2.2.

The primary differences between the conventional 24-bit-per-pixel modes and the PackedPixel mode are that in PackedPixel mode only 16 bits are sent per pixel, and TERC4 encoding is only used on logical channel 0 of PackedPixel streams. This reduces the number of bits needed in each packet such that data islands fit into two logical channels instead of three logical channels.

4.2.1 24-Bit-per-Pixel Mode

In the 24-bit-per-pixel mode, the eight data bits in each of the three logical channels are allocated as diagrammed in Figure 4-6. The MHL protocol alternates between transition-minimized TMDS characters and transition-maximized TMDS characters.

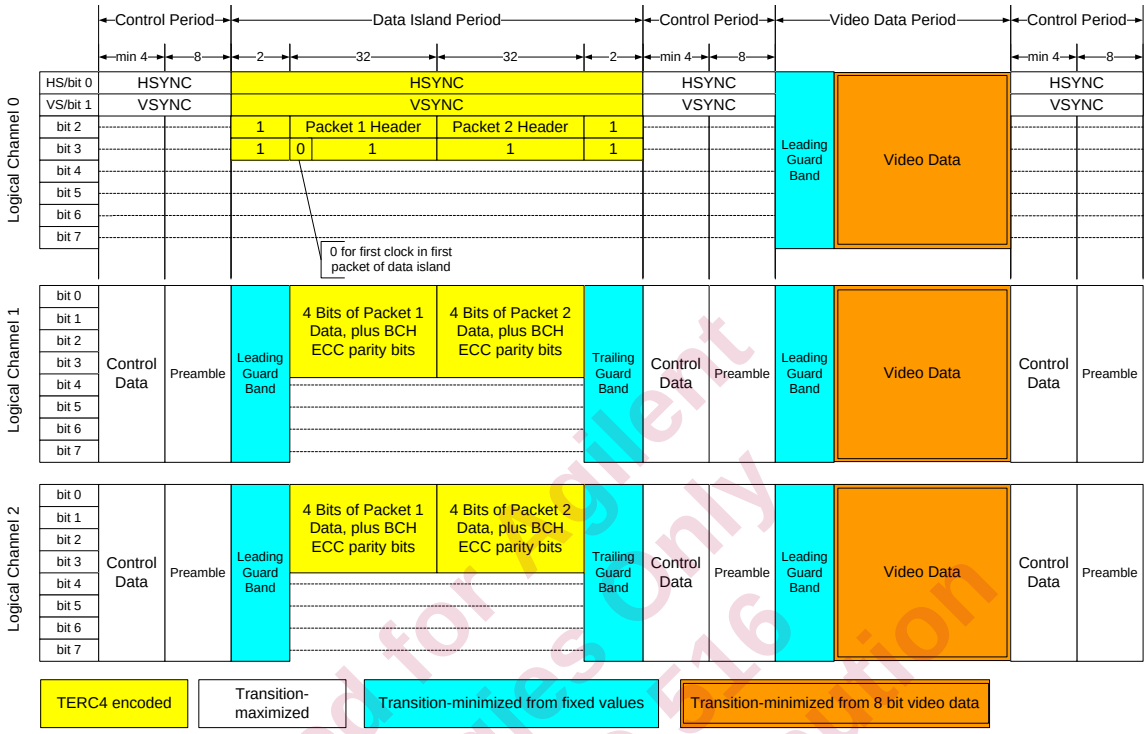


Figure 4-6. 24-Bit-per-Pixel Mode – Preamble and Data Island Channel Bit Allocation

Four types of encoding are shown in Figure 4-6:

Encoding		Section
TERC4 Encoding	These characters are each formed from 4 data bits.	4.4.4
Transition-maximized	These characters are each formed from 2 control bits.	4.4.3
Transition-minimized from fixed values	These characters are each formed from a list of fixed values, according to the type of guard band.	4.2.1.1
Transition-minimized from 8 bit video data	These characters are each formed from 8 data bits.	4.4.5

4.2.1.1 24-Bit-per-Pixel Mode Guard Bands

Each Video Data Period and Data Island Period starts with a Leading Guard Band designed to provide robust determination of the transition from the Control Period to the Video Data Period or to the Data Island Period. This Leading Guard Band consists of two special characters.

The Data Island Period is also protected by a Trailing Guard Band which is designed to provide a robust determination of the transition from the Data Island Period to the Control Period.

The guard band characters are part of the Video Data Period or the Data Island Period, as shown in Figure 4-6.

Table 4-3. Guard Band Values

Guard Band	Logical Channel 0	Logical Channel 1	Logical Channel 2
Video Leading	0b1011001100	0b0100110011	0b1011001100
Data Island Leading	See note 1.	0b0100110011	0b0100110011
Data Island Trailing	See note 1.	0b0100110011	0b0100110011

Notes on Table 4-3:

1. During the Data Island Guard Bands, Logical Channel 0 is encoded as one of four TERC4 values. These data values are 0xC, 0xD, 0xE and 0xF, depending upon the values of HSYNC and VSYNC, with the TERC4 encoding shown on page 49.
2. Leading and trailing guard band characters are not encrypted by HDCP, even when HDCP is enabled and the data packets are encrypted. Refer to the HDCP Specification.
3. The semantics of this syntax for binary fields is explained in Section 1.6.

4.2.1.2 24-Bit-per-Pixel Mode Control Period

Control Period is used for transmission of Preamble characters, which includes the states of VSYNC and HSYNC. The Control Period is also used by the Sink device for character synchronization. Other usage in this period is reserved. See Section 4.4.3 for details on encoding values of CTL0, CTL1, CTL2, CTL3, VSYNC and HSYNC during the Control Period.

4.2.1.2.1 Preamble

Each Control Period ends in a sequence of Preamble characters that indicate whether the upcoming data period is a Video Data Period or is a Data Island Period.

The Preamble contains the Data Period Descriptor which consists of 8 pixels of identical Control characters. The values of CTL0, CTL1, CTL2, and CTL3, indicate the type of data period that follows. The remaining Control signals, HSYNC and VSYNC, can vary during this sequence.

Table 4-4. Preamble Indicator of Data Period Type

CTL0	CTL1	CTL2	CTL3	Data Period Type
1	0	0	0	Video Data Period
1	0	1	0	Data Island Period

Only logical channels 1 and 2 participate in the preamble. Channel 0 is used to send the control characters for HSYNC and VSYNC. The 10-bit values corresponding to each pair of control characters (CTL0/1, CTL2/3) are shown in Section 4.4.3 on page 48.

The Video Data Period preamble indicates that the following data period contains video data in RGB 4:4:4 or YCbCr 4:4:4 or YCbCr 4:2:2 format, beginning with a Video Guard Band.

The Data Island Period preamble indicates that the following data period is an MHL-compliant Data Island, beginning with a Data Island Guard Band.

The transition from TMDS control characters to Guard Band characters following this sequence identifies the start of the Video Data Period or Data Island Period.

4.2.1.3 24-Bit-per-Pixel Mode Video Data Period

Video data periods are used to carry the pixel data of an active video line.

Each Video Data Period is preceded by a Preamble, described in Section 4.2.1.2.1.

Following the Preamble, the Video Data Period begins with a two pixel Video Leading Guard Band, described in Section 4.2.1.1. There is no Trailing Guard Band for the Video Data Period.

During active video periods, 24 bits of pixel data are encoded using TMDS transition minimized encoding.

4.2.1.4 24-Bit-per-Pixel Mode Data Island Period

4.2.1.4.1 Data Island Overview

Data Islands are used to carry packets of audio sample data and auxiliary data. This auxiliary data includes EIA/CEA-861E InfoFrames and other data describing the active audio or video stream or describing the Source device.

Each Data Island is preceded by a Preamble, described in Section 4.2.1.2.1.

Following the Preamble, each Island starts with a Leading Guard Band, described in Section 4.2.1.1. The first packet of the Data Island then follows.

During the Data Island, each of the three Logical channels transmits a series of 10-bit characters encoded from a 4-bit input word, using TMDS Error Reduction Coding (TERC4). TERC4 reduces the error rate on the link by choosing only 10-bit codes with high inherent error avoidance. TERC4 encoding is defined in detail in Section 4.4.4.

During every clock period of the Data Island, including the guard band, bits 0 and 1 of Logical Channel 0 transmit an encoded form of HSYNC and VSYNC.

Bit 2 of Logical Channel 0 is used to transmit the Packet Header. All four bits of Logical Channels 1 and 2 are used for the Packet data as shown in Figure 4-6. Each packet is 32 clocks long and is protected by BCH ECC for error correction and detection purposes.

The last characters of the Data Island, following the last packet, are the Trailing Guard Band.

Following the Data Island, all three channels revert to transmitting control characters.

4.2.1.4.2 Data Island Guard Bands

The first two data characters within the Data Island are the Leading Guard Band. The last two data characters within the Data Island are the Trailing Guard Band.

During the Data Island Guard Bands, Logical Channel 0 is encoded as one of four TERC4 values, depending upon the values of HSYNC and VSYNC. These are shown on page 49, with the bit order [9:0].

4.2.1.4.3 Data Island Packet Construction

All data within a Data Island is contained within 32 byte Packets. Packets consist of a Packet Header, four Subpackets, and associated error correction bits. Each Subpacket includes 56 bits of data and is protected by an additional 8 bits of BCH ECC parity bits.

Subpacket 0 plus its corresponding parity bits make up BCH Block 0. This block is mapped onto bit 0 of both Logical Channels 1 and 2. In this way, the 64 bits of BCH Block 0 are transferred over the course of 32 pixels. Likewise, BCH Block 1 (Subpacket 1 plus parity) is striped on bit 1 of both Logical Channels 1 and 2.

In the tables below, Header bytes are indicated as HB0, HB1, and HB2 and Subpacket bytes are indicated as SB0 to SB6.

Subpacket 0 bytes 0 through 6 (SB0-SB6) are also designated Packet bytes 0 to 6 (PB0-PB6).

Subpacket 1 bytes 0 through 6 (SB0-SB6) are also designated Packet bytes 7 to 13 (PB7-PB13).

Subpacket 2 bytes 0 through 6 (SB0-SB6) are also designated Packet bytes 14 to 20 (PB14-PB20).

Subpacket 3 bytes 0 through 6 (SB0-SB6) are also designated Packet bytes 21 to 27 (PB21-PB27).

This is illustrated in Figure 4-7.

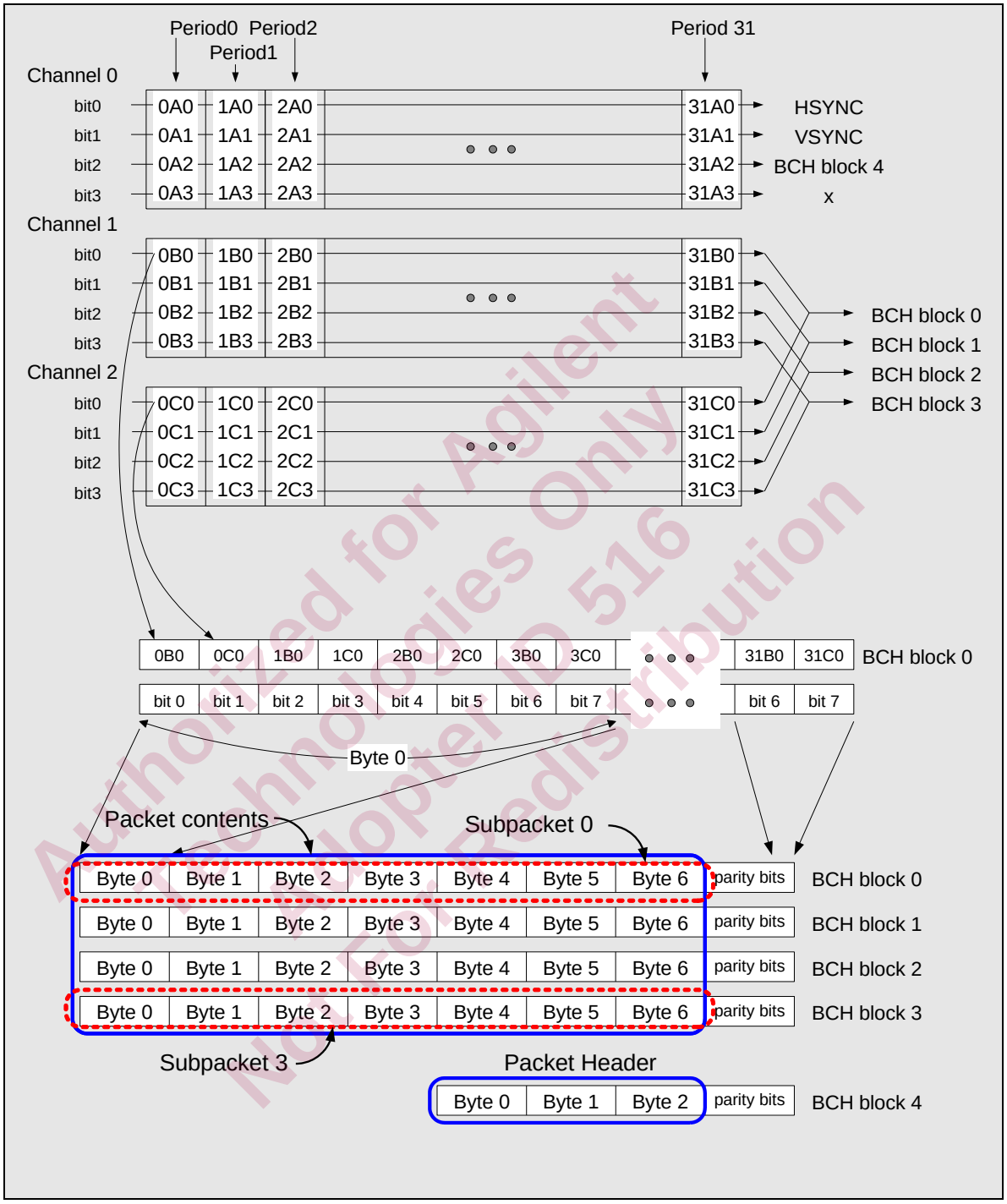


Figure 4-7. 24-Bit-per-Pixel Mode – Data Island Packet and ECC Structure

4.2.2 PackedPixel Mode

In PackedPixel mode, only two logical channels are multiplexed onto the single MHL data pair. The encoding of the data onto the two logical channels is shown in Figure 4-8.

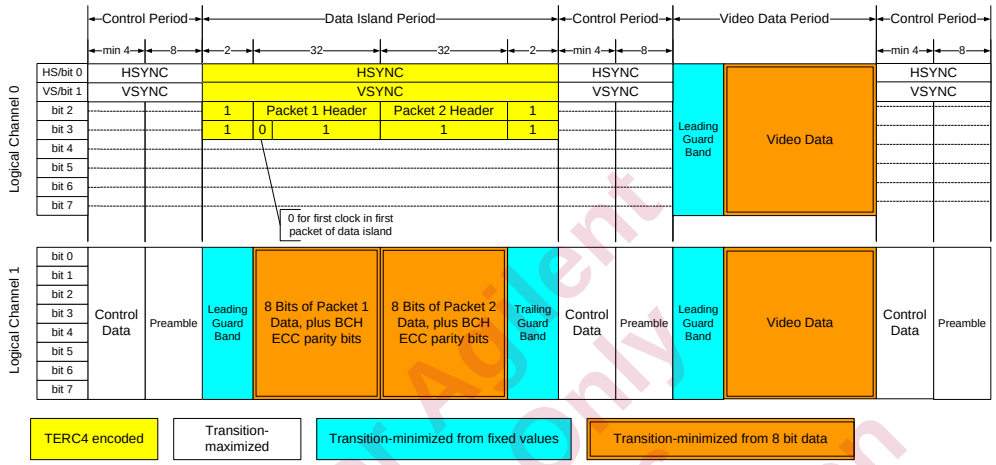


Figure 4-8. PackedPixel Mode – Preamble and Data Island Channel Bit Allocation

Four types of encoding are shown in Figure 4-8:

Encoding	Definition	Section
TERC4 Encoding	These characters are each formed from 4 data bits.	4.4.4
Transition-maximized	These characters are each formed from 2 control bits.	4.4.3
Transition-minimized from fixed values	These characters are each formed from a list of fixed values, according to the type of guard band.	4.2.2.1
Transition-minimized from 8 bit data	These characters are each formed from 8 data bits.	4.4.5

PackedPixel mode is optional for Source, Sink and Dongle devices. Support for it is indicated in the Capability Registers in the SUPP_PPIXEL field. See Table 7-30.

4.2.2.1 PackedPixel Mode Guard Bands

Each Video Data Period and Data Island Period starts with a Leading Guard Band designed to provide robust determination of the transition from the Control Period to the Video Data Period or to the Data Island Period. This Leading Guard Band consists of two special characters.

The Data Island Period is also protected by a Trailing Guard Band which is designed to provide a robust determination of the transition from the Data Island Period to the Control Period.

The guard band characters are part of the Video Data Period or the Data Island Period, as shown in Figure 4-8.

Table 4-5. PackedPixel Mode Guard Band Values

Guard Band	Logical Channel 0	Logical Channel 1
Video Leading	0b0100110011	0b1011001100
Data Island Leading	See note 1.	0b0100110011

Data Island Trailing	See note 1.	0b0100110011

Notes on Table 4-5:

1. During the Data Island Guard Bands, Logical Channel 0 is encoded as one of four TERC4 values. These data values are 0xC, 0xD, 0xE and 0xF, depending upon the values of HSYNC and VSYNC, with the TERC4 encoding shown on page 49.
2. Leading and trailing guard band characters are not encrypted by HDCP, even when HDCP is enabled and the data packets are encrypted. Refer to the HDCP Specification.

4.2.2.2 PackedPixel Mode Control Period

Control Period is used for transmission of Preamble characters, which includes the states of VSYNC and HSYNC. The Control Period is also used by the Sink device for character synchronization. Other usage in this period is reserved. See Section 4.4.3 for details on encoding values of CTL0, CTL1, CTL2, CTL3, VSYNC and HSYNC during the Control Period.

In PackedPixel mode, as there are only two logical channels, the control characters are handled differently than in the 24-bit-per-pixel modes. HSYNC and VSYNC are sent on logical channel 0. CTL2 and CTL3 are sent on logical channel 1. CTL0 and CTL1, which are constant at all times while in this mode, are not transported at all. They are reconstructed in the receiver side of the link.

4.2.2.2.1 Preamble

Each Control Period ends in a sequence of Preamble characters that indicate whether the upcoming data period is a Video Data Period or is a Data Island Period.

The Preamble contains the Data Period Descriptor which consists of 8 pixels of identical Control characters. The values of CTL2, and CTL3, indicate the type of data period that follows. HSYNC and VSYNC, can vary during this sequence.

Table 4-6. Preamble Indicator of Data Period Type

CTL2	CTL3	Data Period Type
0	0	Video Data Period
1	0	Data Island Period

Only logical channel 1 participates in the preamble. Channel 0 is used to send the control characters for HSYNC and VSYNC. The 10-bit values corresponding to each pair of control characters (CTL2/3) are shown in Section 4.4.3 on page 48.

The Video Data Period preamble indicates that the following data period contains video data in the PackedPixel format, beginning with a Video Guard Band.

The Data Island type indicates that the following data period is an MHL-compliant Data Island, beginning with a Data Island Guard Band.

The transition from TMDS control characters to Guard Band characters following this sequence identifies the start of the Data Period.

4.2.2.3 PackedPixel Mode Video Data Period

Video data periods are used to carry the pixel data of an active video line.

Each Video Data Period is preceded by a Preamble, described in Section 4.2.2.2.1.

Following the Preamble, the Video Data Period begins with a two pixel Video Leading Guard Band, described in Section 4.2.2.1. There is no Trailing Guard Band for the Video Data Period.

During active video periods, 16 bits of pixel data are encoded using TMDS transition minimized encoding.

4.2.2.4 PackedPixel Mode Data Island Period

4.2.2.4.1 Data Island Overview

Data Islands are used to carry packets of audio sample data and auxiliary data. This auxiliary data includes EIA/CEA-861E InfoFrames and other data describing the active audio or video stream or describing the Source device.

Each Data Island is preceded by a Preamble, described in Section 4.2.2.2.1.

Following the Preamble, each Island starts with a Leading Guard Band, described in Section 4.2.2.1. The first packet of the Data Island then follows.

During the Data Island, logical channel 0 transmits a series of 10-bit characters encoded from 4 bits of information. TERC4 coding is used.

During the Data Island, logical channel 1 transmits a series of 10-bit characters encoded from an 8-bit input word using TMDS Video Data Coding Algorithm (see Figure 4-12). Note that TERC4 encoding is not used.

During every clock period of the Data Island, including the guard band, bits 0 and 1 of Logical Channel 0 transmit an encoded form of HSYNC and VSYNC.

Bit 2 of Logical Channel 0 is used to transmit the Packet Header. All eight bits of Logical Channel 1 are used for the Packet data as shown in Figure 4-8. Each packet is 32 clocks long and is protected by BCH ECC for error correction and detection purposes.

The last characters of the Data Island, following the last packet, are the Trailing Guard Band.

Following the Data Island, the two channels revert to transmitting control characters.

4.2.2.4.2 Data Island Guard Bands

The first two data characters within the Data Island are the Leading Guard Band. The last two data characters within the Data Island are the Trailing Guard Band.

During the Data Island Guard Bands, Logical Channel 0 is encoded as one of four TERC4 values, depending upon the values of HSYNC and VSYNC. These are shown on page 49.

4.2.2.4.3 Data Island Packet Construction

All data within a Data Island is contained within 32 byte Packets. Packets consist of a Packet Header, four Subpackets, and associated error correction bits. Each Subpacket includes 56 bits of data and is protected by an additional 8 bits of BCH ECC parity bits.

Subpacket 0 plus its corresponding parity bits make up BCH Block 0. This block is mapped onto bit 0 and bit 4 of Logical Channel 1. In this way, the 64 bits of BCH Block 0 are transferred over the course of 32 pixels. Likewise, BCH Block 1 (Subpacket 1 plus parity) is striped on bit 1 and bit 5 of Logical Channel 1.

In the tables below, Header bytes are indicated as HB0, HB1, and HB2 and Subpacket bytes are indicated as SB0 to SB6.

Subpacket 0 bytes 0 through 6 (SB0-SB6) are also designated Packet bytes 0 to 6 (PB0-PB6).

Subpacket 1 bytes 0 through 6 (SB0-SB6) are also designated Packet bytes 7 to 13 (PB7-PB13).

Subpacket 2 bytes 0 through 6 (SB0-SB6) are also designated Packet bytes 14 to 20 (PB14-PB20).

Subpacket 3 bytes 0 through 6 (SB0-SB6) are also designated Packet bytes 21 to 27 (PB21-PB27).

This is illustrated in Figure 4-9.

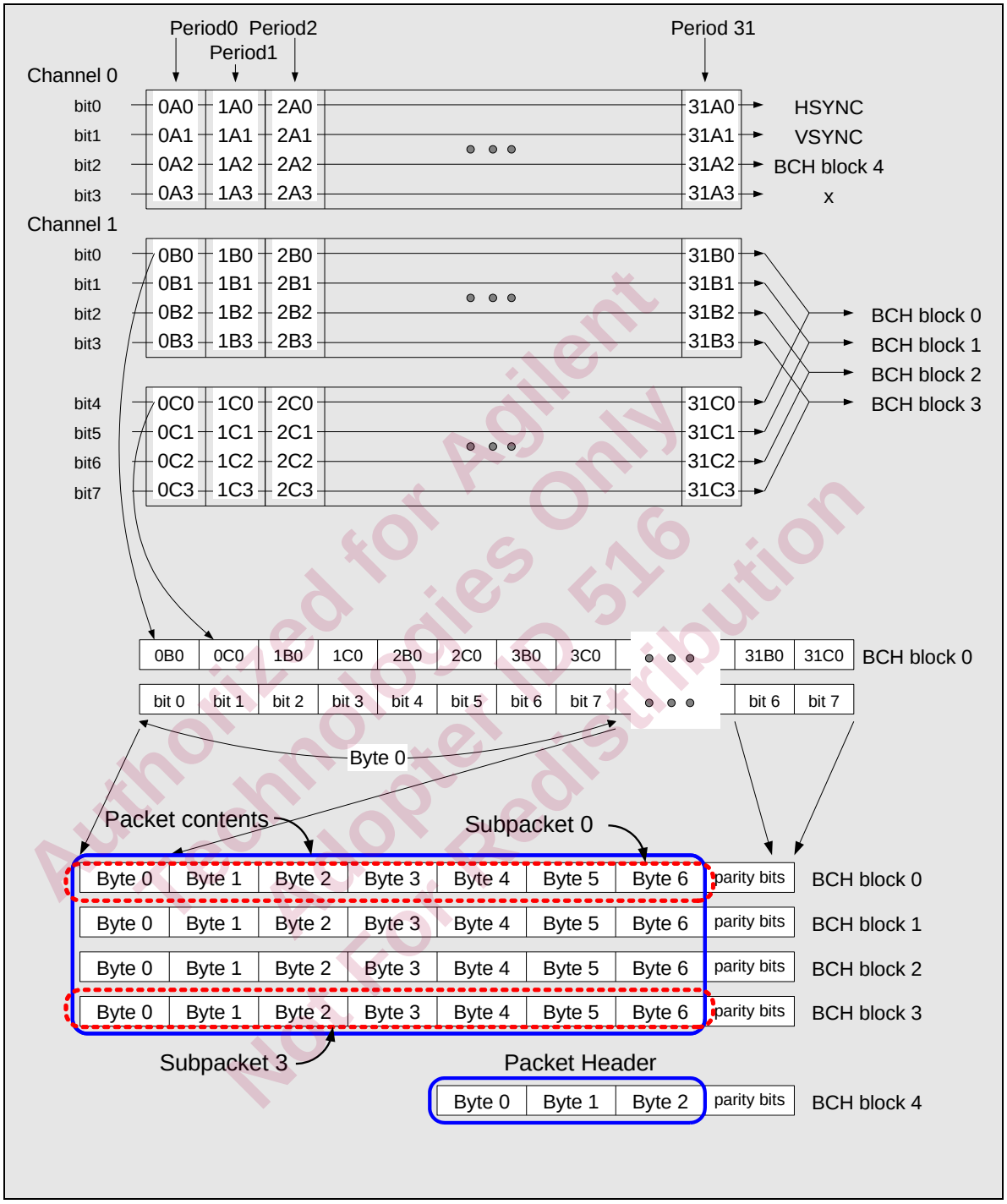


Figure 4-9. PackedPixel Mode – Data Island Packet and ECC Structure

4.3 Data Island Packet Definitions

An MHL Sink shall be able to handle any data island packet which meets the timing and format requirements in this specification. An MHL Sink may discard any packet with a packet type not listed in Table 4-8 and any packet with a checksum whose checksum does not properly match the decoded data.

4.3.1 Packet Header

Packet Headers contain 24 data bits with an additional 8 bits of BCH(32,24) ECC parity. These parity bits are calculated over the 24 bits of the Packet Header.

A Packet Header includes an 8-bit Packet Type and 16 bits of packet-specific data.

Table 4-7. Packet Header

Byte \ Bit #	7	6	5	4	3	2	1	0
HB0	Packet Type							
HB1	packet-specific data							
HB2	packet-specific data							

Table 4-8. Packet Types

Packet Type Value	Packet Type	Section
0x00	Null	4.3.2
0x01	Audio Clock Regeneration (N/CTS)	4.3.3
0x02	Audio Sample	4.3.4
0x03	Content Mute Packet	4.3.5
0x80+InfoFrame Type	EIA/CEA-861E InfoFrame	4.3.6

4.3.2 Null Packet

Null packets can be used by the Source device anytime. It is acceptable to transmit a Null packet whenever a Source device needs to continue a Data Island but cannot deliver useful data. All bytes of a Null packet shall contain only zero values. An MHL Sink shall ignore bytes HB1 and HB2 of the Null Packet header and all data bytes of the Null Packet.

Table 4-9. Null Packet

Byte \ Bit #	7	6	5	4	3	2	1	0
HB0	0	0	0	0	0	0	0	0
HB1	0	0	0	0	0	0	0	0
HB2	0	0	0	0	0	0	0	0
PB0	0	0	0	0	0	0	0	0
PB1	0	0	0	0	0	0	0	0
PB2	0	0	0	0	0	0	0	0
PB3	0	0	0	0	0	0	0	0
PB4	0	0	0	0	0	0	0	0
PB5	0	0	0	0	0	0	0	0
PB6	0	0	0	0	0	0	0	0
PB7	0	0	0	0	0	0	0	0
PB8	0	0	0	0	0	0	0	0
PB9	0	0	0	0	0	0	0	0
PB10	0	0	0	0	0	0	0	0
PB11	0	0	0	0	0	0	0	0
PB12	0	0	0	0	0	0	0	0
PB13	0	0	0	0	0	0	0	0
PB14	0	0	0	0	0	0	0	0
PB15	0	0	0	0	0	0	0	0
PB16	0	0	0	0	0	0	0	0
PB17	0	0	0	0	0	0	0	0
PB18	0	0	0	0	0	0	0	0
PB19	0	0	0	0	0	0	0	0
PB20	0	0	0	0	0	0	0	0
PB21	0	0	0	0	0	0	0	0
PB22	0	0	0	0	0	0	0	0
PB23	0	0	0	0	0	0	0	0
PB24	0	0	0	0	0	0	0	0
PB25	0	0	0	0	0	0	0	0
PB26	0	0	0	0	0	0	0	0
PB27	0	0	0	0	0	0	0	0

- X = Reserved, shall be zero. An MHL Sink device shall ignore these bits.

4.3.3 Audio Clock Regeneration Packet

Audio Clock Regeneration Packets contain both the N and CTS values used in the Audio Clock Regeneration process. The four Subpackets each contain the same Audio Clock Regeneration Subpacket. (Bits in fields are numbered consecutively. Not all bits are shown with labels in Table 4-10, to make the table more readible.)

Table 4-10. Audio Clock Regeneration Packet

Byte \ Bit #	7	6	5	4	3	2	1	0
HB0	0	0	0	0	0	0	0	1
HB1	X	X	X	X	X	X	X	X
HB2	X	X	X	X	X	X	X	X
PB0	0	0	0	0	0	0	0	0
PB1	0	0	0	0	CTS_19	-	-	CTS_16
PB2	CTS_15	-	-	-	-	-	-	CTS_8
PB3	CTS_7	CTS_6	CTS_5	CTS_4	CTS_3	CTS_2	CTS_1	CTS_0
PB4	0	0	0	0	N_19	-	-	N_16
PB5	N_15	-	-	-	-	-	-	N_8
PB6	N_7	N_6	N_5	N_4	N_3	N_2	N_1	N_0
PB7	0	0	0	0	0	0	0	0
PB8	0	0	0	0	CTS_19	-	-	CTS_16
PB9	CTS_15	-	-	-	-	-	-	CTS_8
PB10	CTS_7	CTS_6	CTS_5	CTS_4	CTS_3	CTS_2	CTS_1	CTS_0
PB11	0	0	0	0	N_19	-	-	N_16
PB12	N_15	-	-	-	-	-	-	N_8
PB13	N_7	N_6	N_5	N_4	N_3	N_2	N_1	N_0
PB14	0	0	0	0	0	0	0	0
PB15	0	0	0	0	CTS_19	-	-	CTS_16
PB16	CTS_15	-	-	-	-	-	-	CTS_8
PB17	CTS_7	CTS_6	CTS_5	CTS_4	CTS_3	CTS_2	CTS_1	CTS_0
PB18	0	0	0	0	N_19	-	-	N_16
PB19	N_15	-	-	-	-	-	-	N_8
PB20	N_7	N_6	N_5	N_4	N_3	N_2	N_1	N_0
PB21	0	0	0	0	0	0	0	0
PB22	0	0	0	0	CTS_19	-	-	CTS_16
PB23	CTS_15	-	-	-	-	-	-	CTS_8
PB24	CTS_7	CTS_6	CTS_5	CTS_4	CTS_3	CTS_2	CTS_1	CTS_0
PB25	0	0	0	0	N_19	-	-	N_16
PB26	N_15	-	-	-	-	-	-	N_8
PB27	N_7	N_6	N_5	N_4	N_3	N_2	N_1	N_0

- X = Reserved, shall be zero. An MHL Sink device shall ignore these bits in HB1 and HB2.
- N: value of audio clock regeneration "N" [20 bits]
- CTS: Cycle Time Stamp [20 bits]. Zero indicates no new value of CTS.

4.3.4 Audio Sample Packet

Audio Sample Packets consist of one to four Audio Samples. These may be different samples or different partial samples (e.g., 2 of 6 channels). The configuration of the subpackets is determined by the layout and sample present bits in the header. This is described in detail in Section 6.7, Audio Data Packetization.

- layout: indicates which of two possible subpacket/audio sample layouts are used.
- sample present.spX: indicates if subpacket X contains an audio sample.
- sample flat.spX: indicates if subpacket X represents a "flatline" sample. Only valid if "sample present.spX" is also set.
- For the audio sample data (see Table 4-11):
- B.X Set to 1 if subpacket X contains the first frame in an IEC 60958 block; = 0 otherwise
- L.X Bit corresponding to Time Slot X from first ("left") sub-frame
- R.X Bit corresponding to Time Slot X from second ("right") sub-frame
- V_L,V_R Valid bit from first ("left") sub-frame and second ("right") sub-frame
- U_L,U_R User Data bit from first ("left") sub-frame and second ("right") sub-frame
- C_L,C_R Channel Status bit from first ("left") sub-frame and second ("right") sub-frame
- P_L,P_R Parity bit (even parity) from first ("left") sub-frame and second (*"right") sub-frame
- Sub-frame is defined in IEC 60958-1 Specification.
- (Bits in fields are numbered consecutively. Not all bits are shown with labels in Table 4-11, to make the table more readible.)

Table 4-11. Audio Sample Packet

Byte \ Bit #	7	6	5	4	3	2	1	0
HB0	0	0	0	0	0	0	1	0
HB1	0	0	0	layout	sample present.sp3	sample present.sp2	sample present.sp1	sample present.sp0
HB2	B.3	B.2	B.1	B.0	sample flat.sp3	sample flat.sp2	sample flat.sp1	sample flat.sp0
PB0	L.11	…	…	…	…	…	…	L.4
PB1	L.19	…	…	…	…	…	…	L.12
PB2	L.27	…	…	…	…	…	…	L.20
PB3	R.11	…	…					R.4
PB4	R.19	…	…	…	…	…	…	R.12
PB5	R.27	…	…	…	…	…	…	R.20
PB6	P_R	C_R	U_R	V_R	P_L	C_L	U_L	V_L
PB7	L.11	…	…	…	…	…	…	L.4
PB8	L.19	…	…	…	…	…	…	L.12
PB9	L.27	…	…	…	…	…	…	L.20
PB10	R.11	…	…					R.4
PB11	R.19	…	…	…	…	…	…	R.12
PB12	R.27	…	…	…	…	…	…	R.20
PB13	P_R	C_R	U_R	V_R	P_L	C_L	U_L	V_L
PB14	L.11	…	…	…	…	…	…	L.4
PB15	L.19	…	…	…	…	…	…	L.12
PB16	L.27	…	…	…	…	…	…	L.20
PB17	R.11	…	…					R.4
PB18	R.19	…	…	…	…	…	…	R.12
PB19	R.27	…	…	…	…	…	…	R.20
PB20	P_R	C_R	U_R	V_R	P_L	C_L	U_L	V_L
PB21	L.11	…	…	…	…	…	…	L.4
PB22	L.19	…	…	…	…	…	…	L.12
PB23	L.27	…	…	…	…	…	…	L.20
PB24	R.11	…	…					R.4
PB25	R.19	…	…	…	…	…	…	R.12
PB26	R.27	…	…	…	…	…	…	R.20
PB27	P_R	C_R	U_R	V_R	P_L	C_L	U_L	V_L

4.3.5 Content Mute Packet

The Content Mute Packet allows the source device to suppress the content data while maintaining an active connection to the sink device. Although the content data is blocked, the control signals (HSYNC, VSYNC, CTL0, CTL1, CTL2 and CTL3) shall continue to be transmitted across the link with the same timings as required elsewhere in this specification. In addition, all other data islands shall maintain their structure as defined in this specification, except that the content data itself may be muted. (See the definition of "blank video" on page 17.) In PackedPixel mode, CTL0 and CTL1 are not sent from transmitter to receiver during data island periods, but are re-constituted at the receiver side.

In this context, 'mute' may mean set to all zeroes, set to 'black' (for video) and 'silent' for audio, or any other color and audio value which does not convey the original content stream.

Muted video data shall be kept within the valid range allowed by the link mode. In the case of YCbCr modes, the video data shall neither be below 'black' nor above 'white'.

Muted audio data shall continue to include valid channel status and other 'non-audio'' (V, U, P) bits as defined in Section 4.3.4.

The Content Mute Packet is transmitted once per frame or field, during the vertical blanking interval when a Content Mute Packet is required. Like the Null Packet, the Content Mute Packet consists of four subpackets, each of 7 bytes. All four subpackets shall have identical content. The Content Mute Packet shall be transmitted in its entirety during the vertical blanking Control Period, no earlier than the active edge of the VSYNC signal, and no later than 384 clock periods after the active edge of the VSYNC signal.

The Content Mute Packet UNMUTE bit and MUTE bits shall not both be set to one. A muted state is defined by MUTE = 1 and UNMUTE = 0. An un-muted state is defined by MUTE = 0 and UNMUTE = 1.

An MHL Source may send a Content Mute Packet with both MUTE and UNMUTE set to zero. An MHL Sink which receives a Content Mute Packet which has both MUTE and UNMUTE set to zero may interpret that packet as "no change to the current state of mute". An MHL Sink may automatically reset Mute mode if UNMUTE is not received after a period of time.

The purpose of this function is to block content while changing modes on the link. Normally, if content protection is enabled before such a mode change, the source will stop the content protection while muting and then re-enable content protection when muting is removed. Additional guidelines may be found in the HDCP Specification. This process reduces the loss of HDCP synchronization between the encrypting source and the decrypting sink, which may occur when the link clock changes frequency or control signals (such as VSYNC) are modified.

The Content Mute Packet layout is shown in Table 4-12.

Table 4-12. Content Mute Packet

Byte \ Bit #	7	6	5	4	3	2	1	0
HB0	0	0	0	0	0	0	1	1
HB1	X	X	X	X	X	X	X	X
HB2	X	X	X	X	X	X	X	X
PB0	X	X	X	UNMUTE	X	X	X	MUTE
PB1	X	X	X	X	X	X	X	X
PB2	X	X	X	X	X	X	X	X
PB3	X	X	X	X	X	X	X	X
PB4	X	X	X	X	X	X	X	X
PB5	X	X	X	X	X	X	X	X
PB6	X	X	X	X	X	X	X	X
PB7	X	X	X	UNMUTE	X	X	X	MUTE
PB8	X	X	X	X	X	X	X	X
PB9	X	X	X	X	X	X	X	X
PB10	X	X	X	X	X	X	X	X
PB11	X	X	X	X	X	X	X	X
PB12	X	X	X	X	X	X	X	X
PB13	X	X	X	X	X	X	X	X
PB14	X	X	X	UNMUTE	X	X	X	MUTE
PB15	X	X	X	X	X	X	X	X
PB16	X	X	X	X	X	X	X	X
PB17	X	X	X	X	X	X	X	X
PB18	X	X	X	X	X	X	X	X
PB19	X	X	X	X	X	X	X	X
PB20	X	X	X	X	X	X	X	X
PB21	X	X	X	UNMUTE	X	X	X	MUTE
PB22	X	X	X	X	X	X	X	X
PB23	X	X	X	X	X	X	X	X
PB24	X	X	X	X	X	X	X	X
PB25	X	X	X	X	X	X	X	X
PB26	X	X	X	X	X	X	X	X
PB27	X	X	X	X	X	X	X	X

- X = Reserved, shall be zero. An MHL Sink device shall ignore these bits.

4.3.6 InfoFrame Packet

EIA/CEA-861E InfoFrames are carried across MHL using the MHL InfoFrame packet, with the following restrictions.

- Each InfoFrame shall be at most 30 bytes long, including the Type Code (Packet Type), Version number and InfoFrame length fields.
- InfoFrame Type Codes shall be within the range of 0 to 127.
- Only InfoFrames defined in EIA/CEA-861E or in this specification may be transmitted.

InfoFrames are each carried in their own packet, as shown below.

Table 4-13. InfoFrame Packet Header

Byte \ Bit #	7	6	5	4	3	2	1	0
HB0	1	InfoFrame Type						
HB1	InfoFrame_version							
HB2	0	0	0	InfoFrame_length				

- InfoFrame Type: lower 7 bits of the InfoFrame type code as per EIA/CEA-861E.
- InfoFrame_version: version number of InfoFrame as per EIA/CEA-861E.
- InfoFrame_length: InfoFrame length in bytes as per EIA/CEA-861E. This length does not include any of the bytes in the Packet Header nor the checksum byte. The maximum value for this field is 27 (0x1B).

Table 4-14. InfoFrame Packet Contents

Byte \ Bit #	7	6	5	4	3	2	1	0
PB0	checksum							
PB1	Data Byte 1							
PB2	Data Byte 2							
PB3…PB26	…							
PB27	Data Byte 27							

- checksum: Checksum of the InfoFrame. The checksum shall be calculated such that a byte-wide sum of all three bytes of the Packet Header and all valid bytes of the InfoFrame Packet contents (determined by InfoFrame_length), plus the checksum itself, equals zero.
- Data Byte x: Data Byte X of the InfoFrame as defined in EIA/CEA-861E.
- Each InfoFrame type is described in detail in Section 10.2, beginning on page 201.

4.4 Encoding

4.4.1 Lane Multiplexing and De-multiplexing

The Source shall multiplex the logical data channels into a single channel prior to completing TMDS encoding.

4.4.1.1 Lane Multiplexing in 24-Bit-per-Pixel Mode

The order of transmission for 24-bit-per-pixel mode is shown in Figure 4-10. The first character to be transmitted during a video data period shall be Channel 0, followed by Channel 1 and Channel 2. The same shall be true during control or data island periods.

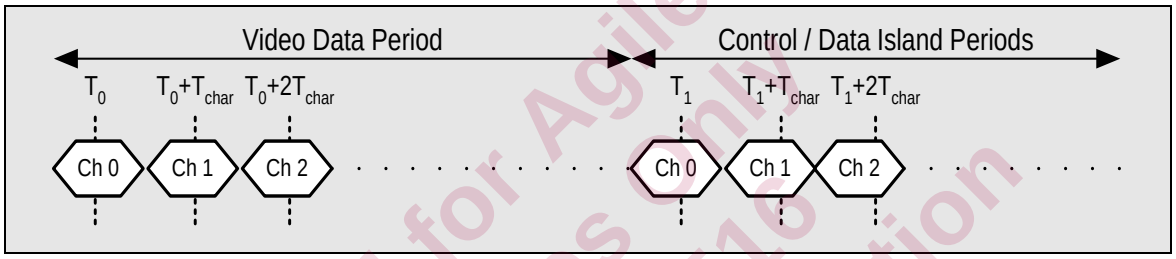


Figure 4-10. 24-Bit-per-Pixel Mode MHL Lane Multiplexing

4.4.1.2 Lane Multiplexing in PackedPixel Mode

The order of transmission for PackedPixel Mode is shown in Figure 4-11. The first character to be transmitted during a video data period shall be Channel 0, followed by Channel 1. The same shall be true during control or data island periods.

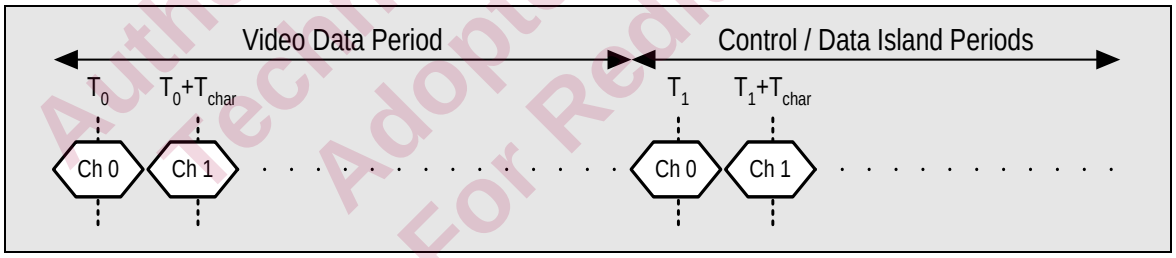


Figure 4-11. PackedPixel Mode MHL Lane Multiplexing

4.4.2 Serialization

The stream of TMDS characters produced by the encoder is serialized for transmission on the TMDS data channel. In the discussions that follow, the least significant bit of each character (q_out[0]) is the first bit to be transmitted and the most significant bit (q_out[9]) is the last.

4.4.3 Control Period Coding

Each Logical Channel has two control signals which are encoded into 10 bits during Control Periods. For each of the three logical channels these signals are shown in Table 4-15.

The two Control signals for each of the three logical channels are encoded as follows:

```
case (C1, C0):
       0, 0:  q_out[9:0] = 0b1101010100;
       0, 1:  q_out[9:0] = 0b0010101011;
       1, 0:  q_out[9:0] = 0b0101010100;
       1, 1:  q_out[9:0] = 0b1010101011;
```

```
endcase;
```

Table 4-15. Control-signal assignment

Logical Channel	C0	C1	Notes
0	HSYNC	VSYNC	
1	CTL0	CTL1	1
2	CTL2	CTL3	

Notes on Table 4-15:

1. CTL0 and CTL1 are constant during PackedPixel mode data islands.

4.4.4 TERC4 Coding

TMDS Error Reduction Coding (TERC4) is used during the Data Island period to encode and decode 4 bits per logical channel into the 10 bits serialized and transmitted.

```
Case (bit3, bit2, bit1, bit0):
    0000: q_out[9:0] = 0b1010011100;
    0001: q_out[9:0] = 0b1001100011;
    0010: q_out[9:0] = 0b1011100100;
    0011: q_out[9:0] = 0b1011100010;
    0100: q_out[9:0] = 0b0101110001;
    0101: q_out[9:0] = 0b0100011110;
    0110: q_out[9:0] = 0b0110001110;
    0111: q_out[9:0] = 0b0100111100;
    1000: q_out[9:0] = 0b1011001100;
    1001: q_out[9:0] = 0b0100111001;
    1010: q_out[9:0] = 0b0110011100;
    1011: q_out[9:0] = 0b1011000110;
    1100: q_out[9:0] = 0b1010001110;
    1101: q_out[9:0] = 0b1001110001;
    1110: q_out[9:0] = 0b0101100011;
    1111: q_out[9:0] = 0b1011000011;
endcase;
```

4.4.5 Video Data Coding

4.4.5.1 Video Data Encoding

The following is a description of the encoding algorithm used during transmission of video data. A detailed description of an encoder is given. Other implementations are possible and are permitted but, given the same sequence of input characters, they are required to produce the same sequence of output (10-bit) characters that is generated by the described encoder.

During video data periods, when each 10-bit character contains 8 bits of pixel data, the encoded characters provide an approximate DC balance as well as a reduction in the number of transitions in the data stream. The encode process for the active data period can be viewed in two stages. The first stage produces a transition-minimized 9-bit code word from the input 8 bits. The second stage produces a 10-bit code word, the finished TMDS character, which will manage the overall DC balance of the transmitted stream of characters.

The 9-bit code word produced by the first stage of the encoder is made up of an 8-bit representation of the transitions found in the input 8 bits, plus a one-bit flag to indicate which of two methods was used to describe the transitions. In both cases the least significant bit of the output matches the least significant bit of the input. With a starting value established, the remaining 7 bits of the output word are derived from sequential exclusive OR (XOR) or exclusive NOR (XNOR) functions of each bit of the input with the previously derived bit. The choice between XOR and XNOR logic is made such that the encoded values contain the fewest possible transitions, and the ninth bit of the code word is used to indicate whether XOR or XNOR functions were used to derive the output code word. The decode of this 9-bit code word is simply a matter of applying either XOR or XNOR gates to the adjacent bits of the code, with the least significant bit passing from decoder input to decoder output unchanged.

The second stage of the encoder performs an approximate DC balance on the transmitted stream by selectively inverting the 8 data bits of the 9-bit code words produced by the first stage. A tenth bit is added to the code word, to indicate when the inversion has been made. The encoder determines when to invert the next character based on the running disparity between ones and zeros that it tracks in the transmitted stream, and the number of ones and zeros found in the current code word. If too many ones have been transmitted and the input contains more ones than zeros, the code word is inverted. This second stage may be performed after the logical channels are multiplexed into the physical channel.

This dynamic encoding decision at the Source device is simply decoded at the receiver by the conditional inversion of the input code word based on the tenth bit of the TMDS character.

The TMDS code mapping is specified by Figure 4-12 with the definitions of Table 4-16. The encoder produces one of 460 unique 10-bit characters. The encoder shall not generate any other 10-bit character during a Video Data Period.

Upon entering a Video Data Period, the data stream disparity (Cnt) shall be considered to be zero by the encoder.

Table 4-16. Encoding Algorithm Definitions

D	The encoder input data set. D is 8-bit pixel data
Cnt	This is a register used to keep track of the data stream disparity. A positive value represents the excess number of "1"s that have been transmitted. A negative value represents the excess number of "0"s that have been transmitted. The expression cnt(t-1) indicates the previous value of the disparity for the previous set of input data. The expression cnt(t) indicates the new disparity setting for the current set of input data.
Q_m	Intermediate value.
Q_out	These 10 bits are the encoded output value.
N1{x}	This operator returns the number of "1"s in argument "x"
N0{x}	This operator returns the number of "0"s in argument "x"

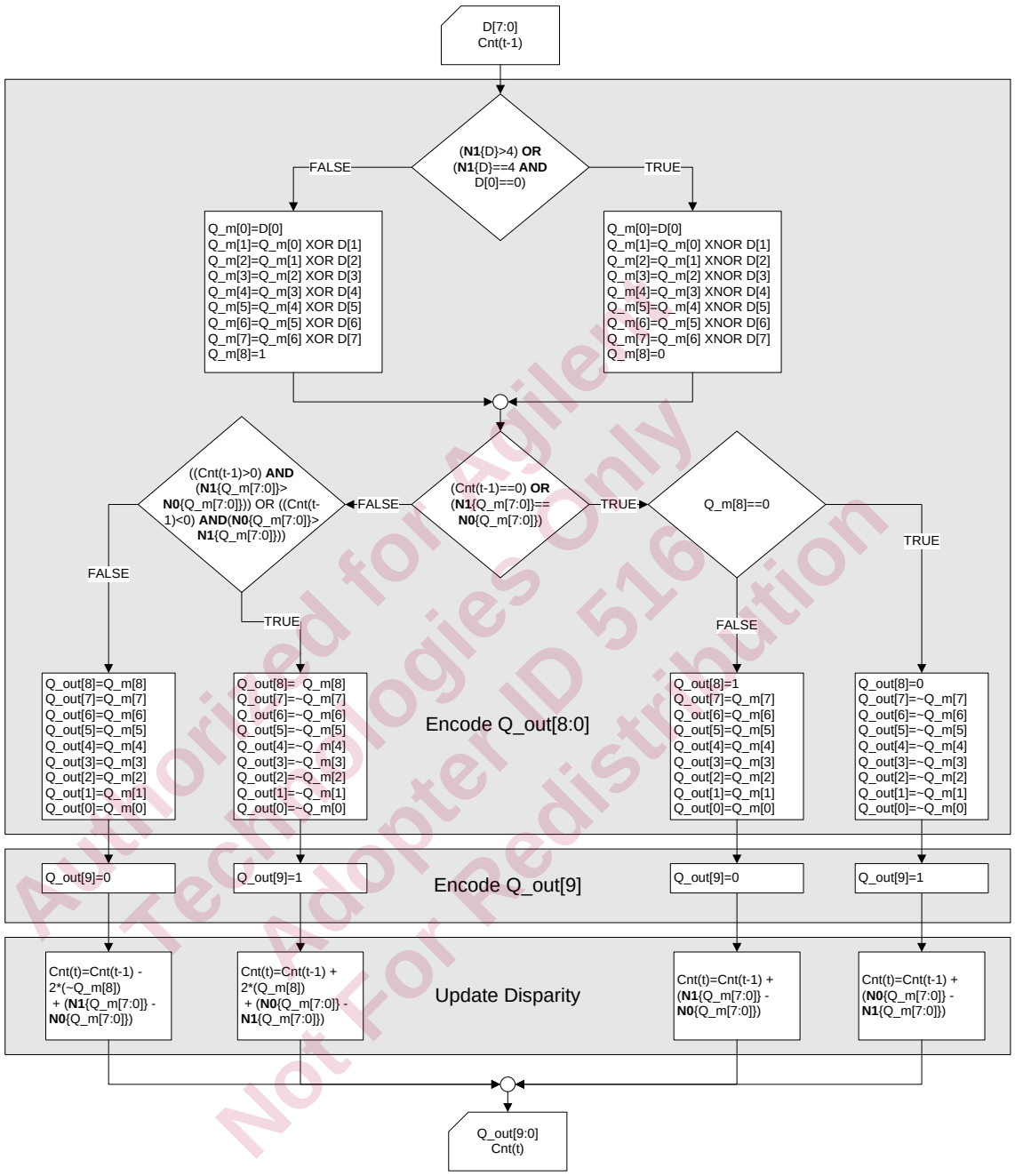


Figure 4-12. TMDS Video Data Encode Algorithm

4.4.5.2 Video Data Decoding

The TMDS decode mapping is specified by Figure 4-13. Alternative implementations are possible but, given the same input data stream, they are required to generate the same output data stream as the described decoder algorithm.

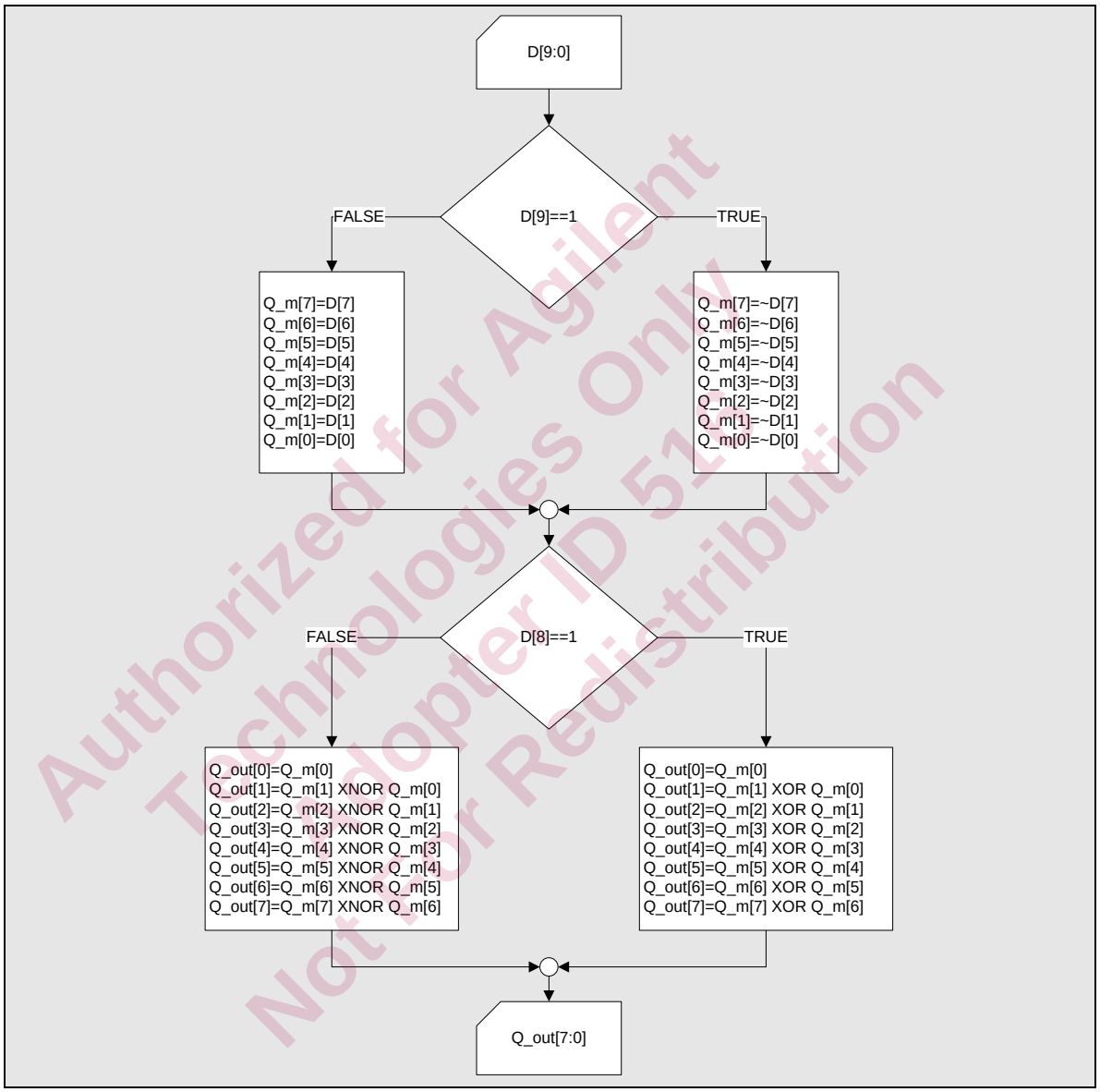


Figure 4-13. TMDS Video Decode Algorithm

4.5 DVI Compatibility

MHL systems cannot communicate directly with DVI systems. However, a bridge can reformat an MHL stream into a DVI compliant stream. Implementation details of such a translation are outside the scope of this specification.

5 Video

5.1 Overview

MHL allows a wide variety of explicitly defined video formats to be transmitted and displayed. These formats define the pixel, line and frame counts and timing and whether the format is interlaced or progressive.

The video pixels carried across the link can be in one of four different pixel encodings: RGB 4:4:4, YCbCr 4:4:4, YCbCr 4:2:2, or PackedPixel.

The MHL Source device determines the pixel encoding and video format and timing of the transmitted signal, based on the characteristics of the source video content, the format and pixel encoding conversions possible at the Source, and the format and pixel encoding capabilities and preferences of the Sink.

5.2 Video Formats

In order to provide maximum compatibility between video sources and sinks, specific minimum requirements have been specified for Source devices and Sink devices.

5.2.1 Format Support Requirements

MHL Source and Sink devices are required to support a minimum set of video formats. The index values shown in Table 5-1 and Table 5-2 are from the EIA/CEA-861E specification.

'Supports' in this context means the ability to send or receive the specified video mode with timings compliant to the EIA/CEA-861E specification.

Other video formats defined in EIA/CEA-861E may be supported. Additional video formats not listed in EIA/CEA-861E may also be supported. See Section 5.2.5.

An MHL Source shall only transmit a video format which the attached Sink has indicated it is capable of receiving. The Sink indicates its supported video modes through its EDID and through the bits in the Video Link Support Register (0x05). See Section 7.9.1.5.

Throughout this specification, many of these video formats are referred to with an abbreviated notation. The video format 720x480p @ 59.94/60 Hz is commonly referred to as 720x480p. The video format 720x576p @ 50 Hz is commonly referred to as 720x576p. The video formats 1920x1080i @ 59.94/60 Hz and 1920x1080i @ 50 Hz are commonly referred to as 1920x1080i. The video formats 1920x1080p @ 59.94/60 Hz and 1920x1080p @ 50 Hz are commonly referred to as 1920x1080p.

5.2.1.1 MHL Source Video Format Support Requirements

The requirements shown in Table 5-1 apply to MHL Sources.

Table 5-1. MHL Source Video Mode Support Requirements

Index	Mode	Refresh	Source	Notes
1	640 x 480 (VGA)	59.94/60	Optional	
2, 3	720 x 480p	59.94/60	Required	1
17, 18	720 x 576p	50	Required	1
4	1280 x 720p	59.94/60	Optional	2
19	1280 x 720p	50	Optional	3
5	1920 x 1080i	59.94/60	Optional	2
20	1920 x 1080i	50	Optional	3
16	1920 x 1080p	59.94/60	Optional	4
31	1920 x 1080p	50	Optional	4
34	1920 x 1080p	30	Optional	
6, 7	720 (1440) x 480i	59.94/60	Optional	5
21, 22	720 (1440) x 576i	50	Optional	5

Notes on Table 5-1:

1. An MHL Source shall support at least one of the following video format timings:
   - 720x480p at 59.94/60Hz refresh
   - 720x576p at 50Hz refresh

   MHL strongly recommends that an MHL Source which supports 720x480p should also support the 720x576p mode. An MHL Source which supports only the 720x480p mode but does not support the 720x576p mode shall indicate to the consumer through the product documentation and the product description that only the 720x480p mode is supported.

   MHL strongly recommends that an MHL Source which supports 720x576p should also support the 720x480p mode. An MHL Source which supports only the 720x576p mode but does not support the 720x480p mode shall indicate to the consumer through the product documentation and the product description that only the 720x576p mode is supported.
2. MHL strongly recommends that an MHL Source which supports the 59.94/60Hz version of this video format timing should also support the output of the 50Hz version of this video format timing. An MHL Source which supports only the 59.94/60Hz version of this video format timing shall indicate to the consumer through the product documentation and the product description that only the 59.94/60Hz mode is supported.
3. MHL strongly recommends that an MHL Source which supports the 50Hz version of this video format timing should also support the output of the 59.94/60Hz version of this video format timing. An MHL Source which supports only the 50Hz version of this video format timing shall indicate to the consumer through the product documentation and the product description that only the 50Hz mode is supported.
4. This video format timing is supported only in PackedPixel mode.
5. This video format timing shall be transmitted using pixel repetition (see Section 5.4). The "pixels per line" count is shown according to the syntax used in EIA/CEA-861E.

5.2.1.2 MHL Sink Video Mode Support Requirements

The requirements shown in Table 5-2 apply to MHL Sinks.

Table 5-2. MHL Sink Video Mode Support Requirements

Index	Mode	Refresh	Sink	Notes
1	640 x 480 (VGA)	59.94/60	Conditional	1
2, 3	720 x 480p	59.94/60	Required	2, 3
17, 18	720 x 576p	50	Required	2, 4
4	1280 x 720p	59.94/60	Conditional	5
19	1280 x 720p	50	Conditional	6
5	1920 x 1080i	59.94/60	Conditional	5
20	1920 x 1080i	50	Conditional	6
16	1920 x 1080p	59.94/60	Optional	7
31	1920 x 1080p	50	Optional	7
34	1920 x 1080p	30	Optional	
6, 7	720 (1440) x 480i	59.94/60	Optional	8
21, 22	720 (1440) x 576i	50	Optional	8

Notes on Table 5-2:

1. An MHL Sink which supports this video format timing on any other input shall support this video format timing on all of its MHL inputs.
2. An MHL Sink which supports this enhanced definition video format timing on any other input shall support this video format timing on all of its MHL inputs.
3. An MHL Sink that accepts 59.94/60Hz video formats shall support this video format on all of its MHL inputs.
4. An MHL Sink that accepts 50Hz video formats shall support this video format on all of its MHL inputs.
5. An MHL Sink that accepts 59.94/60Hz video formats and is HDTV capable shall support index 4 or index 5 video format timings on all of its MHL inputs.
6. An MHL Sink that accepts 50Hz video formats and is HDTV capable shall support index 19 or index 20 video format timings on all of its MHL inputs.
7. This video format timing is supported only in PackedPixel mode.
8. This video format timing shall be transmitted using pixel repetition (see Section 5.4). The "pixels per line" count is shown according to the syntax used in EIA/CEA-861E.

5.2.2 Video Control Signals: HSYNC, VSYNC

During the Data Island period, MHL carries HSYNC and VSYNC using encoded bits on Logical Channel 0. During Video Data periods, MHL does not carry HSYNC and VSYNC and the Sink device should assume that these signals remain constant. During all other periods, MHL carries HSYNC and VSYNC signals through the use of four different control characters on TMDS Logical Channel 0. Refer to Section 4.4.3.

5.2.3 24-Bit Pixel Encoding Requirement

All MHL input and output ports shall be capable of supporting RGB pixel encodings. Only pixel encodings in RGB or the specified YCbCr formats shall be used on MHL.

All MHL inputs on a device shall be capable of supporting both YCbCr 4:4:4 and YCbCr 4:2:2 color space when that device is capable of supporting YCbCr color space from any other analog or digital video input. If an MHL input supports either YCbCr 4:2:2 or YCbCr 4:4:4 then both shall be supported.

All MHL outputs on a device shall support either YCbCr 4:2:2 or YCbCr 4:4:4 color space whenever that device is capable of transmitting a color-difference color space (YCbCr, etc.) across any other analog or digital video interface.

The Sink can indicate if it is capable of accepting YCbCr pixel-encodings. If it indicates support, this indicates that the display can accept both YCbCr pixel-encodings (4:2:2 and 4:4:4). Otherwise, the Sink cannot accept either YCbCr 4:2:2 or YCbCr 4:4:4 encodings. Two fields in the Capability Registers (SUPP_YCBCR444 and SUPP_YCBCR422) are provided to allow a Sink YCbCr support. For completeness, a similar field (SUPP_RGB444) is also provided. See Table 7-30.

An MHL Source device may determine the pixel-encodings that the Sink device understands through the use of the Sink device's Capability Registers. If the Sink indicates that it supports YCbCr –formatted video data and if the Source device is capable of delivering YCbCr data, then the Source may enable the transfer of this data across the link in either 4:2:2 or 4:4:4 format.

5.2.4 PackedPixel Encoding Requirement

An MHL Source may transmit video content using PackedPixel encoding only after it has determined that the connected MHL Sink device is capable to decode PackedPixel encoding.

An MHL Sink which can accept PackedPixel pixel-encodings shall indicate this capability in its Device Capability Registers during the initialization process. A Sink which does not so indicate this capability shall be interpreted by the Source to be unable to receive PackedPixel encoded data.

An MHL Source shall determine the Sink's ability to support PackedPixel pixel-encoding by reading the SUPP_PPIXEL field in the Device Capabilities Registers. See Section 7.9.1.4 for details.

The active link mode shall be indicated by the LINK_MODE field in the Status Registers. An MHL Source shall send an accurate CLK_MODE status value to the MHL Sink using a CBUS command no more than $T_{MODE\_CHG\_DLY}$ after the mode has been changed in the MHL video data stream. A Sink device which does not receive a WRITE_STAT command to set a non-reserved value (i.e., Normal Mode [0b011], or PackedPixel Mode [0b010]) for CLK_MODE, shall default to Normal Mode. See Section 7.9.3 for details.

5.2.5 Vendor-Defined DTV Format

An MHL Source device may support the attached Sink's one or more vendor-defined DTV format under the following conditions:

- MHL Sink device shall use the EDID 18-byte Detailed Timing Descriptor to define each supported, vendor-defined video mode not already defined in a Short Video Descriptor.

The purpose of these vendor-defined DTV formats is to provide a means for manufacturers to send display formats not among those listed in EIA/CEA-861E.

5.3 Video Format Specifications

All specified EIA/CEA-861E video line pixel counts and video field line counts (both active and total) and HSYNC and VSYNC positions and durations shall be adhered to when transmitting a specified video format.

For example, if a Source device is processing material with fewer active pixels per line than the required (i.e. 704 pixels vs. 720 pixels for standard definition MPEG2 material), it could add pixels to the left and right of the supplied material before transmitting across MHL. AVI bar info may need to be adjusted to account for these added pixels.

Detailed timing is found in EIA/CEA-861E for the following primary and optional formats.

5.4 Pixel-Repetition

Video formats with native pixel rates below 25 mega-pixels/second require pixel-repetition in order to be carried across a TMDS link. 720x480i and 720x576i formats shall always be pixel-repeated.

The MHL Source device indicates the use of pixel-repetition with the Pixel Repetition (PRx) field in the AVI InfoFrame. This field indicates to the MHL Sink device how many repetitions of each unique pixel are transmitted. In non-repeated formats, this value is zero.

For pixel-repeated formats, this value indicates the number of pixels that may be discarded by the receiver without losing real image content.

The Source device shall always accurately indicate the pixel repetition count being used. The use of the Pixel Repetition field is optional for MHL Sink devices.

The use of this pixel-repetition count field is more fully described in EIA/CEA-861E.

5.5 Pixel Encodings

There are four different pixel encodings that can be sent across an MHL cable: YCbCr 4:4:4, YCbCr 4:2:2, RGB 4:4:4 and PackedPixel. Whichever encoding is used, it shall conform to one of the methods described in this section.

5.5.1 RGB 4:4:4 Pixel Encoding

Figure 5-1 shows the default encoding, RGB 4:4:4. The R, G, and B components of the first pixel for a given line of video are transferred on the first pixel clock period of the video data period following the Guard Band characters.

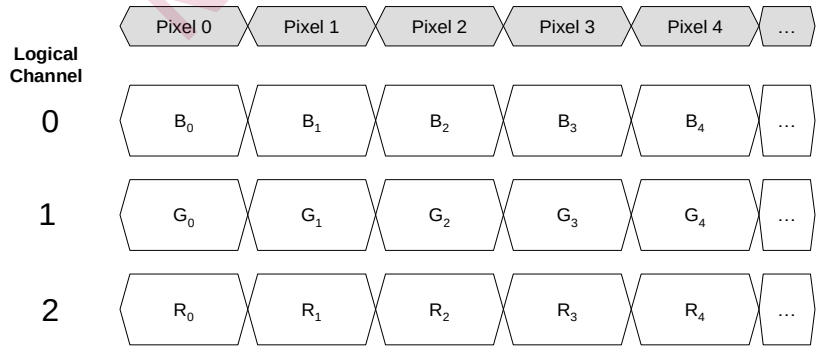


Figure 5-1. Default pixel encoding: RGB 4:4:4, 8 bits/component

5.5.2 YCbCr 4:2:2 Pixel Encoding

Figure 5-2 shows the signal mapping and timing for transferring YCbCr 4:2:2 data across MHL. Because 4:2:2 data only requires two components per pixel clock, more bits are allocated per component. The available 24 bits are split into 12 bits for the Y component and 12 bits for the C components.

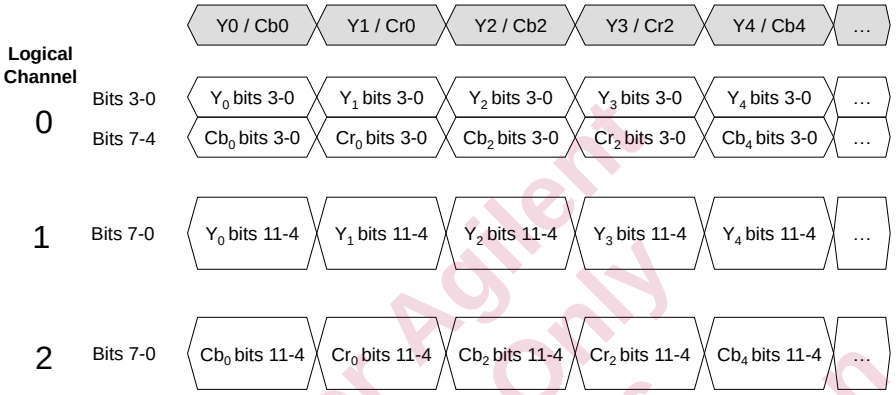


Figure 5-2. YCbCr 4:2:2 Mapping

The YCbCr 4:2:2 mapping onto MHL closely resembles standard ITU-R BT.601 mapping. The high-order 8 bits of the Y samples are mapped onto the 8 bits of Logical Channel 1 and the low-order 4 bits are mapped onto the low-order 4 bits of Logical Channel 0. If fewer than 12 bits are used, the valid bits shall be left-justified (i.e. MSB=MSB) with zeroes padding the bits below the least significant bit.

The first pixel transmitted within a Video Data Period contains three components, Y0, Cb0 and Cr0. The Y0 and Cb0 components are transmitted during the first pixel clock period while Cr0 is transmitted during the second pixel clock period. This second pixel clock period also contains the only component for the second pixel – Y1. In this way, the link carries one Cb sample for every two pixel clocks and one Cr sample for every two pixel clocks.

At the third pixel clock, this process is repeated with the Y and Cb components for the third pixel being transmitted, followed, on the next clock, by the Cr component of the third pixel and the Y component of the fourth pixel.

5.5.3 YCbCr 4:4:4 Pixel Encoding

YCbCr 4:4:4 data is transferred using the scheme illustrated in Figure 5-3.

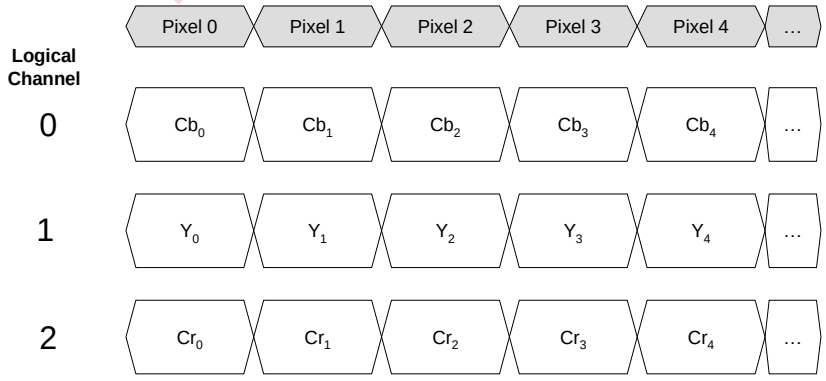


Figure 5-3. 8-bit YCbCr 4:4:4 Mapping

5.5.4 Pixel-Doubling Pixel Encoding

During pixel-doubling (Pixel_Repetition_Count = 1), all of the data sent across during the first pixel clock will be repeated during the second pixel clock. The third clock will then represent the second actual pixel and so on.

Figure 5-4 shows pixel-doubling with an RGB 4:4:4 encoding. Figure 5-5 shows pixel-doubling with a YCbCr 4:2:2 encoding. Figure 5-6 shows pixel-doubling with a YCbCr 4:4:4 encoding.

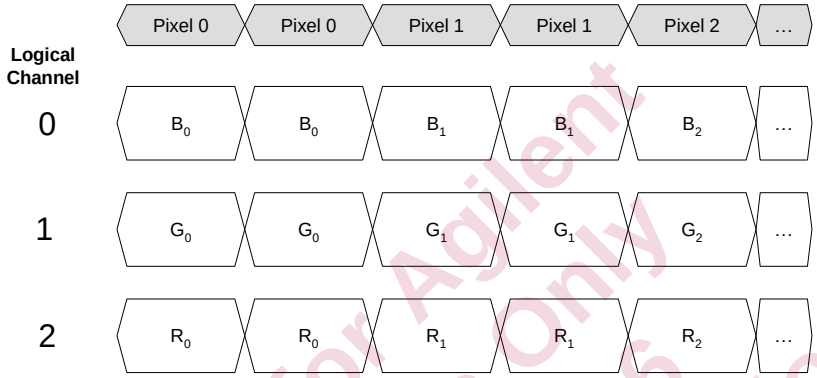


Figure 5-4. RGB with Pixel-Doubling

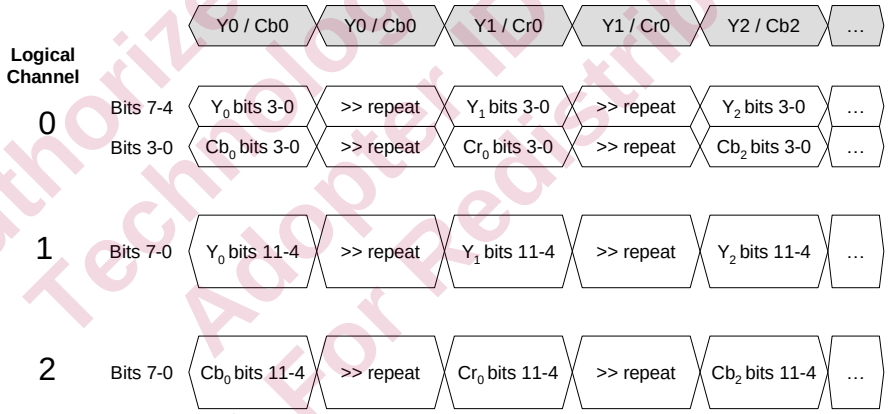


Figure 5-5. YCbCr 4:2:2 with Pixel-Doubling

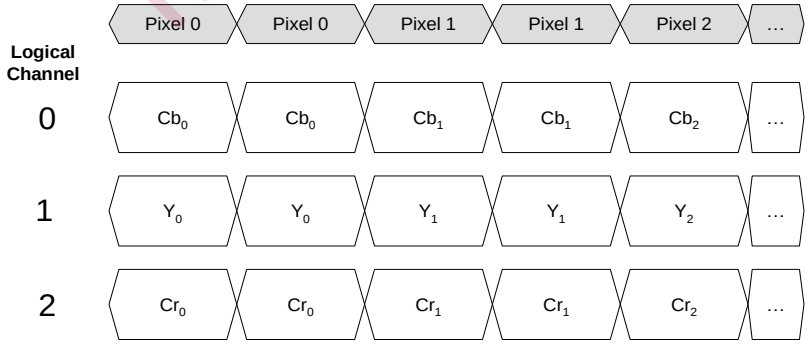


Figure 5-6. YCbCr 4:4:4 with Pixel-Doubling

The explanations for pixel doubling in Figure 5-4 to Figure 5-6 may be expanded to explain pixel repetitions beyond doubling.

5.5.5 PackedPixel Encoding

PackedPixel encoding maps the 8-bit pixel data onto two Logical Channels. This allows transmission of a particular video mode at a lower link clock frequency than is possible using the 24-bit link mode. PackedPixel encoding bit mapping is shown in Figure 5-7.

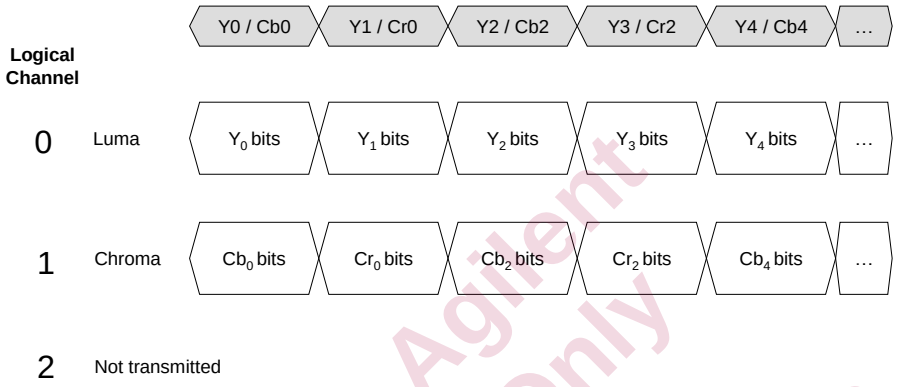


Figure 5-7. PackedPixel Encoding

PackedPixel supports encoding of 8-bit YCbCr 4:2:2 data, but does not support encoding of RGB nor of YCbCr 4:4:4 data. PackedPixel supports the transport of pixel-doubled data, but sends only 8 bits of luma and 8 bits of chroma per pixel.

PackedPixel encoding shall be indicated by the accurately set value in the CLK_MODE field of the Status Registers. See Section 7.9.3.2. PackedPixel is an optional mode. Support for it in any device shall be indicated by the value in SUPP_PPIXEL value of the VID_LINK_MODE Capability Register. See Section 7.9.1.4.

5.6 Video Quantization Ranges

Black and white levels for video components shall either be “Full Range” or “Limited Range.”

EIA/CEA-861E defines two bits in the Video Capability Data Block (QY and QS) which specify the video component range support in the Sink.

YCbCr components may be either Full Range or Limited Range. YCbCr components shall be Limited Range when the Sink’s QY bit is equal to ‘0’. If the QY bit is equal to ‘1’ then YCbCr components may be either Limited Range or Full Range, selectable by the Source using the YQ field in the AVI InfoFrame. If the Sink specifies selectable range with QY=1, then it shall expect Limited Range pixel values when it receives an AVI with YQ=0, and it shall expect Full Range pixel values when it receives an AVI with YQ=1. For other values of YQ, the Sink shall expect pixel values with the default range of the transmitted video format.

RGB components may be either Full Range or Limited Range. While using RGB, Limited Range shall be used for all video formats defined in EIA/CEA-861E, with the exception of VGA (640x480) format which requires Full Range.

5.7 Colorimetry

5.7.1 Standard-definition Formats

The color space used by the standard-definition 480-line and 576-line formats will typically be based on SMPTE 170M.

ITU-R BT.601-5 Section 3.3 shall be used for any color space conversion needed in the course of processing unless a different colorimetry is specified in the AVI InfoFrame.

The encoding parameter values shall be as defined in Table 3 of ITU-R BT.601-5 and as summarized in Section 5.6 above.

5.7.2 High-definition Formats

The color space used by the high-definition 1080i, 1080p and 720p formats will typically be based on ITU-R BT.709-5.

ITU-R BT.709-5 Part 1, Section 4 shall be used for any color space conversion needed in the course of processing unless a different colorimetry is specified in the AVI InfoFrame.

The digital representation shall be as defined in Part 1, Section 6.10 of ITU-R BT.709-4 and as summarized in Section 5.6 above.

5.7.3 Additional Color Space

Refer to EIA/CEA-861E. A Source shall only transmit these additional color spaces if support for the color space is indicated by the Sink in its EDID.

5.7.3.1 xvYCC

xvYCC is defined in IEC 61966-2-4 as extended-gamut luma-chroma-chroma (YCC) color space for video applications.

Note that MHL does not support the transport of xvYCC content across the link, as there is no mechanism defined for transporting the gamut metadata boundary definition. C1 and C0 bits, and EC2, EC1 and EC0 bits in the AVI InfoFrame shall not be set to indicate a stream of xvYCC content.

5.7.3.2 sYCC601

sYCC601 is the YCC color space defined in Annex F of IEC 61966-2-1/Amendment 1. The ITU-R rec. BT.601 color conversion matrix is used to transform RGB values to YCC values. sYCC601 color space can represent colors outside of the sRGB color gamut.

5.7.3.3 AdobeYCC601

AdobeYCC601 is the YCC color space defined in Annex A or IEC 61966-2. The ITU-R Rec. BT.601 color conversion matrix is used to transform RGB values to YCC values.

5.7.3.4 AdobeRGB

AdobeRGB is the RGB color space defined in IEC 61966-2-5. See also the Adobe RGB (1998) Color Image Decoding Specification.

5.8 Required Video Patterns

Some tests in the MHL Compliance Test Specification require the generation of specific video patterns. Refer to the MHL Compliance Test Specification for details.

5.9 3D Video

3D video content may be transported over the MHL 2 link, using the same TMDS coding as in MHL 1 video. The following subsections define requirements for selecting, formatting, and signaling 3D video.

5.9.1 3D Video Support Detection

MHL 2 uses Scratchpad Memory and the WRITE_BURST command to allow a Sink device to indicate to the connected Source the list of 3D video modes which are supported by the Sink. The details of supported modes are communicated entirely within the data of one or more WRITE_BURST transfers from Sink to Source across CBUS.

The Source device which supports 3D output shall, after discovery but not before DCAP_RDY and verification of MHL_VERSION greater than or equal to "0x20", request 3D video mode support data from the Sink by writing the 3D_REQ interrupt bit in Interrupt Register 4 to '1'. See Section 7.9.2.1.

5.9.1.1 WRITE_BURST Usage for 3D Support

3D video mode support is indicated using a sequence of WRITE_BURST transfers, formatted as shown in Table 5-3.

Table 5-3. 3D Video Format Data Structure in WRITE_BURST

Offset	Symbol	Value	Definition	Notes
0x00	BURST_ID_H		High-order byte of BURST_ID	1
0x01	BURST_ID_L		Low-order byte of BURST_ID	1
0x02	CHECK_SUM		Checksum for WRITE_BURST	2
0x03	TOT_ENT		Total Entries in Sequence	3
0x04	SEQ		Sequence Index	4
0x05	NUM_ENT		Number of Entries in this burst	5
0x06	VDI_0_H		High-order byte of First VDI	6
0x07	VDI_0_L		Low-order byte of First VDI	6
0x08	VDI_1_H		High-order byte of Second VDI	6
0x09	VDI_1_L		Low-order byte of Second VDI	6
0x0A	VDI_2_H		High-order byte of Third VDI	6
0x0B	VDI_2_L		Low-order byte of Third VDI	6
0x0C	VDI_3_H		High-order byte of Fourth VDI	6
0x0D	VDI_3_L		Low-order byte of Fourth VDI	6
0x0E	VDI_4_H		High-order byte of Fifth VDI	6
0x0F	VDI_4_L		Low-order byte of Fifth VDI	6

Notes on Table 5-3:

1. BURST_ID values shall be used as follows for 3D:

   3D video modes corresponding to 2D video listed as Short Video Descriptors in the Sink's EDID shall be listed in the WRITE_BURST using the 3D_VIC value of BURST_ID (both high and low bytes).

3D video modes corresponding to 2D video listed as Detailed Timing Descriptors in the Sink's EDID shall be listed in the WRITE_BURST using the 3D_DTD value of BURST_ID (both high and low bytes).

2. The checksum shall be calculated such that a byte-wide sum of all data bytes of the WRITE_BURST, including the checksum itself, equals zero. The checksum precedes the other payload bytes so that a WRITE_BURST of less than sixteen bytes may be used.
3. TOT_ENT indicates the total number of entries in the set of WRITE_BURST transfers using either 3D_VIC or 3D_DTD. If a Sink requires two WRITE_BURST transfers to send all seven video mode definitions with 3D_VIC, then the Sink sets TOT_ENT to 0x07 in each of the two WRITE_BURST transfers. The WRITE_BURST transfers with 3D_DTD use their own TOT_ENT setting. Setting TOT_ENT to 0x00 indicates that there are no entries for 3D video modes, either in the case of 3D_VIC or 3D_DTD or both.
4. SEQ indicates the sequence number in the series of WRITE_BURST transfers with the same BURST_ID. SEQ is set to 0x01 for the first in the series, and increments by one for each subsequent WRITE_BURST.
5. NUM_ENT indicates the number of entries in this WRITE_BURST, from zero to five. Entries shall be used in a contiguous series of from one to five entries, beginning with WRITE_BURST bytes 0x06 and 0x07.
6. See Section 5.9.1.2 for details.

The Sink device shall be responsible for proper formatting of all fields in each WRITE_BURST.

An example of the use of WRITE_BURST to represent 3D support from an EDID is shown in Figure 5-8. The listing includes support for SXGA/60 (Left-Right and Top-Bottom), 1080i/60 (Left-Right), 720p/60 (Frame Sequential and Top-Bottom), and 1080p/24 (Frame Sequential and Top-Bottom).

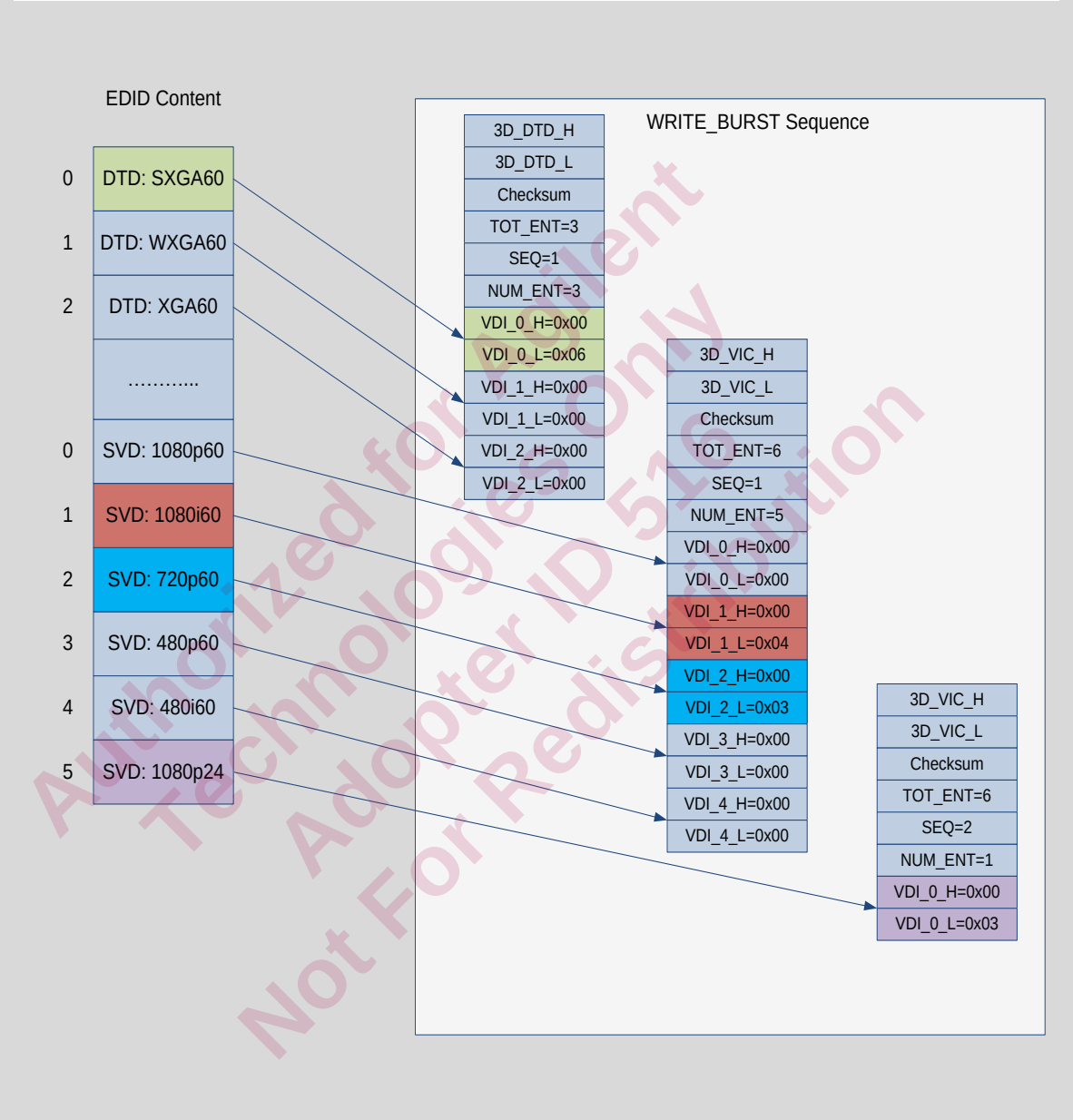


Figure 5-8. Example of WRITE_BURST to EDID Mapping for 3D Video

5.9.1.2 3D Video Descriptor for 3D Support

Each Video Descriptor (VDI) is structured as shown in Table 5-4.

Table 5-4. 3D Video Descriptor Data Structure

	Bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VDI_L	Reserved					LR_SUPP	TB_SUPP	FS_SUPP
VDI_H	Reserved							

Each supported 3D video mode shall be indicated in one 3D Video Descriptor

- FS_SUPP: If set to '1', Device supports the Frame Sequential version of this video mode.
- TB_SUPP: If set to '1', Device supports the Top-Bottom version of this video mode.
- LR_SUPP: If set to '1', Device supports the Left-Right version of this video mode.

Reserved bits shall be set to '0'.

Notes on Use of WRITE_BURST for 3D:

Note that WRITE_BURST transfers for 3D_VIC and 3D_DTD may be intermixed in sequence on CBUS, just as other WRITE_BURST transfers with ADOPTER_ID or BURST_ID different from 3D_VIC and 3D_DTD may be transferred in the middle of WRITE_BURST transfers for 3D video.

A Sink device shall send a minimum of two WRITE_BURST transfers: one with 3D_VIC and a TOT_ENT setting of 0x00; and one with 3D_DTD and a TOT_ENT setting of 0x00 – to explicitly indicate to the Source that no 3D video modes are supported. This "null support" may be indicated in the case of an MHL 2 device which does not itself render video, but which is able to handle MHL 3D video streams, yet is connected downstream to a non-3D video display device.

5.9.2 Active 3D Video Mode Indicator

The Source device shall use an MHL Vendor-Specific InfoFrame (VSIF) to indicate to the Sink the exact 3D video mode being transmitted. The VSIF is structured as shown in Table 5-5.

Table 5-5. Vendor-Specific InfoFrame Packet

Byte \ Bit #	7	6	5	4	3	2	1	0
HB0	1	InfoFrame Type = 0x81						
HB1	InfoFrame_version = 0x01							
HB2	0	0	0	InfoFrame_length				
PB0	checksum							
PB1	24-bit IEEE-assigned Organizationally Unique Identifier (OUI) 0x7CA61D							
PB2								
PB3								
PB4	Reserved			MHL_3D_FMT_TYPE			MHL_VID_FMT	
PB5…PB26	Reserved							
PB27	Reserved							

- checksum: Checksum of the InfoFrame. The checksum shall be calculated such that a byte-wide sum of all three bytes of the Packet Header and all valid bytes of the InfoFrame Packet contents (determined by InfoFrame_length), plus the checksum itself, equals zero.
- OUI: A 24-bit value, assigned to MHL, LLC by IEEE. High-order byte value 0x7C is transferred in PB3; middle byte 0xA6 is transferred in PB2; low-order byte 0x1D is transferred in PB1.
- MHL_VID_FMT: A 2-bit value indicating that this InfoFrame Packet contains video format information:
  0b00 = No additional MHL video format in this packet.
  0b01 = 3D Format present.Additional details located in MHL_3D_FMT_TYPE.
  0b10 = Reserved.
  0b11 = Reserved.
- MHL_3D_FMT_TYPE: A field indicating which of the 3D video modes is active in the stream containing this Vendor-Specific InfoFrame:
  0b0000 = Frame Sequential 3D Video
  0b0001 = Top-Bottom 3D Video
  0b0010 = Left-Right 3D Video
  0b0011 to 0b1111 = Reserved.

This value is used in conjunction with the Video Index Code (VIC) in the AVI InfoFrame. If the AVI InfoFrame carries a VIC value of zero, or if there is no AVI InfoFrame received, then the MHL Sink shall detect the video format by measuring the video timings, and associate the settings in the VSIF with that detected format.

The Source device shall transmit an MHL VSIF at least once every two video fields, during the vertical blanking time.

5.9.3 3D Video Timings

3D video timings differ, in some cases, from the 2D video timings shown in CEA-861E or other originating specifications.

5.9.3.1 Top-Bottom Video Timings

The video timing used for each Top-Bottom 3D video mode shall be the same as the video timings for the corresponding 2D video mode. For example, 720p60 Top-Bottom 3D video uses the same timings as 720p60 CEA-861E timings.

For a video mode with N active lines in each frame, the active lines from the first following the active VSYNC pulse to $(N/2)^{th}$ active line are assigned to the left eye. Active lines from $(N/2+1)^{th}$ to the last active line in the frame are assigned to the right eye.

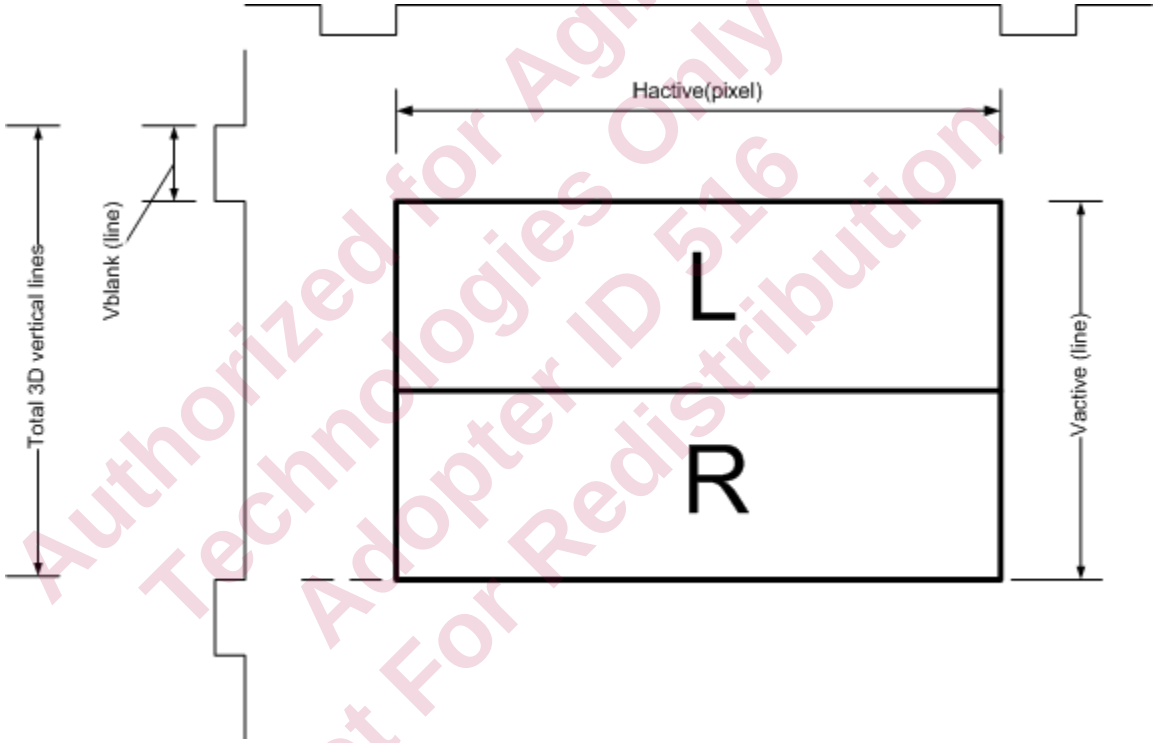


Figure 5-9. Top-Bottom Timings

For example, 720p60 frames contain 720 active lines: 360 assigned to the left eye, and 360 assigned to the right eye.

5.9.3.2 Left-Right Video Timings

The video timing used for each Left-Right 3D video mode shall be the same as the video timings for the corresponding 2D video mode. For example, 1080i60 Left-Right 3D video uses the same timings as 1080i60 CEA-861E timings.

For a video mode with M pixels in each video data period, the first pixel to the $(M/2)^{th}$ pixel are assigned to the left eye. Pixels from the $(M/2 + 1)^{th}$ to the last pixel are assigned to the right eye.

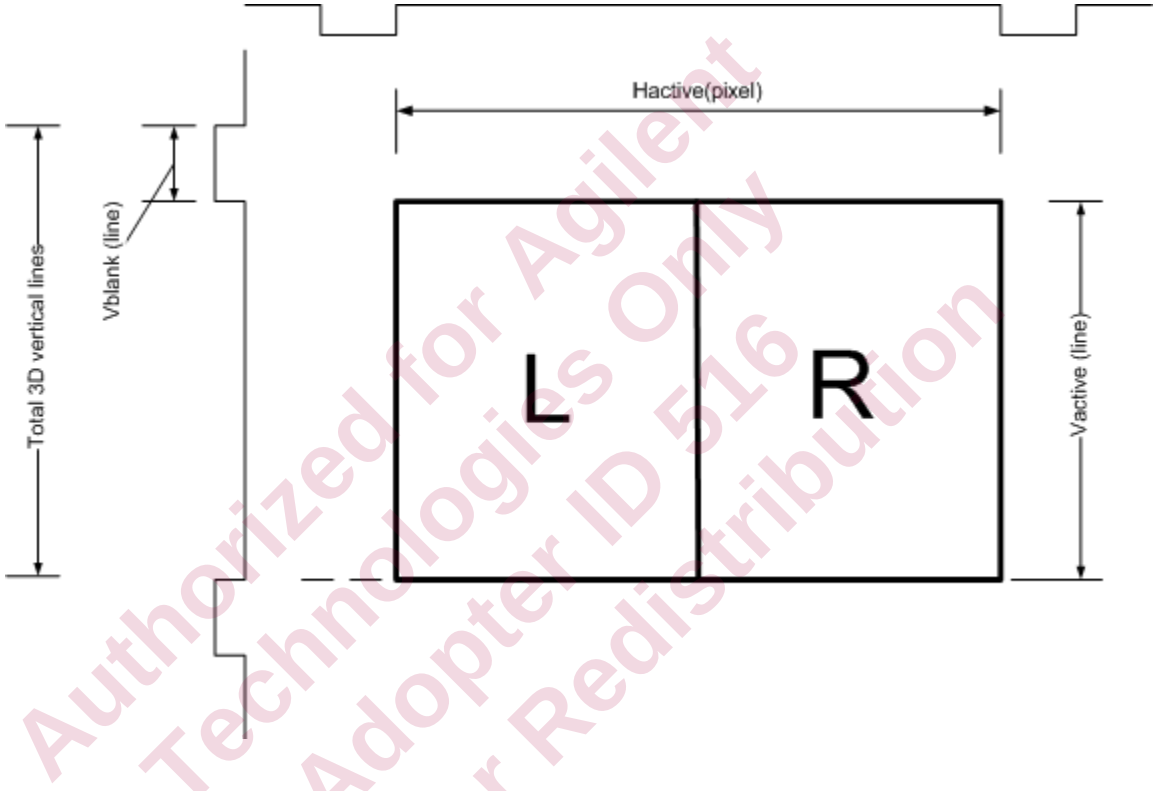


Figure 5-10. Left-Right Timings

For example, 1080i60 fields contain 1920 pixels per line: 960 assigned to the left eye, and 960 assigned to the right eye.

5.9.3.3 Frame Sequential Video Timings

Frame Sequential timings are different from 2-D timings, and are shown in Figure 5-11. Frame Sequential timings are allowed only for progressive video modes.

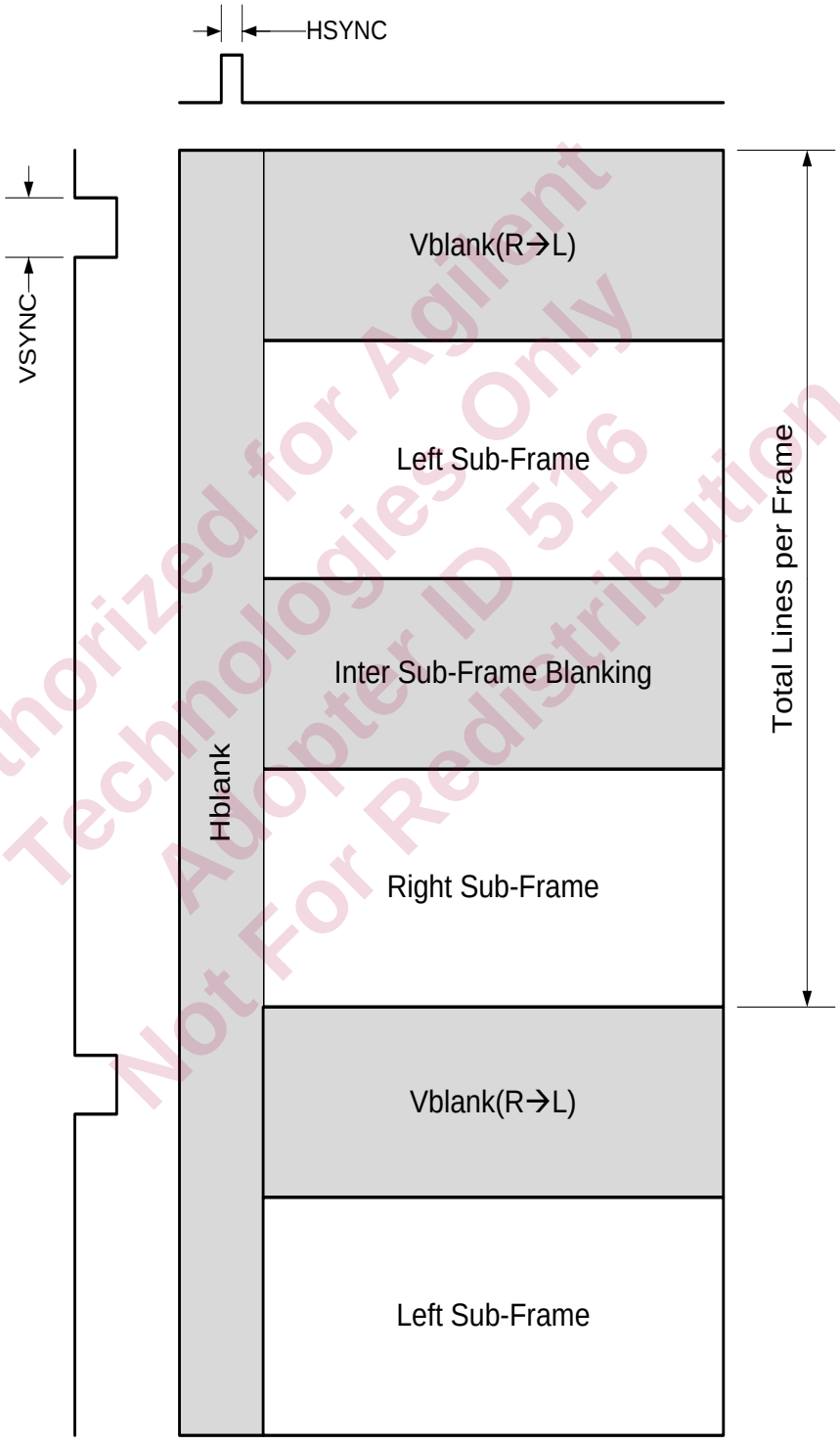


Figure 5-11. Frame Sequential Timings

The Source shall provide in Frame Sequential video the following timings:

- The horizontal blanking time preceding each line in the ISFB time is identical to the horizontal blanking time for video lines within the sub-frames. Data islands are allowed in the horizontal blanking time preceding each line in the ISFB time.
- The vertical sync pulse shall only be transmitted during the vertical blanking time between the right sub-frame and the left sub-frame
- Data Islands are prohibited during the ISFB time. The ISFB time shall be exclusively filled with control period coding (Section 4.4.3).
- Pixel data for the left sub-frame is transmitted in the first sub-frame after the vertical sync pulse. Pixel data for the right sub-frame is transmitted in the second sub-frame after the vertical sync pulse.
- Pixel clock frequency shall be twice the pixel clock frequency of the equivalent 2D video mode.

Refer to Table 5-6 and Figure 5-12 for examples of Frame Sequential timings.

Table 5-6. Example Frame Sequential Timings

3D Video Timings	Index	Mode, Refresh	Hactive	Hblank			
					Hfront	Hsync	Hback
Frame Sequential	4	1280x720p, 59.94/60	1280	370	110	40	220
	19	1280x720p, 50	1280	700	440	40	220
	32	1920x1080p, 24	1920	830	638	44	148

Vactive	ISFB	Vblank				Pixel Frequency (MHz)	Vertical Frequency (Hz)
			Vfront	Vsync	Vback		
720	30	30	5	5	20	148.35 / 148.50	59.940 / 60.000
720	30	30	5	5	20	148.50	50.000
1080	45	45	4	5	36	148.50	24.000

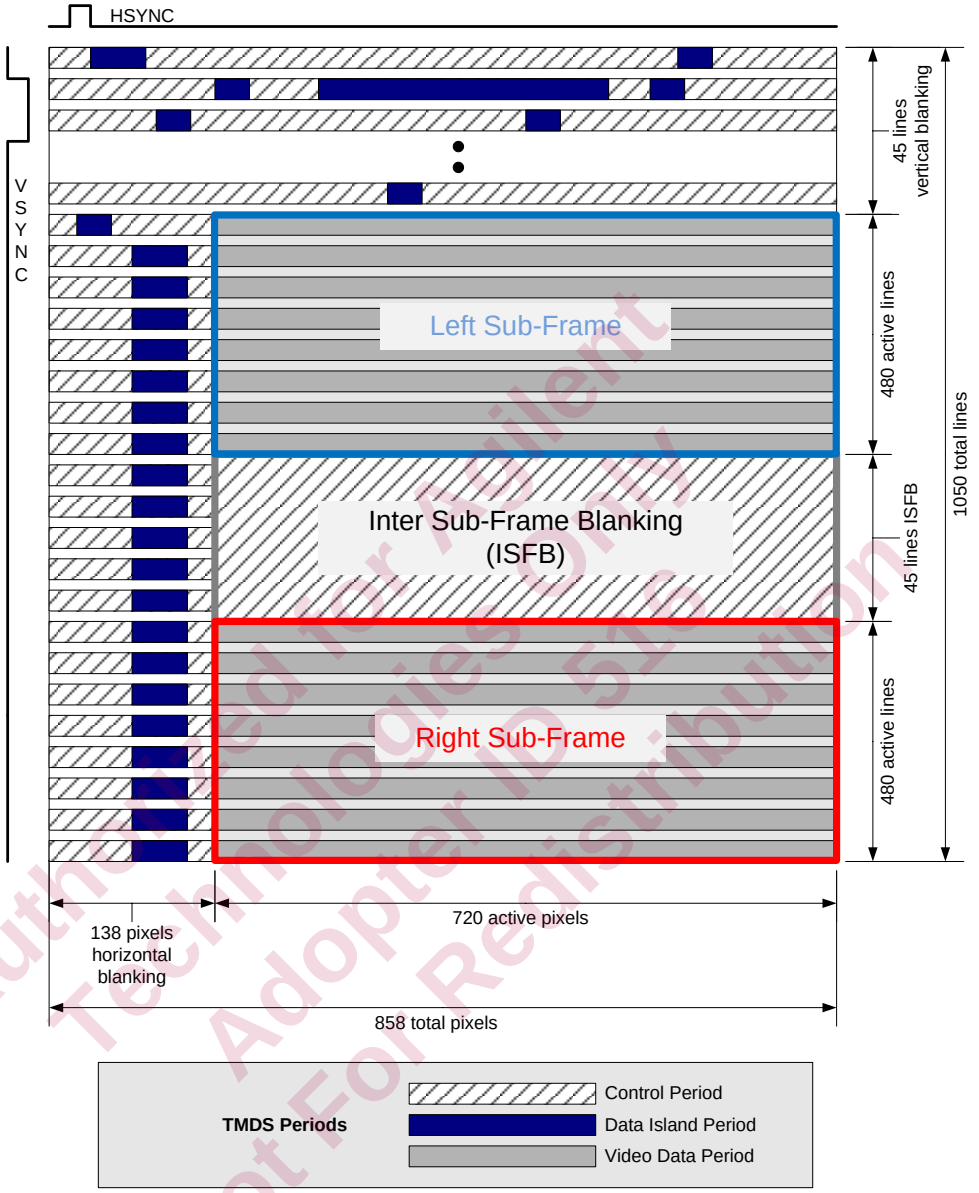


Figure 5-12. Example Frame Sequential Timings

5.9.4 3D Video Support Requirements

MHL Source and Sink devices are required to support a minimum set of 3D video formats according to the details in Sections 5.9.4.1 and 5.9.4.2. The index values shown in Table 5-7 and Table 5-8 are from the EIA/CEA-861E specification.

'Supports' in this context means the ability to send or receive the specified video mode with timings compliant to the EIA/CEA-861E specification, with the additional timings specified in this section.

Other video formats defined in EIA/CEA-861E may be supported. Additional video formats not listed in EIA/CEA-861E may also be supported. See Section 5.2.5. Any such format may be supported in Left-Right, Top-Bottom or Frame Sequential mode.

An MHL Source shall only transmit a video format which the attached Sink has indicated it is capable of receiving. The Sink indicates its supported video modes through its EDID, through the

bits in the Video Link Support Register (0x05), and with WRITE_BURST data. See Section 5.9.1 and 7.9.1.5.

5.9.4.1 MHL Source 3D Video Format Support Requirements

The requirements shown in Table 5-7 apply to MHL Sources. All MHL 2 Sources which are capable of displaying 3D, or are capable of generating 3D content, or are capable of outputting 3D, or any combination of these capabilities, shall support 3D video output on the MHL port, and meet the support requirements explained in Table 5-7.

Table 5-7. MHL Source 3D Video Mode Support Requirements

Index	Mode	Refresh	3D Format	Notes
4	1280 x 720p	59.94/60	Top-Bottom	1
			Frame Sequential	2
19	1280 x 720p	50	Top-Bottom	1
			Frame Sequential	2
5	1920 x 1080i	59.94/60	Left-Right	1
20	1920 x 1080i	50	Left-Right	1
32	1920 x 1080p	24	Top-Bottom	1
			Frame Sequential	2

Notes on Table 5-7:

1. An MHL 2 Source which is capable of displaying 3D, or is capable of generating 3D content, or is capable of outputting 3D, or any combination of these capabilities, shall support at least one of the video modes marked by this note.
2. An MHL 2 Source which is capable of displaying Full-3D, or is capable of generating Full-3D content, or is capable of outputting Full-3D, or any combination of these capabilities, shall support at least one of the video modes marked by this note.

5.9.4.2 MHL Sink 3D Video Mode Support Requirements

The requirements shown in Table 5-8 apply to MHL Sinks. Any MHL 2 Sink that supports 3D video on any other analog or digital input port shall support 3D video on all of its MHL input ports according to the support requirements explained in Table 5-8.

Table 5-8. MHL Sink 3D Video Mode Support Requirements

Index	Mode	Refresh	3D Format	Sink	Notes
4	1280 x 720p	59.94/60	Top-Bottom	Required	1
			Frame Sequential	Conditional	4
19	1280 x 720p	50	Top-Bottom	Required	2
			Frame Sequential	Conditional	5
5	1920 x 1080i	59.94/60	Left-Right	Required	1
20	1920 x 1080i	50	Left-Right	Required	2
32	1920 x 1080p	24	Top-Bottom	Required	3
			Frame Sequential	Conditional	6

Notes on Table 5-8:

1. An MHL 2 Sink that accepts 59.94/60Hz video formats from any other analog or digital video input shall support this video format on all of its MHL inputs.
2. An MHL 2 Sink that accepts 50Hz video formats from any other analog or digital video input shall support this video format on all of its MHL inputs.
3. An MHL 2 Sink shall support this video format on all of its MHL inputs.
4. An MHL 2 Sink that accepts 59.94/60Hz video formats from any other analog or digital video input and that supports PackedPixel mode shall support this video format on all of its MHL inputs.
5. An MHL 2 Sink that accepts 50Hz video formats from any other analog or digital video input and that supports PackedPixel mode shall support this video format on all of its MHL inputs.
6. An MHL 2 Sink that supports PackedPixel mode shall support this video format on all of its MHL inputs.

6 Audio

6.1 Relationship with IEC 60958 / IEC 61937

Audio data is formatted in the Audio Sample Packet as a structure that closely resembles an IEC 60958 frame.

On MHL, each IEC 60958 sub-frame is represented as a 28-bit word. There is no encoding of the preamble type which instead is replaced with a "B" bit (start-of-block) in each Audio Sample packet. The B bit shall be set for a "B, W" frame and shall be clear for an "M, W" frame (IEC 60958-1 Section 4.1.2). No other sub-frame preamble combinations are allowed.

The behavior of all fields within the Audio Sample Subpackets shall follow the corresponding rules specified in the IEC 60958 or IEC 61937 specifications.

6.2 Audio Sample Clock Capture and Regeneration

Audio data being carried across the MHL link, which is driven by a pixel clock only, does not retain the original audio sample clock. The task of recreating this clock at the receiver is called Audio Clock Regeneration.

There are a variety of clock regeneration methods that can be implemented in an MHL receiving system, each with a different set of performance characteristics. This specification does not attempt to define exactly how these mechanisms operate. It does however present a possible configuration and it does define the data items that the MHL Source is required to supply to the MHL destination in order to allow the MHL Sink to adequately regenerate the audio clock. It also defines how that data shall be generated.

In many video source devices, the audio and video clocks are generated from a common clock (coherent clocks). In this situation, there exists a rational (integer divided by integer) relationship between these two clocks. The MHL clock regeneration architecture can take advantage of this rational relationship and can also work in an environment where there is no such relationship between these two clocks, that is, where the two clocks are truly asynchronous or where their relationship is unknown.

Figure 6-1 illustrates the overall system architecture model used by MHL for audio clock regeneration. The Source device will determine the fractional relationship between the video clock and an audio reference clock (128 * audio sample rate) and will pass the numerator and

denominator for that fraction to the receiver across the MHL link. The Sink device will then recreate the audio clock from the pixel clock by using a clock divider and a clock multiplier.

The exact relationship between the two clocks will be:

Equation 2. Pixel Clock to Audio Rate Formula

$$128 \times F_S = \frac{PixelClock \times N}{CTS}$$

The Source device will determine the value of the numerator N as specified in Section 6.2.1. This value N will be used in a clock divider to generate an intermediate clock that is slower than the 128*Fs clock by the factor N. It will determine the value of the denominator CTS (Cycle Time Stamp) by counting the number of pixel clocks in each of the 128*Fs/N clocks.

If there is a constant fractional relationship between these two clocks, and the two clocks are exactly synchronous, then the CTS value will quickly come to a constant value. If the clocks are asynchronous, or there is some amount of jitter between the two clocks, then the CTS value will typically alternate between two or possibly three different values.

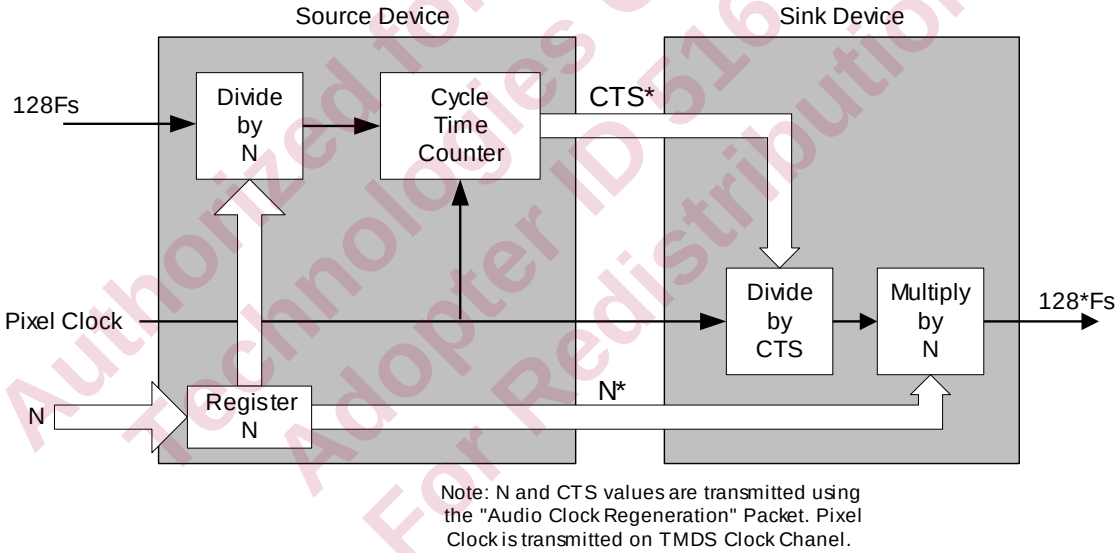


Figure 6-1. Audio Clock Regeneration Model

6.2.1 N parameter

N shall be an integer number and shall meet the following restriction:

Equation 3. Audio N Value Range

$$\frac{128 \times F_S}{1500Hz} \le N \le \frac{128 \times F_S}{300Hz}$$

with a recommended optimal value of :

Equation 4. Audio N Value Recommendation

$$N \approx \frac{128 \times F_S}{1000Hz}$$

For coherent audio and video clock systems, Table 6-1, Table 6-2 and Table 6-3 should be used to determine the value of N. For non-coherent systems or systems where coherency is not known, the equations above should be used.

6.2.2 CTS parameter

CTS shall be an integer number which satisfies the following:

Equation 5. Audio CTS Value Formula

$$Mean\{CTS\} \approx \frac{PixelClock \times N}{128 \times F_s}$$

6.2.3 Recommended N and Expected CTS Values

The recommended value of N for several standard pixel clocks are given in Table 6-1, Table 6-2, and Table 6-3. It is recommended that systems with non-coherent clocks use the values listed for a pixel clock of "Other".

Table 6-1. Recommended N and Expected CTS for 32kHz Audio

	32 kHz	
Pixel Clock (MHz)	**N**	**CTS**
27	4096	27000
54	4096	54000
74.25/1.001	11648	210937-210938*
74.25	4096	74250
148.5/1.001	11648	421875
148.5	4096	148500
Other	4096	measured

*Note: This value will alternate because of restriction on N.

Table 6-2. Recommended N and Expected CTS for 44.1kHz and Multiples

	44.1 kHz		88.2 kHz		176.4 kHz	
Pixel Clock (MHz)	**N**	**CTS**	**N**	**CTS**	**N**	**CTS**
27	6272	30000	12544	30000	25088	30000
54	6272	60000	12544	60000	25088	60000
74.25/1.001	17836	234375	35672	234375	71344	234375
74.25	6272	82500	12544	82500	25088	82500
148.5/1.001	8918	234375	17836	234375	35672	234375
148.5	6272	165000	12544	165000	25088	165000
Other	6272	measured	12544	measured	25088	measured

Table 6-3. Recommended N and Expected CTS for 48kHz and Multiples

	48 kHz		96 kHz		192 kHz	
Pixel Clock (MHz)	**N**	**CTS**	**N**	**CTS**	**N**	**CTS**
27	6144	27000	12288	27000	24576	27000
54	6144	54000	12288	54000	24576	54000
74.25/1.001	11648	140625	23296	140625	46592	140625
74.25	6144	74250	12288	74250	24576	74250
148.5/1.001	5824	140625	11648	140625	23296	140625
148.5	6144	148500	12288	148500	24576	148500
other	6144	measured	12288	measured	24576	measured

6.3 Audio Sample Rates and Support Requirements

If an MHL Source device supports audio transmission across any output, then it shall support MHL audio transmission.

If an MHL Source device supports any MHL audio transmission, then it shall support 2 channel L-PCM using an IEC 60958 subpacket structure, with either 32kHz, 44.1kHz or 48kHz sampling rate and a sample size of 16 bits or more.

An MHL Source device is permitted to transmit L-PCM or encoded audio data at sample rates of 32kHz, 44.1kHz, 48kHz, 88.2kHz, 96kHz, 176.4kHz and 192kHz using either IEC 60958 format or IEC 61937 format.

If an MHL Sink device supports audio reception across any input, then it shall support audio reception from all MHL inputs.

An MHL Sink device that is capable of accepting any audio format is required to accept two channel IEC 60958-formatted L-PCM audio at sample rates of 32kHz, 44.1kHz, and 48kHz.

For EIA/CEA-861E references to Source devices, "Basic Audio" is defined as two channel L-PCM audio at sample rates of 32kHz, 44.1kHz, *or* 48kHz, with a sample size of at least 16 bits. For EIA/CEA-861E references to DTV devices, "Basic Audio" is defined as two channel L-PCM audio at sample rates of 32kHz, 44.1kHz, *and* 48kHz. There is no sample size usage restriction for DTV devices.

An MHL Sink device may optionally accept audio at sample rates of 88.2kHz, 96kHz, 176.4kHz and/or 192kHz using either IEC 60958 format or IEC 61937 format, and should indicate these capabilities in the EDID data structure.

Whenever transmitting a valid audio stream, MHL Source devices shall always include valid sample rate information in the Channel Status bits of the audio sample packets, per Table 6-4.

In some cases, pixel-repetition may be required to increase the available bandwidth for audio transmission. For instance, when transmitting a 720x480p video format, it is required to pixel double in order to transmit 6 channels at 96kHz.

Table 6-4. Channel Status values for audio sample frequencies

Channel Status Bit Number				Sample Frequency
24	**25**	**26**	**27**	
1	1	0	0	32 kHz
0	0	0	0	44.1 kHz
0	0	0	1	88.2 kHz
0	0	1	1	176.4 kHz
0	1	0	0	48 kHz
0	1	0	1	96 kHz
0	1	1	1	192 kHz

6.3.1 Video Dependency

Available audio bandwidth depends upon the pixel clock frequency, the video format, and whether content protection re-synchronization is needed.

EIA/CEA-861E shows the available audio sample rates for 2-channel (Layout 0) and 8-channel (Layout 1) audio transmission at the various video formats specified in EIA/CEA-861E, assuming that 58 clocks of the horizontal blanking interval are required for content protection re-synchronization.

6.4 Characteristics Indication

It is possible for a Source device to send audio without determining any of the audio characteristics of the Sink device. A Source device may transmit Basic Audio without determining any of the audio characteristics of the Sink device. However, the Source device may wish to determine certain audio characteristics of the Sink device in order to enable transmission of more advanced audio capabilities.

Sink system audio characteristics and supported capabilities are indicated in the Sink's EDID data. This data includes a list of audio encodings supported by the Sink device and parameters associated with each of those encodings.

A Source indicates characteristics of the active audio stream through the use of the IEC 60958 Channel Status bits and through the use of the Audio InfoFrame. Whenever an active audio stream is being transmitted, an accurate Audio InfoFrame shall be transmitted at least as often as T_{AIF}.

Upon the start of a new audio stream or upon any change in the audio stream that can be indicated by the Audio InfoFrame, a modified, accurate Audio InfoFrame shall be transmitted quickly. Preferably this should occur just before the first affected audio sample is transmitted but if this is not possible, then it shall occur within T_{AIF0} from that point.

The Audio InfoFrame transmission is not limited by any video-related timing restrictions beyond those pertinent to the transmission of any Data Island.

6.5 Channel / Speaker Assignment

MHL allows a Sink device to indicate the configuration of attached speakers through the use of the Speaker Allocation Data Block described in Section 10.4.2, EDID Timing Extension. In addition, the Source device specifies the speaker assignment for each of the channels in the audio stream delivered to the Sink device.

EIA/CEA-861E specifies the available speaker assignment for active audio channels on MHL. This indication is carried in the Audio InfoFrame. (See Section 10.2.2 on page 204.)

6.6 Audio, Video Synchronization

For a variety of reasons, an MHL link may add a delay to the audio and/or video.

An MHL Source is required to transmit audio and video data streams with no more than T_{AV_SKEW} of audio delay relative to the video. Due to the uneven transmission of audio data, the delay shall be considered to be the average delay of all of the audio sample packets over the course of 3 steady-state video frames, where "steady-state" means a stream of video frames with stable timings.

6.7 Audio Data Packetization

Each subpacket of an Audio Sample Packet contains zero or one frame of an IEC 60958 or IEC 61937 block. There are two defined subpacket layouts. No others are permitted.

Table 6-5. Audio Sample Packet Layout and Layout Value

Layout Value	Max Num Channels	Samples	Subpkt 0	Subpkt 1	Subpkt 2	Subpkt 3
0	2	4	Channel 1,2 Sample 0	Channel 1,2 Sample 1	Channel 1,2 Sample 2	Channel 1,2 Sample 3
1	8	1	Channel 1,2 Sample 0	Channel 3,4 Sample 0	Channel 5,6 Sample 0	Channel 7,8 Sample 0

Layout 0 can carry up to four samples from a single 2-channel IEC 60958 stream of audio, or a single IEC 61937 stream.

Layout 1 can carry only one sample from 1, 2, 3, or 4 IEC 60958 streams of audio and is required for all groups with greater than 2 channels of LPCM audio.

6.7.1 Sample Present Bits in Layout 0 Audio Sample Packets

There are only five valid configurations of Present bits for an Audio Sample Packet using layout 0. They are shown in Table 6-6.

Table 6-6. Valid Present bit configurations

P0	P1	P2	P3	Description
0	0	0	0	No Subpackets contain samples.
1	0	0	0	Only Subpacket 0 contains a sample.
1	1	0	0	Only Subpackets 0 and 1 contain samples.
1	1	1	0	Only Subpackets 0, 1 and 2 contain samples.
1	1	1	1	All Subpackets contain samples.

An MHL Source is required to place the data shown into the specified subpackets and to identify the layout in the Audio Sample Packet Header.

6.7.2 Sample Present Bits in Layout 1 Audio Sample Packets

The sample present bits in Audio Sample Packets using layout 1 shall be set to match the CA0…CA7 field setting in the Audio InfoFrame. Refer to Section 10.2.2.

An MHL Source is required to place the data shown into the specified subpackets, to identify the layout in the Audio Sample Packet Header, and to make the sample present bits in each Audio Packet consistent with the setting of the CA0…CA7 field in the Audio InfoFrame.

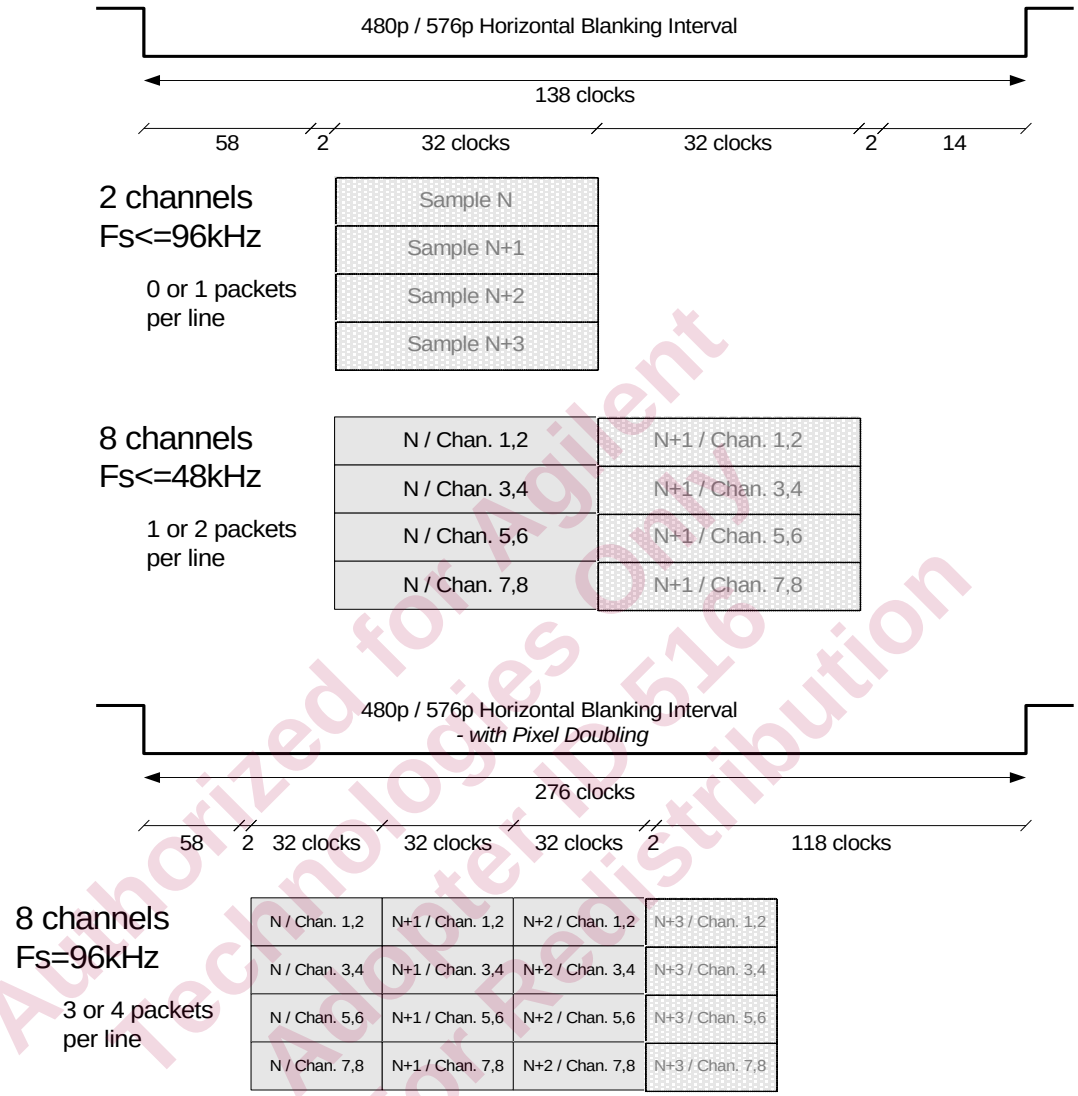


Figure 6-2. Example Audio Sample Timing

In Figure 6-2, the 58 leading clocks are required for some content protection schemes. The next 2 clocks represent the leading guard band characters present at the beginning of each data island.

There are four sample_present bits in the Audio Sample Packet Header, one for each of the subpackets. Each indicates that one subpacket contains an audio sample. In addition, there are four sample_flat bits which are set if no useful audio data was available at the Source during the time period represented by that sample. This may occur during sample rate changes or temporary stream interruptions.

6.8 Packet Delivery Rules

6.8.1 Audio Sample Packets

All audio samples that are stored in a source buffer shall be sent as soon as possible while still fulfilling requirements for audio/video synchronization, Data Island timing and placement, and Audio Clock Regeneration packet delivery.

Relative to an ideal constant-frequency clock, the jitter present in the Audio Sample Packet transmission timing shall not exceed one horizontal line period plus an additional period

represented by all of the audio samples within one Audio Packet. When packets corresponding to Layout 1 are being used (see section 6.7) then this additional period shall be one audio sample period. When Layout 0 is used, then this additional period will be 1, 2, 3, or 4 audio sample periods, depending upon the number of audio samples contained in the packets.

6.8.2 Audio Clock Regeneration Packets

The Source device shall transmit a newly determined CTS data value before transmitting the relevant audio samples.

6.9 Required Audio Patterns

Some tests in the MHL Compliance Test Specification require the generation of specific audio content streams. Refer to the MHL Compliance Test Specification for details.

7 Link Control Bus

7.1 Link Control Bus Overview

The point-to-point single-wire Link Control Bus (CBUS) provides the following functionality:

- A mechanism for the Source and Sink devices to discover connectivity to an MHL-compliant Sink and Source device respectively;
- A channel for the communication of DDC commands, used by an MHL Source device to determine the capabilities and characteristics of the Sink device by reading the EDID data structure;
- A channel for the communication of DDC commands, used by an MHL Source device to initiate all register reads and writes necessary for content protection; and
- An MHL Sideband Channel (MSC) is provided for higher-level user functions such as automatic setup tasks or tasks typically associated with infrared remote control usage.

The Link Control Bus has sufficient headroom to be used for other protocols in the future.

The CBUS interface is composed of two layers: the link layer and the translation layer. The link layer describes a method and protocol for sending and receiving data between two devices using a single wire interface. The translation layer describes how DDC and other protocols should be mapped to this single wire interface.

The link layer is a point-to-point bidirectional bus using bi-phase coding to transmit bits. The protocol is byte-oriented. The start of each byte is signaled by 1 sync pulse (2 bit times duration), followed by 2 bits of header, 1 bit of control, 8 bits of data, 1 bit of parity, and 1 pulse for ack (2 bit times duration). This leads to a total of 16 bit times per byte transfer. The link layer interfaces to the translation layer.

Upon power-on, the bus control logic enters a discovery phase in which it detects its connectivity and functionality. Once connectivity is established, either the Source or the Sink may arbitrate for control of the bus. The device that wins arbitration is considered the initiator; the other device is the follower.

The terms 'requester' and 'responder' are used throughout the definition of CBUS. All commands begin with one or more packets driven from the requesting device – the requester. Most commands include one or more packets driven from the second device – the responder – in reaction to the packets arriving from the requester.

For the MSC channel (see Section 7.4), either the Source or the Sink can be the requester for most commands. The opposite device is therefore the responder.

In the case of the DDC channel (see Section 7.5), only the Source can be the requester, and therefore only the Sink can be the responder.

In addition to the terms 'requester' and 'responder', the specification uses the terms 'initiator' and 'follower' to denote the device driving the CBUS at the link layer. As explained in the following sections, a packet sent from one device to another device includes sync, header and data bits driven from the initiator, and an acknowledge bit driven from the follower.

When a packet is sent from requester to responder, the requester is the initiator and the responder is the follower. When a packet is sent from the responder to the requester (to complete a command), the responder is the initiator of that packet and the requester is the follower.

The specification describes the CBUS in the following sections:

- The Link Layer, including protocol, arbitration, flow control, bit timings, packet timings
- The Translation Layer, including flow control commands
- MHL Sideband Channel commands
- DDC Transactions
- Sub-Command Transactions, including:
  - Remote Control Protocol messages
- Device Register Space, accessible through CBUS

7.2 Link Layer

7.2.1 Bi-phase mark encoding

Data on the CBUS shall be encoded using bi-phase mark encoding. The first state of a symbol is always different from the second state of the preceding symbol. The second state of the symbol is identical to the first if the bit being transmitted is a logical '0'; it is different if the bit is a logical '1'. This encoding results in one transition per bit time for logical '0', and two transitions per bit time for logical '1'.

7.2.2 Bus Driver States

The CBUS shall be actively driven to both high and low states in order to meet the necessary rise and fall times. However, in order to reduce the duration of any possible collisions, the bus shall be driven high for no more than necessary during the bit time.

When transmitting a '0' bit, the state on CBUS changes only at the beginning of the bit time. When driving high (rising edge), the bus shall therefore be driven high for no more than $T_{DRV\_HI\_CBUS}$, after which it shall be tri-stated until the next transition. The high state is maintained by the pull-up on CBUS.

When transmitting a '1' bit, the state on CBUS changes at the beginning and at the mid-point of the bit time. Whenever CBUS transitions from a low level to a high level, it shall be driven high for no more than $T_{DRV\_HI\_CBUS}$. CBUS stays high for the remaining part of the half-bit time by the pull-up on CBUS.

The shaded portions of the waveform in Figure 7-1 indicate the periods when the high level is maintained by the pull-up, and CBUS is not actively driven high.

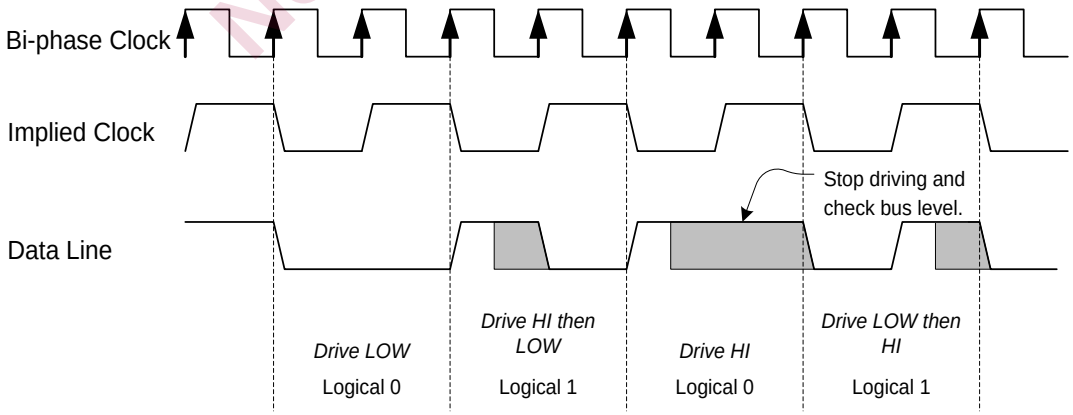


Figure 7-1. Bi-Phase Mark Encoding and CBUS Driver States

7.2.3 Packet structure

Data is sent over the CBUS in packets. A packet consists of a synchronization pulse (of period 2 bit times), 2 bits header, 1 bit control, 8 bits of data or command, 1 bit parity, and 1 acknowledge bit (of period 2 bit times), for a total of 16 bit times. Except for the synchronization bit and the ACK bit, all data is encoded using bi-phase mark encoding. Figure 7-2 shows the link layer packet structure.

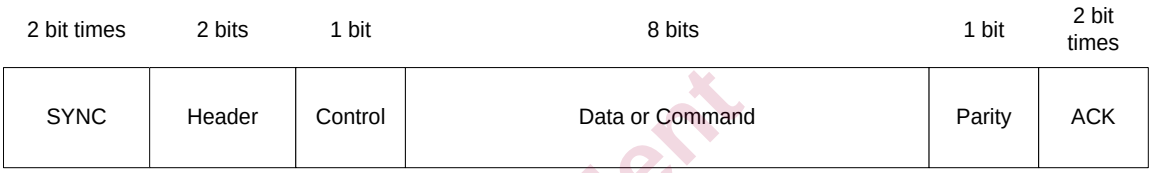


Figure 7-2. Link Layer Packet Structure

For a field with more than one bit, the most significant bit of the field is sent first across CBUS, with each consecutive bit following, down to the least significant bit.

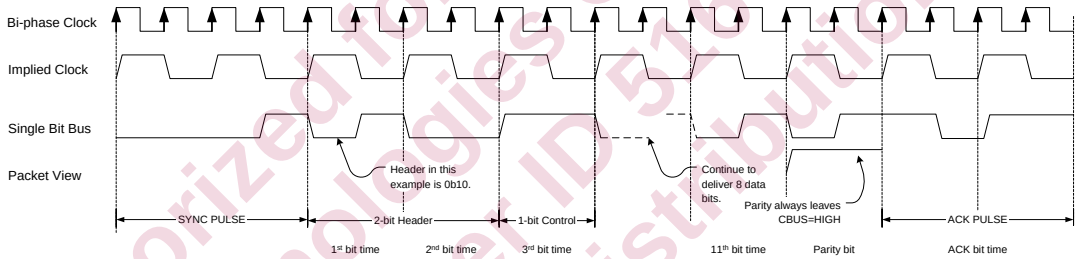


Figure 7-3. Link Layer Packet on the Control Bus

The CBUS is high when idle, due to the pull-up on the Source device.

7.2.3.1 SYNC pulse

A packet always starts with a SYNC pulse. This is a special pulse, a low for 1½ bit times followed by a high for ½ bit time. Note that the SYNC edges are timed using the bit time of the initiator.

7.2.3.2 Header bits

The SYNC pulse is followed by two header bits, which are used to encode one out of four possible channels for the command being transmitted. The DDC channel is used to transfer DDC Read and DDC Write commands across the CBUS. The MHL Sideband Channel (MSC) packet is used to transfer MHL Remote Control Protocol and other user-defined packets across the CBUS.

MHL-compliant devices shall not send packets with header bits set to Reserved, and shall discard any such packets that are received.

MHL-compliant devices which receive vendor-specific packets shall discard such packets. The receiving device shall not send a NACK or ABORT packet in response to a vendor-specific packet which it cannot handle, nor they shall issue a response command in such a case.

Table 7-1. Header Bits Encoding

Header Bits	Description
00	DDC packet
01	Vendor-specific packet
10	MHL Sideband Channel (MSC) packet
11	Reserved

The use of vendor-specific packets is prohibited in the current version of this specification.

7.2.3.3 Control Bit

The control bit indicates the context of the following 8 bits. Each command begins with a control packet. Most commands include one or more data packet, and some conclude with a control packet.

Table 7-2. Control Bit Encoding

Control Bit	Description
0	The following 8 bits are translation layer data
1	The following 8 bits are translation layer control

7.2.3.4 Command or Data Bits

The control bit is followed by 8 bits of translation layer data or command. The translation layer commands are defined in Section 7.3. Note that each command is defined as a specific sequence of command and data packets. Sending a packet type which is not expected by the follower will result in the follower sending an ABORT packet in return. See Section 7.3.1.3.

7.2.3.5 Parity

The parity bit shall cover the two header bits, the control bit and the eight data or command bits. Even parity shall be used. This ensures that the bus has transitioned to a high level prior to start of acknowledge as described below.

Informative text: A packet starts from a high state (idle bus) with a sync pulse, and ends in a high state with an ack. Leaving aside the sync pulse and ack, there should be an even number of 0's in the packet in order for the bus level to end up at a high state at the end. (In bi-phase encoding as used here, the bus level at the end of a "1" is the same as that at the beginning; the bus level at the end of a "0" is the opposite of that at the beginning.) Since we have an even number of bits, we also need an even number of 1's – hence even parity.

7.2.3.6 Acknowledge bit

The follower shall finish capturing and storing the data bits and checking parity by the end of the parity bit time. The parity bit is known to end with the bus in a high state. The follower shall start driving the bus low within $T_{CBUS\_ACK\_FALL}$ after the end of the parity bit time to acknowledge correct receipt of the current packet. Acknowledge is a logical "1". If the CBUS level is unchanged during acknowledge time then the initiator shall assume that the follower did not acknowledge.

Note that the ACK bit is timed within a 2 bit time period, with the bit time calculated by the responder device taking the role of initiator. The "bit time" of this device may be different from

the "bit time" of the preceding SYNC, Header, Control Bit, Data and Parity bits, yet both "bit times" shall be within the limits of CBUS bit time specified in Section 13.

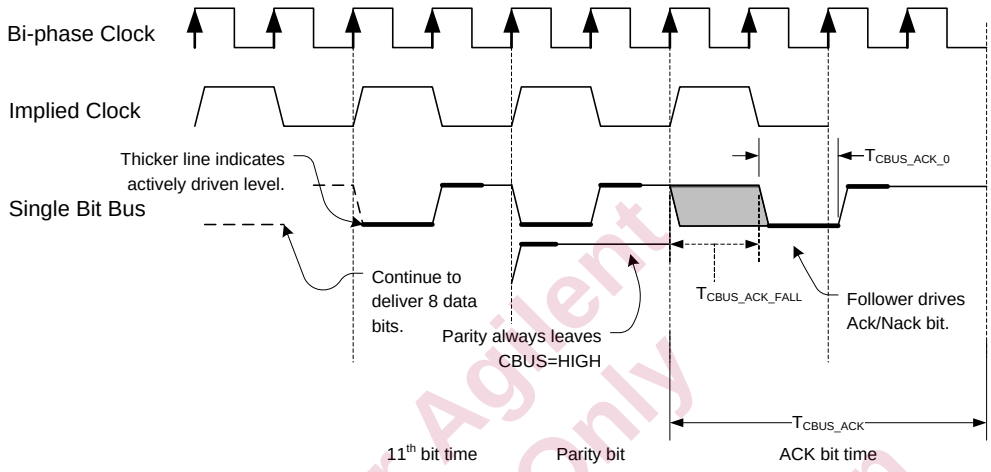


Figure 7-4. ACK Bit Timing on the Control Bus

7.2.4 Arbitration

A device shall arbitrate for CBUS before becoming an initiator.

A device wishing to start arbitration shall first verify that CBUS is still idle, i.e. in a high state, for a minimum duration as shown in one of Figure 7-7, Figure 7-8, Figure 7-9 or Figure 7-10. In case the device last had control of the bus (as initiator), the wait time is measured from the end of the two bit times allocated by the initiator to the acknowledge bit. In all other cases, the wait time is measured from the last rising edge on CBUS. Note that this last rising edge may have been from: acknowledge bit, parity bit, or earlier if the initiator failed to complete driving the bits of the packet.

To arbitrate, an MHL Source device shall drive CBUS low for T_{ARB_SRC}. An MHL Sink device shall drive CBUS low for the T_{ARB_SINK}.

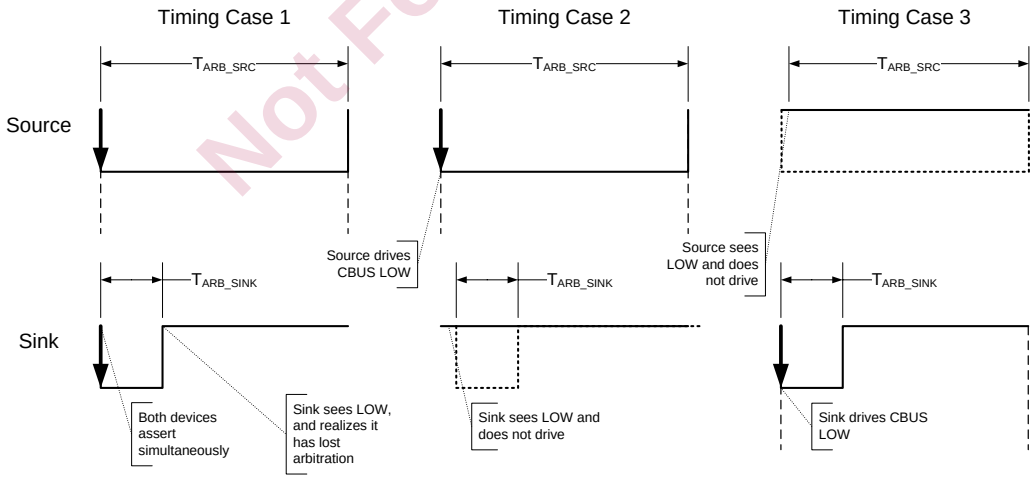


Figure 7-5. Link Layer Arbitration Timing – Source versus Sink Assertion Times

Each device shall check the state of CBUS before driving it LOW. If both devices drive LOW at the same time, then since T_{ARB_SINK} is shorter than T_{ARB_SRC} the Source device will win the arbitration. See "Timing Case 1" in Figure 7-5. "Timing Case 2" shows the Sink checking the

Source's driven LOW on CBUS. "Timing Case 3" shows the Source checking the Sink's driven LOW on CBUS.

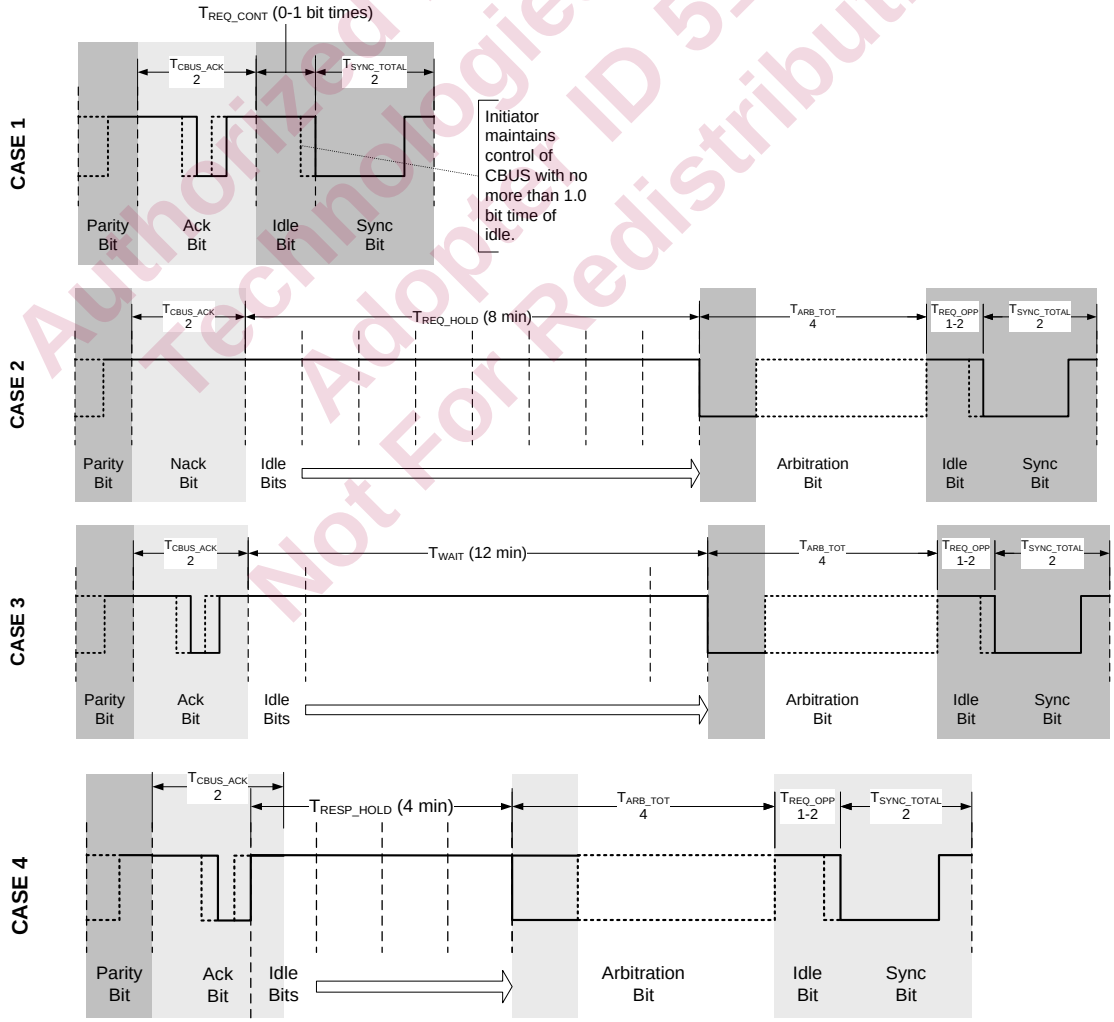
The Sink shall check the state of CBUS during the rest of the T_{ARB_TOT}. Since a device actively drives a "1" only at the beginning of a bit time, if the Sink detects that CBUS is low during the rest of T_{ARB_TOT}, it shall conclude that it has lost arbitration.

Several cases of arbitration require a device to delay asserting CBUS to LOW for at least a minimum number of bit times. If such a device, waiting to assert CBUS, detects a LOW on CBUS at any point during this hold-off time, it shall reset its calculation of hold-off to zero and begin its hold-off delay again. See Figure 7-11 for an example.

If an initiator wins arbitration, it shall send the SYNC pulse within T_{REQ_OPP}. If the follower does not see any activity on CBUS for T_{WAIT} after the bus goes high, it should assume that the initiator has given up control of CBUS.

Each side is guaranteed to get access to CBUS because there is a limit on the maximum allowed number of packets (N_{MAX}) one side can send before giving up control of CBUS. Note that this is a link-layer feature and is independent of any flow control that may be implemented in the translation layer.

There are four cases for re-arbitration, depending on which device in which role wants to win arbitration for the next packet. Each of these is shown as a timing diagram in Figure 7-6, and described in detail in the following sections.



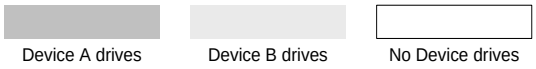


Figure 7-6. Link Layer Arbitration Timing – Four Cases

7.2.4.1 Case 1: Initiator Maintains CBUS Control after Ack Bit

A device which has the role of initiator, and comes to the end of a packet, may maintain control of CBUS using the timing shown in Figure 7-7.

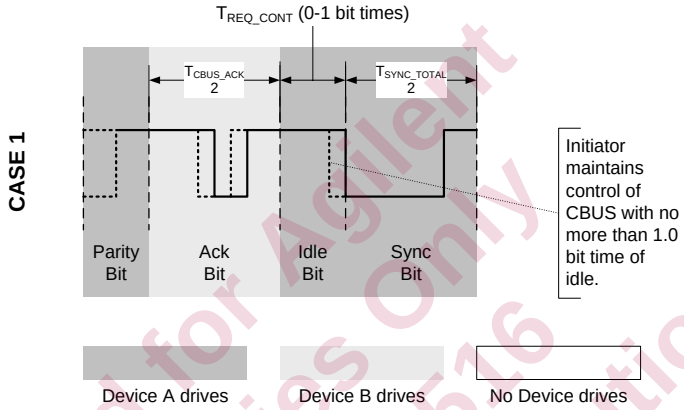


Figure 7-7. Link Layer Arbitration Timing – Initiator Maintains Control of CBUS

After the Ack bit, the initiator shall wait from 0 to T_{REQ_CONT} time before asserting the Sync bit. This period may have any width from 0 to T_{REQ_CONT}. Since the Ack bit is required to end with a HIGH state on CBUS before the end of the Ack bit time, there will always be a HIGH on CBUS before the earliest assertion of the Sync bit's falling edge.

The follower has no opportunity in this case to take control of CBUS through arbitration. The initiator may maintain control of CBUS as initiator for as many as N_{MAX} packets. Such a device shall then allow rearbitration according to the timing in Section 7.2.4.3.

7.2.4.2 Case 2: Initiator Idles Bus After Nack Bit

An initiator which transmits its parity bit to the follower, may receive no Ack bit in return. This Nack condition requires that the initiator re-try transmitting the packet, continuing to re-try for at least the required minimum number of re-tries, N_{RETRY}.

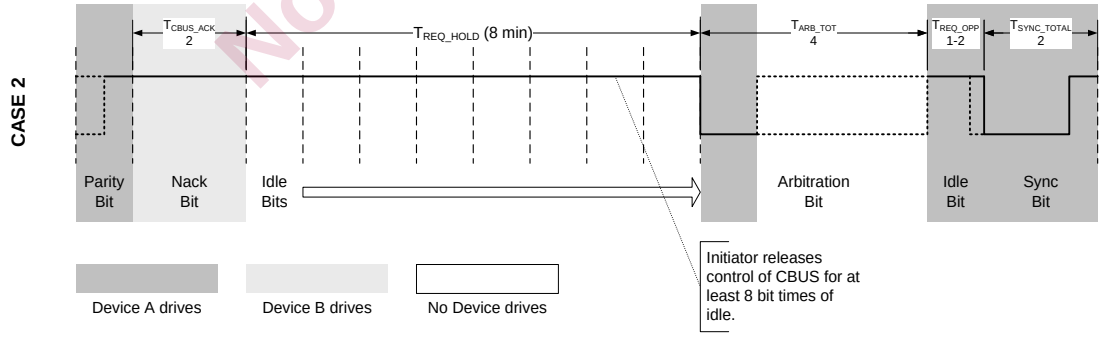


Figure 7-8. Link Layer Arbitration Timing – Initiator Hold-Off After Nack Bit

An initiator which receives a Nack bit shall hold-off driving CBUS to LOW for a minimum of T_{REQ_HOLD}. This hold-off time is shown in the middle of Figure 7-8. The follower device may gain control of CBUS during this time, as described in Section 7.2.4.4.

7.2.4.3 Case 3: Initiator Idles Before Next Arbitration

An initiator which has no pending packet and therefore no immediate need to begin a new packet on CBUS, or an initiator has sent N_{MAX} packets and shall wait before sending another, then the initiator shall hold-off arbitration for at T_{WAIT}, as shown in Figure 7-9.

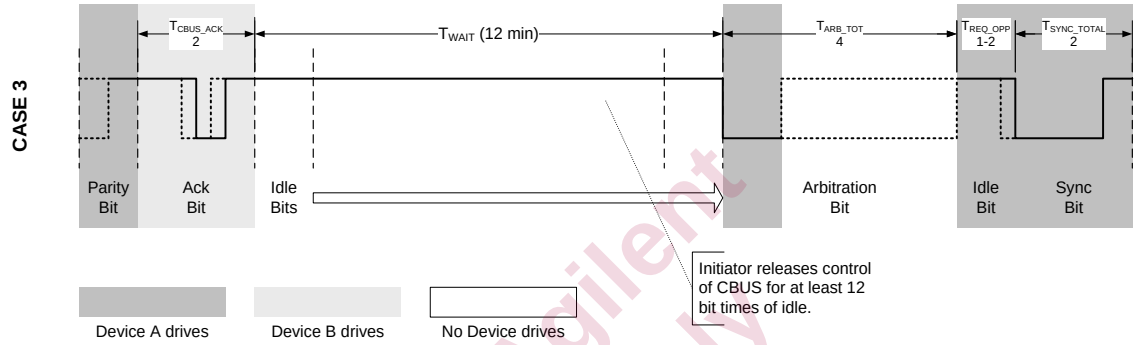


Figure 7-9. Link Layer Arbitration Timing – Initiator Hold-Off with No Pending Packet

7.2.4.4 Case 4: Follower Arbitrates to Gain Bus Control

The follower device which does not lose control of CBUS immediately (see Section 7.2.4.1) may gain control of CBUS by arbitrating after a shorter wait, T_{RESP_HOLD}, shown in Figure 7-10. T_{RESP_HOLD} is measured by the follower from the last CBUS edge. Note that this shorter wait time is only applicable immediately after the last rising edge seen during the receipt of a packet. It is no longer applicable once a new Sync pulse is seen. Since this wait time is shorter than that for an initiator to react to a Nack bit (Section 7.2.4.2), and shorter than that used by initiator without pending packet (Section 7.2.4.3), the follower always has the opportunity to become an initiator and send a packet on CBUS.

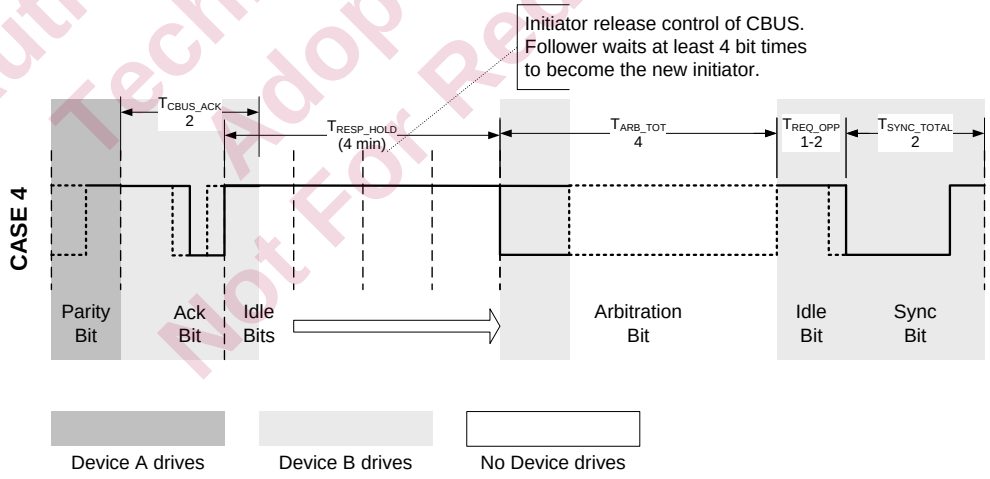


Figure 7-10. Link Layer Arbitration Timing – Follower Wins Arbitration

An example of the follower device (Device B in the figure) gaining control of CBUS during a Nack hold-off by Device A is shown in Figure 7-11.

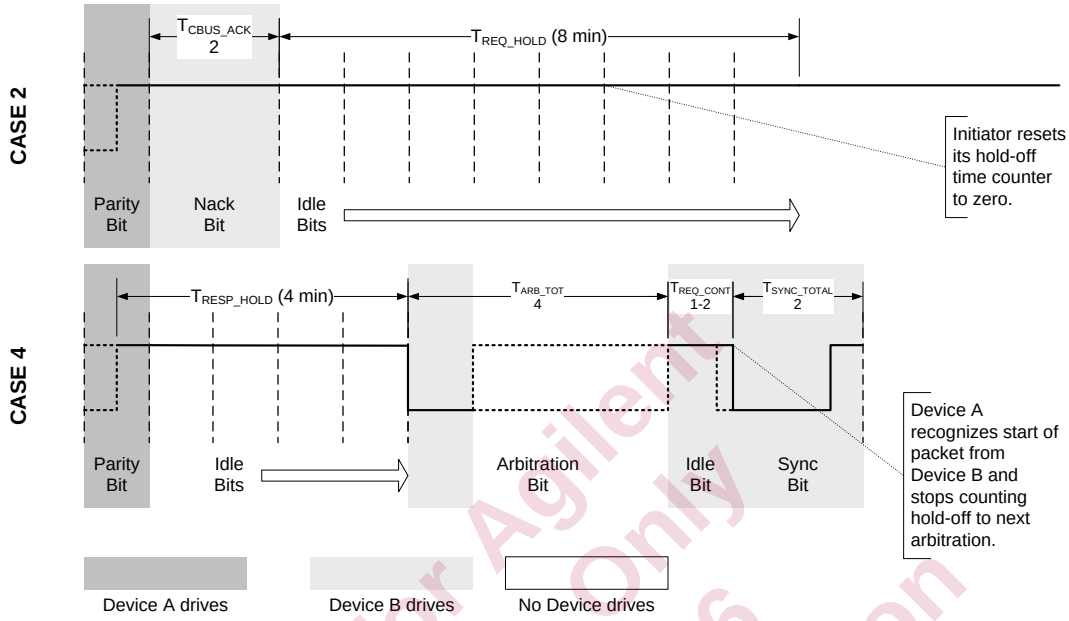


Figure 7-11. Link Layer Arbitration Timing – Follower Preempts Initiator's Post-Nack Hold-Off

7.2.5 Flow Control

The initiator shall stop after N_{MAX} packets and wait for at least T_{WAIT} after the final ACK before re-arbitrating for CBUS.

The follower can arbitrate for CBUS after the previous initiator stops sending packets. If it does so, it gains control of CBUS and can send up to N_{MAX} packets.

The packet maximum N_{MAX} is counted from the first packet transmitted, and increments for each packet sent. If a packet is acknowledged by the follower with a negative acknowledge bit (a Nack bit), the initiator shall re-arbitrate for control of CBUS, and retry sending the same packet. The packet counter is reset to zero after a Nack bit, and incrementing begins again with the first retry packet. The re-arbitration time allows the opposite device the opportunity to win control and use the bus.

7.2.6 Link Control Bus Timing

MHL devices shall comply with the timings defined in Section 13.

7.3 Translation Layer

The translation layer interfaces to the link layer and acts as an interpreter for DDC or MSC commands. In the following description, care must be taken to distinguish a link-layer acknowledge *bit* from a translation layer acknowledge *byte*. The former is part of the 16-bit-time packet. The latter is a control packet, sent by a responder when called for in the command's protocol.

The translation layer must arbitrate among its requesters (DDC, MSC) to allow access to the link layer. This arbitration can be done on a packet-by-packet basis. Each translation layer byte will become a 16-bit-time link layer packet with a 2-bit header identifying its protocol, so packets from multiple channels can be interleaved on the link layer. Note that this arbitration mechanism is independent of the link layer arbitration scheme, which arbitrates for control of the link between two potential initiators.

Table 7-3 and Table 7-8 show the translation layer commands, their codes, and the protocols for which they are used. Note that all commands have the command bit set to 1 in the first packet, and every packet has the appropriate 2-bit header.

The link layer acknowledge or negative acknowledge is not shown in these timing diagrams. This portion of the protocol is described in Section 7.2, and normally handled below the translation layer. The diagrams show sequences of packets directly following one another on the CBUS wire. As explained earlier, the CBUS protocol allows for other packets, on other channels, to interrupt the contiguous flow of packets in an ongoing command. For purposes of clarity, such interruptions are not shown in the timing diagrams in this section.

All MHL devices shall support the control packets listed in Table 7-3 and Table 7-8. Commands not listed in Table 7-3 or Table 7-8 shall not be used on the CBUS. The MSC channel control packets shall only be used in the specific protocols described in Section 7.4.2. The DDC channel control packets shall only be used in the specific protocols described in Section 7.5.3.

After detecting multiple consecutive errors on the bus, a device may disconnect from the connected device, and then attempt a re-discovery. Therefore, each device shall be prepared to handle a disconnection at any time.

MHL devices shall comply with the timings defined in Section 13.

7.3.1 Translation-Layer Flow Control

Several of the control packets listed in Table 7-3 and Table 7-8 may be used to control the flow of data across CBUS. A requester device may send a command on either the DDC or MSC channel whenever the CBUS is available, but shall not send another command on the same channel until it completes one of the protocols defined in Sections 7.4.2 and 7.5.3, and passes through the idle state.

Expectations for ACK packet or NACK packet responses are shown for the DDC channel commands in Section 7.5, and for the MSC channel commands in Section 7.4.1. Note that some commands conclude on receipt of a data packet instead of a flow control packet.

Arbitration for the CBUS operates at the link layer, without regard to any priority of DDC channel over MSC channel, or vice versa. Architectural details to provide adequate bandwidth to both channels are beyond the scope of this specification.

7.3.1.1 ACK Flow Control Packet

An ACK control packet is generated by a responder back across the CBUS in response to a CBUS command from a requester. Note that the ACK control packet is supported by both the DDC and MSC channel.

The ACK control packet indicates to the original requester that its command has been completed. For example, if a requester sends a WRITE_STAT[offset][data] command to a responder, and the responder immediately stores the values and is ready for another command, then the responder will send an ACK control packet back to the requester.

If a device receives an ACK control packet outside the context of a command, it shall interpret that ACK packet as a protocol error, and respond as defined in Section 7.4.1 or Section 7.5.2, according to which channel is being used.

Note that an ACK packet may be returned from the responder in the case of a DDC channel command with an offset address which is not recognized by the downstream DDC device. For example, if a Source sends a DDC Read command to a Dongle, with a device address which is understood by the downstream DDC device, yet the offset address in the command is outside the valid range for that DDC device address, then the downstream DDC device may respond with an ACK bit on DDC, which is passed back through the Dongle as an ACK packet on the DDC channel. HDCP Receiver is an example of a DDC device which has limited register space.

7.3.1.2 NACK Flow Control Packet

A NACK control packet is generated by a responder back across the CBUS in response to a CBUS command from a requester. Note that the NACK control packet is supported by both the DDC and MSC channel.

The NACK control packet indicates to the original requester that its command cannot be completed.

If a device receives a NACK control packet outside the context of a command, it should interpret that NACK packet a protocol error, and respond as defined in Section 7.4.1 or Section 7.5.2, according to which channel is being used.

7.3.1.3 ABORT Flow Control Packet

An ABORT control packet is generated by a device back across the CBUS in response to a CBUS command.

Note that the ABORT control packet is supported by both the DDC and MSC channel.

The ABORT control packet from a responder indicates to the original requester that its command cannot be completed. For example, if a requester sends a WRITE_STAT[offset][data] command to a responder, but the offset value in that command is sent in a control packet instead of a data packet, then the responder will send an ABORT control packet in response back to the requester.

Note that the responder sends the ABORT packet on the CBUS wire to the requester only when the responder wins a re-arbitration. The requester may continue to send multiple packets, up to the packet count limit (N_{MAX}). The requester may stop sending packets from a command on one channel, offer a re-arbitration, win that arbitration, and then begin sending packets from a pending command on the other channel. As a result, the responder may not be able to send its ABORT packet to the requester immediately after the error occurs.

The ABORT control packet from a requester indicates to the responder, in the middle of a command, that the responder's response was in error. For example, if a requester sends a READ_DEVCAP[offset] command to a responder, and the responder sends a data packet back instead of the required ACK control packet, then the requester will send an ABORT control packet to the responder.

A device should not reply to an ABORT packet by sending an ABORT packet in response. That is, if a device receives an ABORT packet from its peer in the middle of a command, the receiving device should not interpret this as a protocol error and issue its own ABORT packet. It should instead move to the idle state, truncating the command.

A device should not respond to an ABORT packet at all when the device is already in the idle state. In this way, repeated ABORT packets will be handled by moving to the idle state on the first ABORT packet, and then staying in the idle state.

A complete list of the error reactions to various conditions is provided in Section 7.4.1 and Section 7.5.2. An error code is latched in the MHL device which detects the error condition. This error code may be read back across the CBUS using one of the following commands, both of which are executed on the MSC channel:

GET_MSC_ERRORCODE

GET_DDC_ERRORCODE

An MHL device which receives, and acknowledges at the link layer, a control packet which has an opcode not shown in Table 7-3 nor in Table 7-8 should reply with an ABORT control packet. The undefined packet shall be discarded with no further response. The device may react to such an undefined packet with an internal error flag. An undefined command or protocol error flag may be fetched using the GET_DDC_ERRORCODE (Section 7.4.3.5) or GET_MSC_ERRORCODE (Section 7.4.3.6) commands on MSC.

7.3.2 Command Description Methodology

Command syntax on the MSC and DDC channels is provided in two ways: a packet flow diagram and a state machine diagram. In the packet flow diagram (for example Figure 7-15), the sequence of packets from requester to responder and responder to requester is shown. A separate flow is shown for some of the commands to indicate how the command may terminate with error. Not all possible error conditions are shown in the packet flow diagram.

In the state diagrams, the exchange of packets is shown step by step (for example Figure 7-13), including most of the error paths. These error paths complicate the overall state diagram, but provide a more complete representation of the logic needed to handle all possible command protocols.

7.3.2.1 Packet Flow Diagram Syntax

Each hexagon in a packet flow diagram represents one CBUS packet, including the SYNC bit, the header bits, the control bit, the data payload bits, the parity bit and the ACK/NACK bit. Each of these packets has its timing checked at the link layer.

A hexagon with a double border is a control packet with the control bit set after the 2 header bits.

A hexagon with a single border is a data packet with the control bit cleared after the 2 header bits.

A right-facing arrow above a hexagon indicates a packet from requester to responder.

A left-facing arrow above a hexagon indicates a packet from responder to requester.

The length of the line joining two adjacent hexagons does not indicate any specific timing relationship. A requester or responder which is to send multiple consecutive packets may do so according to timings in Sections 13.10.2-13.10.3. A device which reacts by sending a packet back to its peer (in either direction) may do so according to the same timing specifications.

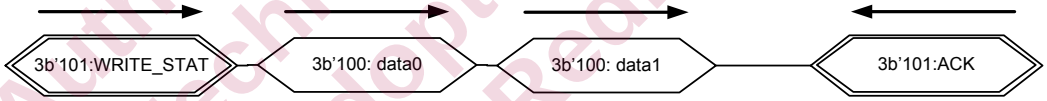


Figure 7-12. Packet Flow Diagram Legend

The control packet's content is shown as the 2-bit header and 1 control bit, then a colon, then the acronym for the control packet. Similarly, the data packet's content is shown as its 2-bit header and 1 control bit, then a colon, followed by the 8-bit payload.

7.3.2.2 State Diagram Syntax

Figure 7-13 illustrates the syntax used in the state diagrams for sub-channels on CBUS. Note that not all elements of the syntax may be used on all sub-channels. Each state in a state diagram is shown as a heavy vertical line with a title bar at the top. A device moves from one state to another on receipt of a packet, by issuing a packet, by timeout, or by some other error condition. Each command state diagram begins at the idle state and ends back at the same idle state. Note that direction of flow is indicated by the arrowheads, and the next state may be to the right, left or below the current state.

Each rectangle in the diagram represents a packet on the CBUS. A packet moves from requester (D1) to responder (D2). The lightly shaded rectangles are transmitted by the requester. A packet may also move from responder (D2) to requester (D1). The darker shaded rectangles are transmitted by the responder.

The label in each rectangle indicates the name of the packet, and if it is a control packet (cPacket) or a data packet (dPacket).

State transitions which do not involve a packet (such as a timeout) use heavier arrows with open arrowheads, and include a label on the arrow itself.

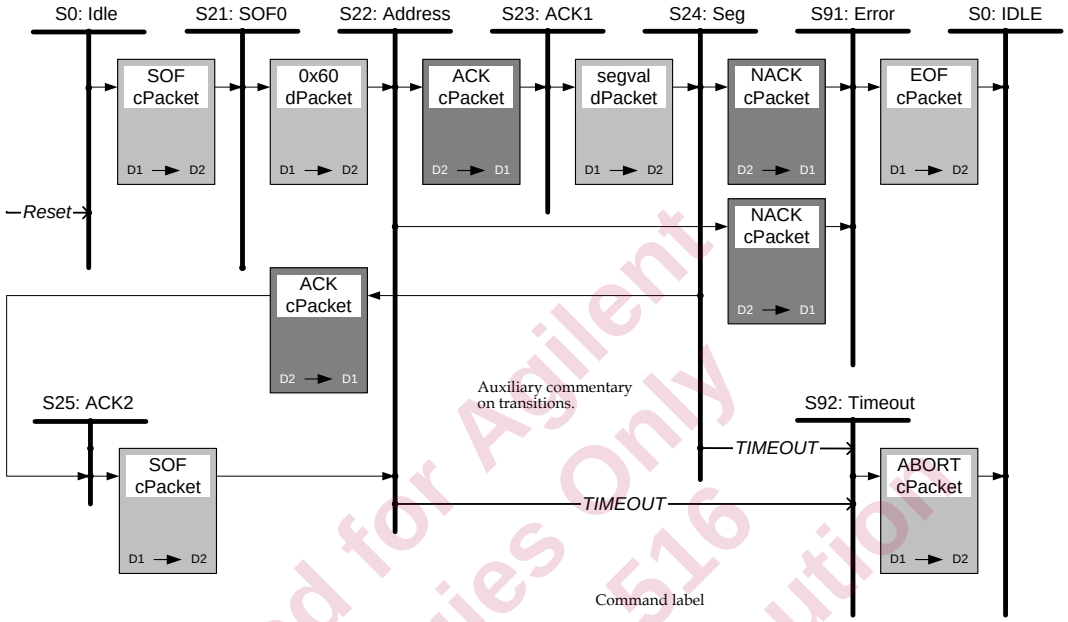


Figure 7-13. Example State Diagram Legend

State transitions are driven in most cases by the transmission or receipt of a packet. In some cases, the incoming packet may be by the sending device at the state transition indicated, but that packet may not actually be recognized by the receiving device until later in the command. A device which continues to act as initiator by maintaining control of the CBUS (i.e., not losing arbitration to the opposite device) may not provide a free CBUS time for the opposite device to send a packet. For example, the WRITE_STAT MSC command (see Section 7.4.3.8) normally sends a control packet followed by two data packets. It does not need to give up the CBUS until it begins to wait for the ACK packet in return.

The opposite device may see an error in the first data packet (invalid offset) and post or queue an ABORT packet. (See the abort packet rules in Section 7.4.1). The ABORT packet may not be recognized by the requester until it begins to wait for the ACK packet.

For WRITE_STAT, the state transition for ABORT is shown connected to the offset data packet, to show the "cause and effect" – the ABORT packet comes from device 'D2' in reaction to the bad offset received from device 'D1'. Note that not all commands include all of these paths.

Throughout this specification, a status value associated with a Sink device is denoted with the suffix "{Sink}", and a status value associated with a Source device is denoted with the suffix "{Source}" to remove confusion. Note that the value of a Sink's status is written by the Sink into the Source, and accessible internally in the Source – perhaps as a registered value (and vice versa for a Source's status in the Sink).

7.4 MSC Commands

Each MSC command is constructed from a sequence of control packets and data packets. The control packets available on MSC are listed in Table 7-3. Codes not listed in Table 7-3, if used, shall receive an ABORT from the opposite device.

Table 7-3. MSC Translation Layer Control Packets

Code	Control Packet	Description	Section	Notes
0x33	ACK	Command/Data byte acknowledged	7.3.1.1	1
0x34	NACK	Command/Data byte not acknowledged	7.3.1.2	1
0x35	ABORT	Transaction abort	7.3.1.3	1
0x60	WRITE_STAT	Write one byte to responder's status bytes.	7.4.3.8	2
	SET_INT	Write one or more interrupt bits in a byte.	7.4.3.9	2
0x61	READ_DEVCAP	Read one byte from responder's registers.	7.4.3.7	
0x62	GET_STATE	Read state from responder.	7.4.3.1	
0x63	GET_VENDOR_ID	Read vendor ID value from responder.	7.4.3.2	
0x64	SET_HPD	Set Hot Plug Detect in responder.	7.4.3.3	
0x65	CLR_HPD	Clear Hot Plug Detect in responder.	7.4.3.4	3
0x68	MSC_MSG	Send RCP, RAP or UCP code.	7.4.3.11	
0x69	GET_SC1_ERRORCODE	Get channel 1 command error code.		4
0x6A	GET_DDC_ERRORCODE	Get DDC channel command error code.	7.4.3.5	
0x6B	GET_MSC_ERRORCODE	Get MSC command error code.	7.4.3.6	
0x6C	WRITE_BURST	Write 1-16 bytes to responder's Scratchpad Registers.	7.4.3.10	
0x6D	GET_SC3_ERRORCODE	Get channel 3 command error code.		4
0x32	EOF	End of Frame.	7.4.3.10	1

Notes on Table 7-3:

1. These packets shall only be used in the specific protocols described in Section 7.4.1.
2. WRITE_STAT and SET_INT use the same opcode, and depend on the offset data packet to distinguish the action to be taken at the follower's end.
3. If a CLR_HPD command is sent, it shall not be followed by a SET_HPD command within less than T_{HPD_WIDTH}.
4. This command is reserved for future use. Response behavior is not defined. "Channel 1" means header bits 0b01. "Channel 3" means header bits 0b11, as shown in Table 7-1.

7.4.1 MSC Command Reactions

Many scenarios exist in which a command does not complete as expected. The responder and the requester are required to react to these situations in predictable ways.

Table 7-4. Summary of MSC Translation Layer Flow Control

Scenario	Responder or Requester Reaction	Code	Notes
Bad offset value in command.	Send ABORT packet, go to IDLE.	0x10	1
Bad opcode in command.	Send ABORT packet, go to IDLE.	0x08	2, 3
Too few packets in command.	Wait until timeout, go to IDLE.	0x04	4, 9
Incomplete packet.	Timeout, go to IDLE.	0x04	5
Opposite device is busy and cannot process new request.	Send NACK packet, go to IDLE.	0x20	6
Protocol error.	Send ABORT packet, go to IDLE.	0x02	7
Re-try threshold exceeded.	Timeout, go to IDLE.	0x01	8

Notes on Table 7-4:

1. Responder shall interpret this as an illegal command, for the reason that a data packet has a bad offset value. A "bad offset" is one which does not align to the MSC Offset Range limits shown in Table 7-21 on page 140.
2. "Bad opcode" has two meanings. It indicates that a valid opcode is not supported by the responder or requester. Indicating an illegal command allows legacy devices to flag commands defined in later versions of this specification which the device does not support. It also can mean that the value in the first control packet of a command is invalid.
3. An idle responder which receives a data packet at the beginning of a command (all commands begin with a control packet) returns this error code.
4. Responder waits maximum time for requester to send a packet ($T_{PKT\_SENDER\_TIMEOUT}$), then responder goes to the idle state with confidence that the MSC channel is free from both ends. This also applies to requester waiting $T_{PKT\_RECEIVER\_TIMEOUT}$ for responder to send packet. The missing packet can be an expected data packet, or an expected control packet (such as EOF from Requester or ACK from Responder), which does not arrive before timeout.
5. Follower begins to receive a new packet from the Initiator, but the Initiator stops sending bits before the end of the packet. CBUS is HIGH. Follower goes to idle.
6. When N_{MAX} (maximum packets allowed before arbitration) is higher than 1, allowing a Requester to send a series of packets without allowing the Responder to win arbitration , it is possible that the Responder's NACK packet may not be sent out while remaining packets arrive from the Requester. If the Responder cannot send out NACK packet for "Device is busy" (0x20), the Responder may see the additional incoming packets from the Requester to be a "Protocol Error" (0x02). In such cases, the Requester will receive ABORT packet (code 0x02) instead of the NACK packet.
7. A bad packet sequence can be an extra packet or a packet of the wrong type: a control packet when a data packet is expected in the command protocol, or a data packet when a control packet is expected. (Control versus Data is indicated by the control bit). A control packet with an opcode different than the expected opcode (as per state transition diagrams for each command) shall also be considered as protocol error.
8. Requester re-tries the packet up to the limit, then goes to the IDLE state.
9. Responder waits maximum time for Requester to finish a command ($T_{CMD\_SENDER\_TIMEOUT}${max}), then Responder goes to the idle state with confidence that the

MSC channel is free from both ends. This also applies to Requester waiting $T_{CMD\_RECEIVER\_TIMEOUT}$\{min\} for Responder to finish a command.

Figure 7-14 shows an example of how an error condition in Table 7-4 might occur.

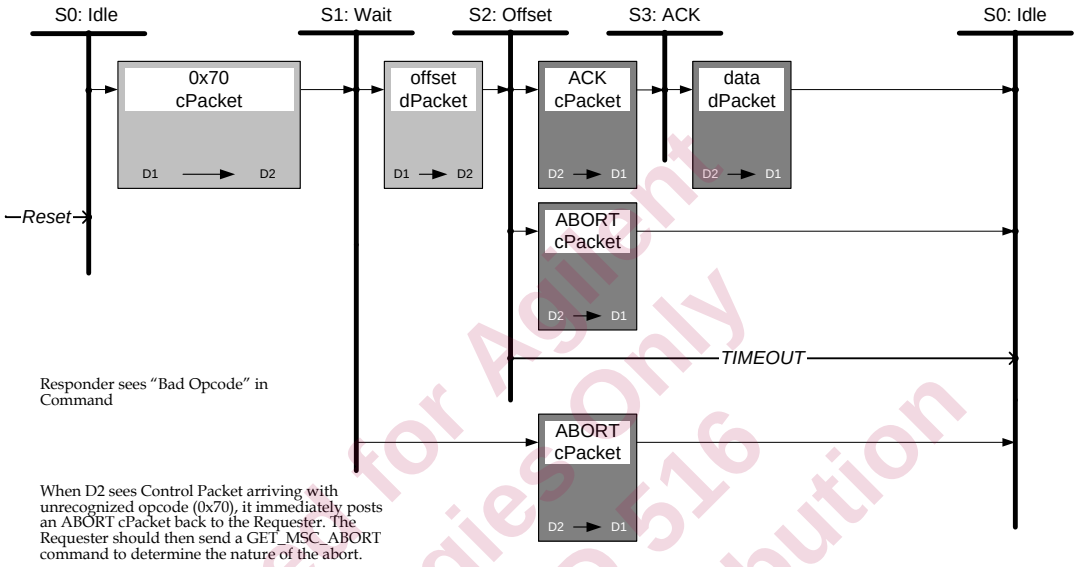


Figure 7-14. Example of MSC "Bad Opcode" Handling

7.4.2 Support Requirements for MSC Commands

All MHL Source or Sink devices shall support all MSC commands listed in Table 7-3 by replying to each with the protocol of packets described in the following sections.

MHL allows Dongles to indicate no support for any or all of RCP sub-commands or RAP sub-commands or UCP sub-commands. See RCP_SUPPORT and RAP_SUPPORT and UCP_SEND_SUPPORT and UCP_RECV_SUPPORT bits, described in Section 7.9.1.10. Other MHL devices are prohibited from sending RCP or RAP or UCP to a device which indicates no support for one or the other of these sub-commands. See Section 7.6.3 through Section 7.8.4

7.4.3 MHL Sideband Channel (MSC) Commands

The CBUS provides a mechanism for MHL-specific commands, including read and write access to MHL-specific registers. Most of these transactions can be initiated by either the Source or the Sink. Exceptions are noted below.

MSC transactions are identified by a header field of 0b10 in each packet at the link layer.

Valid offset values to registers within MHL devices are limited to those registers defined in Table 7-21.

NOTE: This section describes the translation layer commands for the MSC protocol. DDC commands are described in section 7.5.

7.4.3.1 GET_STATE Command

A single byte value is requested from the responder, which indicates the state of the responder. This value is identical to the value in the Capability Register at offset 0x00 (see Section 7.9.1.1 on page 143). Note that the values in the Capability Registers are ready to be read only after the device has sent DCAP_RDY=1. Therefore, GET_STATE may return an uninitialized value if DCAP_RDY status bit has not been sent from the opposite device. See section 7.9.3.1.

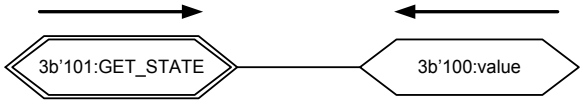


Figure 7-15. MSC GET_STATE Command

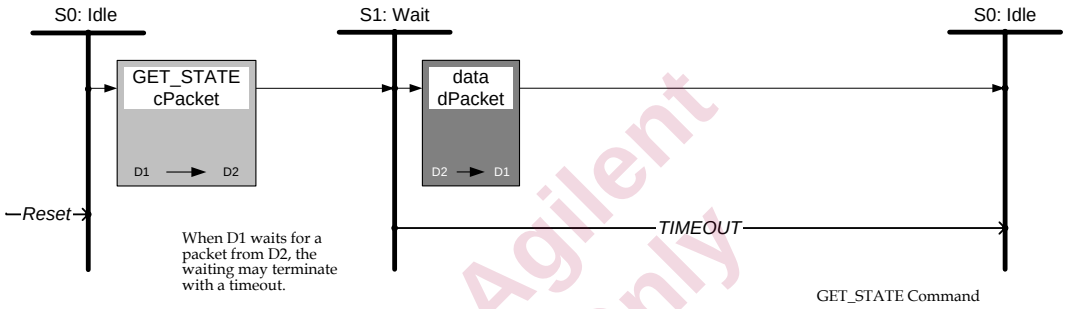


Figure 7-16. MSC GET_STATE State Diagram

7.4.3.2 GET_VENDOR_ID Command

Vendor ID is an eight bit value indicating the type of device providing the response. The vendor ID is associated with the maker of the transmitter, receiver or bridge device, and may not reflect the name of the MHL system-level designer, OEM or ODM.

Figure 7-17. MSC GET_VENDOR_ID Command

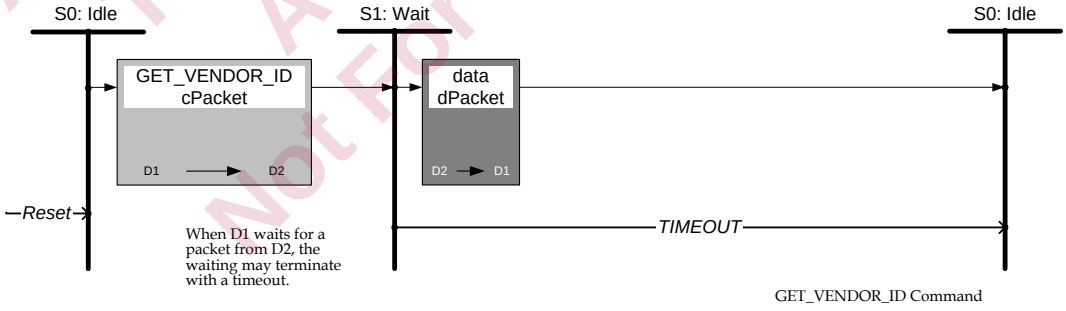


Figure 7-18. MSC GET_VENDOR_ID State Diagram

7.4.3.3 SET_HPD Command

This command shall be initiated only by an MHL Sink or Dongle.

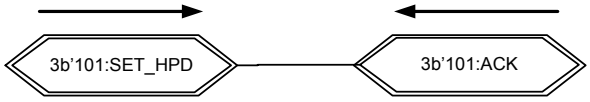


Figure 7-19. MSC SET_HPD Command

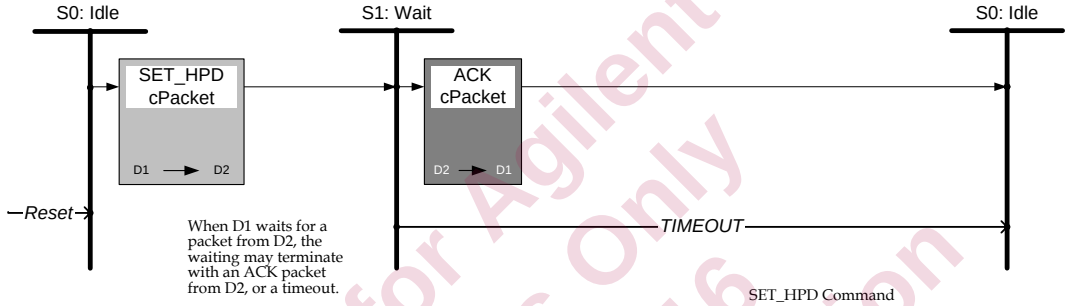


Figure 7-20. MSC SET_HPD State Diagram

A SET_HPD command shall not follow a CLR_HPD command within less than T_{HPD_WIDTH}.

7.4.3.4 CLR_HPD Command

This command shall be initiated only by an MHL Sink or Dongle.

Figure 7-21. MSC CLR_HPD Command

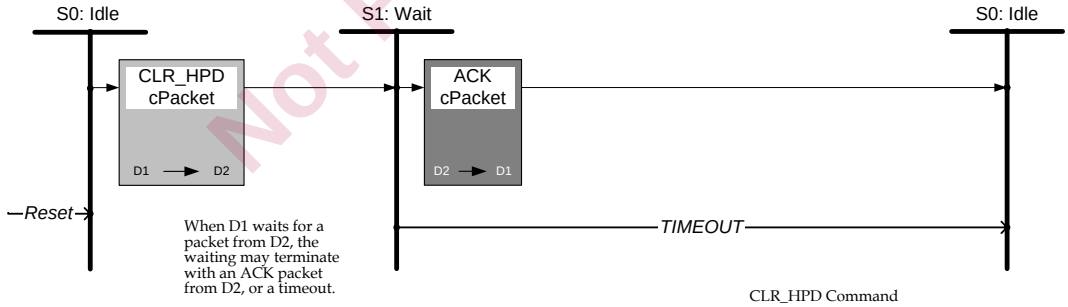


Figure 7-22. MSC CLR_HPD State Diagram

A SET_HPD command shall not follow a CLR_HPD command within less than T_{HPD_WIDTH}.

7.4.3.5 GET_DDC_ERRORCODE Command

GET_DDC_ERRORCODE retrieves an error code associated with the aborted termination of the preceding command on the DDC channel.

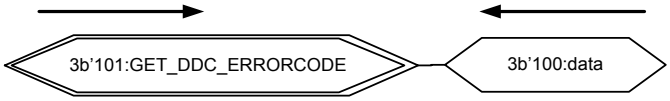


Figure 7-23. MSC GET_DDC_ERRORCODE Command

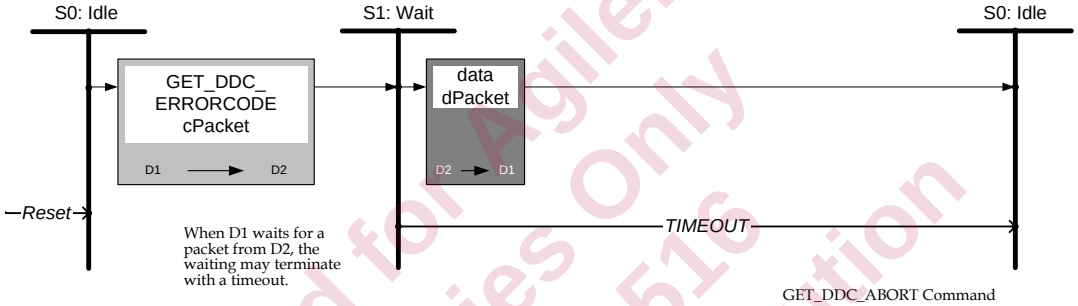


Figure 7-24. MSC GET_DDC_ERRORCODE State Diagram

The error code in the data packet should be one of the values shown in Table 7-5. Not all MHL devices may be able to implement all of these error codes.

Table 7-5. GET_DDC_ERRORCODE Return Codes

Error Code	Explanation
0x00	Normal response, no error.
0x01	Re-try threshold exceeded.
0x02	Protocol error.
0x04	Peer Timed Out.

The 8-bit error code is written each time an error occurs. The value retrieved with the GET_DDC_ERRORCODE command reflects the type of the last error which occurred.

7.4.3.6 GET_MSC_ERRORCODE Command

GET_MSC_ERRORCODE retrieves an error code associated with the aborted termination of the preceding command on the MSC channel. Specific error codes are listed in Table 7-6.

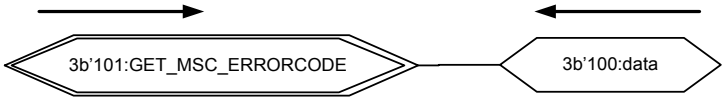


Figure 7-25. MSC GET_MSC_ERRORCODE Command

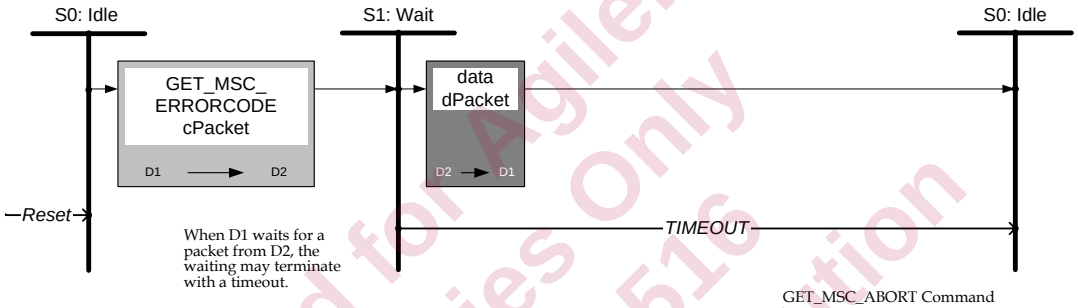


Figure 7-26. MSC GET_MSC_ERRORCODE State Diagram

The error code in the data packet should be one of the values shown in Table 7-6. Not all MHL devices may be able to implement all of these error codes.

Table 7-6. GET_MSC_ERRORCODE Return Codes

Error Code	Explanation
0x00	Normal response, no error.
0x01	Re-try threshold exceeded.
0x02	Protocol Error
0x04	Peer Timed Out.
0x08	Invalid Op Code.
0x10	Bad Offset in Command.
0x20	Peer is busy, cannot process new request.

The entries in Table 7-6 indicate the codes returned with the GET_MSC_ERRORCODE command, and show which of the requester and/or responder scenarios from Section 7.4.1 are covered by each code.

The 8-bit error code is written each time an error occurs. The value retrieved with the GET_MSC_ERRORCODE command reflects the type of the last error which occurred.

7.4.3.7 MSC READ_DEVCAP Command

A single byte may be requested from a responder and returned by the responder using the READ_DEVCAP command. The command begins with a packet prefixed with the 0b10 for the MSC channel, and the 0b1 as a control packet; and completed with the READ_DEVCAP opcode.

This is followed by a data packet with the offset to the desired byte within the address offset space for the Capability Registers (see Section 7.9.1 on page 141).

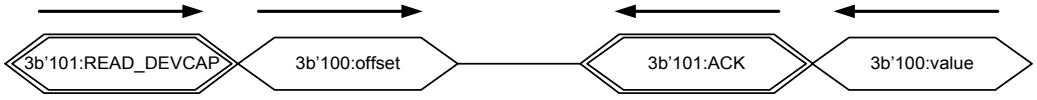


Figure 7-27. MSC Read Device Capability Command

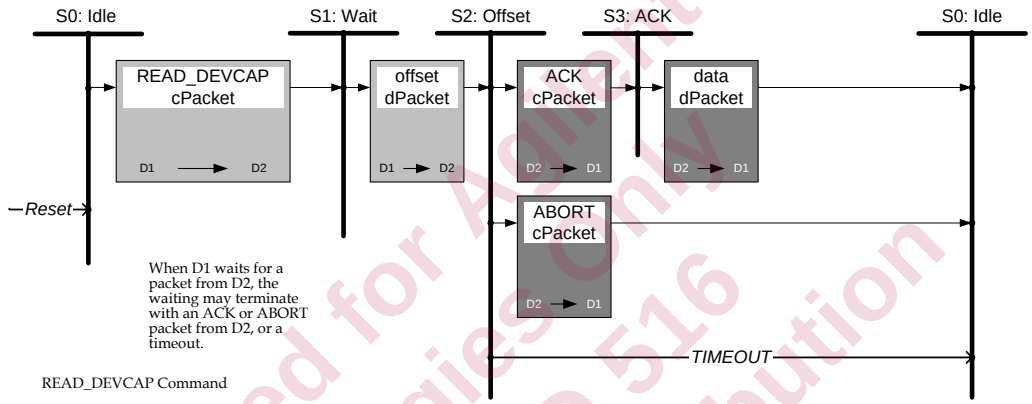


Figure 7-28. MSC READ_DEVCAP State Diagram

The reply from the responder begins with a control packet with the ACK opcode. In the case of a successful read, the ACK is followed by a data packet with the requested single byte value.

If the offset is not within the valid range for Capability Registers, then the responder shall reply with an ABORT control packet, and no data packet. See Table 7-21.

An MHL Source shall re-read (from the opposite, connected MHL Sink or Dongle) all Device Capability registers that are necessary to correctly update its state whenever it receives the DCAP_CHG interrupt, and complete this reading process in no more than $T_{SRC\_READ\_DCAP}$\{max\} (for example the POW bit that is used to disable the VBUS from the Source). The reading device is responsible for comparing the newly-read register values with those previously read to determine which registers have changed value.

An MHL Sink or Dongle may read the Device Capability registers from the Source. There is no maximum time limit specified for such a read by a Sink or Dongle.

In order to send ADOPTER_ID in a WRITE_BURST command, a Sink (or a Dongle supporting WRITE_BURST) shall previously have read the ADOPTER_ID bytes in the Source's Device Capability registers, triggered by the Source's use of DCAP_RDY status bit or DCAP_CHG interrupt.

7.4.3.8 MSC WRITE_STAT Command

A single byte may be written from one device to another using the WRITE_STAT command. The command begins with a packet prefixed with 0b10 for the MSC channel, and the 0b1 as a control packet; and completed with the WRITE_STAT opcode. This is followed by two data packets: the first with the offset, the second with the data value.

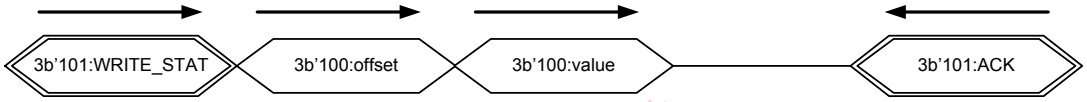


Figure 7-29. MSC Write Status Command

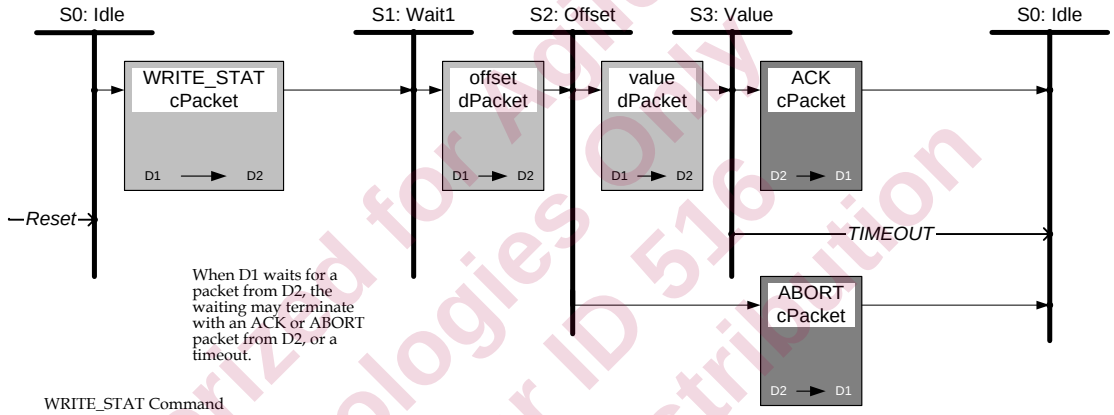


Figure 7-30. MSC WRITE_STAT State Diagram

WRITE_STAT over-writes all bits in the destination offset with the eight bits in the value data packet.

The WRITE_STAT command expects an ACK from the responder, unless no register exists at the specified offset, in which case the responder replies with an ABORT.

If the offset is not within the valid range for Status Registers, then the responder shall reply with an ABORT control packet. If the offset is within the valid range for Status Registers, then the responder shall not reply with an ABORT control packet. See Table 7-21 and Note #1 under Table 7-4.

7.4.3.9 MSC SET_INT Command

One or more bits may be set to '1' in a single byte in one device from another using the SET_INT command. The command begins with a packet prefixed with 0b10 for the MSC channel, and the 0b1 as a control packet; and completed with the SET_INT opcode. This is followed by two data packets: the first with the offset, the second with the data value.

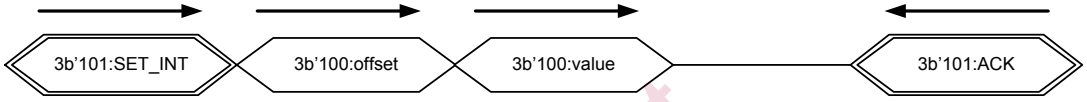


Figure 7-31. MSC Set Interrupt Command

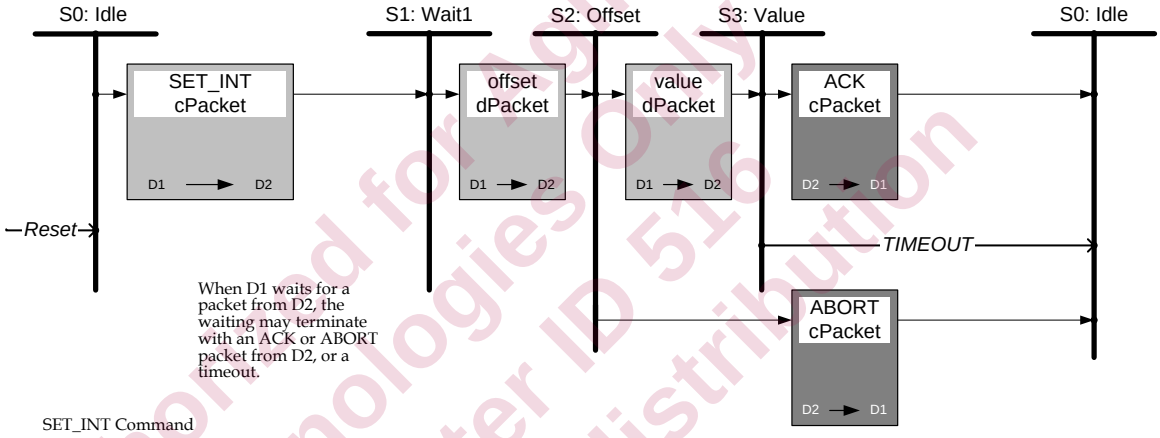


Figure 7-32. MSC SET_INT State Diagram

SET_INT over-writes a bit in the destination offset with a 0b1 if the bit in the value data packet is a 0b1. That is, SET_INT can only set a bit to one, it cannot clear a bit to zero.

The SET_INT command expects an ACK from the responder, unless no register exists at the specified offset, in which case the responder replies with an ABORT.

If the offset is not within the valid range for Interrupt Registers, then the responder shall reply with an ABORT control packet. If the offset is within the valid range for Interrupt Registers, then the responder shall not reply with an ABORT control packet. See Table 7-21 and Note #1 under Table 7-4.

7.4.3.10 MSC Register Write Burst

Up to 16 data bytes can be written from one device to another using one WRITE_BURST command. The command begins with a packet prefixed with 0b10 for the MSC channel, and the 0b1 as a control packet; and completed with the WRITE_BURST opcode. After the offset value, the command requester may send up to sixteen (16) data values. The requester completes the command by sending an EOF control packet.

Note that no explicit count value is sent. The responder accepts data values, beginning at the offset in the 'offset' data packet, and auto-incrementing with each arriving data value packet until it sees the EOF control packet.

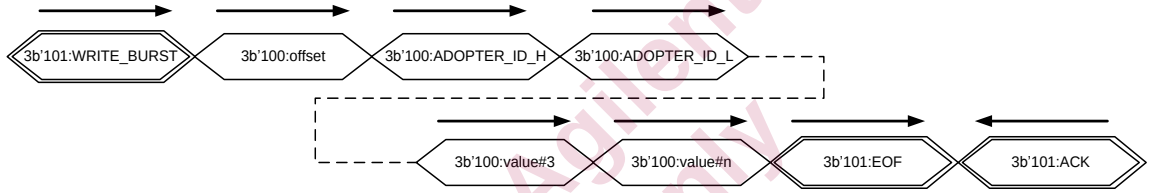


Figure 7-33. MSC Burst Write Command

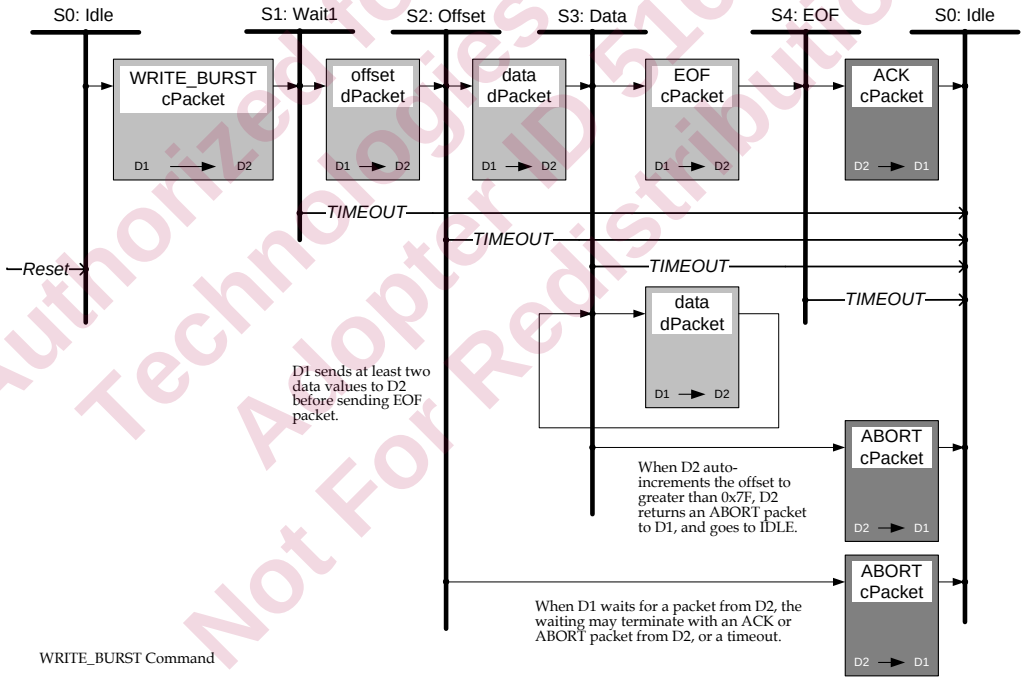


Figure 7-34. MSC WRITE_BURST State Diagram

The reply from the responder begins with a control packet with the ACK or ABORT opcode. If the write is successful, the responder sends an ACK command. If the write is not successful, the responder sends an ABORT command. If sending an ABORT, the responder may write to its local memory those data bytes which correspond to valid offsets within its memory. The responder should treat the Scratchpad Registers as a corrupted memory space unless the WRITE_BURST command is successfully completed.

The requester and responder coordinate their actions using the REQ_WRT and GRT_WRT interrupts. See Sections 0 and 7.9.4.

A responder which receives a WRITE_BURST command which meets any of the following conditions shall reply to the requester with an ABORT packet:

1) The initial offset value is outside the valid range (see Table 7-21).
2) The sequence of data packets after the offset value and up to the EOF control packet causes the auto-incrementing offset to be greater than the maximum allowed offset.
3) The offset packet is immediately followed by a control packet.

In all other cases when WRITE_BURST is supported, the responder shall reply with an ACK packet.Values in the data packets, including ADOPTER_ID, shall not cause the Responder to send an ABORT packet.

See Device Capability register Section 7.9.1.10.

Refer to the register map in Table 7-21 for a list of the required address spaces in an MHL device.

The following three paragraphs apply when using WRITE_BURST to transfer data, except in cases defined in Section 5.9.1.1.

The first two data packets after the offset packet in a WRITE_BURST shall contain the ADOPTER_ID_H and ADOPTER_ID_L values, respectively. (See Section 7.9.1.4.)

This ADOPTER_ID value (two bytes total) shall be the ADOPTER_ID of the responder – the device receiving data into its Scratchpad Registers. By sending the responder's ADOPTER_ID, the requester demonstrates that it has already read the responder's ADOPTER_ID from the responder's Device Capability registers, and that the Scratchpad Register data is intended for a device with that specific ADOPTER_ID.

A responder which receives a WRITE_BURST command with an ADOPTER_ID which does not match its own ADOPTER_ID should discard the WRITE_BURST data. Methods for a responder to indicate to the requester that it has discarded data are beyond the scope of this specification.

7.4.3.11 MSC Message

The MSC channel provides the top-level command (MSC_MSG), under which other sub-commands may be used under the general heading of "messaging".

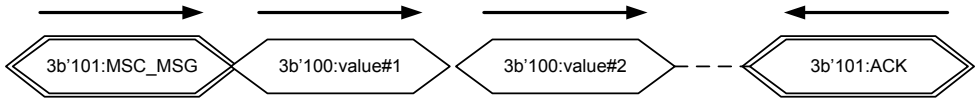


Figure 7-35. MSC MSG Command

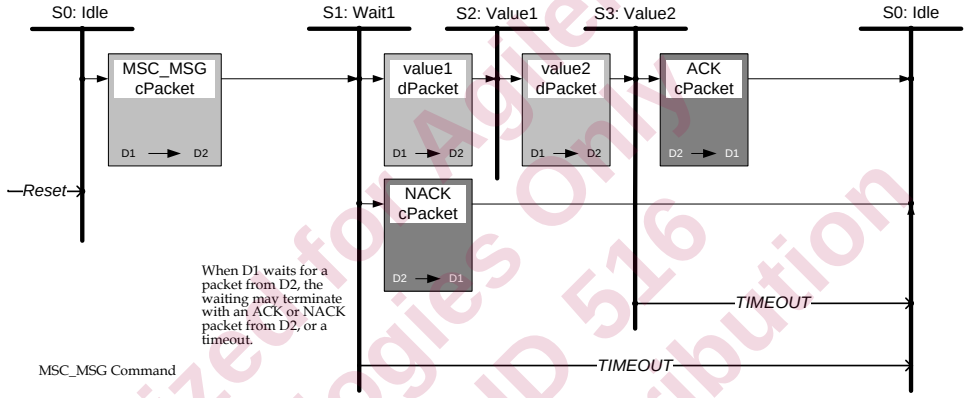


Figure 7-36. MSC MSC_MSG State Diagram

The current version of the MHL Specification provides only the sub-commands listed in Table 7-7. Additional sub-commands may be added in future revisions of this Specification. Support for these will be indicated through a mechanism defined in that future version.

Note that the packet containing the sub-command code is a data packet, not a control packet. Source or Sink systems may choose to implement the processing of sub-commands in firmware. In such cases, the firmware must provide flow control to prevent multiple MSC_MSG commands from arriving at a Source from a Sink (or vice versa) and over-writing each other. CBUS provides hardware flow control only at the link layer. Thus, for example, an MSC_MSG requester shall wait until it receives an acknowledgment command (RCPK, RCPE, RAPK or MSGE) in return from the responder before the requester sends another MSC_MSG sub-command.

Table 7-7. MSC Message Sub-commands

Command	Opcode	Description	Section
MSGE	0x02	MSC_MSG Error sub-command	7.4.3.12
RCP	0x10	Remote Control Protocol sub-command	7.7.1
RCPK	0x11	RCP Acknowledge sub-command	7.7.2
RCPE	0x12	RCP Error sub-command	7.7.3
RAP	0x20	Request Action Protocol sub-command	7.6.1
RAPK	0x21	RAP Acknowledge sub-command	7.6.2
UCP	0x30	UTF-8 Character Protocol sub-command	7.8.1
UCPK	0x31	UCP Acknowledge sub-command	7.8.2
UCPE	0x32	UCP Error sub-command	7.8.3

Since MSC_MSG may transmit more than one type of sub-command, a device shall wait for the acknowledgment sub-command in return before transmitting another MSC_MSG sub-command. That is, for example, after sending an RCP sub-command, a device shall not send another RCP or RAP sub-command until the device has received either RCPK or RCPE, or timed out. The same is true for RAP sub-commands. Examples of proper and improper sub-command sequences are shown in Figure 7-37 to Figure 7-38. *(Note that the MSC_MSG control packet and ACK control packet are not shown in these figures.)*

An MSC_MSG command which sends a sub-command opcode not shown in Table 7-7 shall expect to receive back from the responder an MSGE sub-command, as described in Section 7.4.3.12.

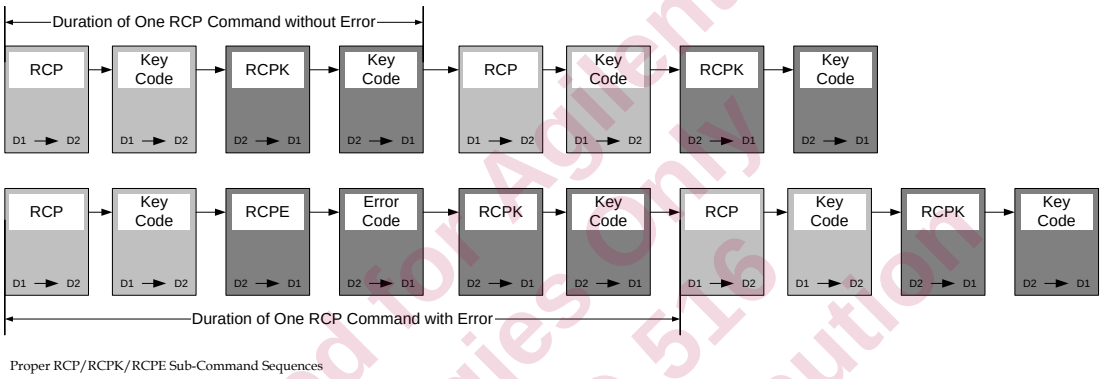


Figure 7-37. Proper MSC MSC_MSG Sub-command Sequencing

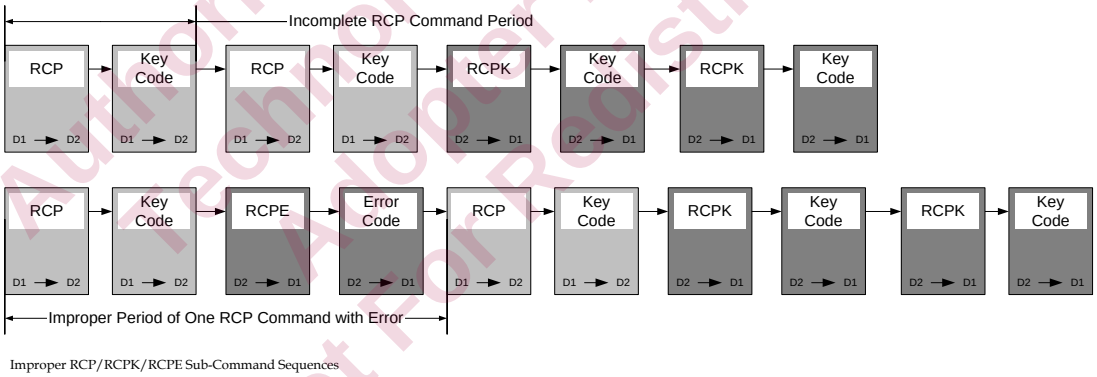


Figure 7-38. Improper MSC MSC_MSG Sub-command Sequencing

The NACK packet back from the responder indicates that the preceding MSC_MSG command has not been processed yet from that device's input queue. In order to prevent collision from sequential MSC_MSG commands, the NACK packet response tells the requester that it needs to re-send the MSC_MSG command. If the receiving device pulls the command from its input stage (after having given each packet a link layer Ack), then it can accept the next MSC_MSG command.

The requester is recommended to wait a period of $T_{NACK\_RETRY\_NEXT}$ before resending the sub-command. There is no limit to the number of times a sub-command may be resent before receiving a positive response from the responder. See Table 13-36.

Note that an MHL Dongle which indicates no support for RAP, yet receives an RAP command, may respond with a translation layer NACK. An MHL Source which receives such a NACK from a Dongle should not retry the sub-command. See Sections 7.6.3 and 7.7.8.1.

7.4.3.12 MSGE Sub-Command

A device which receives an MSC_MSG command with a sub-command opcode not found in Table 7-7 should respond with an MSC_MSG MSGE sub-command, including the error code value 0x01 in the second data packet.

Figure 7-39 shows a single MSC MSGE command transfer.

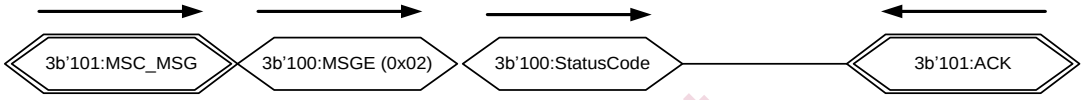


Figure 7-39. MSC MSGE Sub-command

Table 7-7-1. MSC Remote Control Protocol Status Codes

Opcode	Status Codes	Explanation
0x00	No Error	Not allowed.
0x01	Invalid sub-command code	The sub-command code in the MSC_MSG command is not valid.

7.5 DDC Commands

The MHL Source is the logical master of DDC commands. It initiates reads and writes to the MHL receiver, to the EDID and to other devices on the Sink side. The Source does not have any DDC ports – its only connection to the DDC devices is via the CBUS. The Sink device may implement a DDC I2C master to interface between the CBUS and one or more downstream DDC ports.

For both reads and writes, the actual register may be located either at the Sink device (e.g. a BKSV read during part one of an HDCP authentication [refer to HDCP Specification]) or on an external chip (e.g. EDID read). The Source does not need to know the physical location of the register being accessed. The Sink translation layer will handle the command appropriately.

DDC commands shall be initiated only by an MHL Source, not by an MHL Sink.

DDC commands are identified by a header field of 0b00 in each packet at the link layer.

Each DDC command is constructed from a sequence of control packets and data packets. The control packets available on DDC are listed in Table 7-8.

Table 7-8. DDC Channel Translation Layer Control Packets

Code	Control Packet	Description
0x30	SOF	Start of Frame
0x32	EOF	End of Frame
0x33	ACK	Command or Data byte acknowledge
0x34	NACK	Command or Data byte not acknowledge
0x35	ABORT	Transaction abort
0x50	CONT	The transfer keeps going.
0x51	STOP	The transfer approaches to end.

7.5.1 Support Requirements for DDC Commands

All MHL Source or Sink devices shall support all DDC commands listed in Table 7-9 by replying to each with the protocol of packets described in the following sections.

Table 7-9. DDC Command Support in MHL

Command	Source Support	Sink Support	Section
DDC Read	Required	Required	7.5.3.1
DDC Segment Read	Required	Required	7.5.3.2
DDC Short Read	Required	Required	7.5.3.3
DDC Current Read	Required	Required	7.5.3.3
DDC Write	Required	Required	7.5.3.4

7.5.2 DDC Command Reactions

Many scenarios exist in which a command does not complete as expected. The responder and the requester are required to react to these situations in predictable ways.

Table 7-10. Summary of DDC Translation Layer Flow Control

Scenario	Responder or Requester Reaction	Code	Notes
Too few bytes in command.	Wait until timeout, send ABORT packet, go to IDLE.	0x04	1
Protocol error.	Send ABORT packet, go to IDLE.	0x02	2, 3, 4
Incomplete packet.	Wait until timeout, may send ABORT packet, go to IDLE.	0x04	5
Re-try threshold exceeded.	Send ABORT, go to IDLE.	0x01	

Notes on Table 7-10:

1. Responder waits maximum time for requester to send a packet ($T_{PKT\_SENDER\_TIMEOUT}$), then responder sends an ABORT packet and goes to the idle state. Applies also to requester waiting for responder to send a packet ($T_{PKT\_RECEIVER\_TIMEOUT}$). The missing packet can be an expected data packet, or an expected control packet (such as EOF), which does not arrive before timeout.
2. A bad packet sequence can be an extra packet or a packet of the wrong type (control versus data indicated in the packet header).
3. A bad packet can be an expected control packet, but with a valid opcode that is not supported by the responder or requester. Indicating an illegal command allows legacy devices to flag commands defined in later versions of this specification which the device does not support.
4. A Sink may treat an SOF as a valid packet even if the SOF occurs in the middle of a command sequence. In such cases, the Sink may discard the ongoing command sequence, without sending an ABORT packet, and begin a new command with the received SOF packet.
5. Responder begins to receive a new packet from the Requester, but the Requester stops sending bits before the end of the packet. CBUS is HIGH. Responder re-arbitrates for the bus, then sends ABORT packet to Requester indicating an incomplete packet was seen.

Requester can use GET_DDC_ERRORCODE command to retrieve the error code from the responder.

7.5.3 DDC Channel Commands

The commands on the DDC channel are quite similar to each other. Rather than provide a complete state diagram for each command, the most basic command is shown first. Each subsequent command shows the additions to the state machine of the first command, referring to the entry point appropriately. Such common descriptions are not meant to indicate any specific, common logic level implementation of these commands in MHL devices.

7.5.3.1 DDC Read

The DDC Read command provides a means of transferring a sequence of one or more bytes of data from the responder to the requester. There is no 'count' of requested bytes sent from the requester to the responder. Bytes are sent from responder until the requester sends a STOP control packet instead of a CONT control packet. The data at the offset address sent after the device address will be returned first – after the first CONT control packet from the requester. The responder auto-increments the offset internally, returning one data byte from each consecutive offset address each time it receives a CONT control packet.

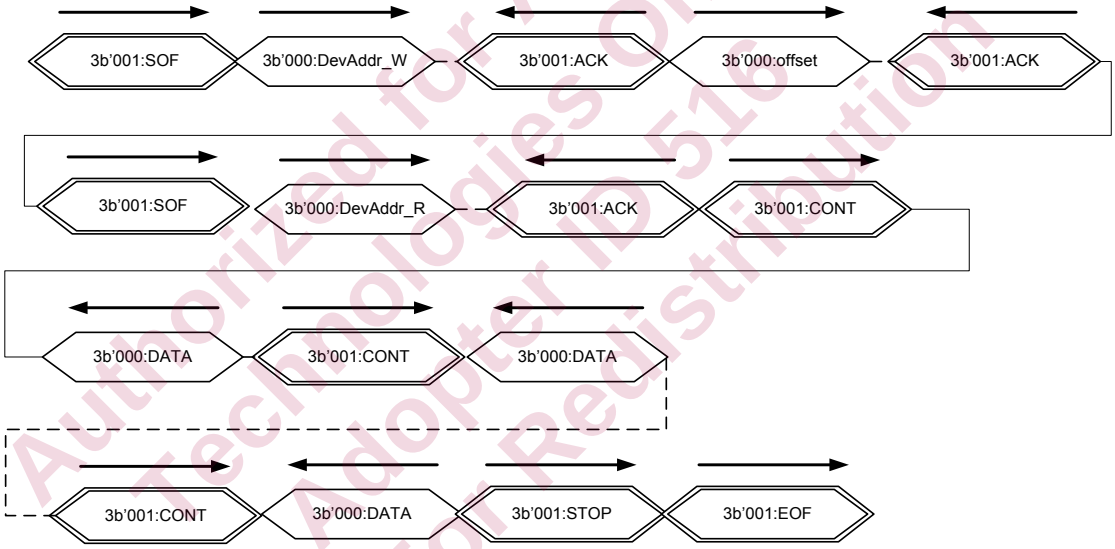


Figure 7-40. DDC Read Command

1. MHL Source performs arbitration and gets control of the bus.
2. MHL Source sends SOF control packet in order to set up a DDC frame.
3. MHL Source sends DDC device address with READ bit cleared, as data packet.
4. MHL Sink sends ACK control packet to MHL Source.
5. MHL Source sends DDC offset as data packet to MHL Sink.
6. MHL Sink sends ACK control packet to MHL Source.
7. MHL Source sends SOF control packet to MHL Sink.
8. MHL Source sends DDC device address with READ bit set, as data packet.
9. MHL Sink sends ACK control packet to MHL Source.
10. MHL Source sends CONT control packet to MHL Sink.
11. MHL Sink sends byte data as data packet to MHL Source.
12. Step 10 and 11 are repeated until Source has received all requested data.
13. MHL Source sends STOP control packet to MHL Sink.
14. MHL Source sends EOF control packet to close a DDC frame.

If, at any point in the command, the requester cannot accept more packets, it shall send an STOP control packet, followed by an EOF control packet; instead of the next CONT or STOP control

packet to the responder. This situation may occur, for example, when an MHL transmitter reads data from a downstream MHL receiver and tries to send that received data upstream across a DDC bus to an upstream DDC-equipped transmitter.

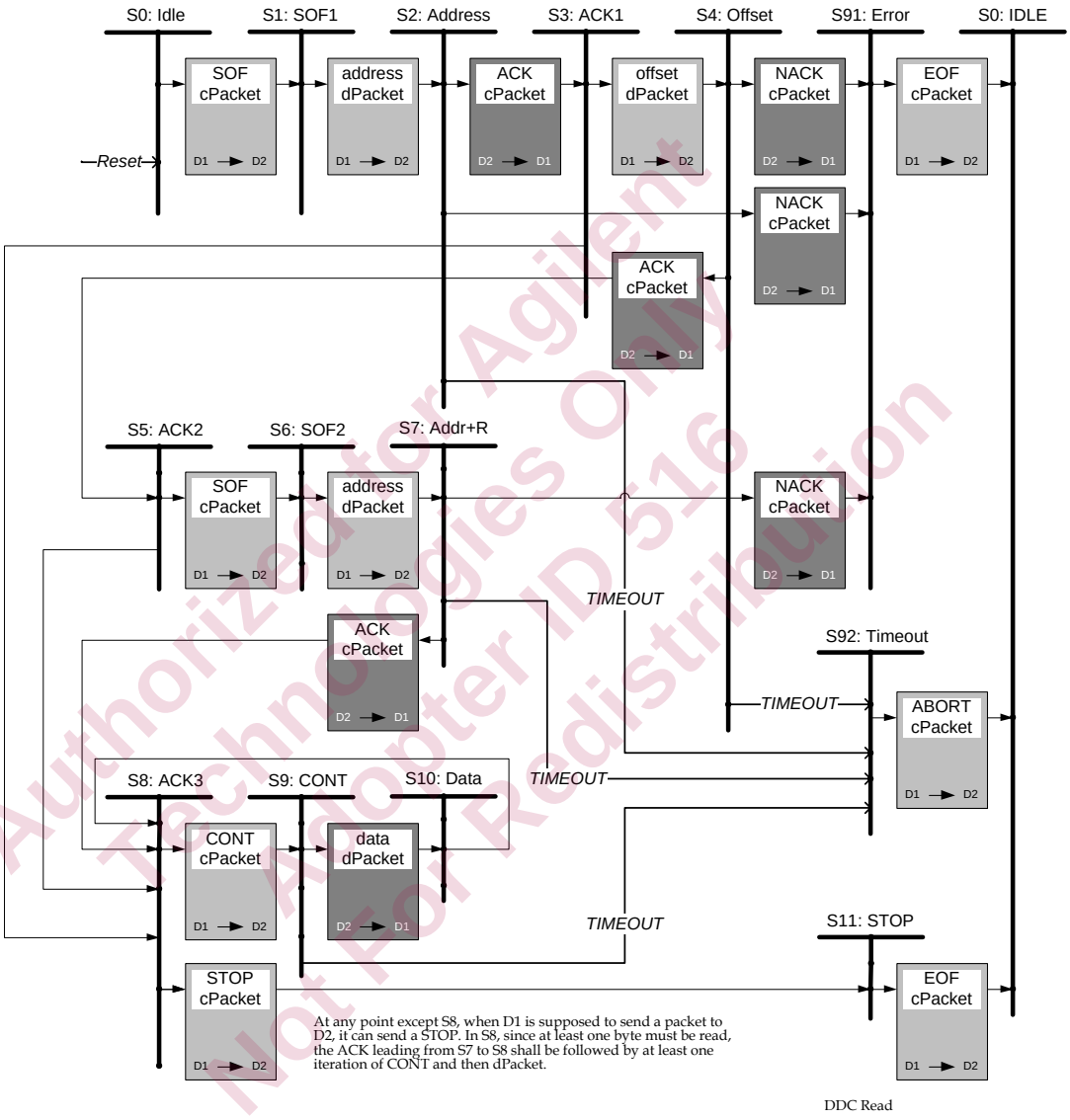


Figure 7-41. MHL DDC Read State Diagram

7.5.3.2 DDC Segment Read

The DDC Segment Read command is executed in one of two ways, according to whether the Sink device supports the Segment Pointer. Support for the Segment Pointer is not required in Sinks and Dongles which have an EDID no larger than 256 bytes. Figure 7-42 illustrates the packet sequence between a Source and a Sink which supports Segment Pointer – as shown by the ACK packet returned from the Sink after the Source's '0x60_W' and 'seg_val' data packet is each sent. Figure 7-43 illustrates the packet sequence between a Source and a Sink which does not support the Segment Pointer. The Sink returns a NACK packet after the '0x60_W' address data packet, which causes the Source to send an EOF packet and immediately move to the next SOF.

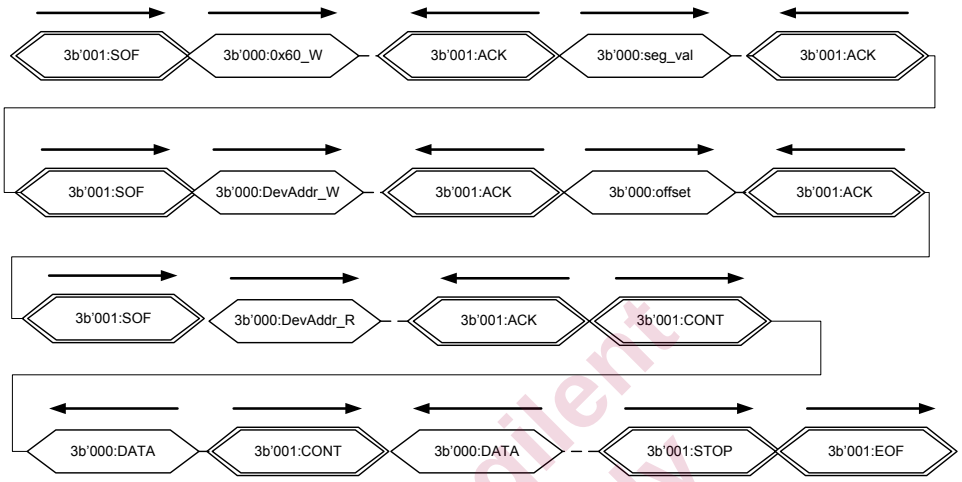


Figure 7-42. DDC Segment Read Command to Sink Supporting Segment Pointer

1. MHL Source performs arbitration and gets control of the bus.
2. MHL Source sends SOF control packet in order to set up a DDC frame.
3. MHL Source sends DDC device address 0x60 with READ bit cleared.
4. MHL Sink sends ACK control packet to MHL Source.
5. MHL Source sends segment value as data packet to MHL Sink.
6. MHL Sink sends ACK control packet to MHL Source.
7. MHL Source sends SOF control packet to MHL Sink.
8. MHL Source sends DDC device address 0xA0 (or 0xA4) with READ bit cleared.
9. MHL Sink sends ACK control packet to MHL Source.
10. MHL Source sends DDC offset as data packet to MHL Sink.
11. MHL Sink sends ACK control packet to MHL Source.
12. MHL Source sends SOF control packet to MHL Sink.
13. MHL Source sends DDC device address with READ bit set.
14. MHL Sink sends ACK control packet to MHL Source.
15. MHL Source sends CONT control packet to MHL Sink.
16. MHL Sink sends byte data as data packet to MHL Source.
17. Step 15 and 16 are repeated until Source has received all requested data.
18. MHL Source sends STOP control packet to MHL Sink.
19. MHL Source sends EOF control packet to close a DDC frame.

Note: An MHL Sink device may send a NACK packet (step #6) to indicate its non-support for the requested segment value. The MHL Source device then sends an EOF packet (step #19), to which the MHL Sink may respond with an ABORT packet.

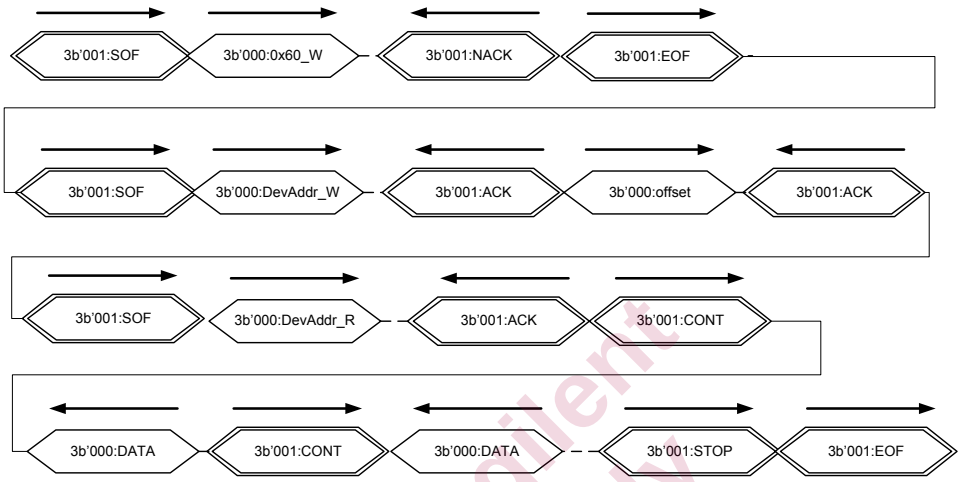


Figure 7-43. DDC Segment Read Command to Sink Not Supporting Segment Pointer

1. MHL Source performs arbitration and gets control of the bus.
2. MHL Source sends SOF control packet in order to set up a DDC frame.
3. MHL Source sends DDC device address 0x60 with READ bit cleared.
4. MHL Sink sends NACK control packet to MHL Source.
5. MHL Source sends EOF control packet to MHL Sink.
6. MHL Source sends SOF control packet to MHL Sink.
7. MHL Source sends DDC device address 0xA0 (or 0xA4) with READ bit cleared.
8. MHL Sink sends ACK control packet to MHL Source.
9. MHL Source sends DDC offset as data packet to MHL Sink.
10. MHL Sink sends ACK control packet to MHL Source.
11. MHL Source sends SOF control packet to MHL Sink.
12. MHL Source sends DDC device address with READ bit set.
13. MHL Sink sends ACK control packet to MHL Source.
14. MHL Source sends CONT control packet to MHL Sink.
15. MHL Sink sends byte data as data packet to MHL Source.
16. Step 15 and 16 are repeated until Source has received all requested data.
17. MHL Source sends STOP control packet to MHL Sink.
18. MHL Source sends EOF control packet to close a DDC frame.

Note: An MHL Sink device may send an ABORT packet in reaction to receiving an EOF packet from the Source (step #5), after the Sink has already signaled "non-support" by sending a NACK packet to the Source (step #4).

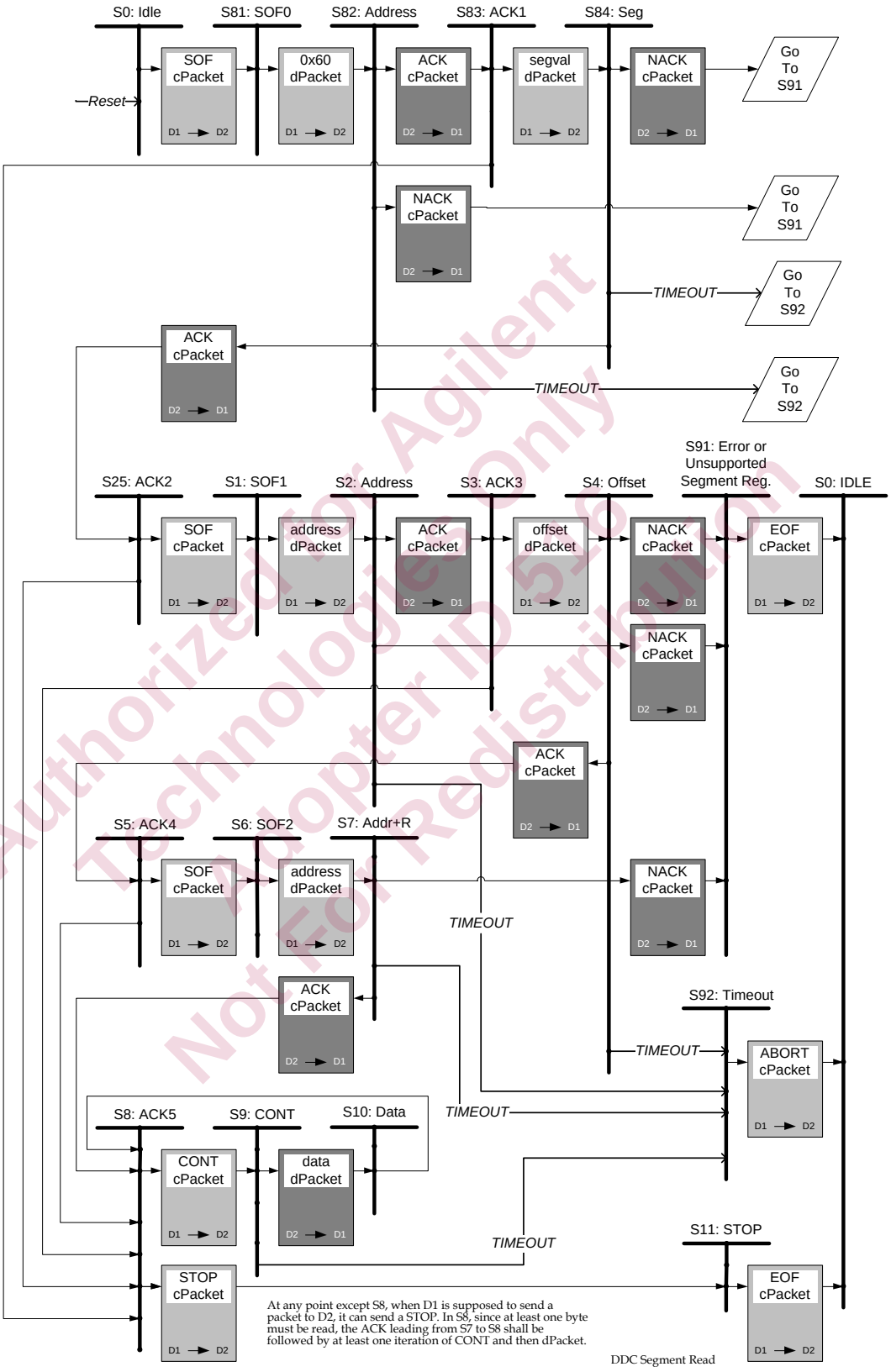


Figure 7-44. MHL DDC Segment Read State Diagram

7.5.3.3 DDC Short Read and DDC Current Read

The Short Read command supports reads only from the device address of the HDCP Receiver. All other device addresses and offsets are invalid for the Short Read command. The offset is automatically set to 0x08 without being explicit in the command.

The Current Read command supports reads only from the device address of the EDID. All other device addresses and offsets are invalid for the Current Read command. The offset is automatically incremented without being explicit in the command.

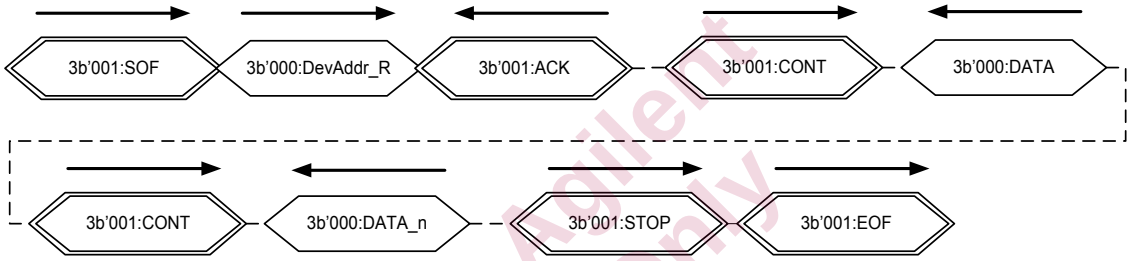


Figure 7-45. DDC Short Read and Current Read Commands

1. MHL Source performs arbitration and gets control of the bus.
2. MHL Source sends SOF packet in order to set up a DDC frame.
3. MHL Source sends DDC device address with READ bit set, to MHL Sink.
4. MHL Sink sends ACK control packet to MHL Source.
5. MHL Source sends CONT control packet to MHL Sink.
6. MHL Sink sends byte data to MHL Source.
7. Repeat steps 5 and 6 for the number of bytes to be received.
8. MHL Source sends STOP control packet to MHL Sink.
9. MHL Source sends EOF packet to close a DDC frame.

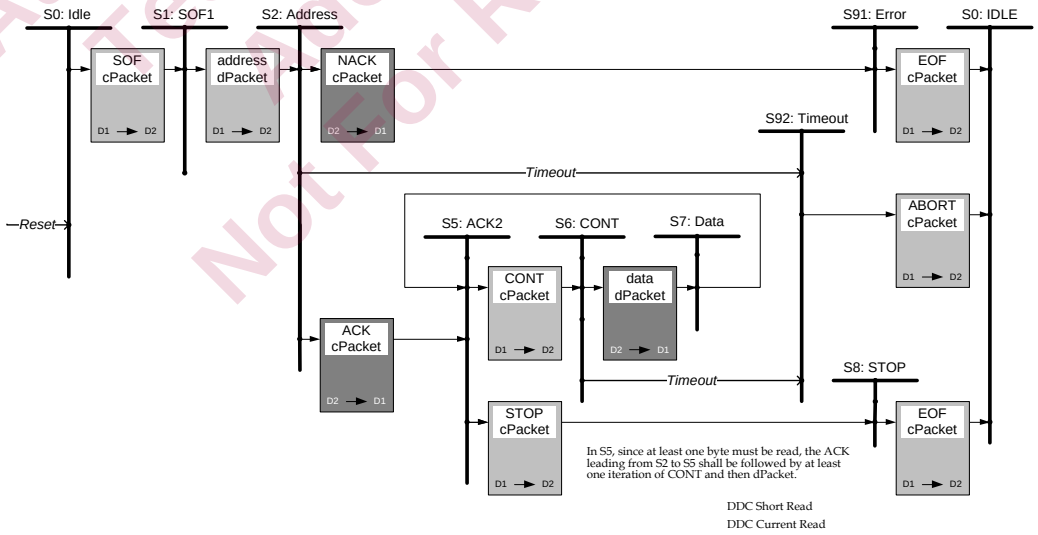


Figure 7-46. MHL DDC Short Read and Current Read State Diagram

7.5.3.4 DDC Write

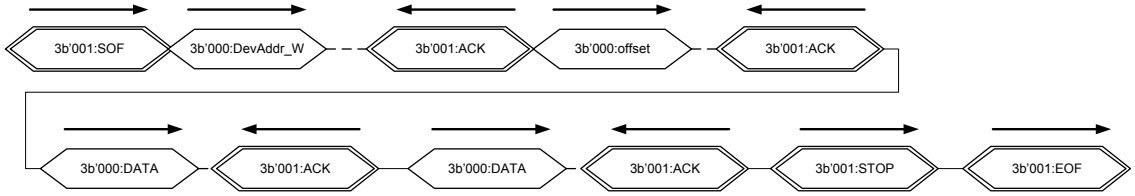


Figure 7-47. DDC Write Command

1. MHL Source performs arbitration and gets control of the bus.
2. MHL Source sends SOF control packet in order to set up a DDC frame.
3. MHL Source sends DDC device address data packet with READ bit cleared, to MHL Sink.
4. MHL Sink sends ACK control packet to MHL Source.
5. MHL Source sends DDC offset data packet to MHL Sink.
6. MHL Sink sends ACK control packet to MHL Source.
7. MHL Source sends byte data as data packet to MHL Sink.
8. MHL Sink sends ACK control packet to MHL Source.
9. Steps 7 and 8 are repeated while Source wants to write further bytes to the Sink.
10. MHL Source sends STOP control packet to MHL Sink.
11. MHL Source sends EOF control packet to MHL Sink to close a DDC frame.

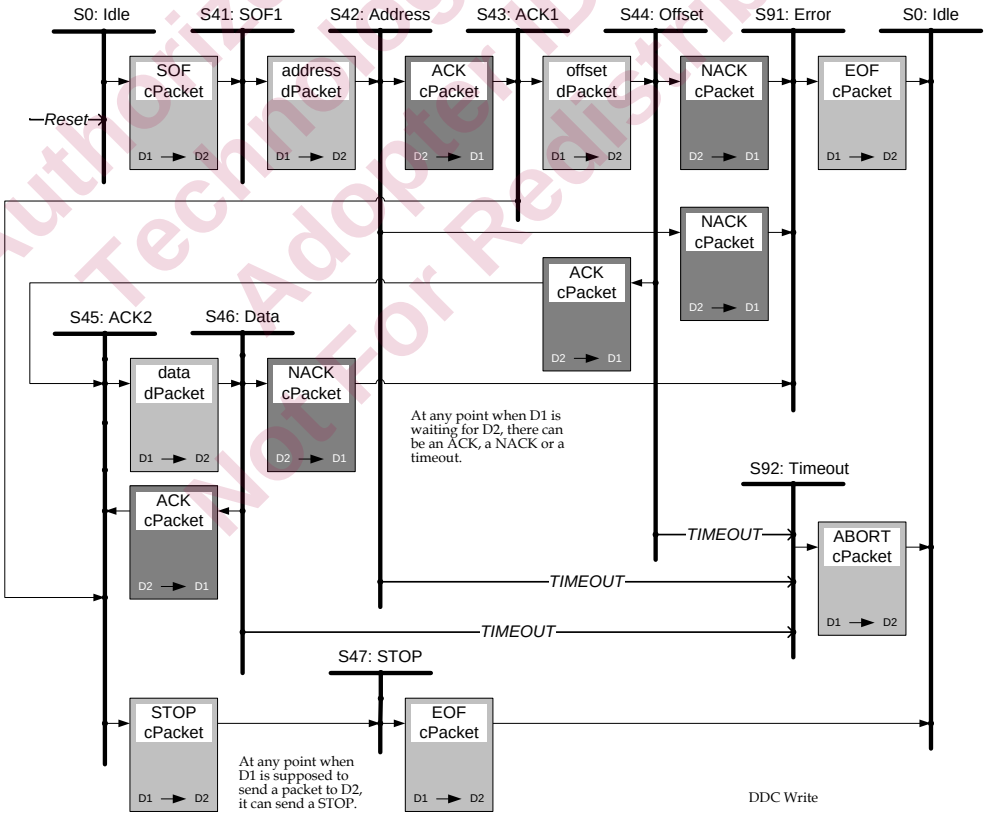


Figure 7-48. MHL DDC Write State Diagram

7.6 Request Action Protocol Sub-commands

The MSC Request Action Protocol (RAP) is a protocol that sends a request for action from a requester to a responder over the MHL Sideband Channel. RAP sub-commands are confirmed by the RAPK (acknowledge) sub-command.

Each RAP sub-command is a complete MSC command, ending with an ACK packet. As such, the RAP sub-command may be followed by another MSC command from the same device. There is no translation layer requirement that each RAP sub-command be acknowledged with RAPK before the next MSC command is issued. However, in order to clearly identify each RAP sub-command with its RAPK sub-command response, a device shall not issue another RAP sub-command until it has either received an RAPK sub-command in return from the first RAP sub-command, or timed out.

7.6.1 RAP Sub-Command

Figure 7-49 shows a single MSC RAP Sub-Command transfer.

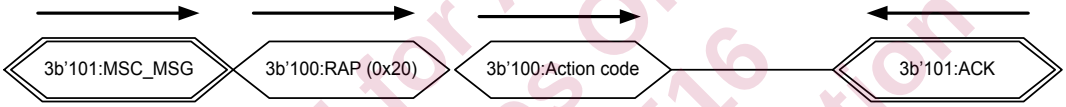


Figure 7-49. MSC RAP Sub-Command

Table 7-11. MSC Request Action Protocol Sub-Command Action Codes

Opcode	Action Codes
0x00	Poll
0x10	Change to CONTENT_ON state.
0x11	Change to CONTENT_OFF state.

The "POLL"action code is used as a 'ping' to the opposite device, to elicit the ACK packet, and the subsequent RAPK sub-command in return.

The RAP sub-command uses two action codes, "CONTENT_ON" and "CONTENT_OFF" to set the opposite device to that configuration. Each of these states is defined as follows:

A device in the "CONTENT_ON" state has CBUS connected and enables the transport of the video and/or audio content. For a Source, this means that the TMDS output is enabled (SRC6 or SRC12 state). For a Sink, this means that the content is rendered (SINK3 state). For a Dongle, this means that the content arriving at the MHL input port is output on the output port.

A device in the "CONTENT_OFF" state has CBUS connected but disables the transport of the video and/or audio content. For a Sink or Dongle, this means that the content arriving at the MHL input port is ignored.

A device shall be in the "CONTENT_ON" state when it is connected through discovery to another MHL device, until the state is changed with an RAP command.

Note that a device in the "Standby" state has no CBUS connection, and therefore cannot directly transfer an RAP sub-command.

Note that a Source shall not enable its TMDS outputs unless it has received SET_HPD, sees active RxSense and sees PATH_EN{Sink}=1.

7.6.2 RAPK Sub-Command

Figure 7-50 shows a single MSC RAPK sub-command transfer.

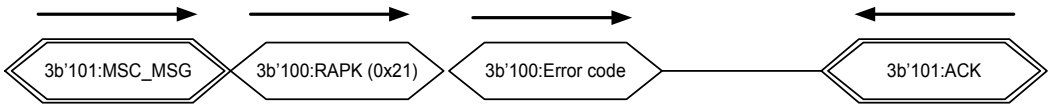


Figure 7-50. MSC RAPK Sub-Command

Table 7-12. MSC Request Action Protocol Status Codes

Opcode	Error Codes	Explanation
0x00	No Error	The previous command was recognized without error.
0x01	Unrecognized Action Code	The previous command contained an action code which is not defined in the MHL specification.
0x02	Unsupported Action Code	The previous command contained an action code which is defined in this specification, but is not supported in the responder.
0x03	Responder Busy	The responder is not able to process the action code as it is busy with other in-process transactions.

7.6.3 MSC Request Action Protocol Support Requirements

An MHL Source device or an MHL Sink device shall support generation of all sub-commands in Table 7-11. An MHL Source or Sink device shall recognize the sub-commands in Table 7-11.

An MHL Dongle is not required to support RAP sub-commands. Such a Dongle shall indicate in the Capability Register RAP_SUPPORT field whether it supports or does not support RAP sub-commands.

An MHL Dongle which chooses to support RAP sub-commands shall also support RCP sub-commands. See Section 7.7.8.1.

An MHL device shall use the RAP command to command a state in a connected MHL device, rather than a specific RCP key code.

If an MHL Source responds with "No Error" status to the "Change to CONTENT_ON state" command, then the Source shall wake up (if not already awake) and activate its MHL output to the connected Sink, as soon as RxSense is active and PATH_EN{Sink} is set to '1'.

If an MHL Sink responds with "No Error" status to the "Change to CONTENT_ON state" command, then the Sink shall wake up (if not already awake) and switch its input to the input with the MHL Source.

An MHL Source shall not send any RAP sub-command to an MHL Dongle if the MHL Dongle indicates with RAP_SUPPORT=0 that it does not support RAP. See Section 7.9.1.10. A Dongle which indicates no support for RAP, yet receives an RAP command from the Source, shall reply to the Source with a NACK packet.

7.7 Remote Control Protocol Sub-commands

The MSC Remote Control Protocol (RCP) is a protocol that transfers remote controller key codes over the MHL Sideband Channel. RCP sub-commands are confirmed by the RCPK (acknowledge) sub-command or the RCPE (error response) sub-command. This protocol, including time-out behavior, is shown in Figure 7-51. (For the syntax, see Section 7.3.2.)

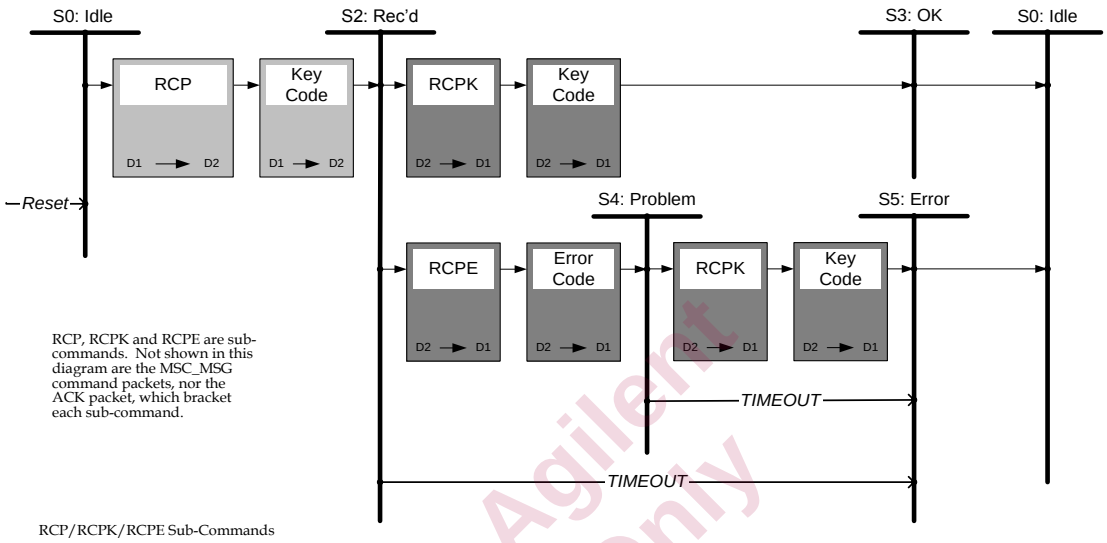


Figure 7-51. MSC_MSG RCP/RCPK/RCPE State Diagram

An RCP command and its key code, when received and recognized by the follower, shall cause the follower to send an RCPK command with the same key code back to the original initiator.

An RCP command and its key code, when received but not recognized by the follower, shall cause the follower to send an RCPE command with an error code, and then an RCPK command with the original key code. Through this means, the original initiator recognizes that the key code failed and confirms the actual value of the key code received by the follower and echoed back.

Each RCP sub-command is a complete MSC command, ending with an ACK packet. As such, the RCP sub-command may be followed by another MSC command from the same device. There is no translation layer requirement that each RCP sub-command be acknowledged with RCPK or RCPE before the next MSC command is issued. However, in order to clearly identify each RCP sub-command with its RCPK or RCPE sub-command response, a device shall not issue another RCP sub-command until it has either received an RCPK or RCPE sub-command in return from the first RCP sub-command; or timed out with T_{RCP_WAIT}.

The key codes used in RCP and RCPK sub-commands are based upon the CEA-931C specification. Refer to Table 7-14. The timeouts in Figure 7-51 are each based on exceeding T_{RCP_WAIT}. Refer to Sections 7.7.2 and 7.7.3.

7.7.1 RCP Sub-Command

An MHL device that receives an RCP sub-command shall acknowledge the sub-command by sending an RCPK sub-command, or a sequence of one RCPE and then one RCPK sub-command.

Figure 7-52 shows a single MSC RCP Sub-Command transfer.

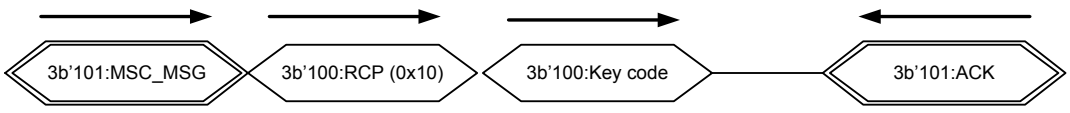


Figure 7-52. MSC RCP Sub-Command

An RCP command shall wait at least T_{RCP_WAIT} for the RCPK response from the other device. If neither an RCPK nor an RCPE sub-command is received in response to an RCP sub-command

before T_{RCP_WAIT}, then the device which issued the RCP sub-command shall assume that the RCP sub-command failed. The device is allowed to re-issue the RCP sub-command.

If an RCPE sub-command is received by the device which issued the RCP sub-command, then that device shall wait at least T_{RCP_WAIT} for the RCPK sub-command. If the device does not receive an RCPK sub-command after the RCPE sub-command before T_{RCP_WAIT} has elapsed, then the device shall assume that the RCP sub-command failed. The device is allowed to re-issue the RCP sub-command.

An MHL device shall not send an RCP sub-command in reaction to receipt of an unexpected RCPK or RCPE sub-command.

7.7.2 RCPK Sub-Command

An MHL device shall use the RCPK sub-command to indicate that the RCP sub-command received was recognized as valid and complete. This acknowledgment does not mean that the action requested by the RCP sub-command (such as Power On) has been completed.

Figure 7-53 shows a single MSC RCPK command transfer.

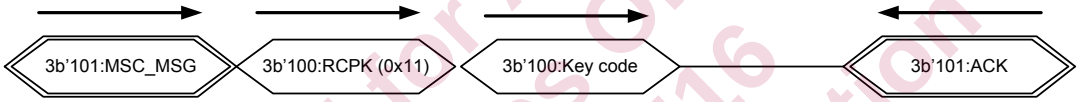


Figure 7-53. MSC RCPK Sub-Command

The acknowledging device shall send the RCPK sub-command within T_{RCP_WAIT} from sending the ACK packet at the end of the original RCP command.

7.7.3 RCPE Sub-Command

An MHL device shall use the RCPE sub-command to indicate that the RCP sub-command received was unrecognized, that the responder was busy, or any of the non-zero error codes listed in Table 7-13.

Figure 7-54 shows a single MSC RCPE command transfer.

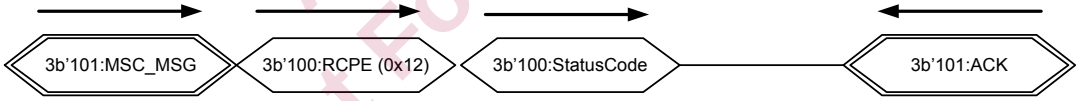


Figure 7-54. MSC RCPE Sub-command

The acknowledging device shall send the RCPE sub-command within T_{RCP_WAIT} from sending the ACK packet at the end of the original RCP command.

Table 7-13. MSC Remote Control Protocol Status Codes

Opcode	Status Codes	Explanation
0x00	No Error	Not allowed. See RCPK acknowledgment.
0x01	Ineffective Key Code	The key code in the RCP sub-command is not recognized or not currently supported by the receiving device.
0x02	Responder Busy	The receiving device cannot execute the behavior associated with the received key code, and therefore the key code has been discarded by the receiving device. The initiator may send another RCP command in response to this status code.

7.7.4 MSC Remote Control Protocol Key Codes

For a more complete definition of the behavior associated with each key ID, refer to Table 7-15. Each key code is an eight bit value, with the most-significant bit in the byte used for "press" and "release" flag. The low-order seven bits are the 'key ID'. The flag, and the timeouts associated with it, are defined in CEA-931C.

Each hex value shown in Table 7-14, Table 7-15, Table 7-17, Table 7-18 and Table 7-19 is a seven bit value. The maximum key ID value is 0x7F.

Table 7-14. MSC Remote Control Protocol Sub-command 7-Bit Key Code

Key ID	Key Name		Key ID	Key Name
0x00	Select		0x40	Reserved
0x01	Up		0x41	Volume Up
0x02	Down		0x42	Volume Down
0x03	Left		0x43	Mute
0x04	Right		0x44	Play
0x05	Right-Up		0x45	Stop
0x06	Right-Down		0x46	Pause
0x07	Left-Up		0x47	Record
0x08	Left-Down		0x48	Rewind
0x09	Root Menu		0x49	Fast Forward
0x0A	Setup Menu		0x4A	Eject
0x0B	Contents Menu		0x4B	Forward
0x0C	Favorite Menu		0x4C	Backward
0x0D	Exit		0x4D-0x4F	Reserved
0x0E-0x1F	Reserved		0x50	Angle
0x20	Numeric 0		0x51	Subpicture
0x21	Numeric 1		0x52-0x5F	Reserved
0x22	Numeric 2		0x60	Play_Function
0x23	Numeric 3		0x61	Pause_Play_Function
0x24	Numeric 4		0x62	Record_Function
0x25	Numeric 5		0x63	Pause_Record_Function
0x26	Numeric 6		0x64	Stop_Function
0x27	Numeric 7		0x65	Mute_Function
0x28	Numeric 8		0x66	Restore_Volume_Function
0x29	Numeric 9		0x67	Tune_Function
0x2A	Dot		0x68	Select_Media_Function
0x2B	Enter		0x69	Reserved
0x2C	Clear		0x6A	Reserved
0x2D-0x2F	Reserved		0x6B-0x70	Reserved
0x30	Channel Up		0x71	F1 (Blue)
0x31	Channel Down		0x72	F2 (Red)
0x32	Previous Channel		0x73	F3 (Green)

Key ID	Key Name		Key ID	Key Name
0x33	Sound Select		0x74	F4 (Yellow)
0x34	Input Select		0x75	F5
0x35	Show Information		0x76-0x7D	Reserved
0x36	Help		0x7E	Vendor_Specific
0x37	Page Up		0x7F	Reserved
0x38	Page Down			
0x39-0x3F	Reserved			

7.7.5 RCP Key Code Behavior Definitions

Key codes sent using the RCP sub-command shall be used to elicit the behaviors according to the definitions in the following sections. The intent is that a remote control should control the MHL Source through RCP commands just as if that remote control were signaling the MHL Source device directly (such as through infrared commands, or through commands entered on the Source itself). These behavioral definitions have been adapted from the behaviors described in the CEA-931C specification.

Table 7-15. Key Code Behaviors

Key ID	Key Name	Behavioral Description	Notes
0x00	Select	Selects the item focused by the cursor.	
0x01	Up	Moves cursor up.	9
0x02	Down	Moves cursor down.	9
0x03	Left	Moves cursor left.	9
0x04	Right	Moves cursor right.	9
0x05	Right-Up	Moves cursor upper-right direction.	9
0x06	Right-Down	Moves cursor lower-right direction.	9
0x07	Left-Up	Moves cursor upper-left direction.	9
0x08	Left-Down	Moves cursor lower-left direction.	9
0x09	Root Menu	Displays main menu or 'desktop'.	5
0x0A	Setup Menu	Displays configuration or setup menu.	
0x0B	Contents Menu	Displays menu listing contents.	
0x0C	Favorite Menu	Displays menu listing favorites.	
0x0D	Exit	Closes current menu and go back to previous menu. If executed while at root menu, 'exit' causes a return to the previous state.	
0x0E-0x1F	Reserved		2
0x20	Numeric 0		
0x21	Numeric 1		
0x22	Numeric 2		

Key ID	Key Name	Behavioral Description	Notes
0x23	Numeric 3		
0x24	Numeric 4		
0x25	Numeric 5		
0x26	Numeric 6		
0x27	Numeric 7		
0x28	Numeric 8		
0x29	Numeric 9		
0x2A	Dot	Used with numeric 0-9 for entering a decimal point.	
0x2B	Enter	Finish the entered numerical sequence and use that value as a response.	4
0x2C	Clear	Cancel the numerical value. Return to the state before the first numerical key.	
0x2D-0x2F	Reserved		2
0x30	Channel Up	Select the channel one higher numerically from the current channel.	9
0x31	Channel Down	Select the channel one lower numerically from the current channel.	9
0x32	Previous Channel	Select the channel used before most recent new channel selection.	
0x33	Sound Select	Alternate among audio selections, such as multiple language tracks in the media stream. If playing media with one or no audio tracks, this key has no effect.	
0x34	Input Select	Switch to the next input port on the device.	6
0x35	Show Information	Display information about the current media content (such as title), or the channel number (if from a tuner).	
0x36	Help	Display help instructions.	
0x37	Page Up	Scroll up the display, or move ahead in the selection menu by more than an increment of one.	9
0x38	Page Down	Scroll down the display, or move back in the selection menu by more than a decrement of one.	9
0x39-0x3F	Reserved		2
0x40	Reserved		
0x41	Volume Up	Increase volume.	9
0x42	Volume Down	Decrease volume.	9

Key ID	Key Name	Behavioral Description	Notes
0x43	Mute	Alternate between mute (zero volume) and previously-set volume level, as a 'toggle' control.	
0x44	Play	Start or resume playback of content at normal speed.	
0x45	Stop	Stop playback of content.	
0x46	Pause	Pause playback of content, or – if already paused – resume playback as if 'play' had been pressed.	
0x47	Record	Begin recording of content.	
0x48	Rewind	Move toward beginning of recorded content, if a recording device. Otherwise, move backward in the content at a speed faster than the forward movement with the 'play' mode.	8, 9
0x49	Fast Forward	Move forward in the content at a speed faster than the 'play' mode.	9
0x4A	Eject	Ejects the media from the device, or may close the door for loading the media.	10
0x4B	Forward	Move to the next media track, chapter selection, or similar division in the media to be played.	9
0x4C	Backward	Move to the previous track, chapter selection, or similar division in the media to be played.	9
0x4D-0x4F	Reserved		2
0x50	Angle		1
0x51	Subpicture		1
0x52-0x5F	Reserved		2
0x60	Play_Function	Starts or resumes playing back the content at normal speed. Repeated codes do not further affect speed.	
0x61	Pause_Play_Function	Pause playback. Repeated codes do not further alter the paused state.	
0x62	Record_Function	Start recording from the currently selected input.	
0x63	Pause_Record_Function	Pause recording.	
0x64	Stop_Function	Stop either recording or playback, depending on the previous state of the device.	

Key ID	Key Name	Behavioral Description	Notes
0x65	Mute_Function	Mute audio output. Repeated codes do not unmute the audio.	
0x66	Restore_Volume_Function	Restore the volume to the level before the mute.	
0x67	Tune_Function		1
0x68	Select_Media_Function	Prepares to select the disk, in a media device with multiple disks. The index of the disk to be used is entered after this key code using the numerical keys.	
0x69-0x70	Reserved		2
0x71	F1	Corresponds to use of Blue button.	11
0x72	F2	Corresponds to use of Red button.	11
0x73	F3	Corresponds to use of Green button.	11
0x74	F4	Corresponds to use of Yellow button.	11
0x75	F5		1
0x76-0x7D	Reserved		2
0x7E	Vendor_Specific		3
0x7F	Reserved		2

Notes on Table 7-15:

1. These codes are defined in the CEA-931C, but have no equivalent function in MHL. A device sending this code should expect to receive back an RCPE sub-command with the error code set to "ineffective key code".
2. These codes are shown as 'reserved' in CEA-931C, and shall not be used in MHL. A device sending this code should expect to receive back an RCPE sub-command with the error code set to "ineffective key code".
3. This code is reserved for vendor-specific use. Mechanisms for coordinating vendor-specific use among multiple vendors is not defined in this version of the specification.
4. 'Enter' shall complete a numerical entry sequence. Example: pressing '4', then '1', then 'enter' causes the numerical value forty-one (41) to be used. A numerical sequence may also be ended by a timeout after the last digit.
5. 'Root Menu' shall move to the highest level selection menu in a device with multi-level menus. In a device with one level of menus, or when displaying the highest level of menu, 'root menu' shall have no effect. Use 'exit' to return out of the highest menu.
6. 'Input Select' shall have no effect on a device which has only one MHL input.
7. 'Mute' toggles between no volume output and the level previously set. The 'Mute_Function' code (0x65) forces mute. 'Restore_Volume_Function' code (0x66) forces restoration of volume.
8. 'Rewind' serves the "fast backward" function, on a non-recording device.
9. Refer to Section 7.7.6.
10. 'Eject' toggles between ejecting the media from the device, and closing the door to load the media into the device.

11. Receiving device should inform the user of the behavioral meaning of this key ID, especially if the meaning changes between applications on the device, or between the application and the operating system.

The power shall be commanded using the RAP command. See Section 7.6.1 to 7.6.3.

7.7.6 RCP Press and Hold

A remote control key, when held in the pressed position by the user, may cause a series of RCP key codes – all identical – to be sent from one MHL device to another. The key IDs marked in Table 7-15 allow press and hold behavior. The most-significant bit in the 8-bit key code field is set to '0' when the button is pressed, and cleared to '1' when the button is released. Refer to CEA-931C for more details on how press and release states are communicated as part of the key code, from initiator to follower.

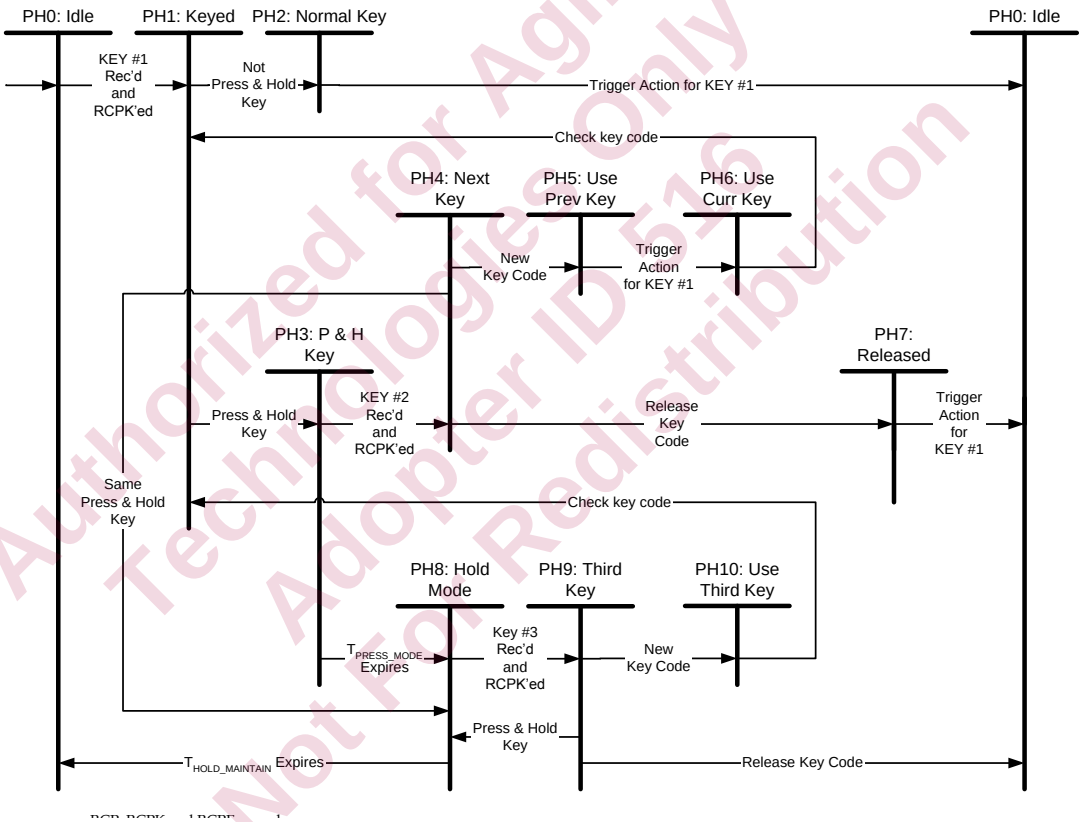


Figure 7-55. MSC_MSG RCP Press-and-Hold Behavior State Diagram

Table 7-16. MSC_MSG RCP Press-and-Hold States and Transitions

	State or Transition Description	Notes
PH0	Waiting for remote control protocol keypress.	
PH0-PH1	First key (KEY #1) arrives in RCP, acknowledged in RCPK.	
PH1	KEY #1 is pending.	
PH1-PH2	KEY #1 is not "press and hold".	
PH1-PH3	KEY #1 is "press and hold" (with 'press' bit asserted).	
PH2	KEY #1 is pending, waiting execution as a 'normal' key.	
PH2-PH0	KEY #1 action is triggered.	1

	State or Transition Description	Notes
PH3	Reset "press and hold timer" to wait for confirmation or expiration.	
PH3-PH4	Second key (KEY #2) arrives in RCP, acknowledged in RCPK.	
PH3-PH8	No second key code is received before T_{PRESS_MODE} elapses.	2
PH4	KEY #2 is pending.	
PH4-PH5	KEY #2 is not the release code nor press code for KEY #1.	
PH4-PH7	KEY #2 is the release code for KEY #1.	
PH4-PH8	KEY #2 is the same as KEY #1 – repeated 'press' code.	
PH5	KEY #2 is new key code, not release of KEY #1. Both KEY #1 and KEY #2 are pending.	
PH5-PH6	KEY #1 action is triggered, since it was first action key recognized.	
PH6	KEY #2 is still pending. Since KEY #1 has been used, KEY #2 comes to the top of the queue as the new KEY #1.	
PH6-PH1	New KEY #1 to be checked for 'normal' or new "press and hold".	1
PH7	"Press and hold state" is cleared. KEY #1 is pending.	
PH7-PH0	KEY #1 action is triggered.	1
PH8	Begin or continue "press and hold" mode, based on KEY #1 key code and the elapsing of at least T_{PRESS_MODE} without a second key press.	3
PH8-PH0	Timer $T_{HOLD\_MAINTAIN}$ elapses without new key press, causing the "press and hold" mode to exit.	4
PH8-PH9	Third key (KEY #3) arrives in RCP, acknowledged in RCPK.	5
PH9	KEY #3 is pending.	
PH9-PH0	KEY #3 is the release code for KEY #1, causing exit from "press and hold" mode.	6
PH9-PH8	KEY #3 is the press code for KEY#1, continuing the "press and hold" mode. The timer for $T_{HOLD\_MAINTAIN}$ is reset and begins again.	7
PH9-PH10	KEY #3 is a new key code, causing exit from "press and hold" mode.	8
PH10	KEY #3 is pending.	9
PH10-PH1	New KEY #1 to be checked for 'normal' or new "press and hold".	

Notes on Table 7-18:

1. For a key press which corresponds to an action (such as 'Up'), the action is triggered in this transition. The action is performed in the MHL device independent of the RCP state machine timing.
2. If, after acknowledging a key ID which is on the "press and hold list" (see Table 7-15), the MHL device does not see another RCP command for at least T_{PRESS_MODE}, then the device shall enter the press and hold mode. (Modeled on CEA-931C timings.)
3. Handling of action related to KEY #1 while in "press and hold" mode is the responsibility of the MHL device, and is outside the scope of this specification. The device may execute the action corresponding to KEY #1 zero or more times during the count down of $T_{HOLD\_MAINTAIN}$. Not all "press and hold" requests will cause repeated behavior with the same periodicity. For example, "Volume Up" may increment the volume in the device each 500 msec while in "press and hold" mode, but "Channel Up" may increment the channel selection in the device each 1000 msec.

4. If, after entering "hold mode" (PH8), the MHL device does not receive an RCP command with the same key code which initiated the hold mode (KEY #1 in Figure 7-55) before $T_{HOLD\_MAINTAIN}$ elapses, then the device shall return to the idle state.
5. Note that KEY #3 is actually the second key in the sequence, after KEY #1. KEY #3 arrives after T_{PRESS_MODE} timeout, whereas KEY #2 arrives before T_{PRESS_MODE} timeout.
6. Receiving the release code for KEY#1 causes the device to exit "press and hold" mode.
7. Receiving another key code which matches the code in KEY #1, before $T_{HOLD\_MAINTAIN}$ elapses, shall cause the MHL device to remain in the hold mode.
8. Receiving another key code while in hold mode, which does not match the code in KEY #1, cause the MHL device to exit hold mode, and check the value of KEY #3.
9. Since KEY #1 has been used, KEY #3 comes to the top of the queue as the new KEY #1.

Note that Table 7-18 does not show the detailed levels of MSC_MSG control packet, ACK control packet, timeouts, RCPE and RCPK handling, etc. Refer to other sections for those details.

7.7.7 Media Selection Mode

If an LD_MEDIA device supports holding multiple physical media internally, then the LD_MEDIA device shall support the following commands. The Select_Media Function (0x68) command without operands enables entry into the Media Selection Mode. After entry into this mode, one or more numeric key ID '0' through '9' followed by the 'Enter' key ID selects the new media by numeric index. A 'Forward' or 'Backward' key ID directs the next media or previous media, by index, to be selected. An 'Enter', 'Forward' or 'Backward' key ID exits from the Media Selection Mode. An explicit 'Select_Media Function' key ID, after any numeric key ID or IDs, causes escape from the Media Selection Mode without changing the selected media. The LD_MEDIA device shall exit from Media Selection Mode if no further media selection command is received within 5 seconds. Refer to Figure 7-56.

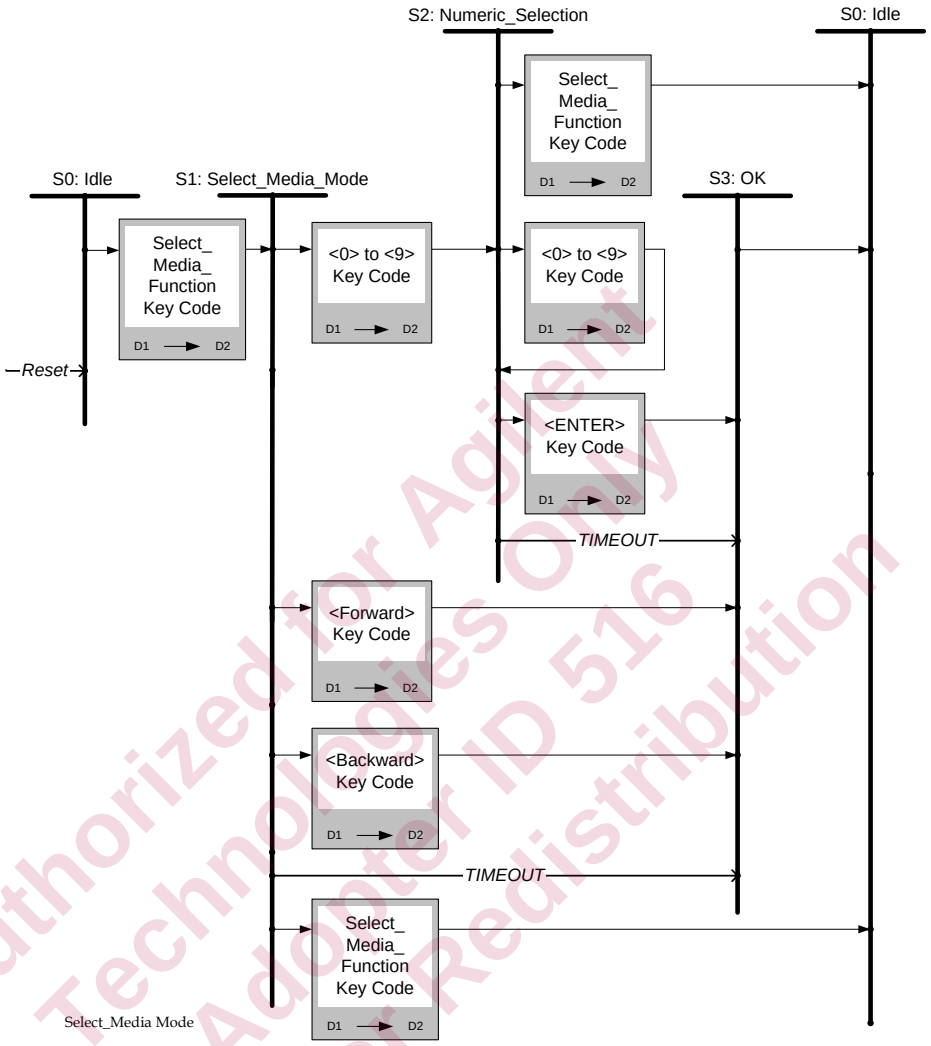


Figure 7-56. Entering and Exiting Media Selection Mode

7.7.8 MSC Remote Control Protocol Support Requirements

MHL devices are required to receive RCP commands and perform the behavior associated with each command, according to the details in Section 7.7.8.1. MHL Sink devices are additionally required to support sending of RCP commands according to the list in Section 7.7.8.2.

7.7.8.1 Support when Receiving RCP Command

An MHL Source device or Sink device shall support receipt all RCP sub-commands listed in Table 7-17 which are marked with an 'X', according to the logical device role or roles the device claims. See also the LOG_DEV_MAP register in Section 7.9.1.8.

MHL Dongle devices are not required to support RCP sub-commands, and shall respond to an incoming RCP sub-command with a translation layer NACK packet, if the Dongle indicates no support. See Section 7.4.3.11. Any MHL Dongle which claims support for RCP sub-commands shall, however, fulfill all the RCP requirements of an MHL Sink.

An MHL Dongle which chooses to support RCP sub-commands shall also support RAP sub-commands. See Section 7.6.3.

'Support' in this context means receiving a key code; acknowledging it with an RCPK sub-command (unless the device is busy); and performing the behavior listed for the 7-bit key ID in that key code Table 7-15.

An MHL Source shall not send any RCP sub-command to an MHL Dongle if the MHL Dongle indicates with RCP_SUPPORT=0 that it does not support RCP. See Section 7.9.1.10.

A device is allowed to support a key ID which is not marked as required in Table 7-17.

Table 7-17. Key Code Receiving Support Map

		Logical Device Type								
Key ID	**Key Name**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Notes**
		LD_DISPLAY	**LD_VIDEO**	**LD_AUDIO**	**LD_MEDIA**	**LD_TUNER**	**LD_RECORD**	**LD_SPEAKER**	**LD_GUI**	
0x00	Select								X	
0x01	Up								X	
0x02	Down								X	
0x03	Left								X	
0x04	Right								X	
0x05	Right-Up									
0x06	Right-Down									
0x07	Left-Up									
0x08	Left-Down									
0x09	Root Menu								X	
0x0A	Setup Menu									
0x0B	Contents Menu									
0x0C	Favorite Menu									
0x0D	Exit								X	
0x0E-0x1F	Reserved									
0x20	Numeric 0		X	X	X	X				1
0x21	Numeric 1		X	X	X	X				1
0x22	Numeric 2		X	X	X	X				1
0x23	Numeric 3		X	X	X	X				1
0x24	Numeric 4		X	X	X	X				1
0x25	Numeric 5		X	X	X	X				1
0x26	Numeric 6		X	X	X	X				1
0x27	Numeric 7		X	X	X	X				1
0x28	Numeric 8		X	X	X	X				1

		Logical Device Type								
Key ID	**Key Name**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Notes**
		LD_DISPLAY	**LD_VIDEO**	**LD_AUDIO**	**LD_MEDIA**	**LD_TUNER**	**LD_RECORD**	**LD_SPEAKER**	**LD_GUI**	
0x29	Numeric 9		X	X	X	X				1
0x2A	Dot									
0x2B	Enter		X	X	X	X				1
0x2C	Clear		X	X	X	X				1
0x2D-0x2F	Reserved									
0x30	Channel Up					X				
0x31	Channel Down					X				
0x32	Previous Channel					X				
0x33	Sound Select			X						
0x34	Input Select									
0x35	Show Information									
0x36	Help									
0x37	Page Up									
0x38	Page Down									
0x39-0x3F	Reserved									
0x40										
0x41	Volume Up							X		
0x42	Volume Down							X		
0x43	Mute							X		
0x44	Play		X	X						
0x45	Stop		X	X			X			
0x46	Pause		X	X			X			
0x47	Record						X			
0x48	Rewind		X	X						
0x49	Fast Forward		X	X						
0x4A	Eject				X					
0x4B	Forward		X	X	X					1
0x4C	Backward		X	X	X					1
0x4D-0x4F	Reserved									
0x50	Angle									
0x51	Subpicture									

Key ID	Key Name	Logical Device Type								Notes
		0	1	2	3	4	5	6	7	
		LD_DISPLAY	LD_VIDEO	LD_AUDIO	LD_MEDIA	LD_TUNER	LD_RECORD	LD_SPEAKER	LD_GUI	
0x52-0x5F	Reserved									
0x60	Play Function		X	X						
0x61	Pause_Play Function		X	X						
0x62	Record Function						X			
0x63	Pause_Record Function						X			
0x64	Stop Function		X	X			X			
0x65	Mute Function							X		
0x66	Restore_Volume Function							X		
0x67	Tune Function									
0x68	Select_Media Function									1
0x69	Reserved									
0x6A	Reserved									
0x6B-0x70	Reserved									
0x71-0x75	F1-F5									
0x76-0x7D	Reserved									
0x7E	Vendor_Specific									
0x7F	Reserved									

Notes on Table 7-17:

1. Refer to Section 7.7.7.

7.7.8.2 Sink Support Sending RCP Command

An MHL Sink device shall support sending all RCP sub-commands listed in Table 7-18 which are marked with an 'X'. This 'superset' of the commands marked in Table 7-17 allows a Sink to send control commands to any type of MHL Source.

'Support' in this context means sending the appropriate key ID when requesting the behavior listed for that key ID in Table 7-15, triggered by the MHL Sink's remote commander or a command pass-through from some other A/V control protocol initiated by another device.

A Sink device is not required to be able to send the RCP key ID associated with a behavior if it has no means. For example, a Sink device which has no 'Record' button, nor menu selection for

'Record', nor any other means for selecting 'Record' behavior, is therefore not required to be able to send the 'Record' key ID to the Source.

A device is allowed to support a key ID which is not marked as required in Table 7-18.

Table 7-18. Key ID Sending Support Map for Sink

Key ID	Key Name	Sink		Key ID	Key Name	Sink
0x00	Select	X		0x37	Page Up	
0x01	Up	X		0x38	Page Down	
0x02	Down	X		0x39-0x3F	Reserved	
0x03	Left	X		0x40	Reserved	
0x04	Right	X		0x41	Volume Up	
0x05	Right-Up			0x42	Volume Down	
0x06	Right-Down			0x43	Mute	
0x07	Left-Up			0x44	Play	X
0x08	Left-Down			0x45	Stop	X
0x09	Root Menu	X		0x46	Pause	X
0x0A	Setup Menu			0x47	Record	X
0x0B	Contents Menu			0x48	Rewind	X
0x0C	Favorite Menu			0x49	Fast Forward	X
0x0D	Exit	X		0x4A	Eject	X
0x0E-0x1F	Reserved			0x4B	Forward	X
0x20	Numeric 0	X		0x4C	Backward	X
0x21	Numeric 1	X		0x4D-0x4F	Reserved	
0x22	Numeric 2	X		0x50	Angle	
0x23	Numeric 3	X		0x51	Subpicture	
0x24	Numeric 4	X		0x52-0x5F	Reserved	
0x25	Numeric 5	X		0x60	Play Function	X
0x26	Numeric 6	X		0x61	Pause_Play Func.	X
0x27	Numeric 7	X		0x62	Record Function	X
0x28	Numeric 8	X		0x63	Pause_Record Func.	X
0x29	Numeric 9	X		0x64	Stop Function	X
0x2A	Dot			0x65	Mute Function	
0x2B	Enter	X		0x66	Restore_Volume Func.	
0x2C	Clear	X		0x67	Tune Function	
0x2D-0x2F	Reserved			0x68	Select_Media Func.	
0x30	Channel Up	X		0x69	Reserved	
0x31	Channel Down	X		0x6A	Reserved	
0x32	Previous Channel	X		0x6B-0x70	Reserved	
0x33	Sound Select	X		0x71	F1 / Blue	
0x34	Input Select			0x72	F2 / Red	

Key ID	Key Name	Sink		Key ID	Key Name	Sink
0x35	Show Info.			0x73	F3 / Green	
0x36	Help			0x74	F4 / Yellow	
				0x75	F5	
				0x76-0x7D	Reserved	
				0x7E	Vendor_Specific	
				0x7F	Reserved	

7.7.8.3 Source Support Sending RCP Command

An MHL Source device should support sending all RCP sub-commands listed in Table 7-19 which are marked with an 'X', for all key IDs which have a corresponding remote command button or local button on the Source. This 'superset' of the commands marked in Table 7-17 allows a Source to send control commands to any type of MHL Sink, or to an MHL Dongle which indicates support of RCP with RCP_SUPPORT.

'Support' in this context means sending the appropriate key ID when requesting the behavior listed for that key ID in Table 7-15, triggered by the MHL Source's remote commander, or local keys or buttons on the Source.

A device is allowed to support a key ID which is not marked as required in Table 7-19.

Table 7-19. Key ID Sending Support Map for Source

Key ID	Key Name	Source		Key ID	Key Name	Source
0x00	Select	X		0x37	Page Up	
0x01	Up	X		0x38	Page Down	
0x02	Down	X		0x39-0x3F	Reserved	
0x03	Left	X		0x40	Reserved	
0x04	Right	X		0x41	Volume Up	X
0x05	Right-Up			0x42	Volume Down	X
0x06	Right-Down			0x43	Mute	X
0x07	Left-Up			0x44	Play	X
0x08	Left-Down			0x45	Stop	X
0x09	Root Menu	X		0x46	Pause	X
0x0A	Setup Menu			0x47	Record	X
0x0B	Contents Menu			0x48	Rewind	X
0x0C	Favorite Menu			0x49	Fast Forward	X
0x0D	Exit	X		0x4A	Eject	X
0x0E-0x1F	Reserved			0x4B	Forward	X
0x20	Numeric 0	X		0x4C	Backward	X
0x21	Numeric 1	X		0x4D-0x4F	Reserved	
0x22	Numeric 2	X		0x50	Angle	
0x23	Numeric 3	X		0x51	Subpicture	
0x24	Numeric 4	X		0x52-0x5F	Reserved	

Key ID	Key Name	Source
0x25	Numeric 5	X
0x26	Numeric 6	X
0x27	Numeric 7	X
0x28	Numeric 8	X
0x29	Numeric 9	X
0x2A	Dot	
0x2B	Enter	X
0x2C	Clear	X
0x2D-0x2F	Reserved	
0x30	Channel Up	X
0x31	Channel Down	X
0x32	Previous Channel	X
0x33	Sound Select	X
0x34	Input Select	
0x35	Show Info.	
0x36	Help	

Key ID	Key Name	Source
0x60	Play Function	X
0x61	Pause_Play Func.	X
0x62	Record Function	X
0x63	Pause_Record Func.	X
0x64	Stop Function	X
0x65	Mute Function	X
0x66	Restore_Volume Func.	X
0x67	Tune Function	
0x68	Select_Media Func.	
0x69	Reserved	
0x6A	Reserved	
0x6B-0x70	Reserved	
0x71	F1 / Blue	
0x72	F2 / Red	
0x73	F3 / Green	
0x74	F4 / Yellow	
0x75	F5	
0x76-0x7D	Reserved	
0x7E	Vendor_Specific	
0x7F	Reserved	

7.8 UTF-8 Character Protocol Sub-commands

The MSC UTF-8 Character Protocol (UCP) is a protocol that sends one 8-bit character from a sender to a receiver over the MHL Sideband Channel. UCP sub-commands are confirmed by the UCPK (acknowledge) sub-command, or the UCPE (error) sub-command.

Each UCP sub-command is a complete MSC command, ending with an ACK packet. As such, the UCP sub-command may be followed by another MSC command from the same device. There is no translation layer requirement that each UCP sub-command be acknowledged with UCPK before the next MSC command is issued. However, in order to clearly identify each UCP sub-command with its UCPK sub-command response, a device shall not issue another UCP sub-command until it has either (a) received a UCPK or UCPE sub-command in return from the first UCP sub-command, or (b) timed out.

7.8.1 UCP Sub-Command

An MHL device that receives a UCP sub-command shall acknowledge the sub-command by sending a UCPK sub-command, or a sequence of one UCPE and then one UCPK sub-command.

Figure 7-57 shows a single MSC UCP Sub-Command transfer.

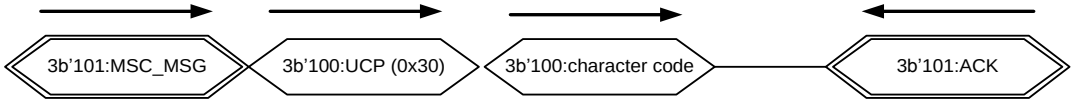


Figure 7-57. MSC UCP Sub-Command

A UCP command shall wait at least T_{RCP_WAIT} for the UCPK response from the other device. If neither a UCPK nor a UCPE sub-command is received in response to a UCP sub-command before T_{RCP_WAIT}, then the device which issued the UCP sub-command shall assume that the UCP sub-command failed. The device is allowed to re-issue the UCP sub-command.

If a UCPE sub-command is received by the device which issued the UCP sub-command, then that device shall wait at least T_{RCP_WAIT} for the UCPK sub-command. If the device does not receive a UCPK sub-command after the UCPE sub-command before T_{RCP_WAIT} has elapsed, then the device shall assume that the UCP sub-command failed. The device is allowed to re-issue the UCP sub-command.

An MHL device shall not send a UCP sub-command in reaction to receipt of an unexpected UCPK or UCPE sub-command.

The character code in the UCP sub-command shall be a code in the range 0 to 255 decimal. Not all codes are allowed. Code usage shall follow the UTF-8 protocol in the Unicode standard. (See Section 1.5.)

7.8.2 UCPK Sub-Command

An MHL device shall use the UCPK sub-command to indicate that the UCP sub-command received was recognized as valid and complete.

Figure 7-58 shows a single MSC UCPK command transfer.

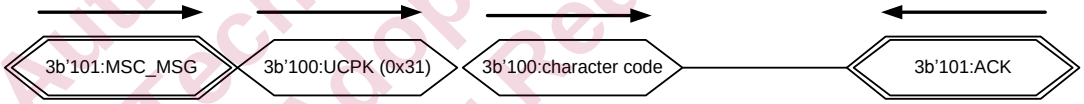


Figure 7-58. MSC UCPK Sub-Command

The acknowledging device shall send the UCPK sub-command within T_{RCP_WAIT} from sending the ACK packet at the end of the original UCP command.

The UCPK sub-command echoes back the 8-bit code it received in the associated incoming UCP sub-command.

7.8.3 UCPE Sub-Command

An MHL device shall use the UCPE sub-command to indicate that the UCP sub-command received was unrecognized, that the responder was busy, or any of the non-zero error codes listed in Table 7-20.

Figure 7-59 shows a single MSC UCPE command transfer.

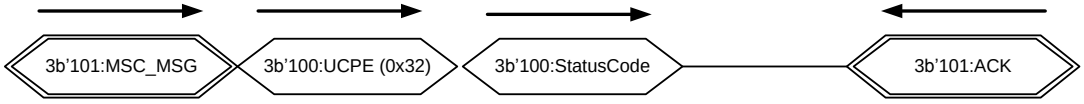


Figure 7-59. MSC UCPE Sub-command

The acknowledging device shall send the UCPE sub-command within T_{RCP_WAIT} from sending the ACK packet at the end of the original UCP command.

Table 7-20. MSC UTF-8 Character Protocol Status Codes

Opcode	Status Codes	Explanation
0x00	No Error	Not allowed. See UCPK acknowledgment.
0x01	Ineffective Key Code	The character code in the UCP sub-command is not recognized or not currently supported by the receiving device.

7.8.4 MSC UTF-8 Character Protocol Support Requirements

7.8.4.1 MSC UCP Source Device Support Requirements

An MHL 2 Source which accepts character input from a user through key board or other input device shall indicate UCP_SEND_SUPPORT=1, and be able to send UCP commands to a connected MHL 2 Sink or MHL 2 Dongle.

An MHL 2 Source which utilizes character input from a user through a key board or other input device shall indicate UCP_RECV_SUPPORT=1, and be able to receive UCP commands from a connected MHL 2 Sink and perform discernible action.

Source shall not send UCP command to Sink unless Sink indicates UCP_RECV_SUPPORT=1.

7.8.4.2 MSC UCP Sink Device Support Requirements

An MHL 2 Sink which provides for user input of characters through remote control shall indicate UCP_SEND_SUPPORT=1, and be able to send UCP commands to a connected MHL 2 Source.

An MHL 2 Sink which utilizes character input from a user through remote control shall indicate UCP_RECV_SUPPORT=1, and be able to receive UCP commands from a connected MHL 2 Source.

Sink shall not send UCP command to Source unless Source indicates UCP_RECV_SUPPORT=1.

7.8.4.3 MSC UCP Dongle Device Support Requirements

An MHL 2 Dongle may set UCP_SEND_SUPPORT=0 and may set UCP_RECV_SUPPORT=0.

7.8.4.4 Requirement for Complete UCP Sub-Command Set Support

An MHL Source or MHL Sink or MHL Dongle which supports sending the UCP sub-command shall also support recognizing the receipt of a properly-formatted UCPK or UCPE sub-command.

7.8.5 MSC UTF-8 Character Sequence Example

The example shown in Figure 7-60 demonstrates the sequence of MSC_MSG commands involved in sending and responding to a character sequence from one MHL device to another. The example includes a terminating null character (0x00). Each character is acknowledged by the receiving device.

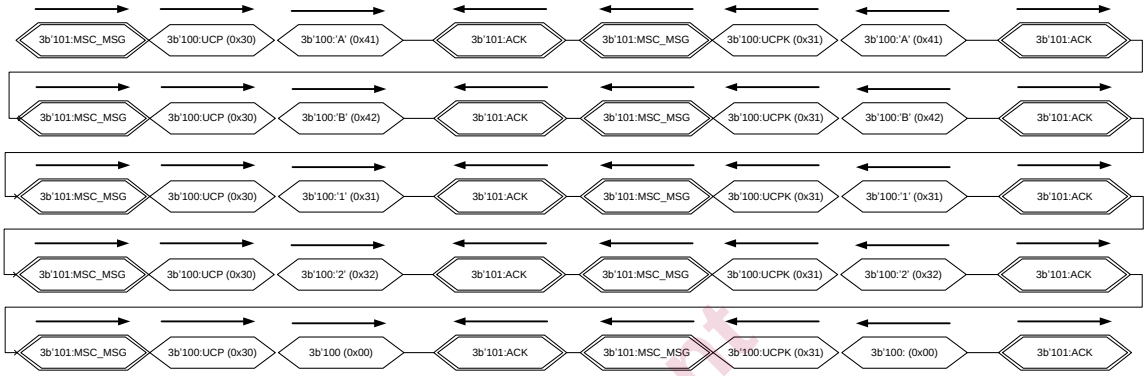


Figure 7-60. MSC_MSG Character Code Sequence "AB12" Example

7.9 Device Register Space

MSC provides register access through the MHL link, and offers four types of registers:

1. Capability Registers – readable across the link, inform each device about the opposite device's features.
2. Interrupt Registers – writeable across the link, inform a device about an event in the sending device, such as a change in link mode.
3. Status Register – writable across the link, inform a device about the readiness of the sending device's register data.
4. Scratchpad Registers – writeable across the link, allows a device to send a message or a series of data bytes to the opposite device.

MHL allows each device on an interconnect to communicate to its opposite a list of its own capabilities. This information is exchanged during the initialization of a connection.

Changes in capabilities or status may be indicated by using the interrupt registers associated with capability discovery. Such interrupt registers are described in Section 7.9.2.

Table 7-21. Device Register Space

Resource Name	MSC Offset Range	Max. Size (bytes)	Req'd Size (bytes)	Size in CAP REG	Usage MHL CBUS	
					Access	Command
Capability Registers	0x00-0x0F	16	16	N/A	Read	READ_DEVCAP
Interrupt Registers	0x20-0x2F	16	4	0x0E[7:4]	Set Bits	SET_INT
Status Registers	0x30-0x3F	16	4	0x0E[3:0]	Write	WRITE_STAT
Scratchpad Registers	0x40-0x7F	64	16	0x0D	Write	WRITE_BURST

Address offsets not listed in Table 7-21 are reserved.

Note that the registers defined throughout Section 7.9 may be implemented in hardware or allocated in software. Commands may be interpreted by logic in hardware, or parsed and interpreted by software.

7.9.1 Device Capability Registers

A Source device and a Sink device shall support exchange of information about their capabilities using the Device Capability Registers read protocol. The required set of Device Capability Registers is given in Table 7-22.

Not all fields in all Device Capability Registers are required to be maintained with accurate values for all device categories. Table 7-23 shows the fields required for Sources, Sinks and Dongles.

Each device shall provide data from Device Capability Registers with offsets from 0x00 to 0x0F, a total of 16 bytes.

These registers are accessed with offsets in the range 0x00 to 0x0F – a total space of 16 bytes. Future versions of this specification may allow for more Device Capability Registers.

Table 7-22. MHL Device Capability Registers

Address	Dev Cap Register Name		Definition
0x00	DEV_STATE	Device State	Identify current connected and powered state.
0x01	MHL_VERSION	MHL Version	Identify level of MHL Spec supported.
0x02	DEV_CAT	Device Category	Identify the type of MHL system.
0x03	ADOPTER_ID_H	Adopter ID (High Byte)	High-order byte of Adopter identifier, assigned by MHL, LLC.
0x04	ADOPTER_ID_L	Adopter ID (Low Byte)	Low-order byte of Adopter identifier, assigned by MHL, LLC.
0x05	VID_LINK_MODE	Video Link Mode Support	List of link modes supported for video.
0x06	AUD_LINK_MODE	Audio Link Mode Support	List of link modes supported for audio.
0x07	VIDEO_TYPE	Video Type Support	List of video types supported.
0x08	LOG_DEV_MAP	Logical Device Map	Map of logical device types.
0x09	BANDWIDTH	Link Bandwidth Limit	Upper bound of MHL link bandwidth.
0x0A	FEATURE_FLAG	Feature Flags	Set flag for each MHL optional feature.
0x0B	DEVICE_ID_H	Device ID (High Byte)	High-order byte of system (Source, Sink or Dongle device) identifier, assigned by Adopter.
0x0C	DEVICE_ID_L	Device ID (Low Byte)	Low-order byte of system (Source, Sink or Dongle device) identifier, assigned by Adopter.
0x0D	SCRATCHPAD_SIZE	Scratchpad Size	Total count of Scratchpad Registers.
0x0E	INT_STAT_SIZE	Interrupt/Status Size	Total count of interrupt and status registers.
0x0F		Reserved for Future Use	

Device Capability registers shall be readable across MHL using the READ_DEVCAP command (see Section 7.4.3.7). Device Capability registers shall not be modifiable through the MHL CBUS connection between two devices. The registers shall be "read only" across the MHL link.

An 'X' in Table 7-23 indicates a field which shall be maintained to an accurate value in the device.

Table 7-23. MHL Capability Register Required Fields by Device Type

Address	Dev Cap Register Name	Field	Source	Sink	Dongle
0x00	DEV_STATE				
0x01	MHL_VERSION		X	X	X
0x02	DEV_CAT		X	X	X
0x03	ADOPTER_ID_H		X	X	X
0x04	ADOPTER_ID_L		X	X	X
0x05	VID_LINK_MODE	SUPP_RGB444	X	X	X
		SUPP_YCBCR444	X	X	X
		SUPP_YCBCR422	X	X	X
		SUPP_PPIXEL	X	X	X
		SUPP_ISLANDS	X	X	X
		SUPP_VGA	X	X	X
0x06	AUD_LINK_MODE	AUD_2CH	X	X	
		AUD_8CH	X	X	
0x07	VIDEO_TYPE			X	X
0x08	LOG_DEV_MAP		X	X	X
0x09	BANDWIDTH			X	X
0x0A	FEATURE_FLAG	RCP_SUPPORT	X	X	X
		RAP_SUPPORT	X	X	X
		SP_SUPPORT	X	X	X
		UCP_SEND_SUPPORT	X	X	X
		UCP_RECV_SUPPORT	X	X	X
0X0B	DEVICE_ID_H		X	X	X
0x0C	DEVICE_ID_L		X	X	X
0x0D	SCRATCHPAD_SIZE		X	X	X
0x0E	INT_STAT_SIZE		X	X	X

7.9.1.1 Device State

Each MHL device shall indicate in the DEV_STATE register the state of its connection to the requesting device.

Table 7-24. Device State Register Details

Register Name		Offset	7	6	5	4	3	2	1	0
DEV_STATE		0x00								
Bit	**Name**									**Notes**
0-7		Reserved								

7.9.1.2 MHL Version

Table 7-25. MHL Version Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
MHL_VERSION	0x01	MHL_VER_MAJOR				MHL_VER_MINOR			

Devices shall set the value of MHL_VERSION to the version of the MHL Specification to which they comply. For example, those devices compliant with MHL Specification Version 2.0 shall set this register to 0x20.

7.9.1.3 Device Category

A device shall use one of the device categories indicated in the DEV_CAT register. The category may change dynamically, such as if a Dongle loses its power supply and becomes an unpowered Dongle.

Table 7-26. Device Category Register Details

Register Name		Offset	7	6	5	4	3	2	1	0
DEV_CAT		0x02		PLIM1	PLIM0	POW	DEV_TYPE			
Bit	**Name**									**Notes**
0-3	DEV_TYPE	Indicates the category of the device. At least one bit shall be set in this field.								
4	POW	When set, device can output power across VBUS to the device connected through the cable. When clear, device cannot output power across VBUS to the device connected through the cable.								
5, 6	PLIM0, PLIM1	PLIM1 (msb) and PLIM0 (lsb) together indicate the minimum VBUS power which the device is designed to deliver to the Source. {PLIM1,PLIM0} {0,0} = Device delivers at least 500mA {0,1} = Device delivers at least 900mA {1,0} = Device delivers at least 1.5A {1,1} = $I_{VBUS\_Dongle\_to\_Source}${min}								
7		Reserved								

Settings for DEV_TYPE, POW and PLIM are defined in Table 7-27 and Table 7-28.

Table 7-27. DEV_TYPE Settings

DEV_TYPE Setting	Value	Explanation	Notes
	0b0000	Not allowed. At least one bit must be set in this field.	
Sink	0b0001		1, 2
Source	0b0010		2, 1
Dongle	0b0011		
	other	Reserved for future use.	

Notes on Table 7-27:

1. Power output is required for these types of devices. See Section 13.9.1.2.
2. RCP and RAP support is required for these types of devices. See Section 7.6.3.
3. A Source which draws power from a connected Sink or Dongle shall meet the requirements in Section 13.9.1.1.

Table 7-28. POW and PLIM Settings

MHL 2 Device Type	Power Provisioning	POW	PLIM1, PLIM0 Setting				Notes
			0,0	0,1	1,0	1,1	
Source	Source which cannot output Power	0					1
	Source which can output Power	1					2
Sink	Sink with AC power	1		OK	OK		3
	DC-powered Sink without external Power Adapter attached	0					4
		1	OK	OK	OK	OK	5
Dongle	with external Power Adapter attached	1				OK	6, 7a
		1		OK	OK		6, 7b
		1	OK				6, 7c
	Unpowered Dongle	0					8

Notes on Table 7-28:

1. A Source which cannot output power shall set POW=0. PLIM1 and PLIM0 values shall be ignored.
2. A Source which can output power shall set POW=1, and meet the requirement of $I_{VBUS\_Source\_to\_Dongle}${min} shown in Table 13-24.
3. A Sink receiving power shall indicate a minimum I_{VBUS} which meets the requirements of $I_{VBUS\_Sink\_to\_Source\_MHL2\_Sink}${min} shown in Table 13-24. The PLIM allowed settings are shown.
4. A DC-powered Sink not receiving power from an external supply may set POW=0. PLIM1 and PLIM0 values shall be ignored.
5. A DC-powered Sink without receiving power from an external supply may supply power to the Source. In that case, the Sink shall indicate a minimum I_{VBUS} that it can provide. The PLIM allowed settings are shown.
6. A Dongle shall indicate a minimum I_{VBUS} which meets the requirements of $I_{VBUS\_Dongle\_to\_Source}${min} shown in Table 13-24.
7. A Dongle shall either (a) set PLIM1, PLIM0 to {1,1} to indicate that it can output the minimum $I_{VBUS\_Dongle\_to\_Source}$, but not the $I_{VBUS\_Sink\_to\_Source\_MHL2\_Sink}${min} shown in Table 13-24; or (b) output at least 900mA or 1.5A, using settings of {0,1} or {1,0} in PLIM, respectively; or (c) output 500mA using setting of {0,0} in PLIM when powered from USB or similar port.
8. An unpowered Dongle shall set POW=0. PLIM1 and PLIM0 values shall be ignored.

7.9.1.4 Adopter ID Registers

Table 7-29. Adopter ID Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
ADOPTER_ID_H	0x03								
ADOPTER_ID_L	0x04								

Bit	Name		Notes
15-0	ADOPTER_ID	Adopter ID uniquely identifies the MHL Adopter (or Promoter).	

The ADOPTER_ID value may be used in conjunction with the DEVICE_ID value to coordinate use of the Scratchpad Registers. Only ADOPTER_ID values as assigned by MHL, LLC shall be used.

An ADOPTER_ID value of 0x0000 indicates that no ADOPTER_ID is available. If the device indicates with SP_SUPPORT=1 that Scratchpad Registers is available, then Scratchpad Registers after the two bytes for the 0x0000 ADOPTER_ID should be used without any coordinated format, as there is no ADOPTER_ID available.

A valid 16-bit ADOPTER_ID has a value of 0x00, as defined above, or from 128 (decimal) to 65535 (decimal). Values from 1 to 127 are reserved for future use.

See Section 7.4.3.10.

7.9.1.5 Video Link Mode Support Register

Table 7-30. Video Link Mode Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
VID_LINK_MODE	0x05								

Bit	Name		Notes
0	SUPP_RGB444	Supports RGB 4:4:4 video link mode.	
1	SUPP_YCBCR444	Supports YcbCr 4:4:4 video link mode.	
2	SUPP_YCBCR422	Supports YcbCr 4:2:2 video link mode with 8, 10 or 12-bit chroma and luma values.	
3	SUPP_PPIXEL	Supports PackedPixel video link mode.	
4	SUPP_ISLANDS	Support Data Islands	1
5	SUPP_VGA	Supports 640x480/60Hz video format timing (VGA).	2
6-7		Reserved	

Notes on Table 7-30:

1. Bit 4 is redundant with bits 1-3, in that those mode also require data islands. A device which supports only RGB 4:4:4 video, and no audio, can explicitly indicate with bit 4 its ability to handle data islands.
2. Bit 5 explicitly indicates that the device supports VGA mode. When set in a Sink or Dongle, and read by the MHL Source, this bit allows the Source to output 640x480/60Hz (VGA) video format timings, even though some downstream protocols through a Dongle do not specifically show VGA support in the EDID. See Section 5.2.1. MHL Sinks and Dongles shall set SUPP_VGA to '1' if they support 640x480/60Hz (VGA) video format timings, and shall clear this bit to '0' if they do not support 640x480/60Hz (VGA) video format timings.

7.9.1.6 Audio Link Mode Support Register

Table 7-31. Audio Link Mode Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
AUD_LINK_MODE	0x06								

Bit	Name		Notes
0	AUD_2CH	Supports 2-channel PCM audio streams (with Layout 0 audio sample packets).	1
1	AUD_8CH	Supports greater than 2 channel PCM audio streams (with Layout 1 audio sample packets).	1
2-7		Reserved	

Notes on Table 7-31:

1. The following three combinations may be used:
   Set AUD_2CH=0 and set AUD_8CH=0: Sink does not support any audio content, or
   Set AUD_2CH=1 and set AUD_8CH=0: Sink supports 2-channel audio only, or
   Set AUD_2CH=1 and set AUD_8CH=1: Sink supports 2-channel audio and more than 2 channel audio.

7.9.1.7 Video Type Support Register

Each device indicates with this register the types of video it supports, in the context of "Content Type" from EIA/CEA-861E. A Dongle may default to indicate no support for content type (SUPP_VT=0), or may deduce the supported types of the downstream device, for example by reading the downstream EDID, and reflect those supported types in this register.

Table 7-32. Video Type Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
VIDEO_TYPE	0x07	SUPP_VT				VT3	VT2	VT1	VT0

Bit	Name		Notes
0	VT_GRAPHICS	When set, indicates that this device recognizes the Graphics type of video content. When cleared, indicates no support for Graphics type.	1
1	VT_PHOTO	When set, indicates that this device recognizes the Photo type of video content. When cleared, indicates no support for Photo type.	1
2	VT_CINEMA	When set, indicates that this device recognizes the Cinema type of video content. When cleared, indicates no support for Cinema type.	1
3	VT_GAME	When set, indicates that this device recognizes the Game type of video content. When cleared, indicates no support for Game type.	1
4-6		Reserved	
7	SUPP_VT	When set, indicates that this device shows its content type support in VIDEO_TYPE[3:0]. When clear, indicates that VIDEO_TYPE[3:0] should be ignored.	2

Notes on Table 7-32:

1. Refer to the EIA/CEA-861E specification for definition of these video types, which are indicated in the AVI InfoFrame with the CN1 and CN0 bits. See Section 10.2.1.
2. An MHL device indicates with this bit that the information in VT3, VT2, VT1, VT0 has not been initialized, and in the case of a Dongle may not reflect the content type support of a downstream device.

7.9.1.8 Logical Device Map Register

Each device indicates with this register the types of functions it supports. With this information, each connected device can choose, across CBUS, to control only those types of functions indicated as being supported. For example, an MHL device which indicates it is only a "display" should not be sent CBUS commands which control content recording or content sourcing.

Every MHL device shall select at least one bit in the LOG_DEV_MAP register, unless the MHL device sets RCP_SUPPORT=0, indicating no support for RCP. A device with RCP_SUPPORT=0 may set LOG_DEV_MAP to 0x00.

A device may support one or more logical devices. Each supported function is indicated in the register by setting the corresponding bit. A zero bit indicates no support for that logical device type.

Table 7-33. Logical Device Map Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
LOG_DEV_MAP	0x08								

Bit	Name		Notes
0	LD_DISPLAY	Device Supports Display Output	2
1	LD_VIDEO	Device Supports Video Output across MHL link	
2	LD_AUDIO	Device Supports Audio Output across MHL link	1
3	LD_MEDIA	Device Supports Media Handling	
4	LD_TUNER	Device Supports Tuner Handling	
5	LD_RECORD	Device Supports Content Recording	
6	LD_SPEAKER	Device Supports Audio Output (audible sound)	1
7	LD_GUI	Device Supports Rendering and Navigation of Menus or other user interface.	3

Notes on Table 7-33:

1. The logical device types are from the point of view of the device itself. That is, a device which claims to be a "audio" device establishes itself as a device which can output audio content on the MHL link. A TV, for example, may also be able to output audio – in audible form – but the TV should not set LOG_DEV_MAP[2] unless it is able to output audio content on a downstream MHL link. A TV with speakers should set LOG_DEV_MAP[6].
2. "Supports display output" means that the device renders video content to a visible form. The device may be a DTV, a projector, etc.
3. Some devices, such as an MP3 player Source or a desktop monitor Sink may not support rendering of a user interface.

7.9.1.9 Bandwidth Limit Register

Table 7-34. Bandwidth Limit Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
BANDWIDTH	0x09								

Each device indicates with this register the maximum Link Clock frequency it can handle across the MHL+ and MHL- differential pair. The value is expressed in terms of 5's of megahertz. For example, an upper limit of 75 MHz would be expressed as the value of 15 decimal, or 0x0F. Refer to F_{MHL} in Table 13-6. If the maximum bandwidth of the system is a value between increments of 5MHz, then the value in the BANDWIDTH register should be set to the next larger increment of 5MHz. For example, a system with a limit of 74.25MHz should set the limit at 75MHz.

In PackedPixel mode, the BANDWIDTH register allows a video stream with a pixel clock two times the link clock rate to be received. For example, a Sink which indicates that it supports 75MHz (BANDWIDTH set to 0x0F) and indicates it supports PackedPixel mode (SUPP_PPIXEL=1) shall be able to receive a video stream with an original pixel clock rate of two times 75MHz, or 150MHz when operating in PackedPixel mode.

A value of 0x00 means that no bandwidth is indicated. A Sink which indicates in its BANDWIDTH register a value of 0x00 shall be able to support all video modes indicated in its EDID using the MHL connection.

A Source which reads the BANDWIDTH register value from a connected MHL Dongle shall respect the lower of the two limits: that set in the Dongle's BANDWIDTH register, and that set by the data in the downstream Sink's EDID – provided that the downstream Sink has an EDID. For example, a Source connected to a Dongle which can handle up to 74.25MHz, which is connected to a downstream Sink which can handle 148.5MHz (1080p/60) shall output to the Dongle only video modes up to 74.25MHz (such as 720p/60 or 1080i/60).

It is the responsibility of the MHL Source device to read the bandwidth maximum of the MHL Sink device attached to it, and to limit the output modes to stay within the stated maximum frequency. No MHL Source shall output a content stream which has an MHL link clock frequency greater than 5 times the value read from the connected MHL Sink's Bandwidth Limit Register.

7.9.1.10 Feature Support Flag Register

Each device indicates with this register the support of features in this MHL specification which are optional. An MHL Source shall not send any RCP sub-command to an MHL Dongle if the MHL Dongle indicates with RCP_SUPPORT=0 that it does not support RCP. An MHL Source shall not send any RAP sub-command to an MHL Dongle if the MHL Dongle indicates with RAP_SUPPORT=0 that it does not support RAP. An MHL Source shall not send any UCP sub-command to an MHL Dongle if the MHL Dongle indicates with UCP_RECV_SUPPORT=0 that it does not support UCP.

Table 7-35. Feature Support Flag Register Details

Register Name		Offset	7	6	5	4	3	2	1	0
FEATURE_FLAG		0x0A								
Bit	**Name**									**Notes**
0	RCP_SUPPORT	1 = Supports RCP, 0 = Does not Support RCP.								1, 3
1	RAP_SUPPORT	1 = Supports RAP, 0 = Does not Support RAP.								2, 3
2	SP_SUPPORT	1 = Supports Scratchpad Registers, 0 = Does not								4

		Support Scratchpad Registers.	
3	UCP_SEND_SUPPORT	1 = Support sending UCP, 0 = Does not Support sending UCP.	5
4	UCP_RECV_SUPPORT	1 = Support receiving UCP, 0 = Does not Support receiving UCP.	5

Notes on Table 7-35:

1. MHL Source and Sink devices shall set this bit to '1', indicating RCP support. MHL Dongles may set this bit to '0', if they do not claim support for RCP. Dongles which set this bit to '1' shall meet the MHL Sink requirements for RCP support.
2. MHL Source and Sink devices shall set this bit to '1', indicating RAP support. MHL Dongles may set this bit to '0', if they do not claim support for RAP. Dongles which set this bit to '1' shall meet the MHL Sink requirements for RAP. Support
3. MHL Dongles shall set either set both RCP_SUPPORT=1 and RAP_SUPPORT=1, or shall set both RCP_SUPPORT=0 and RAP_SUPPORT=0. Other settings are not permitted. See Sections 7.6.3 and 7.7.8.1.
4. MHL Source and Sink devices shall set this bit to '1', indicating support for Scratchpad Registers, WRITE_BURST command, and REQ_WRT and GRT_WRT interrupts. MHL Dongles may set this bit to '0', if they do not claim support for Scratchpad Registers. Dongles which set this bit to '1' shall meet the MHL Sink requirements for Scratchpad Registers support.
5. See Section 7.8.4 for requirements for support of UCP and related sub-commands.

7.9.1.11 Device ID Registers

Table 7-36. Device ID Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
DEVICE_ID_H	0x0B								
DEVICE_ID_L	0x0C								

Bit	Name		Notes
15-0	DEVICE_ID	Device ID is assigned by the Adopter (or Promoter) to identify the Source, Sink or Dongle.	

The DEVICE_ID value may be used in conjunction with the ADOPTER_ID value to coordinate use of the Scratchpad Registers.

7.9.1.12 Scratchpad Size Register

Table 7-37. Scratchpad Size Register Details

Register Name	Offset	7	6	5	4	3	2	1	0
SCRATCHPAD_SIZE	0x0D								

Each device indicates with this register the number of bytes in the Scratchpad Registers. See section 7.9.4. If SP_SUPPORT is set to '0', then the value in SCRATCHPAD_SIZE is "don't care". See section 7.9.1.10.

7.9.1.13 Interrupt and Status Size Register

Table 7-38. Interrupt and Status Size Register Details

Register Name		Offset	7	6	5	4	3	2	1	0
INT_STAT_SIZE		0x0E	STAT_SIZE				INT_SIZE			
Bit	**Name**									**Notes**
0-3	INT_SIZE	Interrupt Register Count: 0x0=1 byte, 0xF=16 bytes.								
4-7	STAT_SIZE	Status Register Count: 0x0=1 byte, 0xF=16 bytes.								

Each device indicates with this register the number of bytes in the interrupt and status memories.

7.9.2 Device Interrupt Registers

A Source device may write a one-byte value to a register in the device interrupt register set of the attached Sink device. Similarly, a Sink device may write a one-byte value to an attached Source. Each device shall provide a minimum of four one-byte device interrupt registers, using the offset range 0x20 to 0x23. That is, each device sets an interrupt in the other, connected device's address space.

A write to a device interrupt register is different from a write to a device status register. A zero bit in the value written to a device interrupt register has no affect on the value in that register's corresponding bit. A one bit in the value written to a device interrupt register causes the corresponding bit to be set to one. Internal to the Source and Sink devices, a set interrupt register bit may create an event, to which the device or its firmware may respond.

Note that internal events in the device may also set interrupts in the same interrupt register space.

The device shall indicate with a one-byte value at offset 0x0E bits 3:0 the total count of device interrupt registers. Registers beyond those at 0x20-0x23 shall be populated beginning at offset 0x24. This provides a maximum interrupt register space of 16 bytes, up to offset 0x2F.

Each MHL device shall support access to its Device Interrupt registers across an MHL link by SET_INT commands. Therefore, an initiator may change only one Device Interrupt register in a follower with each command across the link.

In each register's description, the table shows a column for 'Source' and a column for 'Sink', marked with an 'X'. The 'X' indicates that that type of device benefits by writing that interrupt bit into the other device. A device for which no 'X' is indicated is not prohibited from writing that interrupt bit, but normal usage does not require it to use that bit.

Table 7-39. Mandatory Device Interrupt Registers

Address	Device Interrupt Register Name	Definition
0x20	Device Register Change	Flags a change in Capability or Scratchpad.
0x21	Device State Change	Flags a change in EDID content.
0x22	Reserved for Future Use	
0x23	Reserved for Future Use	

Not all interrupt bits are required for all device types. Table 7-40 shows with an 'X' those bits which are required according to device type: Source, Sink and Dongle. Requiring a bit means that the device shall be able to write that interrupt bit into the opposite device.

Table 7-40. MHL Interrupt Bits Required by Device Type

Address	Dev Cap Register Name	Field	Source	Sink	Dongle
0x20	RCHANGE_INT	DCAP_CHG	X	X	
		DSCR_CHG	X	X	
		REQ_WRT	X	X	
		GRT_WRT	X	X	
		3D_REQ	X		
0x21	DCHANGE_INT	EDID_CHG		X	

7.9.2.1 Device Register Change Interrupt

Table 7-41. Register Change Interrupt Register Details

Register Name		Offset		7	6	5	4	3	2	1	0
RCHANGE_INT		0x20									
Bit	**Name**	**Source**	**Sink**								**Notes**
0	DCAP_CHG	X	X	Device Capability Register value changed.							1
1	DSCR_CHG	X	X	Device Scratchpad Register value changed.							2
2	REQ_WRT	X	X	Request-to-Write.							2
3	GRT_WRT	X	X	Grant-to-Write.							2
4	3D_REQ	X		Request for 3D Information.							3
5-7				Reserved.							

Notes on Table 7-41:

1. Two connected MHL devices coordinate exchange of up-to-date Device Capability Register data using the DCAP_CHG interrupt. A device with one or more new Device Capability Register values shall write the DCAP_CHG interrupt bit in the opposite device. This action triggers a reaction from the opposite device to read the complete set of Device Capability Registers. See Section 7.4.3.7.
   Note that only MHL Sources are required to respond to receipt of a DCAP_CHG interrupt with a sequence of READ_DEVCAP commands. See Section 7.4.3.7.
2. Two connected MHL devices coordinate the transfer of data via their Scratchpad Registers using the REQ_WRT, GRT_WRT and DSCR_CHG bits. An initiator shall request clearance to write data with a WRITE_BURST command using the REQ_WRT. Initiator which sends a Request-to-Write interrupt shall be free to send another Request-to-Write interrupt if that initiator does not receive a Grant-to-Write interrupt from the follower within the $T_{BURST\_WAIT}\{max\}$ time.
   Figure 7-61 illustrates the flow of register values, including the interrupts and the burst data, when using RCHANGE_INT and WRITE_BURST to the Scratchpad Registers.
   Note that MHL Dongles are allowed to indicate no support for Scratchpad Registers, which includes no support for WRITE_BURST, REQ_WRT and GRT_WRT, with the SP_SUPPORT bit in the FEATURE_FLAG register. See Section 7.9.1.10.
3. 3D_REQ is used by a Source device to request the 3D video support information from the Sink. See Section 5.9.1.1.

Refer also to the ADOPTER_ID and DEVICE_ID registers, and their usage in coordinating use of the Scratchpad Registers. See Section 7.9.1.

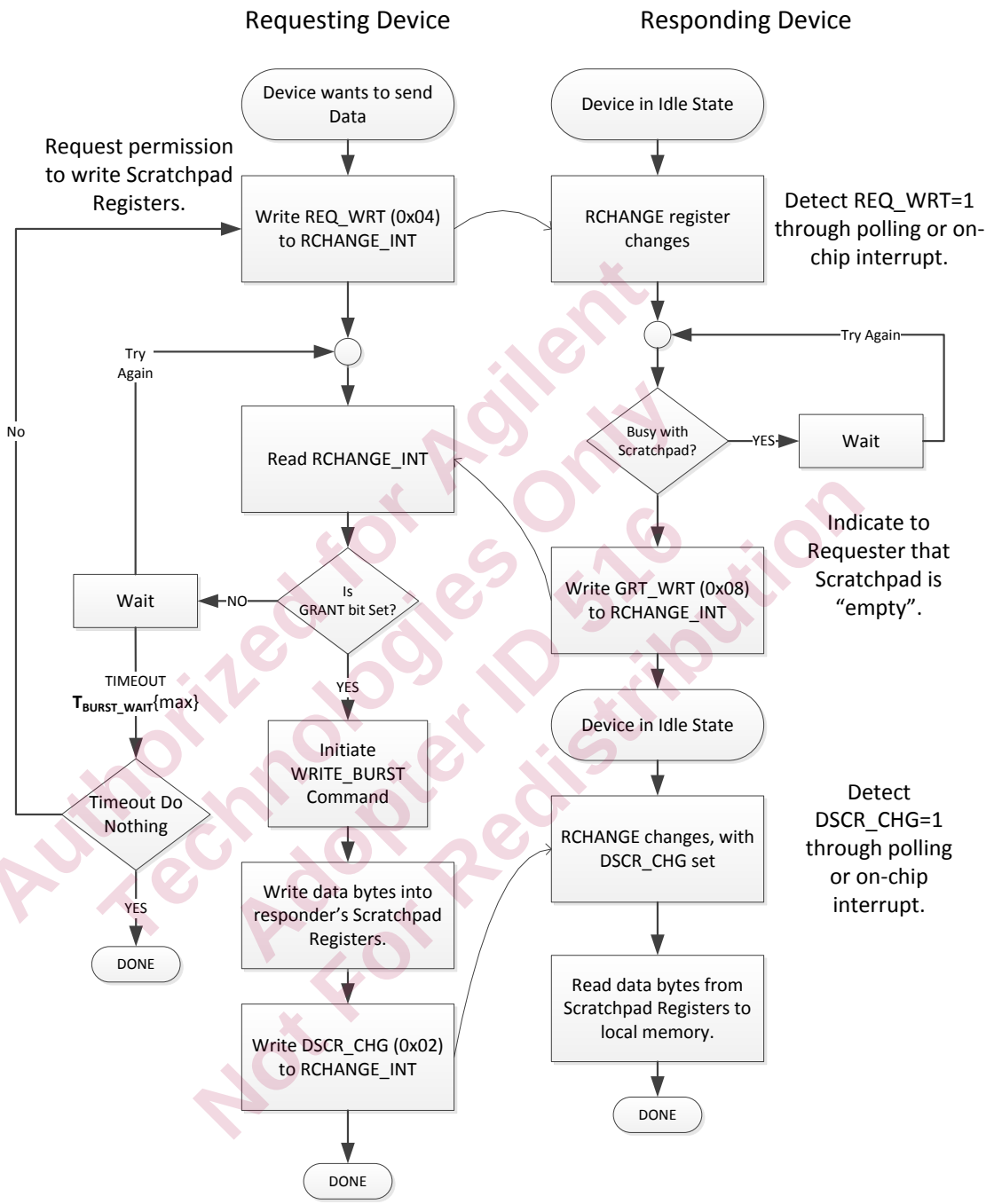


Figure 7-61. MSC WRITE_BURST Coordinated with RCHANGE_INT Interrupt Bits

7.9.2.2 Device State Change Interrupt

Table 7-42. Device State Change Interrupt Register Details

Register Name		Offset		7	6	5	4	3	2	1	0
DCHANGE_INT		0x21									
Bit	**Name**	**Source**	**Sink**								**Notes**
0				Reserved.							
1	EDID_CHG		X	EDID content change on virtual DDC channel.							1, 2
2				Reserved.							
3				Reserved.							
4-7				Reserved.							

Notes on Table 7-42:

1. EDID_CHG is provided to allow a Sink to indicate to a Source that the Sink's EDID is to be read again. This allows the Sink to signal the Source without having to use the virtual Hot Plug Detect signal (through the SET_HPD and CLR_HPD commands). Note that the Sink shall already have established an HPD=asserted state known to the Source by sending to the Source the SET_HPD command prior to asserting the EDID_CHG interrupt. EDID may not be read until HPD is active. See Section 8.5.4.
2. Sink may write EDID_CHG bit to 1 while the Source is reading the EDID content.

7.9.3 Device Status Registers

A Source device may write a one-byte value to a register in the device status register set of the attached Sink device. Similarly, a Sink device may write a one-byte value to an attached Source. Each device shall provide a minimum of four one-byte device status registers, using the offset range 0x30 to 0x33. That is, a device writes its own status into the byte or bytes of the other, connected device: the Source writes the status of the Source device into the status registers in the Sink device, and vice versa.

The device shall indicate with a one-byte value at offset 0x0E bits 3:0 the total count of device status registers. Registers beyond those at 0x30-0x33 shall be populated beginning at offset 0x34. This provides a maximum status register space of 16 bytes, up to offset 0x3F.

Each MHL device shall support access to its device status registers across an MHL link by WRITE_STAT commands.

An 'X' in the columns labeled 'Source' and 'Sink' indicates that that device type shall accurately indicate the field value by writing it to the opposite device with the WRITE_STAT command whenever the status changes and affects the field's value. Indications for 'Sink' apply also to Dongle devices.

Table 7-43. Mandatory Device Status Registers

Address	Device Status Register Name	Definition
0x30	Ready Bits for Connected Device	Flags connected device's memory ready.
0x31	Active Link Mode	Indicates current link mode.
0x32	Reserved for Future Use	
0x33	Reserved for Future Use	

Not all status register fields are required for all device types. Table 7-44 shows with an 'X' those fields which are required according to device type: Source, Sink and Dongle. Requiring a field means that the device shall be able to write that status field into the opposite device.

Table 7-44. MHL Status Register Fields Required by Device Type

Address	Dev Cap Register Name	Field	Source	Sink	Dongle
0x30	CONNECTED_RDY	DCAP_RDY	X	X	X
0x31	LINK_MODE	CLK_MODE	X		
		PATH_EN	X	X	X
		MUTED	X		

7.9.3.1 Connected Device Ready Status

Table 7-45. Connected Device Ready Status Register Details

Register Name			Offset	7	6	5	4	3	2	1	0
CONNECTED_RDY			0x30								
Bit	**Name**	**Source**	**Sink**								**Notes**
0	DCAP_RDY	X	X	Capability Register values are stable.							
1-7				Reserved							

If Device Capability register data is not stable in the device, such as during initialization, then the DCAP_RDY bit shall be set to zero. When the Device Capability register data is stable, then the DCAP_RDY bit shall be set to one. Refer to the DCAP_CHG interrupt bit for details on how one device indicates to a connected device that the Device Capability register data has been changed.

7.9.3.2 Active Link Mode Status

The Source shall maintain an accurate indication of the current MHL link clock mode by setting CLK_MODE. From initial discovery to the first WRITE_STAT from Source to Sink, the Source shall maintain the link in Normal mode, and the Sink shall assume Normal mode on the link.

Sink writes PATH_EN{Sink}=1 into the Source device to enable TMDS transmission from the Source. The Sink shall maintain an accurate indication of the active use of the Sink's MHL input port by setting or clearing PATH_EN in its LINK_MODE status register.

Source may only enable its TMDS output after the Sink writes PATH_EN{Sink}=1, and shall disable its TMDS output after the Sink writes PATH_EN{Sink}=0. On detection of the Sink writing PATH_EN{Sink}=1, Source shall write PATH_EN{Source}=1. On detection of the Sink writing PATH_EN{Sink}=0, the Source shall write PATH_EN{Source}=0. Source shall not change PATH_EN{Source} at other times. After Sink writes PATH_EN{Sink}=0, Sink shall not set its termination to $Z_{RXSENSE\_TERM}$ until after it detects the Source writes PATH_EN{Source}=0.

Table 7-46. Link Mode Status Register Details

Register Name		Offset	7	6	5	4	3	2	1	0
LINK_MODE		0x31				MUTED	PATH_EN	CLK_MODE		
Bit	**Name**	**Source**	**Sink**	**Notes**						
0-2	CLK_MODE	X		The clock mode on the link.				1, 2, 3, 4		
3	PATH_EN	X	X	Device's TMDS path is in use.						
4	MUTED	X		Device's content stream is muted.				5		
5-7				Reserved						

Notes on Table 7-46:

1. The MHL link is currently operating in the clock mode indicated here:
   Bits[2:0] = 0b000 = Reserved
   Bits[2:0] = 0b001 = Reserved
   Bits[2:0] = 0b011 =Normal (24-bit) clock mode
   Bits[2:0] = 0b010 = PackedPixel clock mode
   Bits[2:0] = 0b100 through 111 = Reserved
2. An MHL Source shall update CLK_MODE to match the actual mode on the content stream within $T_{MODE\_CHG\_DLY}$ of a change in clock mode on the link.
3. An MHL Sink shall assume Normal (24-bit) clock mode on the link from discovery until the Sink receives an update to LINK_MODE using WRITE_STAT from the Source. Thereafter, while in CBUS-connected states, link mode on the Sink side shall be maintained based only on the value in the CLK_MODE field.
4. Reserved values received in the CLK_MODE field shall be discarded by the Sink and interpreted as "no change" in the link mode.
5. The Source may set this bit to indicate that stream contains no content: video is a blank screen, audio is silent. This state is commonly used when changing from one video mode to another while encrypting, to prevent invalid pixel data during the transition time. When clear, this bit indicates that the stream contains the content.

7.9.4 Device Scratchpad Registers

A Source device may write a one-byte value into the Scratchpad Registers space in the attached Sink. Similarly, a Sink may write to the attached Source. This provides a memory space which either device may use to communicate to the other device. Each device shall provide a sixteen byte writeable memory space at offset 0x40-0x4F. An eight-bit value written to one of these registers causes the register to change to the eight-bit value.

The device shall indicate with a one-byte value at offset 0x0D (SCRATCHPAD_SIZE) the total count of Scratchpad Registers. Registers beyond those at 0x40-0x4F shall be populated beginning at offset 0x50 up to offset 0x7F. This provides a maximum Scratchpad Register space of 64 bytes.

MHL devices use the Scratchpad Registers in combination with the REQ_WRT and GRT_WRT status bits. Refer to Section 7.9.2.1 for details.

A more sophisticated mechanism for sharing array data across the CBUS may be added in a future version of this specification.

Scratchpad Registers support is required for MHL Sources and Sink. MHL Dongles may indicate no support for Scratchpad Registers with the SP_SUPPORT bit in the FEATURE_SUPPORT register (see Section 7.9.1.10).

'Support for Scratchpad Registers' in this context means that the device performs the proper protocol in receiving and sending a WRITE_BURST command – the only MSC command which accesses the Scratchpad Registers; and allows its own REQ_WRT and GRT_WRT interrupt bits to be set by a SET_INT command from the opposite device. Indicating support for Scratchpad Registers does not itself indicate that data received from a WRITE_BURST command is used by the device. This determination is made at the application layer. If there is no application layer support, then the device may discard data received from a WRITE_BURST command. (See Section 7.4.3.10.)

7.10 BURST_ID

WRITE_BURST has expanded uses in MHL 2 through the definition of values which take the place of the ADOPTER_ID bytes in the WRITE_BURST transfer. Rather than inserting the ADOPTER_ID of the Responder (as defined in Section7.9.1.4), the Requester may insert any of the BURST_ID values found in Table 7-47.

7.10.1 BURST_ID Codes

Each BURST_ID provides for a different use of the payload bytes in the WRITE_BURST transfer. The Responder receiving the WRITE_BURST data understands from the BURST_ID value the type of data being received.

A valid BURST_ID has a value in the range 1 (decimal) to 127 (decimal), if it is also defined in Table 7-47.

Note that this set of BURST_ID values does not limit the MHL user from setting these two bytes in a WRITE_BURST to a valid ADOPTER_ID value. The ranges of valid BURST_ID and valid ADOPTER_ID values do not overlap.

Table 7-47. BURST_ID Codes

Symbol	Value	Description	Notes
3D_VIC	0x0010	Indicates Video Index Codes for 3D	1
3D_DTD	0x0011	Indicates Detailed Timing Descriptors for 3D	1

Notes on Table 7-47:

1. 3D_VIC and 3D_DTD are used by Sink in indicating support of 3D. See Section 5.9.1.

7.10.2 MHL Support Requirements for BURST_ID

Support for each BURST_ID value is defined in the section corresponding to the feature associated with that BURST_ID.

8 Device Discovery

The CBUS provides mechanisms for the Source and Sink device to discover connectivity to an MHL-compliant Sink and Source device respectively. There are different mechanisms for the Source device and the Sink device.

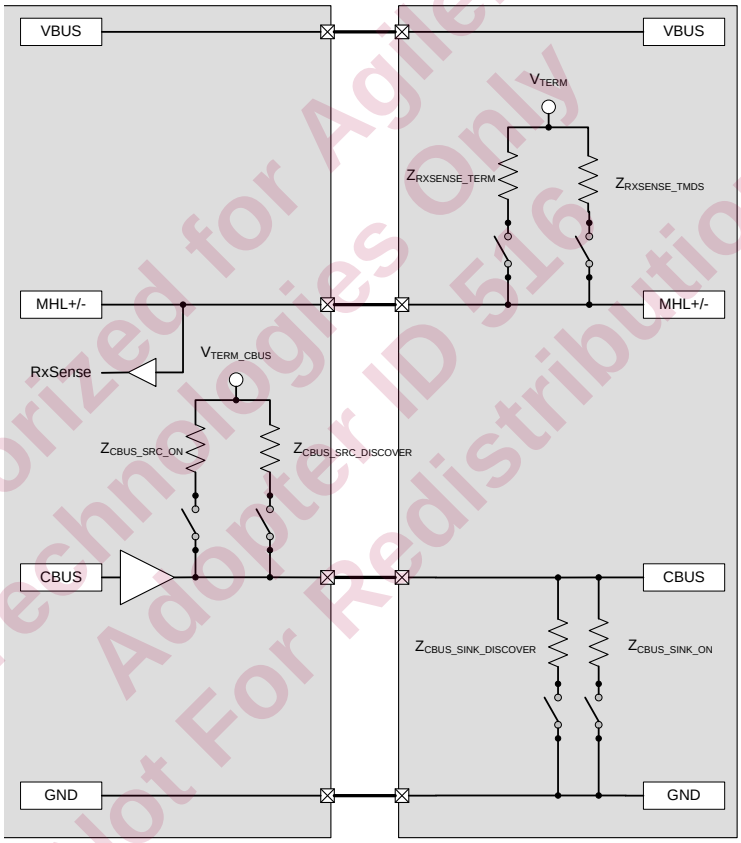


Figure 8-1. Connection of an MHL Source to an MHL sink

The definition of discovery is divided into two segments: "electrical discovery", which enables both VBUS and CBUS function; and "extended discovery", which uses CBUS commands to exchange information for complete identification of Source and Sink.

Similarly, disconnection involves multiple mechanisms, to detect physical disconnection, disablement of power, etc.

Section	Material
8.1	Discovery and Disconnection Mechanisms
8.2	Discovery and Disconnect
8.3	Discovery Timelines
8.4	Electrical Discovery Sequence
8.5	Data Exchange during Discovery Sequence

8.6	Disconnection Sequence
8.7	Sharing MHL Connector Pins

8.1 Discovery and Disconnection Mechanisms

MHL utilizes a set of mechanisms to reliably detect both connection between Source and Sink devices, and disconnection between Source and Sink devices.

8.1.1 CBUS Impedance

CBUS impedance is controlled to allow the Source to discover the Sink as an MHL device. After discovery, the CBUS impedance is changed to allow signaling with packets (see Section 7).

Details of CBUS impedance handling are shown in the state diagrams Figure 8-2 to Figure 8-7, and details later in Section 8.

8.1.2 Receiver Sense (RxSense)

Receiver Sense (RxSense) is a signal measured by the MHL Source. RxSense is active (HIGH) when either of the two TMDS signals from Source to Sink (or Dongle) is terminated to an active voltage. RxSense is inactive (LOW) when both of the two TMDS signals are unterminated. This may be due to an inactive power supply in the Sink, or due to a disconnected cable. The termination resistance is defined as $Z_{RXSENSE}$. There are three ranges for $Z_{RXSENSE}$: $Z_{RXSENSE\_TERM}$, $Z_{RXSENSE\_TMDS}$, and high impedance (meeting the $I_{RXSENSE\_LEAK}$ specification).

See Sections 8.4.4 and 8.6.2.

An MHL Source shall be able to complete the discovery pulse process with an MHL Sink or Dongle without regard to the Sink or Dongle terminating the TMDS signals or the state of RxSense.

An MHL Source shall be able to read EDID from the discovered MHL Sink or Dongle without regard to the state of RxSense.

An MHL Source which implements HDCP shall meet the HDCP Specification requirements by observing the state of RxSense as it is specified in this document.

An MHL Source or Sink or Dongle shall be able to send commands on MSC without regard to the state of RxSense.

8.1.3 Hot Plug Detect (HPD)

MHL provides a virtual Hot Plug Detect (HPD), signaled from Sink to Source, by means of commands on CBUS. The virtual HPD signal is used by the Source in reading the Sink's EDID and for content protection purposes. The virtual HPD simplifies interconnection, such as through an MHL Dongle, with other protocols that may run alongside MHL.

An MHL Source shall consider HPD deasserted whenever CBUS is disconnected. An MHL Source shall consider HPD asserted on receipt of a SET_HPD command from the connected Sink or Dongle.

An MHL Sink shall be able to transmit its HPD state to the connected MHL Source using the SET_HPD and CLR_HPD commands on the MSC channel of CBUS (see Sections 7.4.3.3 and 7.4.3.4).

An MHL Source and Sink shall be able to exchange packets on the CBUS MSC sub-channel irrespective of the state of HPD. For a Dongle, this allows an MHL Dongle to communicate with an MHL Source, even if the downstream link from the Dongle to the connected device does not indicate a fully-active state and the HPD signal (or equivalent) on this link is not asserted.

An MHL Source and Sink shall exchange packets on the CBUS DDC sub-channel while HPD is HIGH, asserted by a SET_HPD command from Sink (or Dongle) to Source, without a subsequent CLR_HPD command. It should be noted, however, that if the MHL DDC sub-channel depends on the availability of a DDC bus on the downstream side (e.g., from a Dongle), then the MHL DDC commands may terminate with an error. This termination may occur at the link layer or at the translation layer. Such termination should not indicate that the MHL DDC Sink or Dongle is itself in error, but that there may be a downstream dependency.

A Sink which modifies the contents of its EDID shall notify the Source by sending a CLR_HPD command followed subsequently by a SET_HPD command (or alternatively through setting of the EDID_CHG bit if the last command sent was a SET_HPD command resulting in an HPD=asserted state for the Source). See Section 7.9.2.2.

A Sink shall not send SET_HPD until its EDID is available to be read. See Section 8.5.4.

The HDCP Specification includes a dependency on HPD. Authentication may not begin until the Source detects an active Hot Plug Detect (HPD=HIGH) signal from the Sink or Repeater. Therefore, MHL Sources which implement HDCP shall meet the HDCP Specification requirements by observing the state of HPD as set from the Sink (or Dongle) via the SET_HPD and CLR_HPD commands.

8.1.4 MHL Cable Discovery

Sink devices which support one or more protocols in addition to the MHL protocol through a port shall ensure while the MHL cable is attached that such an input port meets the requirements of Section 8.

8.2 Discovery and Disconnect State Diagrams

The state diagrams in the following sections provide detail on the requirements for connection, discovery and disconnection handling.

In the diagrams, the states on the VBUS, CBUS, MHL+ and MHL- pins are defined with respect to how they are to be set in MHL mode. If an output port on a Source is shared with another protocol, or an input port on a Sink or Dongle is shared with another protocol, where noted in Section 8.2, the states on the pins used by VBUS, CBUS, MHL+ and MHL- may be defined to other conditions than those described. System design of an MHL device shall be such as to prevent damage to MHL-related signals or voltage pins when the port is shared with another protocol. Refer to Section 13.12.

8.2.1 Source Viewpoint

Two possible implementations are shown in the following sections: one for a Source which has the ability to output VBUS (to power a Dongle, for example); the other for a Source which does not have the ability to output VBUS.

8.2.1.1 MHL Source with Possible VBUS Output

An MHL Source which can output VBUS shall process events received from devices connected to it according to the state transitions and timings shown in Figure 8-2, and further defined in Table 8-1 and Table 8-2.

See Section 8.2.1.2 for an abbreviated description for a Source which cannot output VBUS.

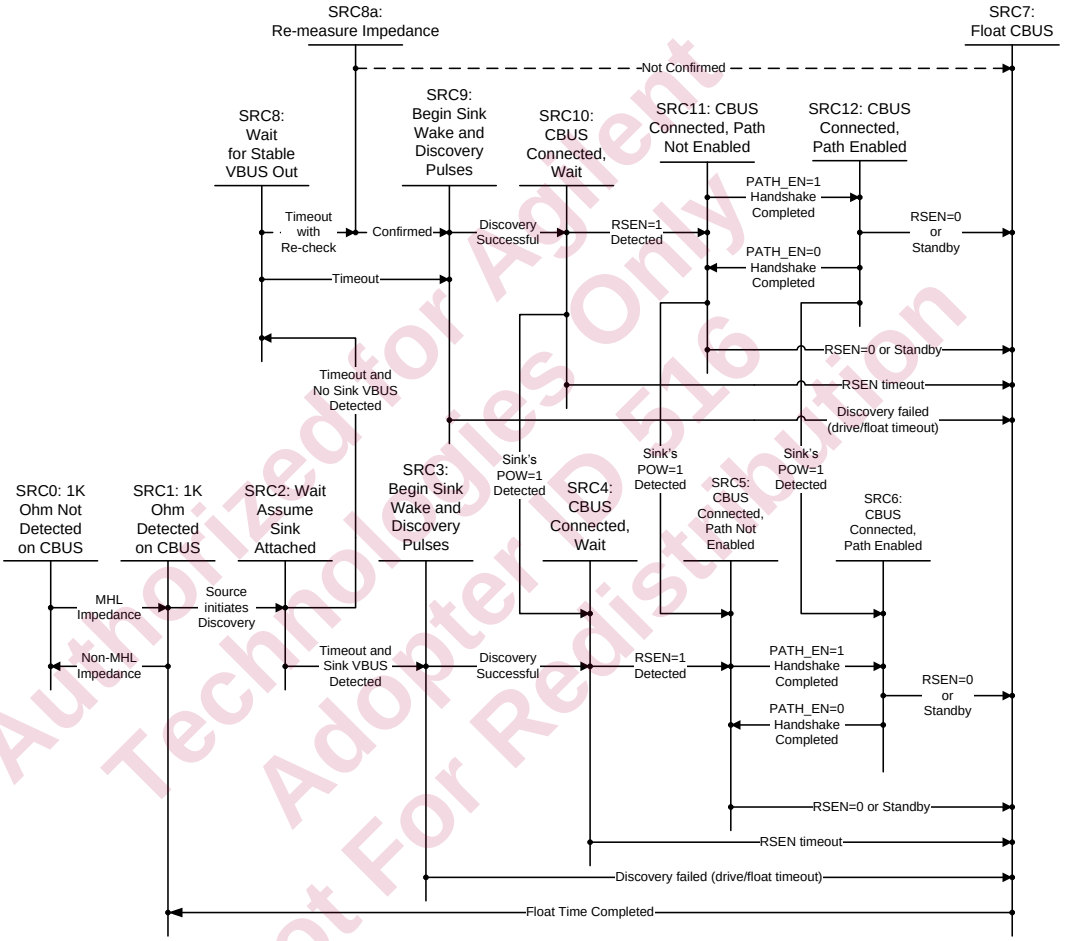


Figure 8-2. Source Discovery and Disconnect State Diagram

Table 8-1. MHL Source – Signals During Discovery

State	VBUS Output	Source CBUS Impedance	MHL Pair	Notes
SRC0	Disabled	Impedance checking	Inactive	1
SRC1	Disabled	Impedance checking	Inactive	1
SRC2	Disabled	Meet $I_{CBUS\_LEAK\_SRC}$ specification	Inactive	
SRC3	Disabled	$Z_{CBUS\_SRC\_DISCOVER}$	Inactive	2
SRC4	Disabled	$Z_{CBUS\_SRC\_ON}$	Inactive	3
SRC5	Disabled	$Z_{CBUS\_SRC\_ON}$	Inactive	
SRC6	Disabled	$Z_{CBUS\_SRC\_ON}$	Active	4
SRC7	Disabled	Meet $I_{CBUS\_LEAK\_SRC}$ specification	Inactive	

State	VBUS Output	Source CBUS Impedance	MHL Pair	Notes
SRC8	Active	Meet $I_{CBUS\_LEAK\_SRC}$ specification	Inactive	
SRC8a	Active	Impedance checking	Inactive	
SRC9	Active	$Z_{CBUS\_SRC\_DISCOVER}$	Inactive	2
SRC10	Active	$Z_{CBUS\_SRC\_ON}$	Inactive	3
SRC11	Active	$Z_{CBUS\_SRC\_ON}$	Inactive	
SRC12	Active	$Z_{CBUS\_SRC\_ON}$	Active	4

Notes on Table 8-1:

1. A device which shares the connector with another protocol may use state SRC0 and/or SRC1 to check for attachment of a device using that other protocol, during which time VBUS output and MHL pair may be activated. See Section 8.4.1.
2. See Section 8.4.3.
3. See Section 8.4.4.
4. MHL pair 'Active' condition depends on state of HPD, as explained in Section 7.6.1.

Table 8-2. MHL States and Transitions from Source During Discovery

	State or Transition Description	Notes
SRC0	Unknown impedance on CBUS. VBUS disabled. MHL pair inactive.	1
SRC0-SRC1	Source measures CBUS impedance as $Z_{CBUS\_SINK\_DISCOVER}$.	
SRC1	Source sees $Z_{CBUS\_SINK\_DISCOVER}$ ohms. VBUS disabled. MHL pair inactive.	1
SRC1-SRC0	Source sees CBUS impedance change to non-MHL value.	
SRC1-SRC2	A Source shall initiate discovery within $T_{SRC\_VBUS\_CBUS\_STABLE}${max} when CBUS impedance changes to $Z_{CBUS\_SINK\_DISCOVER}$ from a state of $I_{CBUS\_LEAK\_SINK}$ after being in impedance state of $I_{CBUS\_LEAK\_SINK}$ for at least $T_{SINK\_CBUS\_FLOAT}${min} (arrives at SRC1 from SRC0 based on this condition) See Sink transition SINK5→SINK1 in Section 8.2.2. Otherwise Source may choose to initiate discovery (arrives at SRC1 from SRC7).	
SRC2	Source waits ($T_{SRC\_VBUS\_CBUS\_STABLE}$) for stable VBUS input from Sink or Dongle. Source VBUS is disabled. Source MHL pair is inactive. CBUS is set to impedance $I_{CBUS\_LEAK\_SRC}$.	
SRC2-SRC3	Wait completed. Source detects active VBUS from Sink or Dongle.	
SRC2-SRC8	Wait completed. Source detects no active VBUS from Sink or Dongle. Source begins to drive VBUS output.	
SRC3	Source changes impedance on CBUS to $Z_{CBUS\_SRC\_DISCOVER}$. Source begins sending wake sequence and discovery pulses. Source VBUS output disabled. MHL pair inactive.	
SRC3-SRC4	Source recognizes response from Sink or Dongle, completes wake and discovery and transitions to connected state.	
SRC3-SRC7	Source discover pulse count exceeds maximum, indicating no responsive Sink or Dongle is detected.	3

	State or Transition Description	Notes
SRC4	Source changes impedance on CBUS to $Z_{CBUS\_SRC\_ON}$. Source waits T_{ARB_SRC} before being able to act as initiator on CBUS. Transitions on RxSense shall be deglitched per $T_{SRC\_RXSENSE\_DEGLITCH}$. Source checks until RxSense becomes active, or until $T_{SRC\_RXSENSE\_CHK}$ expires. Source VBUS disabled. MHL pair inactive.	
SRC4-SRC5	Source detects active RxSense.	
SRC4-SRC7	$T_{SRC\_RXSENSE\_CHK}$ wait time completes without detecting active RxSense.	
SRC5	Source is connected without MHL+/MHL- activated. $Z_{CBUS\_SRC\_ON}$ impedance on CBUS. Source VBUS output disabled.	
SRC5-SRC6	Source acknowledges Sink's PATH_EN{Sink}=1 by sending PATH_EN{Source}=1.See Section 7.9.3.2.	
SRC5-SRC7	Source detects inactive RxSense or Source decides to change to Standby mode.	
SRC6	Source is connected with enabled content output path. MHL pair is activated. $Z_{CBUS\_SRC\_ON}$ impedance on CBUS. VBUS output disabled.	
SRC6-SRC5	Source sends PATH_EN{Source}=0 to Sink when Source's TMDS is disabled. See Section 7.9.3.2.	
SRC6-SRC7	Source detects inactive RxSense or Source decides to change to Standby mode.	
SRC7	Source changes CBUS to high-impedance ($I_{CBUS\_LEAK\_SRC}$) and waits at least $T_{SRC\_CBUS\_FLOAT}$ for CBUS impedance to stabilize. Source disables the content output on the MHL pair within $T_{SRC\_CBUS\_TMDS\_DIS}$ if content output on MHL pair was active.	
SRC7-SRC1	Source returns to monitoring impedance on CBUS.	
SRC8	Source VBUS output is enabled; wait $T_{SRC\_VBUS\_OUT\_TO\_STABLE}$ for VBUS output to stabilize. MHL pair inactive.	
SRC8-SRC8a	After wait $T_{SRC\_VBUS\_OUT\_TO\_STABLE}$ exceeded, Source may re-check impedance on CBUS with its VBUS output enabled. MHL pair inactive. CBUS impedance set to $Z_{SRC\_CBUS\_DISCOVER}$.	
SRC8a	Source may re-measure impedance on CBUS. This state is optionally implemented.	
SRC8a-SRC7	CBUS impedance is confirmed not to be $Z_{CBUS\_SINK\_DISCOVER}$.	
SRC8a-SRC9	CBUS Impedance confirmed to be $Z_{CBUS\_SINK\_DISCOVER}$.	
SRC8-SRC9	$T_{SRC\_VBUS\_OUT\_TO\_STABLE}$ wait time exceeded.	
SRC9	Source impedance on CBUS is $Z_{CBUS\_SRC\_DISCOVER}$. Source begins sending wake sequence and discovery pulses. Source VBUS output enabled. MHL pair inactive.	
SRC9-SRC10	Source recognizes response from Sink or Dongle, completes wake and discovery, and transitions to connected state.	
SRC9-SRC7	Source discover pulse count exceeds maximum, indicating no responsive Sink or Dongle is detected.	3

	State or Transition Description	Notes
SRC10	Source changes impedance on CBUS to $Z_{CBUS\_SRC\_ON}$. Source waits T_{ARB_SRC} before being able to act as initiator on CBUS. Transitions on RxSense shall be deglitched per $T_{SRC\_RXSENSE\_DEGLITCH}$. Source checks until RxSense becomes active, or until $T_{SRC\_RXSENSE\_CHK}$ expires. Source VBUS enabled. MHL pair inactive.	
SRC10-SRC7	$T_{SRC\_RXSENSE\_CHK}$ wait time completes without detecting active RxSense.	
SRC10-SRC4	Source detects POW=1 from Sink's capability register. Source stops driving VBUS output.	2
SRC10-SRC11	Source detects active RxSense.	
SRC11	Source is connected without MHL+/MHL- activated. CBUS impedance is $Z_{CBUS\_SRC\_ON}$. Source VBUS output is enabled.	
SRC11-SRC5	Source detects POW=1 from Sink's capability register. Source stops driving VBUS output.	2
SRC11-SRC12	Source acknowledges Sink's PATH_EN{Sink}=1 by sending PATH_EN{Source}=1. See Section 7.9.3.2.	
SRC11-SRC7	Source detects inactive RxSense or Source disconnects, such as with a change to Standby mode.	
SRC12	Source is connected with enabled content output path. MHL pair activated. CBUS impedance is $Z_{CBUS\_SRC\_ON}$. Source VBUS output enabled.	
SRC12-SRC6	Source detects POW=1 from Sink's capability register. Source stops driving VBUS output.	2
SRC12-SRC7	Source detects inactive RxSense or Source disconnects, such as with a change to Standby mode.	
SRC12-SRC11	Source sends PATH_EN{Source}=0 to Sink when Source's TMDS is disabled. See Section 7.9.3.2.	

Notes on Table 8-2:

1. A device which shares the connector with another protocol may use state SRC0 and/or SRC1 to check for attachment of a device using that other protocol, during which time VBUS output and MHL pair may be activated. During Source's measurement of impedance on CBUS, Source may float CBUS (i.e. go to high-z impedance) for at least 50 msec (same as floating period before going back to SRC1) between repeated measurements of impedance on CBUS.
2. A Source which detects that the Sink is outputting VBUS, shall turn off the Source's VBUS output. Source determines the Sink's state by reading the Sink's Device Capability register POW bit.
3. On failed completion of the wake sequence and discovery pulse process, a Source should reattempt discovery by re-checking impedance (state SRC1) for proper CBUS discovery impedance, then proceed through the states to re-try the discovery pulse process (state SRC3 or SRC9). This loop of states should be repeated – reattempting discovery using wake and discovery pulse sequences – multiple times for as long as the Source requirement for discovery persists.

8.2.1.2 MHL Source with No VBUS Output

An MHL Source which cannot output VBUS shall process events received from devices connected to it according to the state transitions and timings shown in Figure 8-3, and further defined in Table 8-3 and Table 8-4.

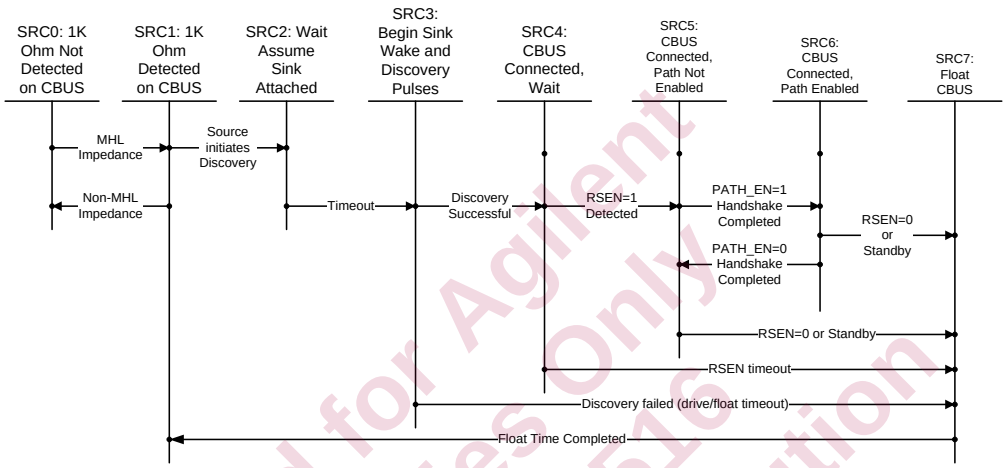


Figure 8-3. Source without VBUS Output – Discovery and Disconnect State Diagram

Table 8-3. MHL Source without VBUS Output – Signals During Discovery

State	VBUS Output	Source CBUS Impedance	MHL Pair	Notes
SRC0	Disabled	Impedance checking	Inactive	1
SRC1	Disabled	Impedance checking	Inactive	1
SRC2	Disabled	Meet $I_{CBUS\_LEAK\_SRC}$ specification	Inactive	
SRC3	Disabled	$Z_{CBUS\_SRC\_DISCOVER}$	Inactive	2
SRC4	Disabled	$Z_{CBUS\_SRC\_ON}$	Inactive	3
SRC5	Disabled	$Z_{CBUS\_SRC\_ON}$	Inactive	
SRC6	Disabled	$Z_{CBUS\_SRC\_ON}$	Active	4
SRC7	Disabled	Meet $I_{CBUS\_LEAK\_SRC}$ specification	Inactive	

Notes on Table 8-3:

1. A device which shares the connector with another protocol may use state SRC0 and/or SRC1 to check for attachment of a device using that other protocol, during which time VBUS output and MHL pair may be activated. See Section 8.4.1.
2. See Section 8.4.3.
3. See Section 8.4.4.
4. MHL pair 'Active' condition depends on state of HPD, as explained in Section 7.6.1.

Table 8-4. MHL States and Transitions from Source without VBUS Output - During Discovery

	State or Transition Description	Notes
SRC0	Unknown impedance on CBUS. MHL pair inactive.	1
SRC0-SRC1	Source measures CBUS impedance as $Z_{CBUS\_SINK\_DISCOVER}$.	
SRC1	Source sees $Z_{CBUS\_SINK\_DISCOVER}$ ohms. MHL pair inactive.	1
SRC1-SRC0	Source sees CBUS impedance change to non-MHL value.	
SRC1-SRC2	A Source shall initiate discovery within $T_{SRC\_VBUS\_CBUS\_STABLE}${max} when CBUS impedance changes to $Z_{CBUS\_SINK\_DISCOVER}$ from a state of $I_{CBUS\_LEAK\_SINK}$ after being in impedance state of $I_{CBUS\_LEAK\_SINK}$ for at least $T_{SINK\_CBUS\_FLOAT}${min} (arrives at SRC1 from SRC0 based on this condition).See Sink transition SINK5→SINK1 in Section 8.2.2. Otherwise Source may choose to initiate discovery (arrives at SRC1 from SRC7).	
SRC2	Source waits ($T_{SRC\_VBUS\_CBUS\_STABLE}$) for stable VBUS input from Sink or Dongle. Source MHL pair is inactive. CBUS is set to impedance $I_{CBUS\_LEAK\_SRC}$.	
SRC2-SRC3	Wait completed. Source might detect active VBUS from Sink or Dongle.	
SRC3	Source changes impedance on CBUS to $Z_{CBUS\_SRC\_DISCOVER}$. Source begins sending wake sequence and discovery pulses. MHL pair inactive.	
SRC3-SRC4	Source recognizes response from Sink or Dongle, completes wake sequence and discovery process, and transitions to connected state.	
SRC3-SRC7	Source discover pulse count exceeds maximum, indicating no responsive Sink or Dongle is detected.	2
SRC4	Source changes impedance on CBUS to $Z_{CBUS\_SRC\_ON}$. Source waits T_{ARB_SRC} before being able to act as initiator on CBUS. Transitions on RxSense shall be deglitched per $T_{SRC\_RXSENSE\_DEGLITCH}$. Source checks until RxSense becomes active, or until $T_{SRC\_RXSENSE\_CHK}$ expires. Source VBUS disabled. MHL pair inactive.	
SRC4-SRC5	Source detects active RxSense.	
SRC4-SRC7	$T_{SRC\_RXSENSE\_CHK}$ wait time completes without detecting active RxSense.	
SRC5	Source is connected without MHL+/MHL- active. $Z_{CBUS\_SRC\_ON}$ impedance on CBUS.	
SRC5-SRC6	Source acknowledges Sink's PATH_EN{Sink}=1 by sending PATH_EN{Source}=1.See Section 7.9.3.2.	
SRC5-SRC7	Source detects inactive RxSense or Source decides to change to Standby mode.	
SRC6	Source is connected with enabled content output path. MHL pair active. $Z_{CBUS\_SRC\_ON}$ impedance on CBUS.	
SRC6-SRC5	Source sends PATH_EN{Source}=0 to Sink when Source's TMDS is disabled.See Section 7.9.3.2.	
SRC6-SRC7	Source detects inactive RxSense or Source decides to change to Standby mode.	

	State or Transition Description	Notes
SRC7	Source disables the content output on the MHL pair within $T_{SRC\_CBUS\_TMDS\_DIS}$ if content output on MHL pair was active. Source changes CBUS to high-impedance ($I_{CBUS\_LEAK\_SRC}$) and waits at least $T_{SRC\_CBUS\_FLOAT}$ for CBUS impedance to stabilize.	
SRC7-SRC1	Source returns to monitoring impedance on CBUS.	

Notes on Table 8-4:

1. A device which shares the connector with another protocol may use state SRC0 and/or SRC1 to check for attachment of a device using that other protocol, during which time VBUS output and MHL pair may be activated. During Source's measurement of impedance on CBUS, Source may float CBUS (i.e. go to high-z impedance) for at least 50 msec (same as floating period before going back to SRC1) between repeated measurements of impedance on CBUS.
2. On failed completion of the wake sequence and discovery pulse process, a Source should reattempt discovery by re-checking impedance (SRC1) for proper CBUS discovery impedance then proceeding through the states to retry the discovery pulse process (SR3). This loop of states should be repeated – reattempting discovery using wake and discovery pulse sequences – multiple times for as long as the Source requirement for discovery persists.

8.2.2 Sink Viewpoint

An MHL Sink shall process events received from Sources connected to it according to the state transitions and timings shown in one of Figure 8-4, Figure 8-5, or Figure 8-6; and further defined in Table 8-5 and Table 8-6.

Figure 8-4 is the state diagram for a Sink which recognizes wake pulses when the Sink is in Standby, but which does not output VBUS when the Sink is in Standby.

Figure 8-5 is the state diagram for a Sink which outputs VBUS when the Sink is in Standby, but which does not recognize wake pulses when the Sink is in Standby.

Figure 8-6 is the state diagram for a Sink which outputs VBUS and recognizes wake pulses when the Sink is in Standby.

Coordination of actual video and audio content flow from Source to Sink, and rendering of video and audio on the Sink, are described in Section 8.6.3.

If the device is a DC-Powered Sink and for the case when the device is not attached to an external power source, then Adopter shall indicate the configurations of the device (if any) in which it outputs VBUS to an attached Source and the level of power output provided. This indication shall be made in the printed or electronic user documentation. For example: "This MHL product, when not attached to a charger, outputs 4.5 watts of power.Once the battery level becomes depleted, the power output is turned off."

8.2.2.1 Sink Recognizes Wake Pulses from Standby Condition

Figure 8-4 specifies the required behavior for a Sink device which recognizes wake pulses from the attached Source while the Sink device is in Standby, but which does not output active VBUS while the Sink device is in Standby.

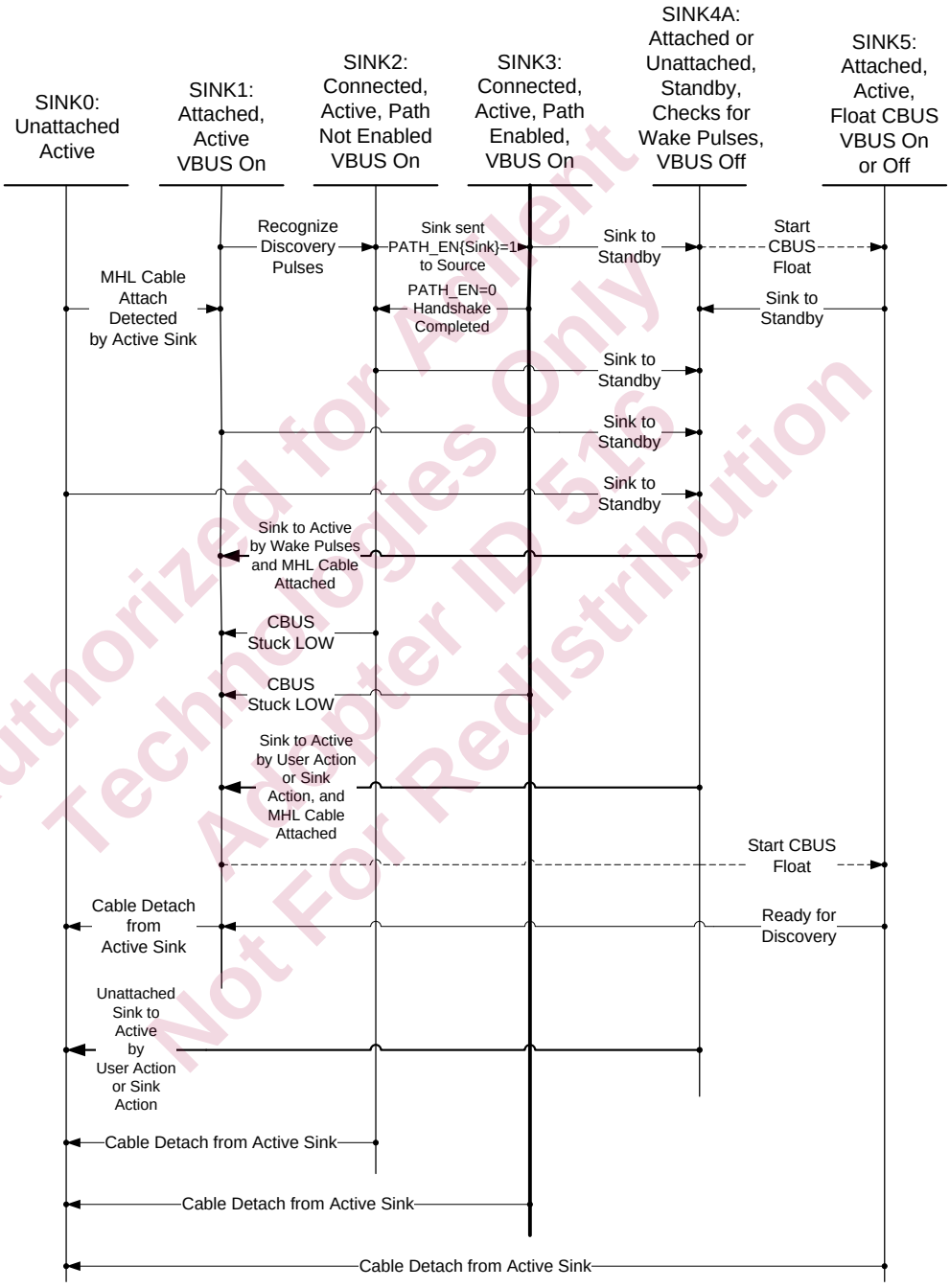


Figure 8-4 Sink Discovery and Disconnect State Diagram #1

8.2.2.2 Sink Outputs VBUS from Standby Condition

Figure 8-5 specifies the required behavior for a Sink device which outputs active VBUS to the attached Source based on the Sink sensing an attached cable, while the Sink device is in Standby, but which does not watch for wake pulses while the Sink device is in Standby.

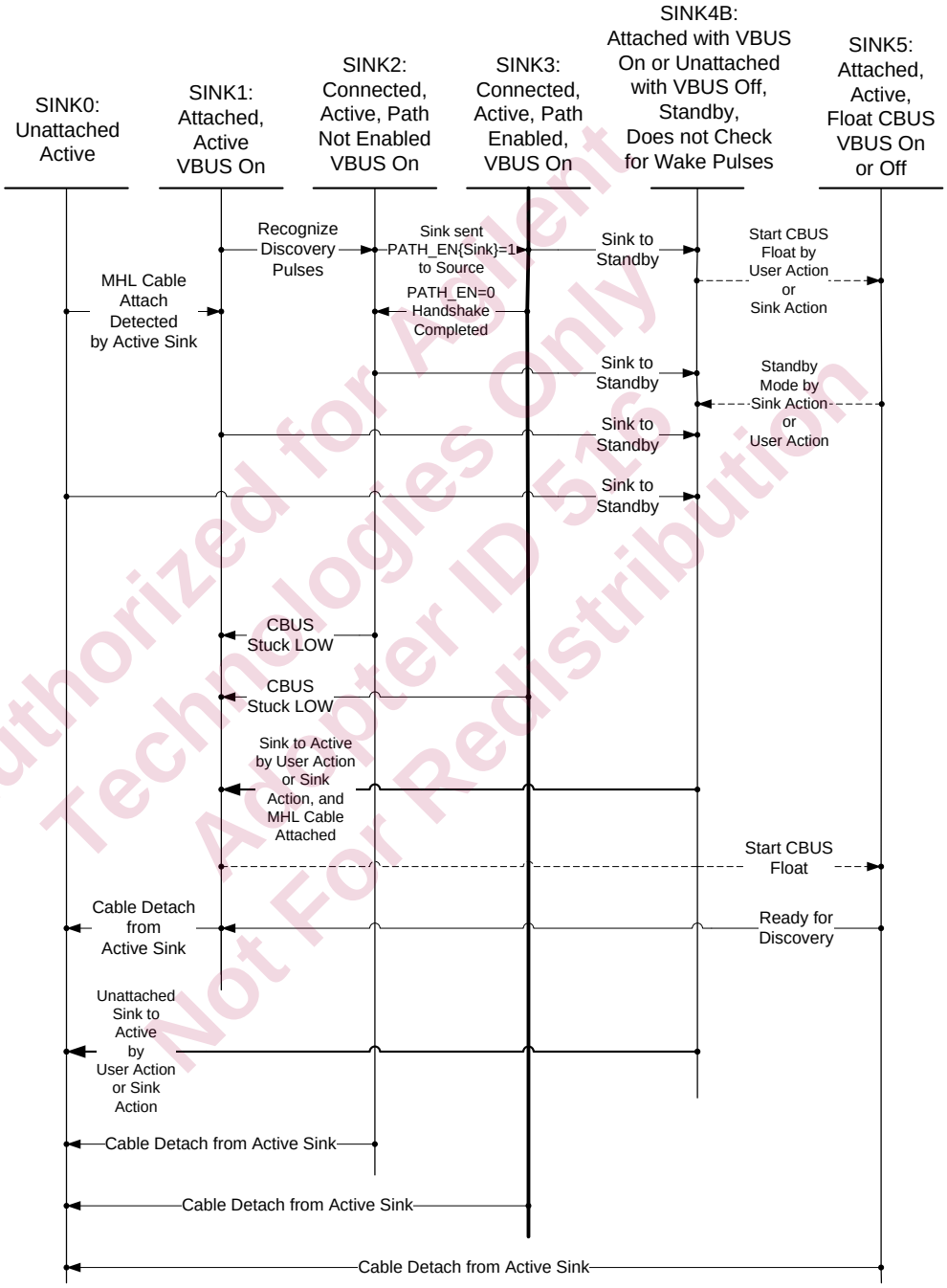


Figure 8-5. Sink Discovery and Disconnect State Diagram #2

8.2.2.3 Sink Outputs VBUS and Recognizes Wake Pulses from Standby Condition

Figure 8-6 specifies the required behavior for a Sink device which recognizes wake pulses from the attached Source while the Sink device is in Standby, and which outputs active VBUS to the attached Source, based on the Sink sensing an attached cable, while the Sink device is in Standby.

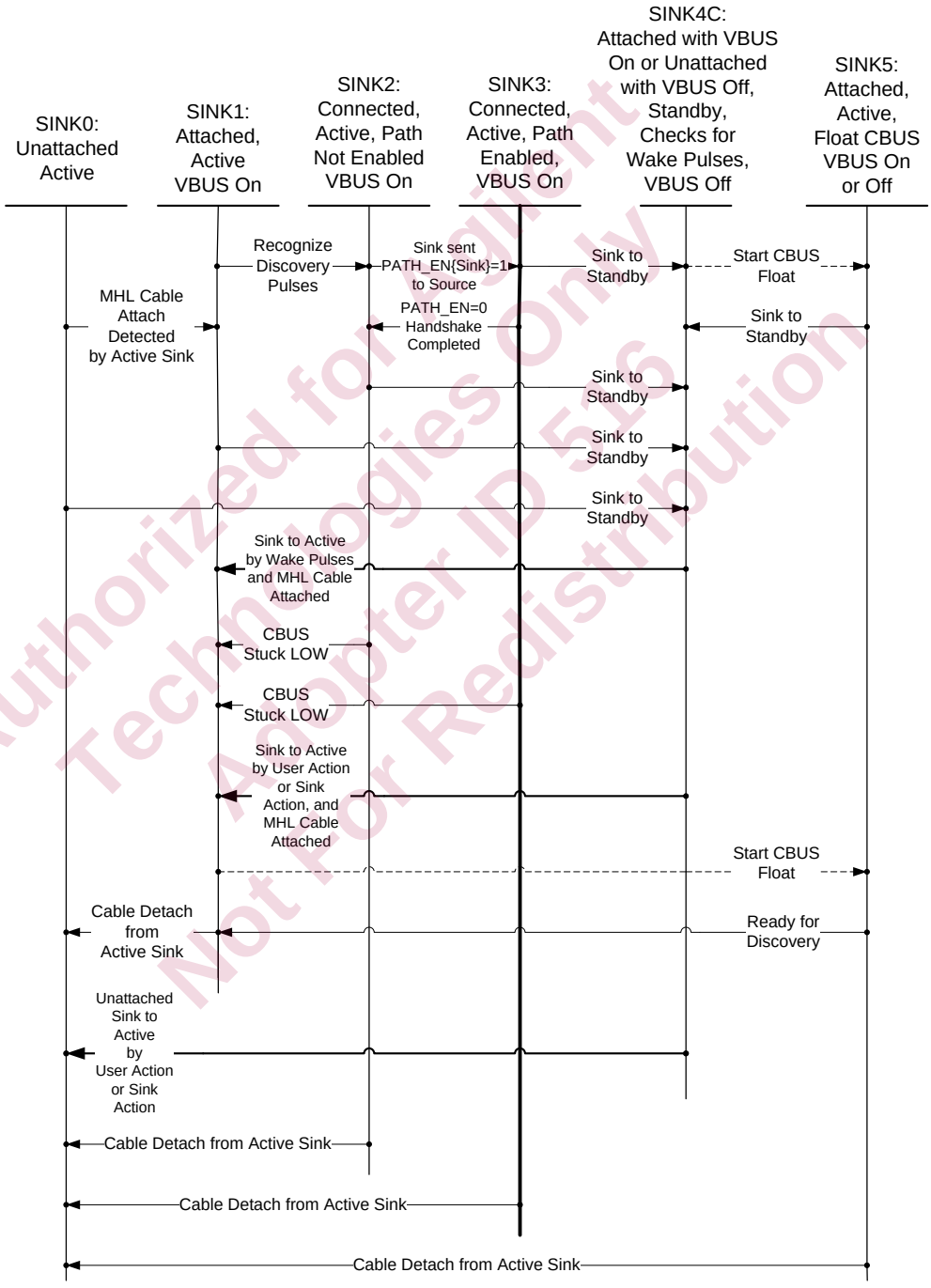


Figure 8-6. Sink Discovery and Disconnect State Diagram #3

Table 8-5. MHL Signals from Sink During Discovery

State	Sink	MHL Cable	VBUS Output when Attached	Sink CBUS Impedance when Attached	RxSense when Attached	Notes
SINK0	Active	Unattached				3
SINK1	Active	Attached	Active	$Z_{CBUS\_SINK\_DISCOVER}$	$I_{RXSENSE\_LEAK}$	1
SINK2	Active	Attached	Active	$Z_{CBUS\_SINK\_ON}$	$Z_{RXSENSE\_TERM}$ or $Z_{RXSENSE\_TMDS}$	1
SINK3	Active	Attached	Active	$Z_{CBUS\_SINK\_ON}$	$Z_{RXSENSE\_TMDS}$	1
SINK4A	Standby	Attached or Unattached	Inactive	$Z_{CBUS\_SINK\_DISCOVER}$	$I_{RXSENSE\_LEAK}$	2, 3
SINK4B	Standby	Attached or Unattached	Active	$Z_{CBUS\_SINK\_DISCOVER}$ or meet $I_{CBUS\_LEAK\_SINK}$ specification	$I_{RXSENSE\_LEAK}$	1, 2, 3, 4
SINK4C	Standby	Attached or Unattached	Active	$Z_{CBUS\_SINK\_DISCOVER}$	$I_{RXSENSE\_LEAK}$	1, 2, 3
SINK5	Active	Attached	Active or Inactive	Meet $I_{CBUS\_LEAK\_SINK}$ specification	$I_{RXSENSE\_LEAK}$	1

Notes on Table 8-5:

1. MHL Sinks which share an MHL input port with other protocols may be required to control VBUS output according to cable detect. Refer to Section 12.
2. Sinks which implement only an Active state may not implement the SINK4A, SINK4B, or SINK4C state.
3. With no attached cable, the impedances on CBUS and MHL+/MHL- pins are not defined, the VBUS state is undefined, and checking for wake pulses is undefined.
4. Sinks which implement SINK4B in Standby may meet the impedance of $I_{CBUS\_LEAK\_SINK}\{max\}$ specification on CBUS during the SINK4B state.

Table 8-6. MHL Sink States and Transitions During Discovery with VBUS from Sink

State or Arc		State or Transition Description	Notes
SINK0		Cable is not attached. Sink is in Unattached, Active state. This state is implemented when necessary to meet the requirements of Section 8.1.4.	
SINK0-SINK1		Cable attachment is detected and Sink power state is Active. Sink moves to "Attached, Active" state.	
SINK0-SINK4	SINK0-SINK4A	Sink moves from Active to Standby power state. Sink makes this transition when Sink power changes from Active to Standby.	
	SINK0-SINK4B	Sink moves from Active to Standby power state. Sink makes this transition when Sink power changes from Active to Standby.	
	SINK0-SINK4C	Sink moves from Active to Standby power state. Sink makes this transition when Sink power changes from Active to Standby.	
SINK1		Sink is in Active power state, and is ready for discovery and checks for discovery pulses from the MHL Source.	
SINK1-SINK0		Cable detach is detected. Return to Unattached, Active state.	
SINK1-SINK2		Sink recognizes sufficient discovery pulses, moves to the connected state.	
SINK1-SINK4	SINK1-SINK4A	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
	SINK1-SINK4B	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
	SINK1-SINK4C	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
SINK1-SINK5		Sink may float CBUS, and subsequently transition back to SINK1 state to prompt Source to re-initiate the discovery process. This transition is optionally taken, based on the requirement of the Sink.	
SINK2		Sink is in Active power state, and is connected to an MHL Source. The path for TMDS signaling on the MHL pair is inactive. See Section 7.9.3.2.	
SINK2-SINK0		Cable detach is detected. Return to Unattached, Active state.	
SINK2-SINK1		Sink returns to attached (but not connected) state when detecting CBUS low for $T_{SINK\_CBUS\_DISCONN}$.	
SINK2-SINK3		Source and Sink negotiate activation of the MHL pair based on using MSC protocols. See Section 7.9.3.2.	

State or Arc		State or Transition Description	Notes
SINK2-SINK4	SINK2-SINK4A	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
	SINK2-SINK4B	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
	SINK2-SINK4C	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
SINK3		Sink is in Active power state, and is connected to an MHL Source. MHL pair is active.	
SINK3-SINK0		Cable detach is detected. Return to Unattached, Active state.	
SINK3-SINK1		Sink returns to Attached, Active (but not connected) state when detecting CBUS low for $T_{SINK\_CBUS\_DISCONN}$.	1
SINK3-SINK2		Sink and Source negotiate deactivation of the MHL pair using MSC protocols. See Section 7.9.3.2.	
SINK3-SINK4	SINK3-SINK4A	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
	SINK3-SINK4B	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
	SINK3-SINK4C	Sink moves from Active to Standby power state. Sink makes this transition when Sink power state changes from Active to Standby.	
SINK4A		Sink is in Standby power state and checks for the wake pulse sequence from MHL Source, but does not output active VBUS.	2
SINK4A-SINK0		Sink moves from Standby to Active power state. Sink makes this transition when MHL Cable is not attached, and Sink power state changes from Standby to Active (by user action or by Sink action). Return to Unattached, Active state.	
SINK4A-SINK1		Sink moves from Standby to Active power state. Sink makes this transition when MHL Cable is attached and either the Sink requirement changes from Standby to Active (by user action or Sink action), or on detection of the wake pulse sequence from MHL Source. Return to Attached, Active state.	1, 3
SINK4A-SINK5		Sink may float CBUS and subsequently transition back to SINK4A or SINK1 state to prompt the Source to re-initiate discovery sequence. This transition is optionally taken, depending on the requirement of the Sink.	
SINK4B		Sink is in Standby power state and outputs active VBUS to the attached Source (according to Table 8-5), but does not check for the wake pulse sequence from MHL Source.	

State or Arc		State or Transition Description	Notes
SINK4B-SINK0		Sink moves from Standby to Active power state. Sink makes this transition when Sink power state changes from Standby to Active by user action or by Sink action, and MHL Cable is not attached. Return to Unattached, Active state.	
SINK4B-SINK1		Sink moves from Standby to Active power state. Sink makes this transition when MHL Cable is attached and either the Sink requirement changes from Standby to Active, or on detection of the wake pulse sequence from MHL Source.	1, 3
SINK4B-SINK5		Sink may float CBUS and subsequently transition back to SINK4B or SINK1 state to prompt the Source to re-initiate discovery sequence. This transition is optionally taken, depending on the requirement of the Sink.	
SINK4C		Sink is in Standby power state and outputs active VBUS (according to Table 8-5) and checks for wake pulse from the attached MHL Source.	2
SINK4C-SINK0		Sink moves from Standby to Active power state. Sink makes this transition when Sink power state changes from Standby to Active by user action or by Sink action, and MHL Cable is not attached. Return to Unattached, Active state.	
SINK4C-SINK1		Sink moves from Standby to Active power state. Sink makes this transition when MHL Cable is attached and either the Sink requirement changes from Standby to Active (by user action or Sink action), or on detection of the wake pulse sequence from MHL Source. Return to Attached, Active state.	1, 3
SINK4C-SINK5		Sink may float CBUS and subsequently transition back to SINK4C or SINK1 state to prompt the Source to re-initiate discovery sequence. This transition is optionally taken, depending on the requirement of the Sink.	
SINK5		Sink shall float CBUS for a period of at least $T_{SINK\_CBUS\_FLOAT}$.	
SINK5-SINK0		Cable detach is detected. Return to Unattached, Active state.	
SINK5-SINK1		Sink returns to Attached, Active state after floating CBUS. The transition back to SINK1 state is to prompt the Source to re-initiate discovery sequence.	1, 4
SINK5-SINK4	SINK5-SINK4A	Sink returns to Standby state after floating CBUS. The transition back to SINK4A state is to prompt the Source to re-initiate discovery sequence. Transition may be from user action or sink action.	4
	SINK5-SINK4B	Sink returns to Standby state after floating CBUS. The transition back to SINK4B state is to prompt the Source to re-initiate discovery sequence. Transition may be from user action or sink action.	4
	SINK5-SINK4C	Sink returns to Standby state after floating CBUS. The transition back to SINK4C state is to prompt the Source to re-initiate discovery sequence. Transition may be from user action or sink action.	4

Notes on Table 8-6:

1. On transition to the SINK1 "Attached, Active" state, the Sink shall meet the following timing parameters : $T_{SINK\_VBUS\_EN}$ and $T_{SINK\_READY\_TO\_DISCOVER}$, and be capable to meet the following timing parameters on discovery: T_{SINK_CONN}, $T_{SINK\_ARBITRATE}$, and $T_{SINK\_RXSENSE\_EN}$. If unable to meet any of these timings, the Sink shall first transition to the SINK5 state, and subsequently transition to the SINK1 state when it is capable of meeting all these timings.
2. The wake pulse sequence on CBUS shall bring a Sink out of the SINK4 "Standby" state, and the Sink shall transition to SINK1 or to SINK5, according to note 1 above.
3. On transition to SINK1 "Attached, Active" state from SINK4 "Standby" state, if the Sink is unable to confirm an MHL Cable is attached while meeting the timing requirements of note 1, then Sink shall first transition to the SINK5 state, and subsequently transition to the SINK1 state after it has confirmed an MHL Cable is attached and it is also capable of meeting the requirements of note 1.
4. See the SRC1→SRC2 transition in Table 8-2 and Table 8-4.

8.2.3 Dongle Viewpoint

An MHL Dongle shall process events received from MHL Source devices connected to it according to the state transitions and timings shown in Figure 8-7, and further defined in Table 8-7 and Table 8-8.

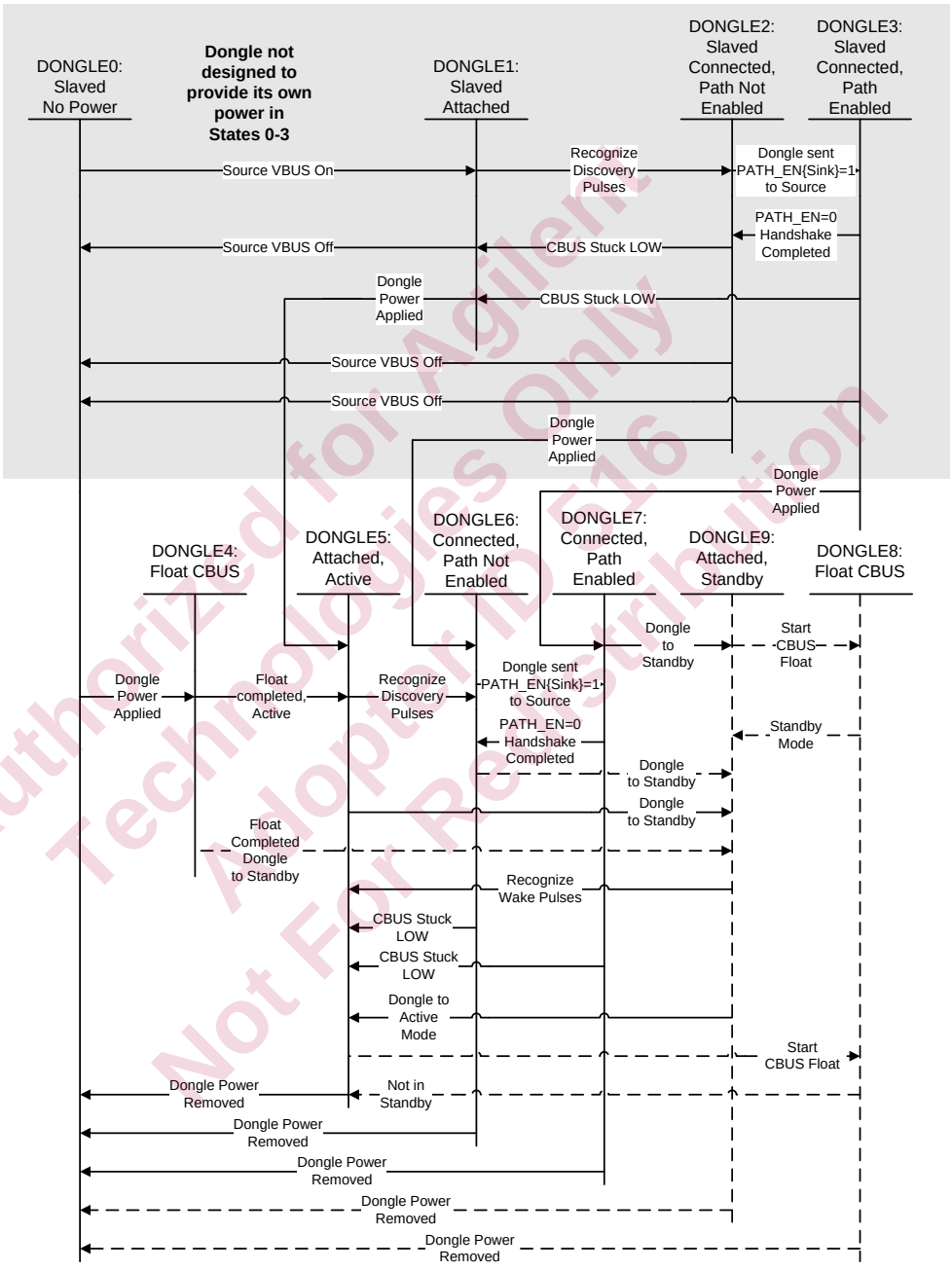


Figure 8-7. Dongle Discovery and Disconnect State Diagram

Table 8-7. MHL Signals from Dongle During Discovery

State	Dongle Power	Dongle VBUS Output	Dongle CBUS Impedance	RxSense	Notes
DONGLE0	No	Inactive	$Z_{CBUS\_SINK\_DISCOVER}$	$I_{RXSENSE\_LEAK}$	
DONGLE1	No	Inactive	$Z_{CBUS\_SINK\_DISCOVER}$	$I_{RXSENSE\_LEAK}$	
DONGLE2	No	Inactive	$Z_{CBUS\_SINK\_ON}$	$Z_{RXSENSE\_TERM}$ or $Z_{RXSENSE\_TMDS}$	
DONGLE3	No	Inactive	$Z_{CBUS\_SINK\_ON}$	$Z_{RXSENSE\_TMDS}$	
DONGLE4	Yes	Active	Meet $I_{CBUS\_LEAK\_SINK}$ specification	$I_{RXSENSE\_LEAK}$	
DONGLE5	Yes	Active	$Z_{CBUS\_SINK\_DISCOVER}$	$I_{RXSENSE\_LEAK}$	
DONGLE6	Yes	Active	$Z_{CBUS\_SINK\_ON}$	$Z_{RXSENSE\_TERM}$ or $Z_{RXSENSE\_TMDS}$	
DONGLE7	Yes	Active	$Z_{CBUS\_SINK\_ON}$	$Z_{RXSENSE\_TMDS}$	
DONGLE8	Yes	Active or Inactive	Meet $I_{CBUS\_LEAK\_SINK}$ specification	$I_{RXSENSE\_LEAK}$	
DONGLE9	Yes	Inactive	$Z_{CBUS\_SINK\_DISCOVER}$	$I_{RXSENSE\_LEAK}$	

Table 8-8. MHL Dongle States and Transitions During Discovery with VBUS from Dongle

	State or Transition Description	Notes
DONGLE0	Dongle does not have its own power and is not supplied with power over VBUS from the Source. Impedance on CBUS is $Z_{CBUS\_SINK\_DISCOVER}$ in this state to allow Source to recognize when the Dongle is attached.	
DONGLE0-DONGLE1	Source asserts VBUS and powers Dongle. Dongle shall ensure possible disturbance to $Z_{CBUS\_SINK\_DISCOVER}$ during power transition settles within $T_{SRC\_VBUS\_OUT\_TO\_STABLE}$.	
DONGLE0-DONGLE4	Dongle's own power is applied.	
DONGLE1	Dongle is ready for discovery and checks for discovery pulses from MHL Source.	
DONGLE1-DONGLE0	Source deasserts VBUS. Dongle returns to unpowered state.	
DONGLE1-DONGLE2	Dongle recognizes sufficient discovery pulses, and moves to connected state.	
DONGLE1-DONGLE5	Dongle's own power is applied. Dongle reacts by asserting POW=1 in the Dongle's Capability Register, and writing the interrupt DCAP_CHG to the Source.	1
DONGLE2	Dongle is connected to an MHL Source. MHL pair is inactive.	
DONGLE2-DONGLE0	Source deasserts VBUS. Dongle returns to unpowered state.	

	State or Transition Description	Notes
DONGLE2-DONGLE1	Dongle returns to attached (but not connected) state when detecting CBUS low for $T_{SINK\_CBUS\_DISCONN}$.	
DONGLE2-DONGLE3	Dongle and Source negotiate activation of the MHL pair using MSC protocols. See Section 7.9.3.2.	
DONGLE2-DONGLE6	Dongle's own power is applied. Dongle notifies Source by asserting POW=1 in the Dongle's Capability Register, and writing the interrupt DCAP_CHG to the Source.	
DONGLE3	Dongle is connected to an MHL Source. MHL pair is active.	
DONGLE3-DONGLE0	Source deasserts VBUS. Dongle returns to unpowered state.	
DONGLE3-DONGLE1	Dongle returns to attached (but not connected) state when detecting CBUS low for $T_{SINK\_CBUS\_DISCONN}$.	
DONGLE3-DONGLE2	Dongle and Source negotiate deactivation of the MHL pair using MSC protocols. See Section 7.9.3.2.	
DONGLE3-DONGLE7	Dongle's own power is applied. Dongle notifies Source by asserting POW=1 in the Dongle's Capability Register, and writing the interrupt DCAP_CHG to the Source.	
DONGLE4	Dongle floats CBUS to prompt Source that Dongle is attached. Dongle shall float CBUS for a period of at least $T_{SINK\_CBUS\_FLOAT}$.	
DONGLE4-DONGLE5	Dongle enters attached state after floating CBUS.	1
DONGLE4-DONGLE9	Dongle enters Standby state after floating CBUS. This transition is optionally implemented.	
DONGLE5	Dongle is ready for discovery and checks for discovery pulses from MHL Source.	
DONGLE5-DONGLE0	Dongle's own power is removed. Dongle returns to unpowered state.	
DONGLE5-DONGLE6	Dongle recognizes sufficient discovery pulse and moves to "Connected, Content Off" state.	
DONGLE5-DONGLE8	Dongle may float CBUS and subsequently transition back to DONGLE5 to prompt the Source to re-initiate the discovery sequence. This transition is optionally implemented.	
DONGLE5-DONGLE9	Dongle moves from Active to Standby state. Dongle makes this transition when Dongle power state changes from Active to Standby. This transition is optionally implemented.	
DONGLE6	Dongle is connected to MHL Source. MHL pair is inactive.	
DONGLE6-DONGLE0	Dongle's own power is removed. Dongle returns to unpowered state.	
DONGLE6-DONGLE5	Dongle returns to attached (but not connected) state when detecting CBUS low for $T_{SINK\_CBUS\_DISCONN}$.	1
DONGLE6-DONGLE7	Dongle and Source negotiate activation of the MHL pair using MSC protocols. See Section 7.9.3.2.	

	State or Transition Description	Notes
DONGLE6-DONGLE9	Dongle moves from "Connected, Content Off" to Standby power state. Dongle makes this transition when Dongle power state changes from Active to Standby.	
DONGLE7	Dongle is connected to MHL Source. MHL pair is active.	
DONGLE7-DONGLE0	Dongle's own power is removed. Dongle returns to unpowered state.	
DONGLE7-DONGLE5	Dongle returns to attached (but not connected) state when detecting CBUS low for $T_{SINK\_CBUS\_DISCONN}$.	1
DONGLE7-DONGLE6	Dongle and Source negotiate deactivation of the MHL pair using MSC protocols. See Section 7.9.3.2.	
DONGLE7-DONGLE9	Dongle moves from "Connected, Content On" to Standby power state. Dongle makes this transition when Dongle power state changes from Active to Standby.	
DONGLE8	Dongle shall float CBUS for a period of at least $T_{SINK\_CBUS\_FLOAT}$. This state is optionally implemented.	
DONGLE8-DONGLE5	Dongle returns to attached state after floating CBUS, the transition back to DONGLE5 state is to prompt the Source to re-initiate discovery sequence. This transition is optionally taken, depending on the requirement of the Dongle.	1
DONGLE8-DONGLE9	Dongle returns to Standby state after floating CBUS, the transition back to DONGLE9 state is to prompt the Source to re-initiate discovery sequence. This transition is optionally taken, depending on the requirement of the Dongle.	
DONGLE9	Dongle is in Standby power state and checks for the wake pulse sequence from MHL Source. This state is optionally implemented.	2
DONGLE9-DONGLE5	Dongle moves from Standby to Active power state. Dongle makes this transition when either the Dongle requirement changes from Standby to Active, or on detection of the wake pulse sequence from MHL Source.	1
DONGLE9-DONGLE8	Dongle may float CBUS and subsequently transition back to DONGLE9 or DONGLE5 state to prompt Source to re-initiate the discovery sequence. This transition is optionally taken, based on the requirement of the Dongle.	

Notes on Table 8-8:

1. On transition to the DONGLE5 "Attached, Active" state, the Dongle shall meet the following timing parameter: $T_{SINK\_READY\_TO\_DISCOVER}$, and be capable to meet the following timing parameters on discovery: T_{SINK_CONN}, $T_{SINK\_ARBITRATE}$, and $T_{SINK\_RXSENSE\_EN}$. If unable to meet any of these timings, the Dongle shall first transition to the DONGLE8 state, and subsequently transition to the DONGLE state when it is capable of meeting all these timings.
2. The wake pulse sequence on CBUS shall bring a Dongle out of the DONGLE9 "Standby" state, and the Dongle shall transition to DONGLE5, or to DONGLE8 if necessary until the Dongle is able to meet the timing requirements in note 1 above.

8.3 Discovery Timelines

The timelines in Section 8.3 illustrate the timings involved in several example discovery scenarios. The timings shown are illustrative of the states and transitions specified in the preceding sections.

8.3.1 Source Successfully Discovers Powered Sink

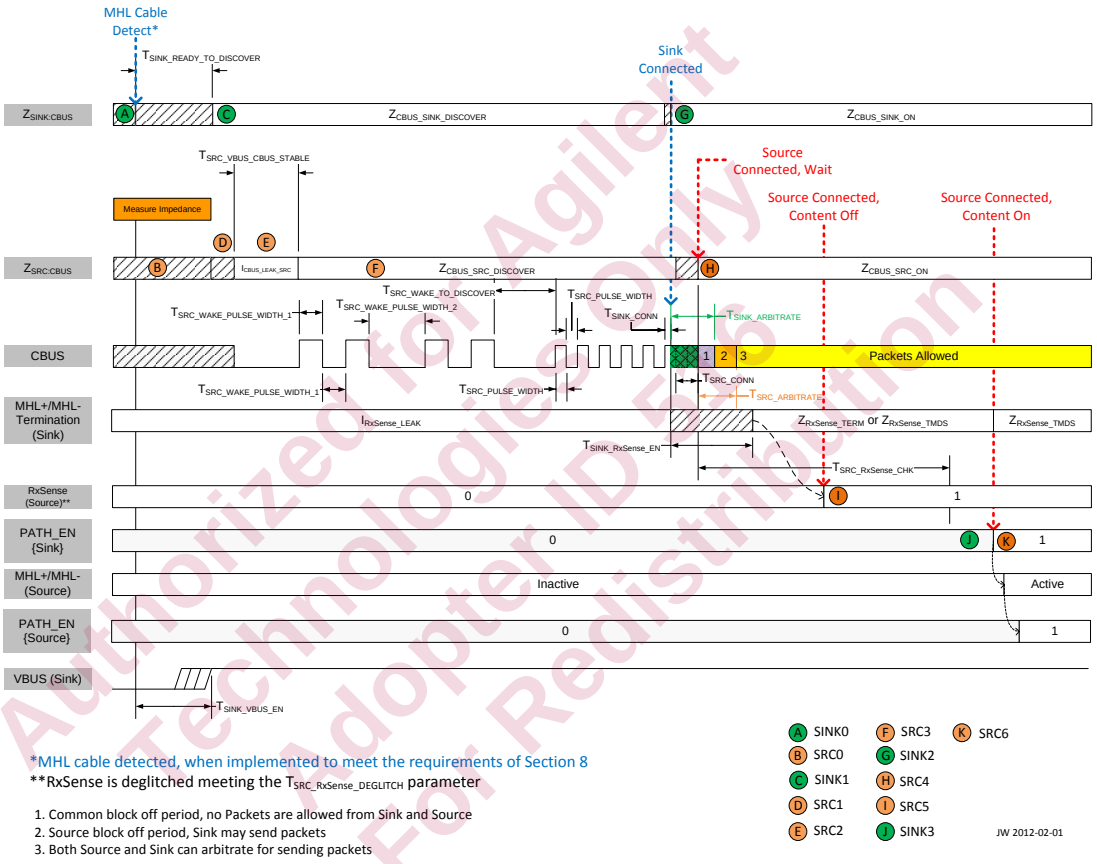


Figure 8-8. Source Successfully Connects to Active Sink – Timing Diagram

8.3.2 Source Successfully Discovers Unpowered Dongle

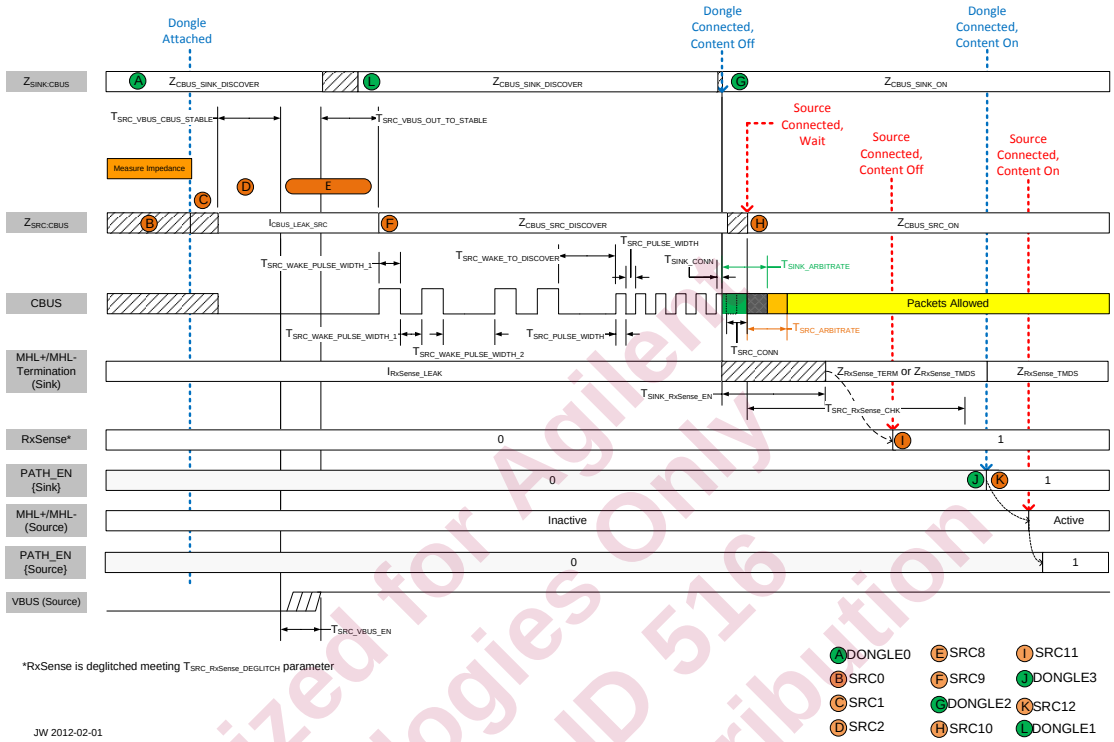


Figure 8-9. Source Successfully Discovers Unpowered Dongle – Timing Diagram

8.3.3 Source Plugs into Standby Sink

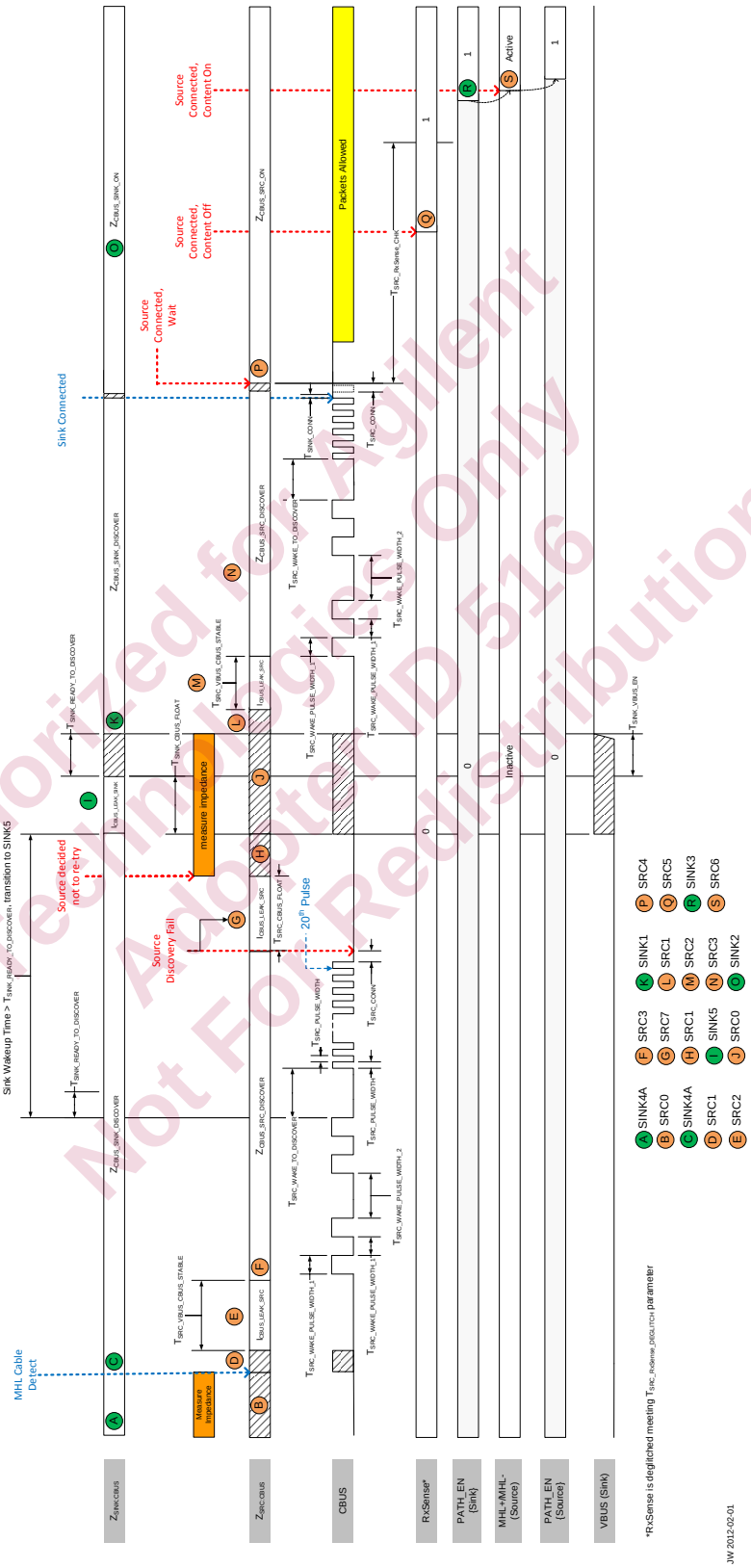


Figure 8-10. Source Plugs into Sink that Recognizes Wake Pulses from Standby – Timing Diagram

8.4 Electrical Discovery Sequence

The diagrams in this section provide a view to some of the steps in the discovery process. Refer to Section 8.2, where these diagrams are referenced in the notes below the state diagram descriptions.

During the discovery process, a HIGH on CBUS shall be a voltage level greater than or equal to V_{IH_CBUS}. A Low on CBUS shall be a voltage level less than or equal to V_{IL_CBUS}.

8.4.1 CBUS Impedance Recognition

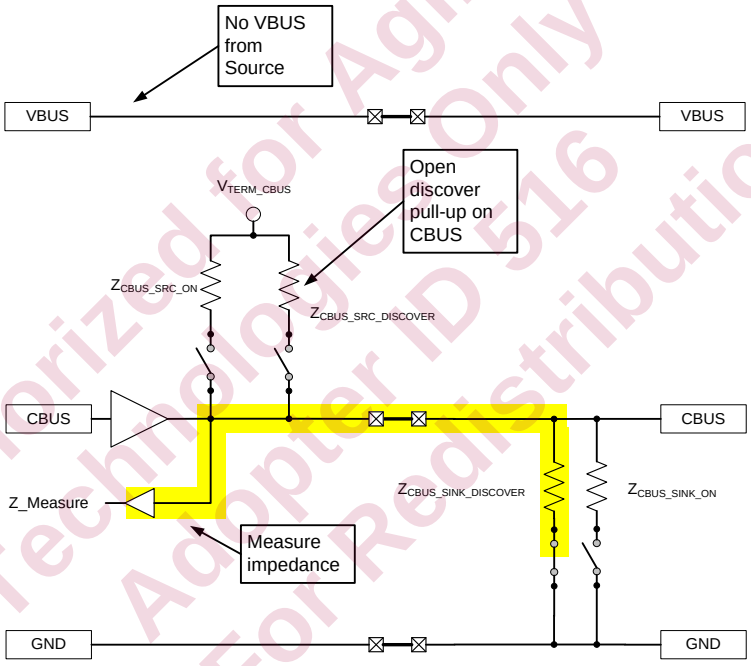


Figure 8-11. Connection –CBUS Pull-down

After detecting an impedance change on the CBUS, and after a recommended wait of $T_{SRC\_START\_IMP\_MEAS}$, the Source measures the impedance on CBUS. If the value falls within the minimum to maximum range of $Z_{CBUS\_SINK\_DISCOVER}$, then the Source proceeds to the next step in MHL discovery. If the impedance is not in this range, then the Source assumes it is connected to a non-MHL device. The Source shall float CBUS (impedance equivalent to meeting $I_{CBUS\_LEAK\_SRC}$) prior to each subsequent impedance measurement.

8.4.2 Wake Pulse Sequence

The attached Source device shall begin a wake pulse sequence after detecting $Z_{CBUS\_SINK\_DISCOVER}$ on the CBUS connection between the Source and Sink or Dongle, and prior to each initiation by the Source of the discovery pulse sequence.

The Source shall create the wake pulse sequence on CBUS shown in Figure 8-12, by actively driving the CBUS high and allowing it to float low, using timings defined in Table 13-30, in order to alert an attached Sink device, whether the Sink is in Active or Standby state, that a discovery pulse sequence is to follow within $T_{SRC\_WAKE\_TO\_DISCOVER}$.

The wake pulse sequence shall consist of two pulses, each of high time $T_{SRC\_WAKE\_PULSE\_WIDTH\_1}$ and low time $T_{SRC\_WAKE\_PULSE\_WIDTH\_1}$, followed by a low time on CBUS of $T_{SRC\_WAKE\_PULSE\_WIDTH\_2}$, then another two pulses with high and low times of $T_{SRC\_WAKE\_PULSE\_WIDTH\_1}$ as before.

An attached Sink device or Dongle device, whether in Active or Standby state, shall react to a valid wake pulse sequence by being ready to recognize the first rising edge of the discovery pulse sequence, which follows within $T_{SRC\_WAKE\_TO\_DISCOVER}$.

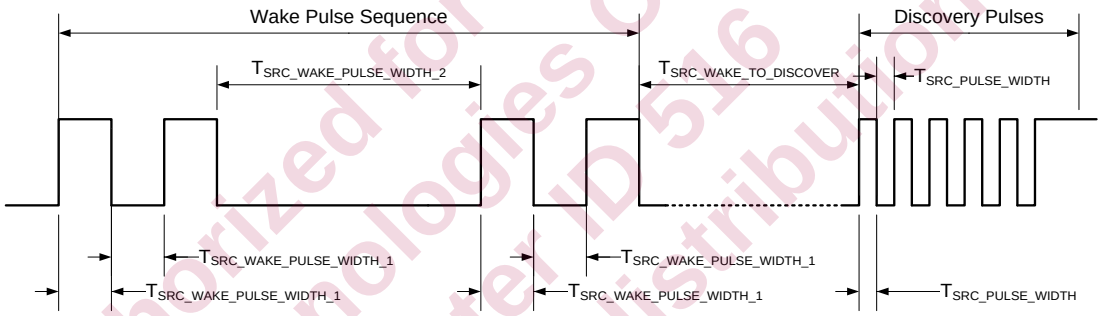


Figure 8-12. Connection –Wake Pulse Sequence

A Sink which detects only a partial wake pulse sequence, or invalid wake pulses is not required to be ready for a discovery pulse sequence. A Source which sends a wake pulse sequence, and then a discovery pulse sequence, but sees no connection to the Sink (potentially because the Sink did not see the complete wake sequence as valid) should re-initiate the discovery with a new wake pulse sequence.

The wake pulse sequence allows a Sink in Standby to recognize the impending attempt by the Source to discover the MHL Sink, without the Sink having to measure and qualify the discovery pulses, which are faster and have tighter timing constraints.

See Sections 8.2.2 and 8.2.3 for reference to wake pulses in the discovery state listings for Sink and Dongle.

8.4.3 CBUS Drive-and-Float "Discovery Pulse" Counting

After the Source has recognized the proper pull-down impedance on CBUS, it shall proceed to distinguish a valid MHL Sink or Dongle from any non-MHL device which uses the same pull-down impedance through the discovery pulse process specified in this section. This discovery pulse counting process is executed by the Source during states SRC3 and SRC9 (see Section 8.2).

The Source executes this process by repeatedly first driving a HIGH level on CBUS, then releasing the driver to monitor the level on CBUS for a pulled-down LOW. Each of these "drive HIGH / float and measure" periods is termed a "discovery pulse". The Source has a specified number of discovery pulses it shall attempt before concluding that the Sink is not an MHL device. The Sink has a minimum number of discovery pulses it must observe before the Sink concludes that the connection is completely discovered.

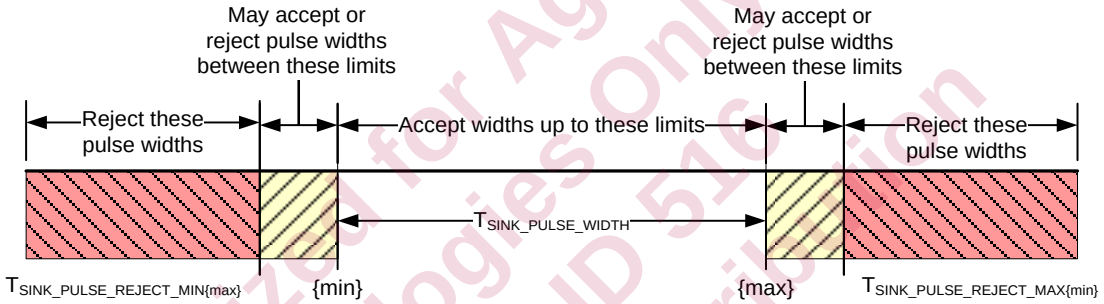


Figure 8-12-1 Timing Diagram for Acceptance and Rejection of Discovery Pulses

Both Source and Sink shall clear their respective discovery pulse counters prior to initiation of this process.

8.4.3.1 Drive CBUS to High from Source

The Source actively drives CBUS to a HIGH state, overcoming the $Z_{CBUS\_SINK\_DISCOVER}$ pull-down. The Source maintains a HIGH state for a pulse of width $T_{SRC\_PULSE\_WIDTH}$.

The Sink shall recognize only pulses of specified width, accepting all HIGH pulses which meet $T_{SINK\_PULSE\_WIDTH}$, and rejecting all pulses shorter than $T_{SINK\_PULSE\_REJECT\_MIN}$ or longer than $T_{SINK\_PULSE\_REJECT\_MAX}$. A Sink may accept or reject HIGH pulse widths between $T_{SINK\_PULSE\_WIDTH}\{min\}$ and $T_{SINK\_PULSE\_REJECT\_MIN}$, or between $T_{SINK\_PULSE\_WIDTH}\{max\}$ and $T_{SINK\_PULSE\_REJECT\_MAX}$. On detecting an invalid pulse width, the Sink shall clear its discovery pulse counter to zero.

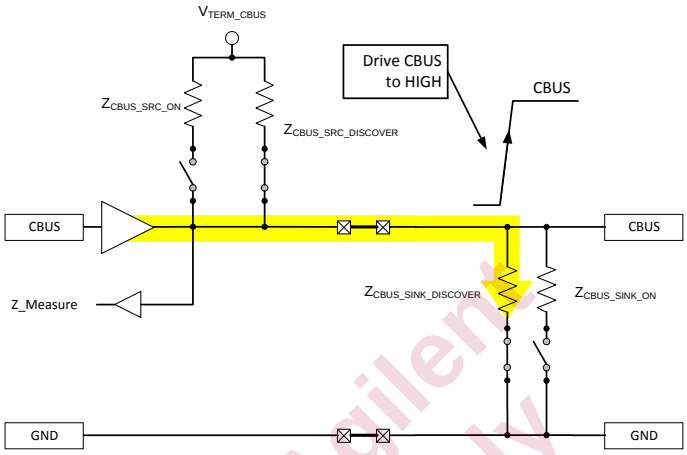


Figure 8-13. Connection –Drive CBUS to HIGH

8.4.3.2 Float CBUS from Source Side and Check Level

After driving HIGH for duration $T_{SRC\_PULSE\_WIDTH}$, the Source stops driving CBUS for a float period of duration $T_{SRC\_PULSE\_WIDTH}$. If, at any time during the float period, the level on CBUS is detected as a LOW, then the Source increments its discovery pulse counter. It concludes that the Sink has not yet been discovered.

The Sink checks float period duration by measuring the LOW pulse width on CBUS. The Sink shall recognize only pulses of specified width, accepting all LOW pulses meeting $T_{SINK\_PULSE\_WIDTH}$, and rejecting all pulses shorter than $T_{SINK\_PULSE\_REJECT\_MIN}$ or longer than $T_{SINK\_PULSE\_REJECT\_MAX}$. A Sink may accept or reject pulses with LOW pulse widths between $T_{SINK\_PULSE\_WIDTH}\{min\}$ and $T_{SINK\_PULSE\_REJECT\_MIN}$ or between $T_{SINK\_PULSE\_WIDTH}\{max\}$ and $T_{SINK\_PULSE\_REJECT\_MAX}$. On detecting an invalid pulse width, Sink shall clear its discovery pulse counter to zero.

The Sink, on observing the HIGH level driven by the Source followed by a LOW (constituting a complete discovery pulse), increments its discovery pulse counter.

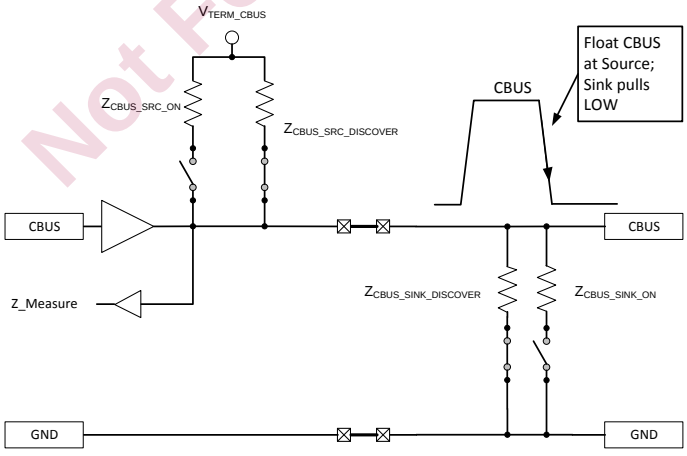


Figure 8-14. Connection –Float CBUS and Count Discovery Pulse if LOW

The process of driving and then floating CBUS shall be repeated until the Source either (1) detects a HIGH on CBUS during one of its float periods of $T_{SRC\_PULSE\_WIDTH}$ to conclude that a connection

has been discovered; or (2) reaches the maximum number of discovery pulses ($N_{SRC\_PULSE\_COUNT}$) and concludes that no ready MHL Sink or Dongle is connected.

Upon reaching $N_{SRC\_PULSE\_COUNT}$, the Source should recheck the CBUS impedance, then reattempt the discovery pulse process multiple times, for as long as the Source requirement for discovery persists.

8.4.3.3 Sink CBUS Float when Ready

The Sink, on observing the next rising edge on CBUS after the required number of discovery pulses, $N_{SINK\_PULSE\_COUNT}$, causes the CBUS to remain HIGH by changing the pull-down on CBUS from $Z_{CBUS\_SINK\_DISCOVER}$ to $Z_{CBUS\_SINK\_ON}$ no later than T_{SINK_CONN} after the rising edge. However, the Sink shall not change to the $Z_{CBUS\_SINK\_ON}$ impedance until it has counted the minimum number of discovery pulses, $N_{SINK\_PULSE\_COUNT}$.

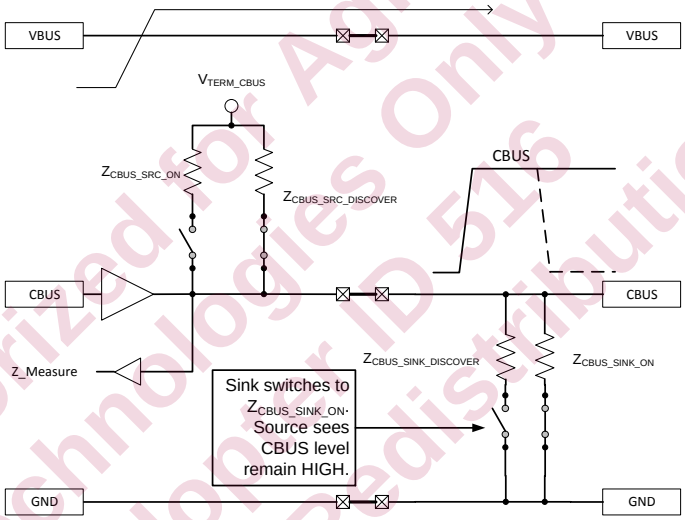


Figure 8-15. Connection –Float CBUS and Count Discovery Pulse

A Source device which observes a HIGH on CBUS during one of its float periods of $T_{SRC\_PULSE\_WIDTH}$, before counting more than $N_{SRC\_PULSE\_COUNT}$ discovery pulses shall conclude it is connected to an MHL Sink.

A Sink (or Dongle) device which counts $N_{SINK\_PULSE\_COUNT}$ discovery pulses shall conclude it is connected to an MHL Source.

8.4.4 MHL+ and MHL- Termination

After the Sink is in the connected state, it shall terminate the MHL+ and MHL- signal paths with an impedance of $Z_{RXSENSE\_TERM}$ or $Z_{RXSENSE\_TMDS}$ within $T_{SINK\_RXSENSE\_EN}$. The connected Source checks for RxSense as specified in Section 8.2, and shown in Figure 8-16.

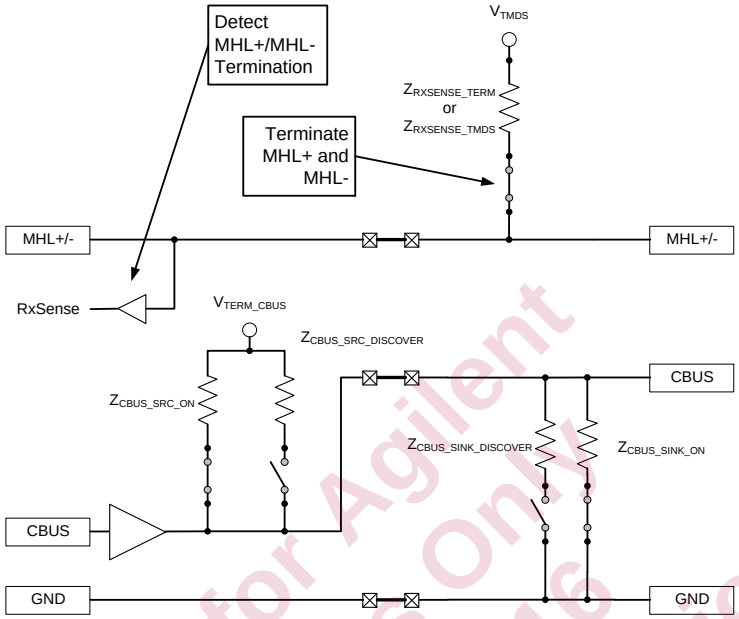


Figure 8-16. Connection –Terminate MHL+/MHL-

8.5 Data Exchange during Discovery Sequence

After a Source and Sink (or Dongle) finish the drive-and-float discovery pulse process, the connection process continues with exchange of information between the two devices.

There are three parts in the data exchange process.

1. Connected devices exchange Capability Register values.
2. Connected devices exchange Status Register values.
3. The Source device reads the EDID data from the Sink or Dongle device.

Other than the dependencies on HPD (see Section 8.1.3), the above steps may be performed in any order.

8.5.1 CBUS First Packet Arbitration

After the Sink is in the connected state, it shall wait at least T_{ARB_SINK} before beginning the first arbitration on CBUS to send a packet.

After the Source is in the connected state, it shall wait at least T_{ARB_SRC} before beginning the first arbitration on CBUS to send a packet.

8.5.2 Exchange Device Capability Registers

The Source and Sink shall exchange the data in each other's Device Capability registers, using the commands defined in Section 7.4. A Dongle should read the Device Capability register values from the connected Source, using the commands defined in Section 7.4.

For two MHL devices, Device A and Device B, attached and connected through the electrical discovery process:

When Device A is ready with stable data in its Device Capability registers, Device A shall write a '1' to the DCAP_RDY bit using the WRITE_STAT command, and write a '1' to the DCAP_CHG interrupt bit using the SET_INT command. See Sections 7.4.3.8 and 7.4.3.9.

When the DCAP_CHG bit in its register space changes from '0' to '1', Device B shall read the Device Capability registers from Device A, using a sequence of READ_DEVCAP commands. If the status of Device A also indicates with DCAP_RDY that the Device Capability data is reliable, then Device B may rely on the values read. See Section 7.4.3.7.

This process shall be performed in the opposite direction as well, from Device B to Device A, whenever Device A's DCAP_CHG bit changes from '0' to '1'.

8.5.3 Exchange Status Registers

The Source and Sink and Dongle shall exchange the data in each other's Status registers, using the WRITE_STAT command defined in Section 7.4.

For two MHL devices, Device A and Device B, attached and connected through the electrical discovery process:

Device A shall write its status into the registers of Device B.

Device B shall write its status into the registers of Device A.

Each device shall update its status across the link whenever any bits in its status change.

8.5.4 Read Sink EDID

The Source shall read the EDID data from the Sink, using the commands defined in Section 7.5. The EDID shall not be read before the Source receives a SET_HPD command from the connected Sink or Dongle.

Whenever the Source sees a change in HPD from deasserted to asserted, by receiving CLR_HPD and SET_HPD commands from the Sink or Dongle (or after initial connection and subsequently receiving the SET_HPD command), the Source shall re-read the entire EDID.

A Source shall also read the EDID when it sees an EDID_CHG interrupt in the Interrupt register.

8.6 Disconnection Sequence

Two different events may change the state of a link to the disconnected state: CBUS may be held LOW by the Source for an extended time (Section 8.6.1), or MHL pair termination may be switched to high-impedance by the Sink (Section 8.6.2).

8.6.1 CBUS Goes Low

A Sink shall monitor the state of its CBUS pin and transition to a disconnected state whenever CBUS goes low for greater than $T_{SINK\_CBUS\_DISCONN}$.

A Sink which detects a continuous LOW on CBUS for $T_{SINK\_CBUS\_DISCONN}$ shall conclude that no active MHL Source is connected, transition to a disconnected state, and change the impedance on CBUS back to $I_{CBUS\_LEAK\_SINK}$.

8.6.2 RxSense Goes Low

A Source shall monitor the impedance on the MHL+ and MHL- lines. The Source shall transition to a disconnected state whenever the impedance on either of these lines exceeds an impedance equivalent to $I_{RXSENSE\_LEAK}$. Refer to Table 13-9 for Sink and Table 13-12 for Dongle. When transitioning to the disconnected state, the Source disables $Z_{CBUS\_SRC\_ON}$ for at least $T_{SRC\_CBUS\_FLOAT}$ (see Table 13-28), while meeting the $I_{CBUS\_LEAK\_SRC}$ specification (see Section 8.2). See

Table 13-27 for the next cable attach impedance measurement.

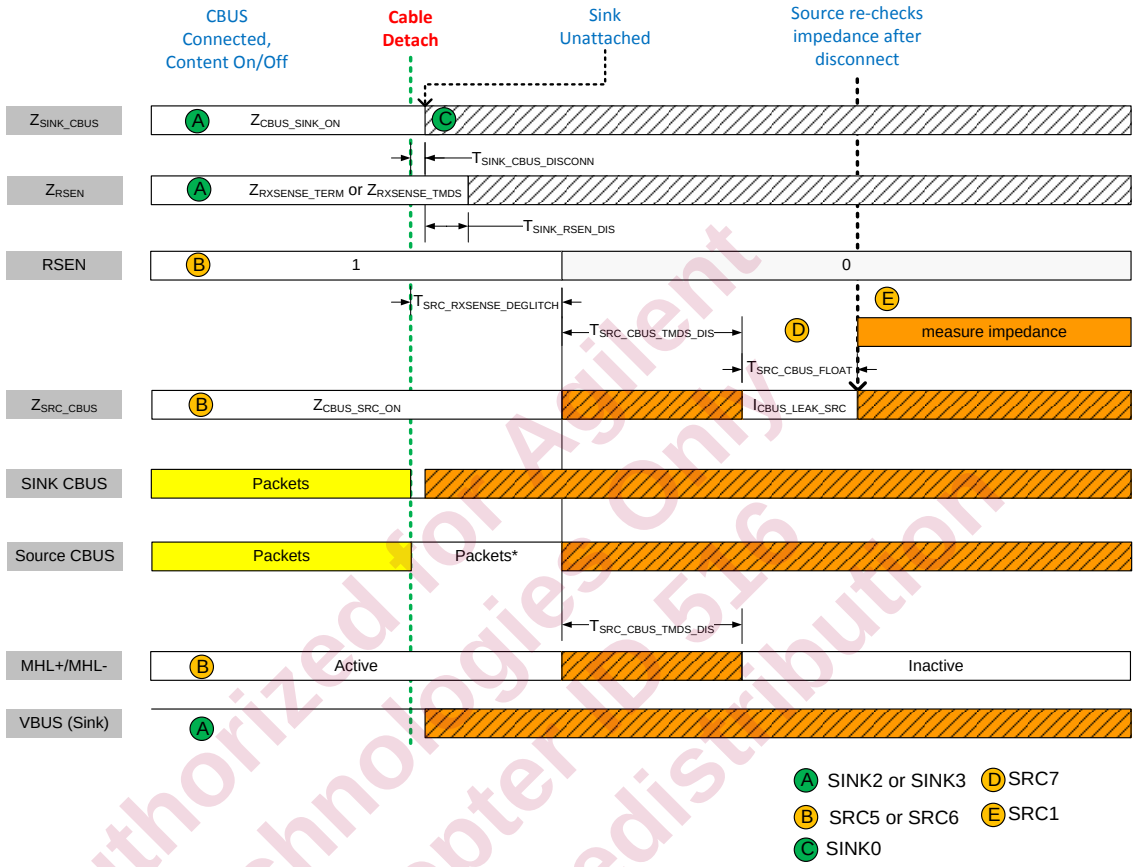


Figure 8-17. "Disconnection on Detach" Timing Diagram

8.6.3 Informative: Coordinating Content Flow to the Screen

Once an MHL Source and Sink are connected, there are many use cases depending on when the Source and Sink establish stable TMDS, and when the Source's content is actually shown in the display.

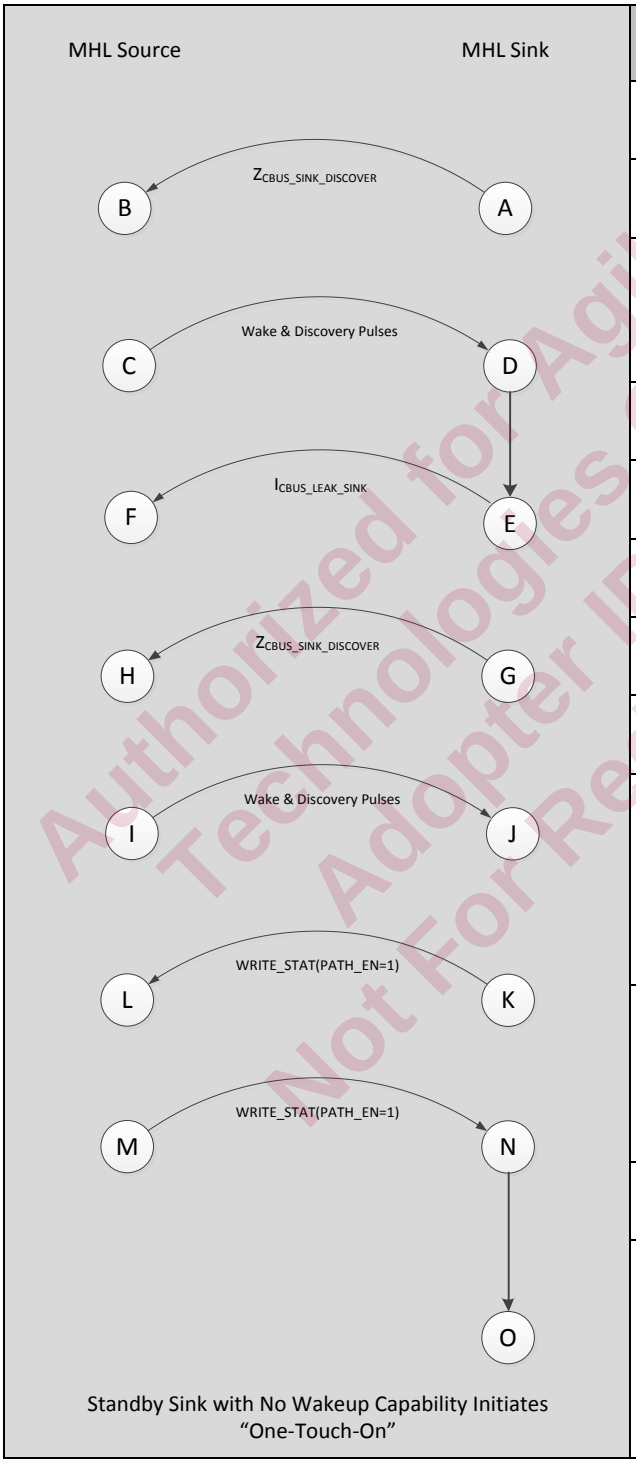
NOTE: This section is provided to describe the use of the RAP sub-command in coordination with PATH_EN and other mechanisms. Refer to the definition sections for those features for the controlling requirements.

8.6.3.1 One-Touch-On

The One-Touch-On feature intends to allow a user to activate an attached MHL Source and show its content by taking one action on the Sink side, such as pressing one key on the Sink remote control.

8.6.3.1.1 One-Touch-On Case #1: Attach to Standby Sink Which Does Not Check for Wake Pulses

In Figure 8-18, the MHL port on the Standby Sink begins in an unselected state, with the MHL pair unterminated, and no Source is attached. The Source is then attached to the Sink (SINK4B). The Source begins drawing power.

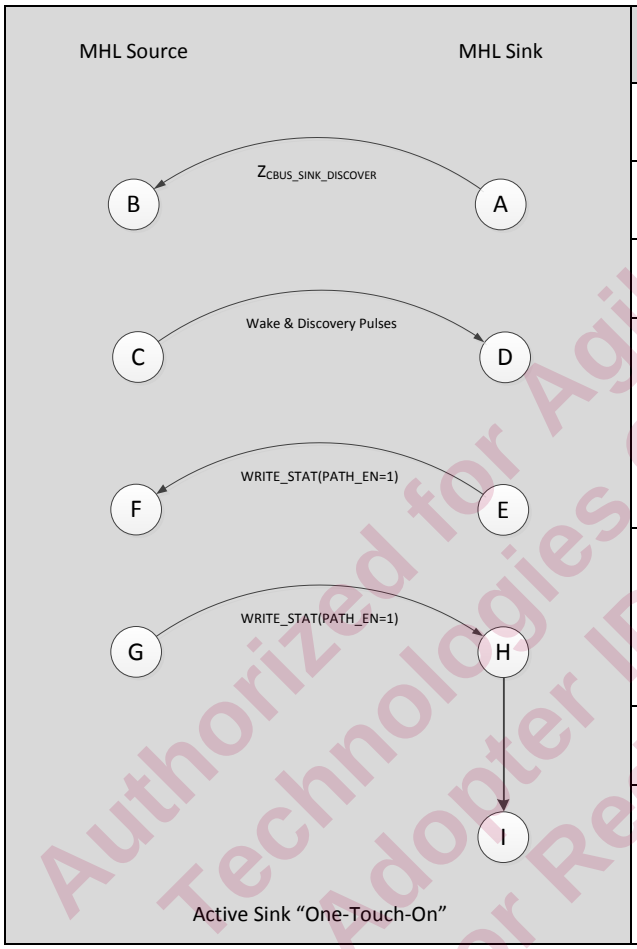


	State or Transition Description
A -> B	Standby Sink presents $Z_{CBUS\_SINK\_DISCOVER}$ to Source.
C -> D	Source initiates discovery protocol.
	Sink remains in standby (see Section 8.2.2.2), Source discovery fails and remains in a CBUS unconnected state.
D -> E	User turns on Sink and Sink floats CBUS.
E -> F	Sink presents $I_{CBUS\_LEAK\_SINK}$ to Source.
G -> H	Active Sink presents $Z_{CBUS\_SINK\_DISCOVER}$ to Source
I -> J	Source initiates discovery protocol.
	Discovery is successful between Source and Sink.
K → L	Sink writes its status of PATH_EN{Sink}==1 to Source, indicating that the MHL port TMDS is to be made active. No content is being used on the Sink side.
M→ N	Source recognizes active status of Sink MHL port, and responds by sending its active MHL port status to the Sink with PATH_EN{Source}==1.
	Source begins flow of active content.
N → O	Sink selects MHL port to which Source is attached, active content being displayed.

Figure 8-18. One-Touch-On – Case #1

8.6.3.1.2 One-Touch-On Case #2: Attach to Active Sink – Automatic Select

In Figure 8-19, the MHL port on the Sink begins in an unselected state, with the MHL pair unterminated, and no Source is attached. The Source is then attached to the Sink (SINK1), and the Source begins drawing power.

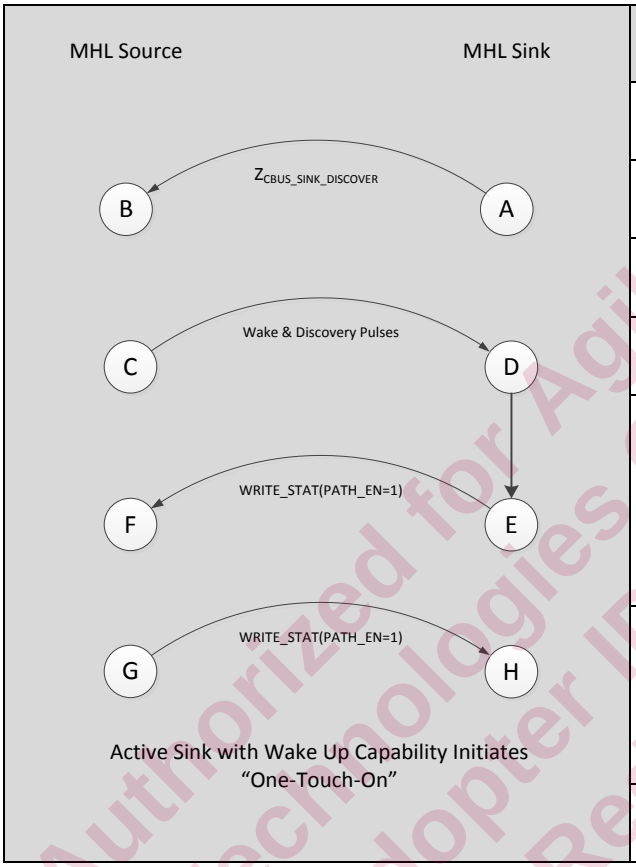


	State or Transition Description
A -> B	Active Sink presents $Z_{CBUS\_SINK\_DISCOVER}$ to Source
C -> D	Source initiate discovery protocol
	Discovery successful between Source and Sink
E → F	Sink writes its status of PATH_EN{Sink}==1 to Source, indicating that the MHL port TMDS is to be made active. No content is being used on the Sink side.
G→ H	Source recognizes active status of Sink MHL port , and responds by sending its active MHL port status to the Sink with PATH_EN{Source}==1.
	Source begins flow of active content
H → I	Sink selects MHL port to which Source is attached, active content being displayed

Figure 8-19. One-Touch-On – Case #2

8.6.3.1.3 One-Touch-On Case #3: Attach to Active Sink – User Selects Port

In Figure 8-20, the MHL port on the Sink begins in an unselected state, with the MHL pair unterminated, and no Source is attached. The Source is then attached to the Sink (SINK1), and the Source begins drawing power.



	State or Transition Description
A -> B	Active Standby Sink presents $Z_{CBUS\_SINK\_DISCOVER}$ to Source.
C -> D	Source initiates discovery protocol.
	Discovery is successful between Source and Sink.
D → E	User selects MHL port to which Source is attached.
E → F	Sink writes its status of PATH_EN{Sink}==1 to Source, indicating that the MHL port TMDS is to be made active. No content is being used on the Sink side.
G→ H	Source recognizes active status of Sink MHL port , and responds by sending its active MHL port status to the Sink with PATH_EN{Source}==1.
	Source begins flow of active content.

Figure 8-20. One-Touch-On – Case #3

Note that the process in Figure 8-20 can also be applied to a Sink device which monitors its MHL input port for wake pulses, and can come out of standby and select its MHL input port based only on wake pulses (D→E transition). In this scenario, no user action is needed at the Sink.

8.6.3.2 One-Touch-Play

The One-Touch-Play feature intends to allow a user to activate an attached MHL Sink and show its content by taking one action on the Source side, perhaps as simple as making the attachment.

In Figure 8-21, two scenarios are shown. In Scenario #1, the MHL Sink does not send PATH_EN=1 to the Source after CBUS connection, for instance when the user has a different input port selected. When the Source sends a RAP{CONTENT_ON} to the SINK (for instance when the user pushes "Play" on the phone's user interface), the Sink then sends PATH_EN=1 to the Source, moves its selected port to the MHL port to which the Source is attached, and the content is viewed on the display.

In Scenario #2, the MHL Sink completes the PATH enabling by sending PATH_EN=1 to the Source immediately after a CBUS connection is established. This allows the MHL Sink to recognize the active video mode coming from the MHL Source, even if the MHL Sink currently does not have this port selected on the display (for instance the user has a different input port selected.) When the Source sends a RAP{CONTENT_ON} to the Sink (for instance when the user pushes "Play" on the phone's user interface), the Sink moves its selected port to the MHL port to which the Source is attached, and the content is viewed on the display.

Note, there may be a situation, for example when a recording is in progress on a different Sink input port, where the Sink responds with a RAPK{BUSY} in response to the RAP{CONTENT_ON} message from the Source. In this event the Sink can display a message on its display for the user indicating the request from the Source. The MHL Source can repeat its request by polling, by waiting for a timer to expire, or prompt the user at the Source side to re-initiate the request.

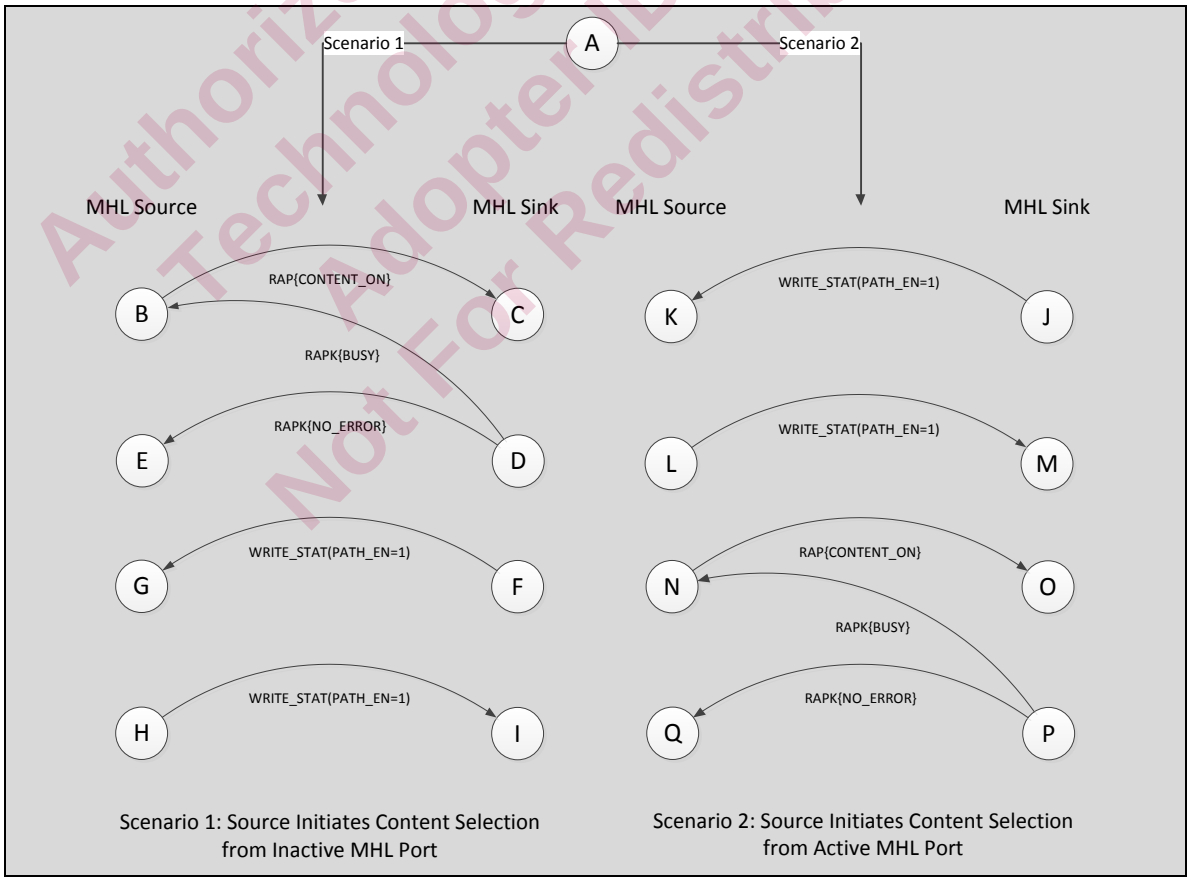


Figure 8-21. One-Touch-Play

Scenario 1	State or Transition Description	Scenario 2	State or Transition Description
A	Source and Sink complete discovery.	A	Source and Sink complete discovery.
B → C	Source device sees complete discovery phase but see PATH_EN{Sink}==0. Source wants to show its content on the Sink, and sends RAP{CONTENT_ON} to Sink	J → K	Sink writes its status of PATH_EN{Sink}==1 to Source, indicating that the MHL port TMDS is to be made active. No content is being used on the Sink side.
D → E	Sink acknowledges Source's request by responding with RAPK{NO_ERROR}.	L → M	Source recognizes active status of Sink MHL port , and responds by sending its active MHL port status to the Sink with PATH_EN{Source}==1.
D → B	Sink acknowledges Source's request by responding with RAPK{BUSY}. Source re-attempts by sending RAP{CONTENT_ON} to Sink (B → C)	N → O	Source wants to show its content on the Sink, and sends RAP{CONTENT_ON} to Sink
F → G	Sink writes its status of PATH_EN{Sink}==1 to Source, indicating that the MHL port TMDS is to be made active. Sink moves its selected input to the MHL port to which the Source is attached.	P → Q	Sink acknowledges Source's request by responding with RAPK{NO_ERROR} and moves its selected input to the MHL port to which the Source is attached.
H → I	Source recognizes active status of Sink MHL port , and responds by sending its active MHL port status to the Sink with PATH_EN{Source}==1.	P → N	Sink acknowledges Source's request by responding with RAPK{BUSY}. Source re-attempts by sending RAP{CONTENT_ON} to Sink (L → M)

Figure 8-22. One-Touch-Play

8.7 Sharing MHL Connector Pins

A Source or Sink which shares the pins of an MHL connector for use with other protocols shall bear the responsibility to prevent damage to MHL-related signals and voltage pins and to test that such functionality does not interfere with the functionality and the compliance of such pins when in MHL mode. This requirement pertains as well during the discovery process, during which a Source device shall choose the proper functional mode for connection to an attached device, without ambiguity and without damage to MHL functionality. Similarly, a Sink device shall choose the proper functional mode for connection to an attached device, without ambiguity and without damage to the MHL functionality.

The details of the discovery process in conjunction with other protocols are outside the scope of this specification.

9 Power Provisioning

9.1 Sink and Dongle Power Output

An MHL Sink is expected to be able to provide power to a connected MHL Source as specified in Section 8 and Section 13.9.1.2. An MHL 2 Sink shall be capable of supplying $I_{VBUS\_Sink\_to\_Source\_MHL2\_Sink}\{min\}$ of power on the VBUS pin of each of its MHL input ports.

A DC-powered Sink (see Section 2.2) is required to meet the VBUS output requirements of a Sink when attached to an external power source.

When not attached to an external power source, a DC-powered Sink may output VBUS to a Source.

An MHL Dongle with its own power source may output VBUS to the MHL Source immediately after power is applied to the Dongle (Section 8.2), which may occur before MHL discovery completes. An MHL Dongle that is outputting power on VBUS after MHL discovery completes shall have the POW bit set to 1 (Section 7.9.1.3). See also $T_{SRC\_VBUS\_DIS}$.

The sequence of use of a Source and Dongle may proceed as follows:

1) User attaches unpowered Dongle to Source.
2) Source sees discovery impedance on Dongle and begins to output VBUS to power the Dongle.
3) Source and Dongle proceed with the discovery process. Source reads the POW bit from the Dongle as part of the Device Capability register read sequence. The Source see POW=0 from the Dongle and continues to output VBUS to the Dongle.
4) Meanwhile, the user attaches a power supply (charger) to the Dongle.
5) After discovery completes, the Dongle, having recognized its active charger, writes its own POW bit to '1', and sends a DCAP_CHG interrupt to the discovered Source.
6) The Source recognizes the DCAP_CHG interrupt, reads the Device Capability registers of the Dongle, sees the asserted POW bit, and disables the Source's own VBUS output.

The timing parameter $T_{SRC\_VBUS\_DIS}$ limits the duration of power contention on VBUS. The worst case time is the sum of:

$$T_{DONGLE\_DCAP\_CHG} + T_{SRC\_READ\_DCAP} + T_{SRC\_VBUS\_DIS}$$

9.2 Source Power Output

MHL allows a Source device to output power onto the interconnecting cable. This power may be useful to a Dongle, eliminating the need for an external power source for the Dongle, yet allowing it to connect to an input port on the Sink side which cannot provide power.

A Source device, once attached to a Dongle, may output VBUS when $T_{SRC\_VBUS\_CBUS\_STABLE}$ expires without detecting a stable VBUS from the Dongle. The Source shall respond to the assertion of the Source's DCAP_CHG interrupt bit, or the Source's DCAP_RDY Status bit by reading the Device Capability registers in the Dongle. If the Dongle shows a POW bit set to '1', then the Source shall disable its output VBUS within $T_{SRC\_VBUS\_DIS}$ after reading the POW bit. See Table 13-34.

The Source may see a DCAP_RDY once connected, with a POW=0 state in the DC-powered Sink or Dongle's Device Capability register if there is no power available from the DC-powered Sink or Dongle. In this case, the Source may continue to output VBUS to the DC-powered Sink or Dongle. The DC-powered Sink or Dongle may indicate with its POW bit set to 0 that its external power is removed; or it may output power without external power applied, with its POW bit set to 1. See Figure 8-7.

An MHL Source which outputs VBUS to an attached device ceases to output VBUS when it determines that the other device is a Sink with POW=1, as specified in Section 8.2.1.1.

9.3 External Power to MHL Source via Dedicated Charging Device

Power may be applied to an MHL Source through the Source's MHL connector using a device other than an MHL Sink, such as a dedicated charger, provided such a device does not damage other pins on the interface. Implementation of such shared use of the MHL connector is beyond the scope of this specification. Please refer to application notes or other documentation as may be available for connection to other devices.

9.4 Power Configurations

Table 9-1 defines configurations in which the Sink or the Source provide VBUS power to the link. Refer to the description of DEV_CAT in Section 7.9.1.3. The DEV_TYPE values are the values shown by the Sink or Dongle in each case.

Table 9-1. Power Configurations

Case	Device 1	Device 2	Power Flow	Dongle or Sink DEV_TYPE	Notes
1	Source with Power Output	Unpowered Dongle	From Source to Dongle	0x3	1
2a	Source with Power Output	Dongle without Local Power, can be powered from Source	From Source to Dongle	0x3	2
2b		Dongle with Local Power	From Dongle to Source	0x3	
3a	Source without Power Output	Dongle without Local Power	No Power Flow		3
3b		Dongle with Local Power	From Dongle to Source	0x3	
4a	Source with Power Output	Dongle without Local Power, but cannot be powered from Source	No Power Flow		4
4b		Dongle with Local Power	From Dongle to Source	0x3	
5	Source with or without Power Output	Powered Sink	From Sink to Source	0x1	5

Notes on Table 9-1:

1. The Source is designed to output power to a Dongle. The Dongle is designed to be powered only from the Source. Power flows from the Source into the Dongle, as the Dongle has no other power source.
2. The Source is designed to output power to a Dongle. The Dongle is designed to be powered from a Source or from a local power supply. In 2a, power flows from the Source into the Dongle, as the Dongle has no charger plugged into it. In 2b, when the active charger is plugged into the Dongle, the Source and Dongle change power sourcing, using the DEV_CAT register, so that the Dongle provides power to the Source.
3. The Source is not designed to output power to a Dongle. The Dongle may or may not be designed to draw power from the Source. If the Dongle has no local power (3a), then no power can flow to the Source. If the Dongle has local power (3b), then power flows from Dongle to Source.
4. The Source is designed to output power to a Dongle. The Dongle is not designed to draw power from the Source. If the Dongle has no local power (4a), then no power can flow to the Source. If the Dongle has local power (4b), then power flows from Dongle to Source.
5. The Sink has power. Power flows from the Sink to the Source.

10 Control And Configuration

10.1 Overview

MHL includes two separate communications channels: TMDS and CBUS.

TMDS is used to carry all audio and video data as well as auxiliary data (such as InfoFrame packets) that describes the active audio and video streams.

The CBUS provides a mechanism for detection of device connectivity. It also provides a virtual DDC channel, used by an MHL Source to determine the capabilities and characteristics of the Sink by reading the E-EDID data structure. In addition it functions as an MHL Sideband Channel (MSC), used for higher-level user functions such as automatic setup tasks or tasks typically associated with infrared remote control usage.

MHL Source devices are expected to read the Sink's EDID data and to deliver only the audio and video formats which are supported by the Sink. (See also the SUPP_VGA bit in Section 7.9.1.5.) Also, MHL Sink devices are expected to detect AVI and Audio InfoFrames in the TMDS stream and to render the received audio and video data appropriately.

MHL Source devices and Sink devices may also include Device Capability Registers to define the set of features in each system. A Source device reads the Device Capability Registers from the Sink using CBUS commands. A Sink device reads the Device Capability Registers from the Source using CBUS commands.

10.2 EIA/CEA-861E InfoFrames

An InfoFrame packet can carry one InfoFrame. The InfoFrame provided by MHL is limited to 30 bytes plus a checksum byte. MHL Source devices are required to use the AVI InfoFrame and Audio InfoFrame. Other InfoFrames specified in EIA/CEA-861E are optional.

All InfoFrames are described in detail in EIA/CEA-861E section 6. The following describes how these InfoFrames are placed within the InfoFrame Packet structure and any areas where MHL behavior is different than that specified in EIA/CEA-861E.

10.2.1 AVI InfoFrame

Various aspects of the current video stream are indicated by the Source device to the Sink device with an Auxiliary Video Information (AVI) InfoFrame.

If the receiver does not detect any AVI InfoFrame packets from a particular Source device, then the Sink should assume that the video conforms to default characteristics. These are:

- RGB 4:4:4 (8/8/8) pixel encoding
- Aspect ratio is determined by the video format and EDID (see EIA/CEA-861E, section 4.1 and 7.1.3 for details)
- Under-scan/over-scan should be determined by the video format (640x480p is under-scanned, all other EIA/CEA-861E formats are over-scanned)
- Pixel repetition should be determined by the video format (pixel-doubled for 480i and 576i, no repetition otherwise)
- Colorimetry should be determined by the video format (see EIA/CEA-861E, section 5 for details)

The packetization of the AVI InfoFrame is shown in Table 10-1.

Table 10-1. AVI InfoFrame Packet

Packet Byte #	EIA/CEA-861E Byte #	7	6	5	4	3	2	1	0
HB0	Data Byte 1	Packet Type = 0x82							
HB1	Data Byte 1	Version = 0x02							
HB2	Data Byte 1	Length = 13							
PB0	N. A.	Checksum							
PB1	Data Byte 1	Rsvd (0)	Y1	Y0	A0	B1	B0	S1	S0
PB2	Data Byte 2	C1	C0	M1	M0	R3	R2	R1	R0
PB3	Data Byte 3	ITC	EC2	EC1	EC0	Q1	Q0	SC1	SC0
PB4	Data Byte 4	Rsvd (0)	VIC6	VIC5	VIC4	VIC3	VIC2	VIC1	VIC0
PB5	Data Byte 5	YQ1	YQ0	CN1	CN0	PR3	PR2	PR1	PR0
PB6	Data Byte 6	Line Number of End of Top Bar (lower 8 bits)							
PB7	Data Byte 7	Line Number of End of Top Bar (upper 8 bits)							
PB8	Data Byte 8	Line Number of start of Bottom Bar (lower 8 bits)							
PB9	Data Byte 9	Line Number of start of Bottom Bar (upper 8 bits)							
PB10	Data Byte 10	Pixel Number of End of Left Bar (lower 8 bits)							
PB11	Data Byte 11	Pixel Number of End of Left Bar (upper 8 bits)							
PB12	Data Byte 12	Pixel Number of Start of Right Bar (lower 8 bits)							
PB13	Data Byte 13	Pixel Number of Start of Right Bar (upper 8 bits)							
PB14-PB27	Data Bytes 14-27	Reserved (0)							

- Y0, Y1 RGB or I indicator. See EIA/CEA-861E for details.
- A0 Active Information Present. See EIA/CEA-861E for details.
- B0, B1 Bar Info data valid. See EIA/CEA-861E for details.
- S0, S1 Scan Information (i.e. overscan, underscan). See EIA/CEA-861E for details. See also restrictions below.
- C0, C1 Colorimetry (ITU BT.601, BT.709 etc.). See EIA/CEA-861E for details.
- M0, M1 Picture Aspect Ratio (4:3, 16:9). See EIA/CEA-861E for details.
- R0…R3 Active Format Aspect Ratio. See EIA/CEA-861E for details.
- ITC IT Content. See EIA/CEA-861E for details.
- EC2...EC0 Extended Colorimetry. (See Section 5.7.3 and note on xvYCC in section 5.7.3.1.) See EIA/CEA-861E for details.
- Q1, Q0 RGB Quantization Range. See EIA/CEA-861E for details.
- VIC0…VIC6 Video Format Identification Code. See EIA/CEA-861E for details.
- PR0…PR3 Pixel Repetition factor. See EIA/CEA-861E for details.

- CN1, CN0 Content Type. See EIA/CEA-861E for details.
- YQ1, YQ0 YCC Quantization. See EIA/CEA-861E for details.
- SC1, SC0 : Non-uniform Picture Scaling. See EIA/CEA-861E. SC1 and SC0 shall be 0, 0 in order to indicate that no non-uniform scaling has been performed. If accurate aspect ratio information that describes the transmitted video stream is available at the Source, then it must be sent by the Source on every frame using an AVI InfoFrame. Note that this is an exception to the rules specified in EIA/CEA-861E. Note also that Appendix I of EIA/CEA-861E, which describes the need for the SC0 and SC1 bits, illustrates a situation where the Source transmits incorrect aspect ratio information, causing confusion at the display. MHL Sources are not allowed to transmit incorrect aspect ratio information therefore they have no need for the SC0 and SC1 bits.

Table 10-2. MHL Valid Pixel Repeat Values for each format

Video Code	Video Description	Vertical Refresh Rate	MHL Pixel Repeat Values
1	640x480p	60Hz	Pixel sent 1 time
2,3	720x480p	59.94/60Hz	Pixel sent 1 time
4	1280x720p	59.94/60Hz	Pixel sent 1 time
5	1920x1080i	59.94/60Hz	Pixel sent 1 time
6,7	720(1440)x480i	59.94/60Hz	Pixel sent 2 times
10,11	2880x480i	59.94/60Hz	Pixel sent 1 to 10 times
14,15	1440x480p	59.94/60Hz	Pixel sent 1 to 2 times
17,18	720x576p	50Hz	Pixel sent 1 time
19	1280x720p	50Hz	Pixel sent 1 time
20	1920x1080i	50Hz	Pixel sent 1 time
21,22	720(1440)x576i	50Hz	Pixel sent 2 times
25,26	2880x576i	50Hz	Pixel sent 1 to 10 times
29,30	1440x576p	50Hz	Pixel sent 1 to 2 times
32	1920x1080p	23.97/24Hz	Pixel sent 1 time
33	1920x1080p	25Hz	Pixel sent 1 time
34	1920x1080p	29.97/30Hz	Pixel sent 1 time

An MHL Source shall transmit an accurate AVI InfoFrame at least once every two video fields if the Source:

- is ever capable of transmitting an AVI InfoFrame, or
- is ever capable of transmitting YcbCr pixel encoding across MHL, or
- is ever capable of transmitting any video format with a selectable pixel repeat value.

According to these requirements, such an MHL Source shall transmit an accurate AVI InfoFrame even when sending RGB pixel data or when sending non-repeated pixel data.

10.2.2 Audio InfoFrame

The following tables show the Audio InfoFrame.

Table 10-3. Audio InfoFrame Packet

Packet Byte #	EIA/CEA-861E Byte #	7	6	5	4	3	2	1	0
HB0	n. a.	Packet Type = 0x84							
HB1	n. a.	Version Number = 0x01							
HB2	n. a.	Length = 10							
PB0	n. a.	checksum							
PB1	Data Byte 1	CT3	CT2	CT1	CT0	Rsvd	CC2	CC1	CC0
PB2	Data Byte 2	Reserved			SF2	SF1	SF0	SS1	SS0
PB3	Data Byte 3	Format depends on coding type (i.e. CT0…CT3)							
PB4	Data Byte 4	CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0
PB5	Data Byte 5	Reserved	LSV3	LSV2	LSV1	LSV0	Rsvd	LFEPBL1	LFEPBL0
PB6	Data Byte 6	Reserved							
PB7	Data Byte 7	Reserved							
PB8	Data Byte 8	Reserved							
PB9	Data Byte 9	Reserved							
PB10	Data Byte 10	Reserved							
PB11..PB27		X	X	X	X	X	X	X	X

- CC0…CC2 Channel Count. See EIA/CEA-861E for details.
- CT0…CT3 Coding Type. The four CT bits shall always be set to a value of 0. See EIA/CEA-861E for details.
- SS0 to SS1 Sample Size. See EIA/CEA-861E for details. The SS field bits shall always be set to a value of 0, 0 (“Refer to Stream Header”).
- SF0…SF2 Sample Frequency. See EIA/CEA-861E for details. The SF field bits shall always be set to a value of 0, 0, 0 (“Refer to Stream Header”).
- CA0…CA7 Channel/Speaker Allocation. See EIA/CEA-861E for details.
- LSV0…LSV3 Level Shift Value (for down-mixing). See EIA/CEA-861E for details.
- LFEPBL0 to LFEPBL1 LFE Playback Level. See EIA/CEA-861E for details.

10.3 EDID Access

All MHL Sink devices shall contain an EDID data structure accessible through the virtual DDC channel provided by CBUS.

Before sending a content stream, the MHL Source shall determine if the MHL Sink device is capable of supporting that feature. This is done through CBUS reads of E-EDID or through CBUS queries of the HDCP and Capability registers inside of the MHL Sink device.

EDID access described in this section uses the extended EDID addressing capability as defined in VESA "ENHANCED DISPLAY DATA CHANNEL STANDARD Version 1 (September 2, 1999)". All Sink devices are required to support these enhanced EDID features. If a Sink device's EDID structure is longer than 256 bytes, it is required to support the segment pointer. All Source devices shall be capable of using the segment pointer to access EDID segments beyond the first 256 bytes if present.

A Sink device indicates that it is an MHL compliant Sink device by setting the appropriate bits in its Capability Registers.

10.3.1 Data Transfer Protocol

The CBUS provides a virtual Enhanced DDC channel between the Source device and the Sink device. The Source device is a virtual master and the Sink device is a virtual slave.

In Enhanced DDC, a segment pointer is used to allow addressing outside of the normal 256 byte limit of the A0h address. The Enhanced DDC protocol sets the segment pointer before the remainder of the DDC command.

See Section 7.5 for a description on the transport of DDC information over CBUS.

10.3.2 Segment Pointer

Enhanced DDC addressing allows access of up to 32 Kbytes of data. This is accomplished using a combination of the A0h/A1h address pair and a segment pointer. For each value of the segment pointer 256 bytes of data are available at the A0h/A1h address pair. An unspecified segment pointer references the same data as when the segment pointer is zero.

Each successive value of the segment pointer allows access to the next two blocks of EDID (128 bytes each). This is illustrated in Figure 10-1. The value of the segment pointer register cannot be read since it is reset at the completion of each command.

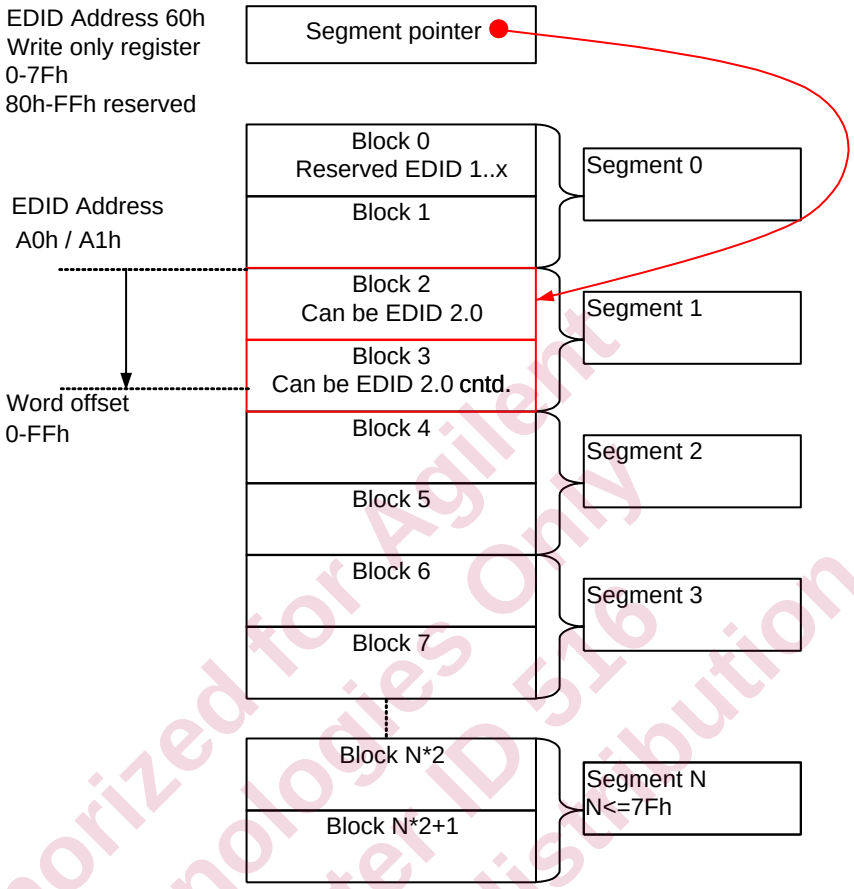


Figure 10-1. Enhanced EDID Segment Pointer and Block Layout

10.3.3 Enhanced EDID Compliant Devices

10.3.3.1 Enhanced EDID Sink device

The Sink device shall be Enhanced EDID read compliant.

The Sink device shall be capable of displaying with EDID 1.x data and up to 255 extension blocks, each 128 bytes (up to 32K bytes total EDID memory).

The Sink device should be capable of providing Enhanced EDID information over the Enhanced DDC channel whenever the MHL link is in a connected state.

10.3.3.2 Enhanced EDID Source device

The Source device uses Enhanced EDID addressing protocols.

The Source device shall be capable of reading EDID 1.x data at address 0xA0.

The Source device reads Enhanced EDID extensions data at address 0xA0 using segment pointer 0x60.

10.4 EDID Data Structures

10.4.1 Overview

The overall structure of the EDID data in the MHL Sink device shall conform to E-EDID structure but shall also meet the additional requirements specified herein.

The first 128 bytes of the EDID data shall contain an EDID 1.3 structure. The contents of this structure shall also meet the requirements of EIA/CEA-861E.

The second 128 bytes shall contain a CEA EDID Timing Extension version 3, defined in Section 10.4.2. The EDID shall not contain an EDID Timing Extension version 1 or version 2.

The Source device should output video and audio signals in the most suitable format supported by the Sink device.

Further details on the requirements of the data structures in the EDID data and implementation examples are given in EIA/CEA-861E.

10.4.2 EDID Timing Extension

EDID Timing Extension version 3 details are described in EIA/CEA-861E section 7.5.

If an MHL Source device needs to determine if a Sink device is MHL compliant, it shall do so in part by checking for the presence of the CEA EDID Timing Extension version 3 or higher. All MHL 1.0 Sink devices will contain version 3 of this structure. See also the bits in the Sink's Capability Registers, described in detail in Section 7.9.1 (page 141).

If a Sink supports either YcbCr 4:2:2 or 4:4:4, it shall support both 4:2:2 and 4:4:4 (see Section 5.2.3). Therefore bits 4 and 5 of byte 3 of the EDID Timing Extension shall be the same value.

11 Content Protection

11.1 Recommendation

Content protection capability is recommended for all MHL compliant devices. An MHL compliant Source device may protect its audiovisual data before making it available at its MHL output port. HDCP is an available copy protection technology which is compatible with MHL.

11.2 HDCP Implementations

HDCP implementations for MHL shall adhere to the HDCP Specification [Normative Reference #11].

NOTE: A system which includes one or more MHL inputs, and one or more digital outputs which are to carry HDCP streams based on the MHL input stream, are required to implement all of the HDCP Repeater features in each of the MHL input ports.

11.3 HDCP Compliance

MHL has been designed to be compliant with HDCP as applied to the HDMI interface. All references to HDMI in the HDCP spec shall be interpreted as being fully applicable to MHL, with the exceptions and clarifications documented in this section.

11.3.1 Data Encryption

HDCP Encryption is applied in the MHL transmitter on the three logical channels. Decryption is applied in the MHL receiver on the three logical channels. Thus, HDCP is applied to 24-bit data.

In the case of PackedPixel mode the encryption may be applied to only two of the three logical channels in the MHL transmitter, since only two logical channels are sent to the MHL receiver. This is an implementation detail within the transmitter and receiver, outside of the multiplexing of logical channels.

11.3.2 Stream Mapping

All references to TMDS Channels 0, 1, and 2 in the HDCP spec shall be interpreted as referring to MHL Logical Channels 0, 1, and 2 respectively.

The following tables are reproduced from the HDCP spec, with the addition of MHL stream mapping information.

Table 11-1. Encryption Stream Mapping

Cipher Output	TMDS Channel per HDCP	MHL: RGB Channel	Video Stream Bits
23:16	2	2 (Red)	Red[7:0]
15:8	1	1 (Green)	Green[7:0]
7:0	0	0 (Blue)	Blue[7:0]

Table 11-2. Encryption Stream Mapping for TERC4 Encoding

Cipher Output	TERC4 Bits per HDCP	MHL Logical Channel Bits
23:20	Unused	Unused
19	Channel 2 bit 3	Channel 2 bit 3
18	Channel 2 bit 2	Channel 2 bit 2
17	Channel 2 bit 1	Channel 2 bit 1
16	Channel 2 bit 0	Channel 2 bit 0
15:12	Unused	Unused
11	Channel 1 bit 3	Channel 1 bit 3
10	Channel 1 bit 2	Channel 1 bit 2
9	Channel 1 bit 1	Channel 1 bit 1
8	Channel 1 bit 0	Channel 1 bit 0
7:3	Unused	Unused
2	Channel 0 bit 2	Channel 0 bit 2
1:0	Unused	Unused

Other color bit mappings are possible when transmitting non-RGB data. Refer to the detailed descriptions of the pixel color bit assignments to each channel in Section 5.5.

11.3.2.1 Stream Mapping and PackedPixel Mode

Devices which handle encrypted streams, and convert from MHL format to non-MHL TMDS-based formats while maintaining encryption, may not be able to convert a PackedPixel input stream. Since the pixel data is encrypted, it is not possible for a conversion to be completed into non-MHL TMDS-based mode without first decrypting, since only sixteen (16) bits of pixel data appear in the stream per pixel. All non-MHL TMDS-based video modes (RGB, YcbCr 4:4:4, and YcbCr 4:2:2) require twenty-four (24) accurate bits per pixel, even though in the case of YcbCr 4:2:2 there may be unused bits. Those unused bits must be zeros.

If an MHL device needs to output a non-MHL, TMDS-based, HDCP encrypted stream; and the stream coming into that MHL device is HDCP encrypted and using PackedPixel mode; then that MHL device shall decrypt the incoming stream before converting to the output stream. The MHL Source device and the converting device shall recognize this limitation by exchanging accurate and complete Capability data. (See Section 7.9.1.) This exchange of Capability data may be necessary whenever HDCP is turned on or turned off for the link between MHL Source and a converting device.

11.3.3 Hot Plug Detect

The Hot Plug Detect function is provided using the SET_HPD and CLR_HPD CBUS commands. See Sections 0 and 7.4.3.4.

12 Physical Layer

12.1 Purpose and Scope

An MHL device may use any connection mapping, including from those contained in this section, chosen by the manufacturer. The Promoters hereby disclaim any and all representations and warranties, explicit and/or implicit, concerning the use of any connection mappings not listed in this section. Use of any non-listed connection is done so at the sole risk of the parties using it.

Authorized Test Centers (ATCs) and their associated test equipment are configured to test only those products using the sample connections listed in this section. When submitting to an ATC any product that uses a non-listed connection mapping, the submitter shall, at its expense and if requested by the ATC, provide the ATC an adapter that connects such non-listed connection mapping to one that is contained herein. Adapter shall be passive and shall not improve the DUT performance when compared to a direct compliance measurement at the DUT.

In the event that the owner(s) of the necessary rights to a given connector, to which a connection mapping is made, mandate that such connector be selected from a list of one or more connectors, such mapping shall be made to a listed connector. For example, mappings to HDMI Type A connectors shall be made to connectors listed on the following webpage: www.hdmi.org/manufacturer/approved_connectors.aspx. Also, mappings to Micro USB 2.0 connectors shall be made to connectors listed on the following webpage: www.usb.org/kcompliance/view.

Adopters should give careful attention to mechanical dimensions of Cable designs to avoid interfering with neighboring ports, and should also apply these limitations to Direct Attach Source designs.

12.2 HDMI Type A Connector

12.2.1 Normative References

The following standards contain provisions that, through reference in this text, constitute normative provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. If the referenced standard is dated, the reader is advised to use the version specified.

1. HDMI, LLC, "High-Definition Multimedia Interface Specification", Version 1.3a, 2006.

To the extent applicable, Adopter is strongly encouraged to obtain certain necessary rights to the connector referenced herein from HDMI Licensing, LLC (see http://www.hdmi.org/index.aspx).

12.2.2 Contact Assignments

MHL Adopters choosing to use the HDMI Type A connector (either plug or receptacle) shall map the five MHL connections to the contact numbers for the HDMI type A connector as shown in Table 12-1. "N/C" indicates a pin which is not used by MHL and is not connected to any other signal or level in the Sink end of the MHL cable.

MHL_Shield, TMDS_GND and VBUS_CBUS_GND are connected at the system level to the GND signal defined in the MHL Specification. Separate names are given here to facilitate cable construction.

MHL_Shield is intended to be closely associated with the MHL+ and MHL- signals in order to meet the electrical requirements of the MHL Specification. VBUS_CBUS_GND is intended to be closely associated with the VBUS and CBUS pins in order to meet the electrical requirements of the MHL Specification.

The HDMI signal names associated with the contact numbers in Table 12-1 may be found in the "High-Definition Multimedia Interface Specification", [Section 12.2.1, Normative Reference #1].

Table 12-1. HDMI Type A Pin Assignment List

Contact Number	MHL Name	Contact Number	MHL Name
1	N/C	11	TMDS_GND
2	CD_SENSE	12	N/C
3	N/C	13	N/C
4	N/C	14	N/C
5	TMDS_GND	15	CD_PULLUP
6	N/C	16	N/C
7	MHL+	17	VBUS_CBUS_GND
8	MHL_Shield	18	VBUS
9	MHL-	19	CBUS
10	N/C	Shell	Shield

12.2.3 Use of the HDMI Type A Connector

The current allowed by MHL when using the HDMI Type A connector may be higher than the current allowed in other uses of the connector.

12.2.3.1 MHL Sinks

An MHL Sink uses a receptacle to connect to an upstream Source via an MHL Cable.

The HDMI Type A receptacle is supported for use on the MHL input of an MHL Sink. MHL Sinks which choose the HDMI Type A connector shall use the Type A receptacle defined in Section 12.2.1, Normative Reference #1 on the MHL input of an MHL Sink.

The Sink shall be responsible for managing the detection of the proper mode or proper Cable, and protecting the pins and contacts from damage during that detection process.

An MHL Sink shall not drive signal levels onto the MHL+, MHL- or CBUS signal pins, nor drive any voltage on VBUS, in a manner which may damage a non-MHL device connected to this receptacle.

12.2.3.1.1 HDMI Type A Electrical Specification for Sinks

In addition to the requirement in the preceding section, MHL Sinks which use the Type A receptacle shall use a receptacle which guarantees the rated $I_{VBUS\_Sink\_to\_Source\_MHL2\_Sink}$ of their Sink design on the VBUS and GND pins. Sink shall not output more than $I_{CABLE\_VBUS\_MAX}$.

12.2.3.2 MHL Cables

The HDMI Type A plug is supported for use on the downstream end of an MHL Cable. MHL Cables which choose the HDMI Type A connector shall use the Type A plug defined in Section 12.2.1, Normative Reference #1 on the downstream end of the Cable.

12.2.3.2.1 HDMI Type A Electrical Specification for Cables

In addition to the requirement in paragraph 1 of Section 12.2.3.2, MHL Cables which use the Type A plug shall use a plug which guarantees $I_{MINIMUM\_VBUS\_CURRENT\_FOR\_PLUG}\{min\}$ on the VBUS and GND pins.

Table 12-2. Electrical Specifications for HDMI Type A Receptacle for Cables

Item	Symbol	Min	Max	Unit
Minimum VBUS Current for Type A Plug	$I_{MINIMUM\_VBUS\_CURRENT\_FOR\_PLUG}$	1.8		A

12.2.4 HDMI Type A Cable Detect Mechanism

12.2.4.1 MHL Sink and HDMI Type A Cable Detect

MHL Sinks which choose to use the HDMI Type A receptacle shall incorporate the circuit shown in Figure 12-1 and detect the presence or absence of an attached MHL Cable using the input voltage levels shown in Table 12-3. If the voltage on CD_SENSE (Pin #2) of the HDMI Type A receptacle is less than or equal to $V_{CABLE\_DETECT\_IL\_TYPE\_A}$, then the MHL Sink shall conclude that no MHL Cable is attached. If the voltage on CD_SENSE (Pin #2) of the HDMI Type A receptacle is greater than or equal to $V_{CABLE\_DETECT\_IH\_TYPE\_A}$, then the MHL Sink shall conclude that an MHL Cable is attached.

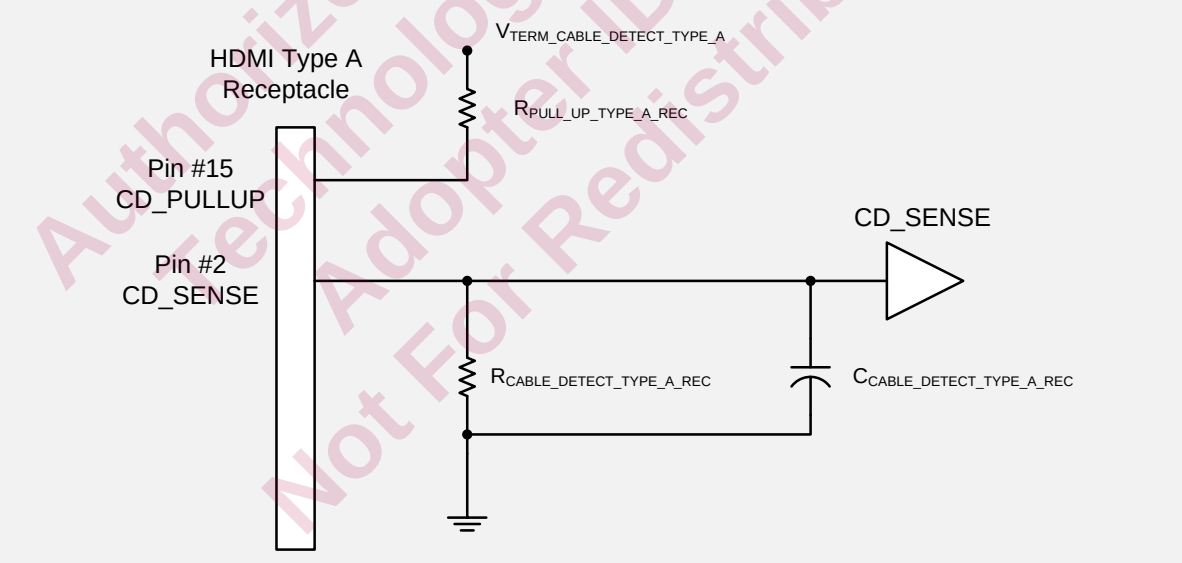


Figure 12-1. Cable Detect Circuit for MHL Sink with HDMI Type A Receptacle

Table 12-3. Cable Detect AC Electrical Specifications for MHL Sink with HDMI Type A Receptacle

Item	Symbol	Min	Typ	Max	Unit	Notes
Cable Detect Pull-up Resistor	$R_{PULL\_UP\_TYPE\_A\_REC}$	42.3k	47k	51.7k	ohms	1
Cable Detect Pull-down Resistor	$R_{CABLE\_DETECT\_TYPE\_A\_REC}$	270k	300k	330k	ohms	
Cable Detect Pull-down Capacitor	$C_{CABLE\_DETECT\_TYPE\_A\_REC}$	40	47		nF	
Termination Voltage	$V_{TERM\_CABLE\_DETECT\_TYPE\_A}$	2.97		5.3	V	
Detection Input High Voltage	$V_{CABLE\_DETECT\_IH\_TYPE\_A}$	2.0			V	2
Detection Input Low Voltage	$V_{CABLE\_DETECT\_IL\_TYPE\_A}$			0.8	V	

Notes on Table 12-3:

1. The total resistance from pin #15 to $V_{TERM\_CABLE\_DETECT\_TYPE\_A}$ shall be in this range. If an existing circuit in the MHL Sink on pin #15 already provides a termination pull-up, it may not be necessary to add an additional resistor.
2. MHL Sinks which choose to use HDMI Type A receptacle may drive VBUS only when CD_SENSE=HIGH.

12.2.4.2 MHL Cable and HDMI Type A Cable Detect

MHL Cables which choose to use the HDMI Type A plug shall incorporate the circuit shown in Figure 12-2, using components specified in Table 12-4. This circuit, in combination with the method for MHL Sinks defined in Section 12.2.4.1, allows an MHL Sink to detect an attached MHL Cable.

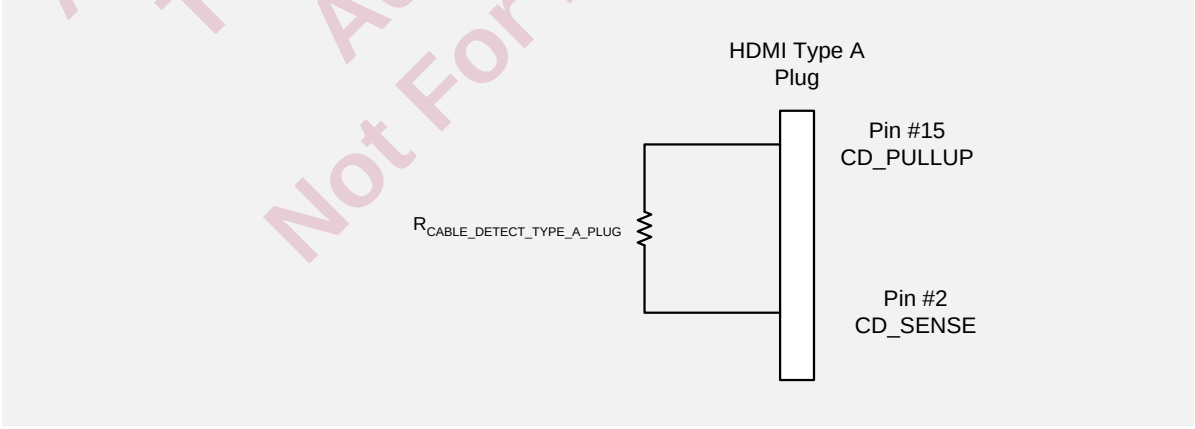


Figure 12-2. Cable Detect Circuit for MHL Cable with HDMI Type A Plug

Table 12-4. Cable Detect AC Electrical Specifications for MHL Cable with HDMI Type A Plug

Item	Symbol	Min	Typ	Max	Unit
Cable Detect Plug Resistor	$R_{CABLE\_DETECT\_TYPE\_A\_PLUG}$	2.97	3.3	3.63	Kohms

12.3 Micro USB Connector

12.3.1 Normative References

The following standards contain provisions that, through reference in this text, constitute normative provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. If the referenced standard is dated, the reader is advised to use the version specified.

1. USB Implementers Forum, "Universal Serial Bus Micro-USB Cables and Connectors Specification, Revision 1.01" (April 2007)

To the extent applicable, Adopter is strongly encouraged to obtain certain necessary rights to the connectors referenced herein from the USB Implementers Forum, Inc. (see http://www.usb.org/home).

12.3.2 Contact Assignments

MHL Adopters choosing to use the Micro USB connector (either plug or receptacle) shall map the five MHL connections to the contact numbers for the Micro USB connector as shown in Table 12-5.

Table 12-5. Pin Assignment List

Contact Number	MHL Name
1	VBUS
2	MHL-
3	MHL+
4	CBUS
5	GND
Shell	Shield

The USB signal names associated with the contact numbers in Table 12-5 may be found in the "Universal Serial Bus Micro-USB Cables and Connectors Specification, Revision 1.01" (April 2007) [Section 12.3.1, Normative Reference #1.]

12.3.3 Use of the Micro USB Connector

12.3.3.1 MHL Sources

An MHL Source uses a receptacle to connect to a downstream Sink via an MHL Cable or Dongle.

The Source shall be responsible for managing the detection of the proper mode and protecting the pins and contacts from damage during that detection process.

If an MHL Source with the Micro USB connector, when operating in the non-MHL mode, may not expect 5 volts on VBUS, then the MHL Source shall prevent VBUS from actively driving the VBUS pin until MHL mode is conclusively detected.

Similarly, an MHL Source shall not drive signal levels onto the MHL+, MHL- or CBUS signal pins in a manner which may damage a non-MHL device connected into this receptacle.

The Source may choose to test the port in the non-MHL mode first, passing control to the MHL discovery logic only when an attached downstream device is not found to be used in the non-MHL mode.

12.3.3.1.1 Micro-AB Receptacle

A Micro-AB receptacle defined in Section 12.3.1, Normative Reference #1 allows the Source device to accept power from the downstream device (Sink or Dongle), or to output power to a Sink or Dongle.

12.3.3.1.2 Micro-B Receptacle

A Micro-B receptacle Normative Reference #1 allows the Source device to accept power from the downstream device (Sink or Dongle), but does not allow the Source to output power to a Sink or Dongle.

12.3.3.2 MHL Cables

The Micro-B plug is supported for use on the upstream end of an MHL Cable.

MHL Cables which choose the Micro USB connector shall use the Micro-B plug defined in Section 12.3.1, Normative Reference #1 on the upstream end of the Cable, since all MHL Sinks provide VBUS power to the Source.

12.3.3.3 MHL Dongles

12.3.3.3.1 Micro-A Plug

The Micro-A plug is supported for use on the upstream end of an MHL Dongle. MHL Dongles which choose the Micro USB connector and which are capable to operate only with power from an attached Source shall use the Micro-A plug defined in Normative Reference #1 on the upstream end of the Dongle. Such a Dongle operates only when plugged into an MHL Source with the Micro-AB receptacle on its output port.

12.3.3.3.2 Micro-B Plug

The Micro-B plug is supported for use on the upstream end of an MHL Dongle. MHL Dongles which choose the Micro USB connector and which can provide power to an attached MHL Source shall use the Micro-B plug defined in Normative Reference #1 on the upstream end of the Dongle. Such a Dongle may operate when plugged into an MHL Source with the Micro-AB receptacle on its output port, or when plugged into an MHL Source with the Micro-B receptacle on its output port.

13 Electrical Specification

Schematic diagrams contained in this section are for illustration only and do not represent the only feasible implementation.

13.1 MHL Overview

Some timing parameter values in this specification are based on the clock time period (T_{MHL}) and data bit time (T_{BIT}) of the link and the pixel clock time period (T_{PIXEL}) while others are based on absolute values. One tenth of the 'character time' is called the bit time, or T_{BIT}. In 24-bit mode, one pixel occupies thirty bit times (30 T_{BIT}) on the link. In PackedPixel mode, one pixel occupies twenty bit times (20 T_{BIT}) on the link.

The conceptual schematic of the MHL differential pair is shown in Figure 13-1. The MHL data signal is carried from the source to the sink on a differential pair as a TMDS signal that uses a current drive to develop the low voltage differential signal at the sink side of the DC-coupled transmission line or interconnect. The MHL clock signal is transmitted as a common-mode signal of the differential signal.

The following four impedances shall be met:

- the interconnect common-mode characteristic impedance (Z_{OC})
- the interconnect differential characteristic impedance (Z_{OD})
- the sink common-mode termination impedance (Z_{CT}).
- the sink differential termination impedance (Z_{DT})

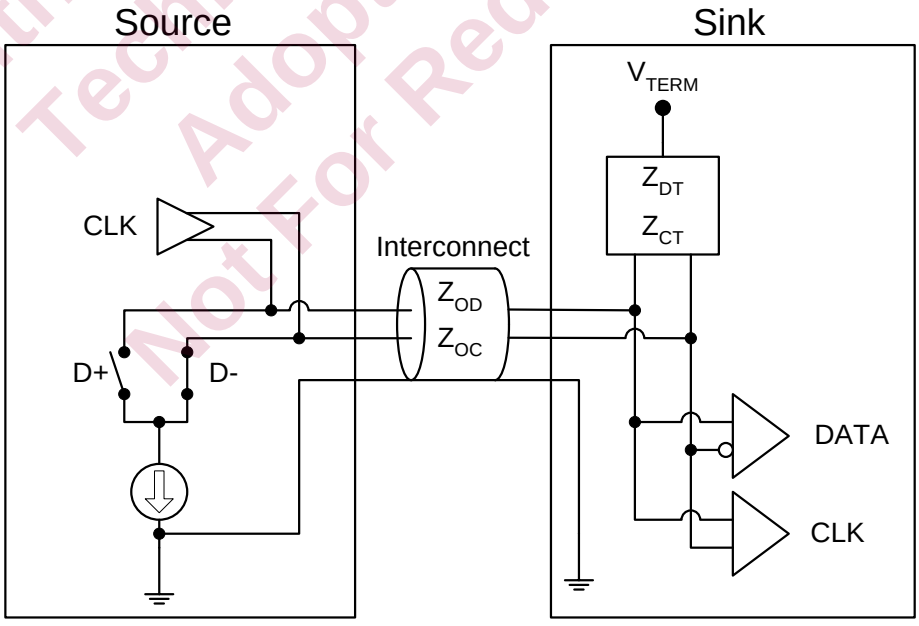


Figure 13-1. Conceptual Schematic for an MHL Differential Pair

Single-ended signals of the differential pair, representing the positive and negative terminals of the differential pair, are illustrated in Figure 13-2. The high-level and low-level voltages of the single-ended signal are determined by termination voltage (VTERM), differential swing voltage (VDFSWING) and common-mode swing voltage (VCMSWING). The differential signal is obtained by subtracting the negative line single-ended signal from the positive line single-ended signal as shown in Figure 13-2. The common-mode signal is obtained by averaging the two single-ended signals. The DC voltage level for each single-ended signal (VDC) is obtained by averaging each single-ended signal over a period of at least one horizontal line time.

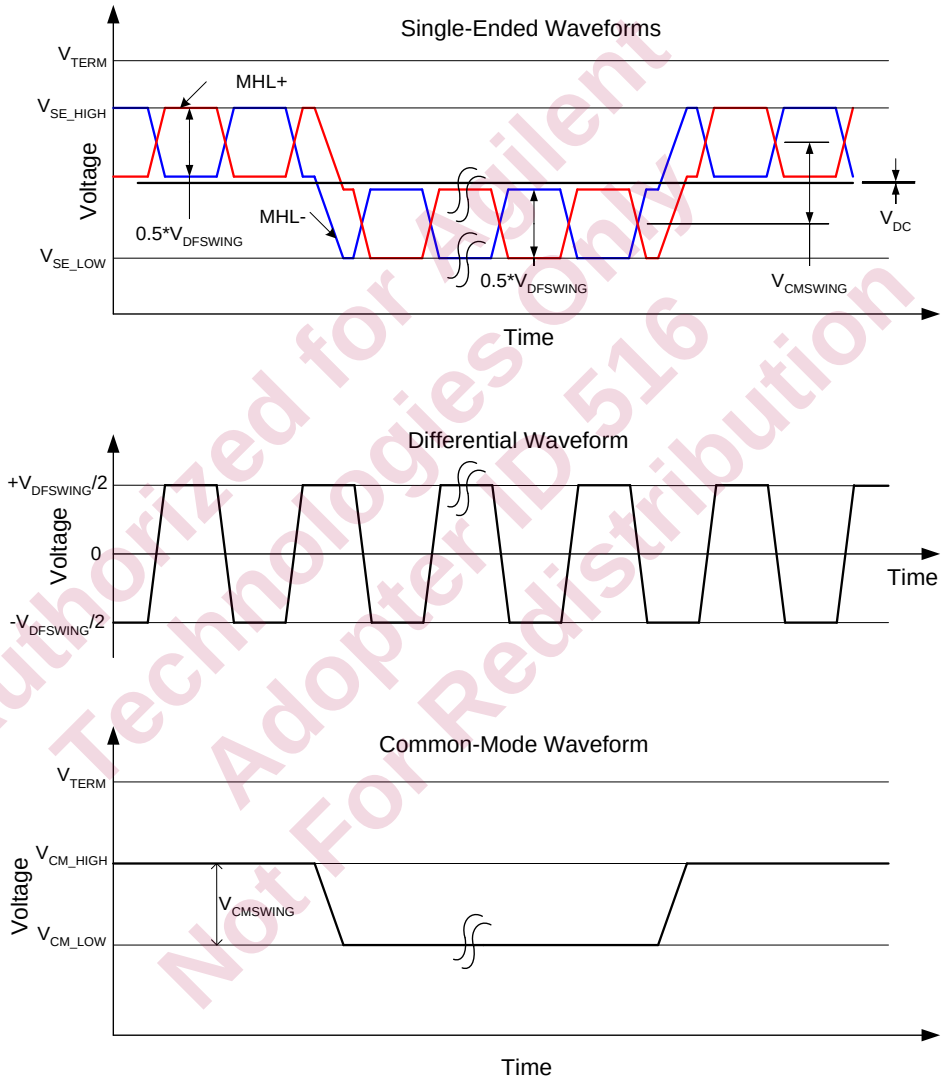


Figure 13-2. Single-ended, Differential and Common-Mode Signals

13.2 Operating Conditions

The MHL interface shall work across the operating conditions specified in Table 13-1:

Table 13-1. Operating Conditions

Item	Symbol	Value
Termination Supply Voltage	V_{TERM}	3.3 Volts, ±5%
Termination Impedance, Differential	Z_{DT}	100 ohms, ±10 ohms
Termination Impedance, Common Mode	Z_{CT}	30 ohms, ±5 ohms

13.3 Electrical Specification Methodology

The following sections define the parametric measurements required to assess MHL compliance, and define the measurement methods for those parameters.

13.3.1 MHL Measurement Methods

Note that power may be required from the attached test fixture when testing an MHL Source. Since power may flow from Sink to Source after the negotiation of the interconnect (refer to Sections 8 and 9), the test fixture may also require the functionality of MHL discovery in order to provide power to the Source-under-test.

13.3.1.1 MHL Signal Test Points

The signal test points for an MHL link are shown in Figure 13-3. TP1 is used for testing of MHL Source devices and transmitter components. TP2 is used for testing of MHL Sink devices and receiver components. TP1 and TP2 together are also used for testing of cables.

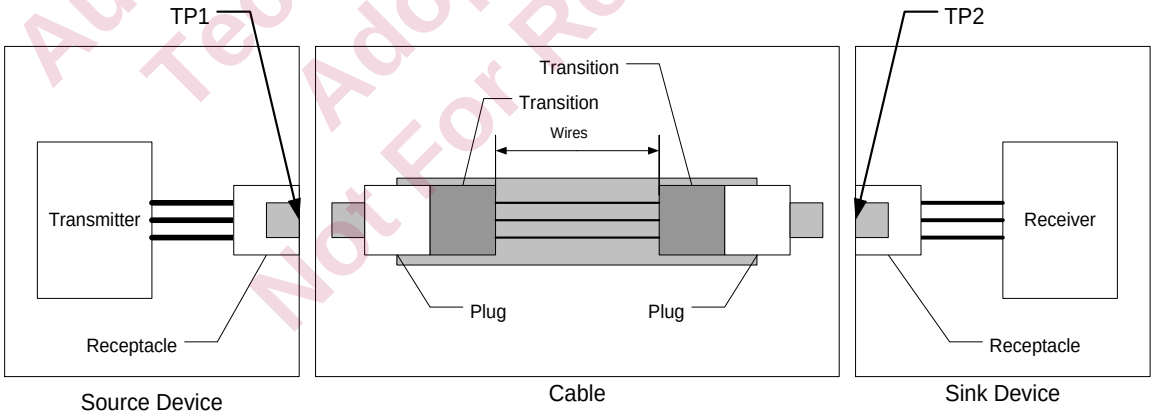


Figure 13-3. MHL Link Test Points

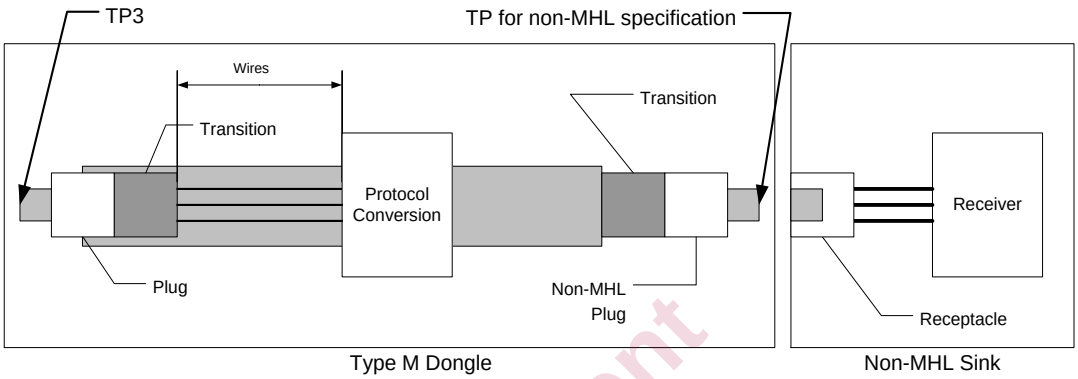


Figure 13-4. MHL Type M Dongle Test Points

A Type M Dongle is a Dongle with male plugs on both ends.

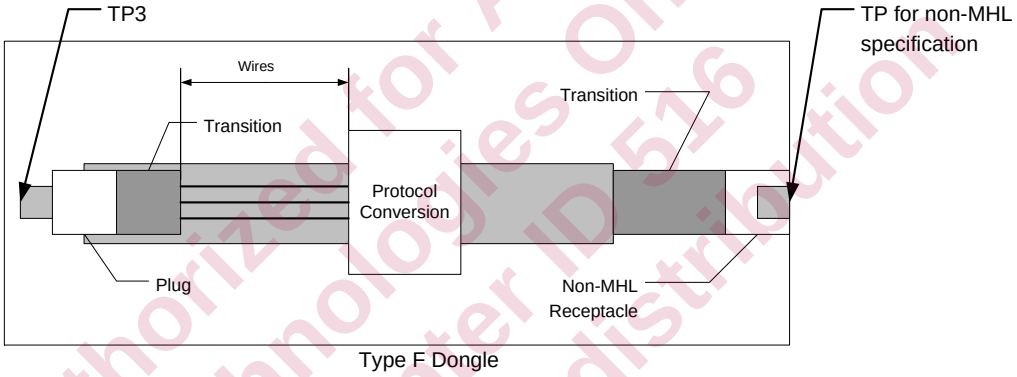


Figure 13-5. MHL Type F Dongle Test Points

A Type F Dongle is a Dongle with a male plug on the upstream side (to connect to the upstream Source which has a female receptacle), and a female receptacle on the downstream, non-MHL side. The downstream side is connected to a downstream Sink using a conventional male-to-male cable.

13.3.1.2 Source Device Test Points

MHL Source Device electrical testing shall be performed using a test signal analyzer as shown in Figure 13-6.

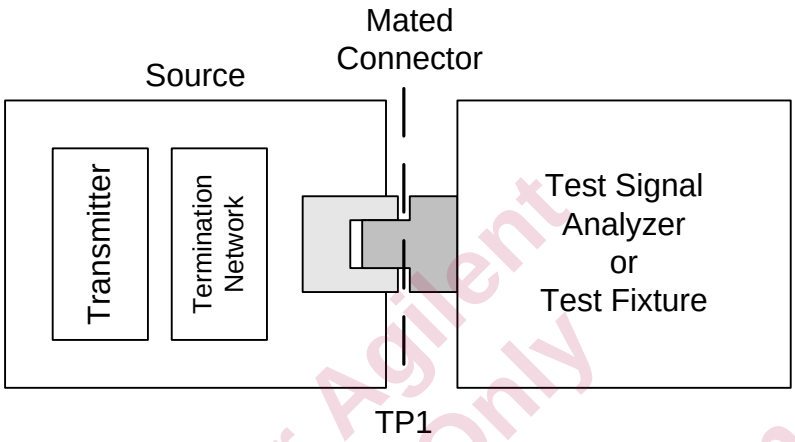


Figure 13-6. Source Device Test Points

13.3.1.3 Source Device Test Load

Source device electrical testing shall be performed using the test load shown in Figure 13-7. TP1 represents the connection point of the mated connector.

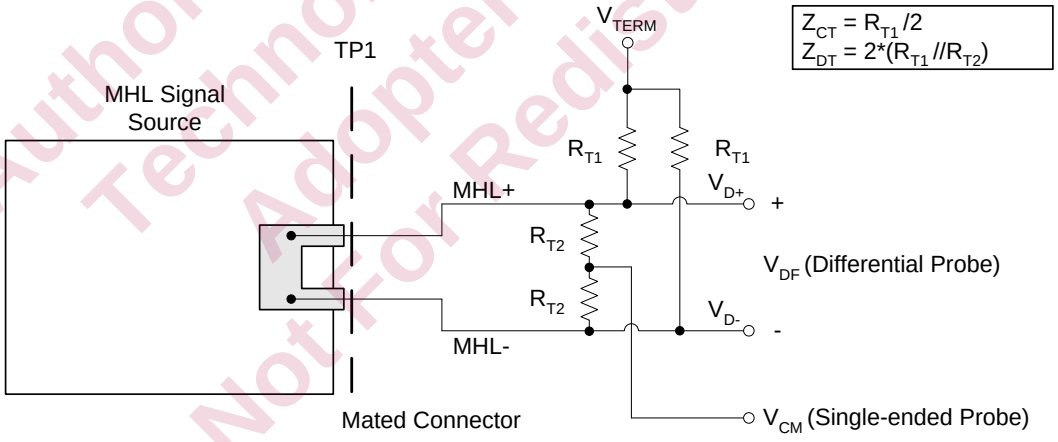


Figure 13-7. Source Device Test Load

13.3.1.4 Test Signal Generator Qualification Load

TP2 and TP3 testing shall be performed using the load shown in Figure 13-8, applied to the test signal generator to qualify its output signal.

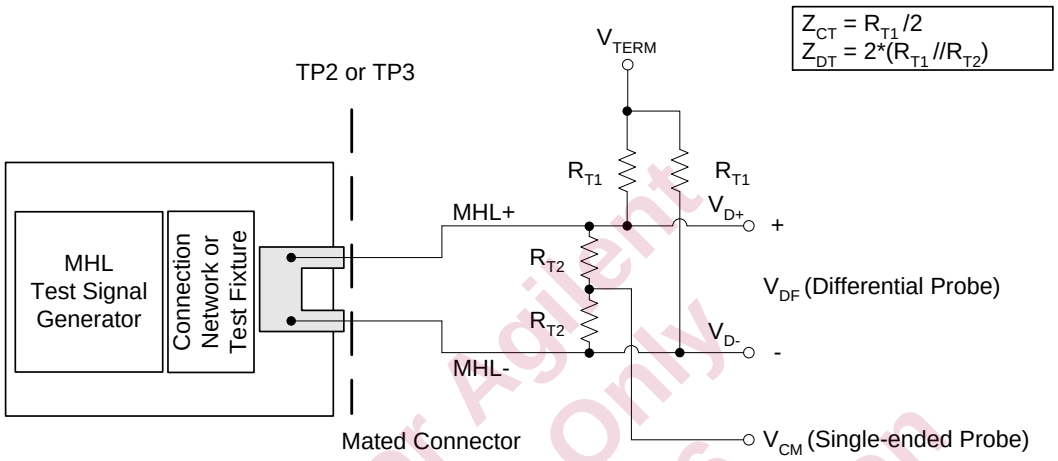


Figure 13-8. Test Signal GeneratorTest Load

13.3.1.5 Sink Device Test Points

MHL Sink Device electrical testing shall be performed using a test signal generator as shown in Figure 13-9. TP2 represents the connection point of the mated connector.

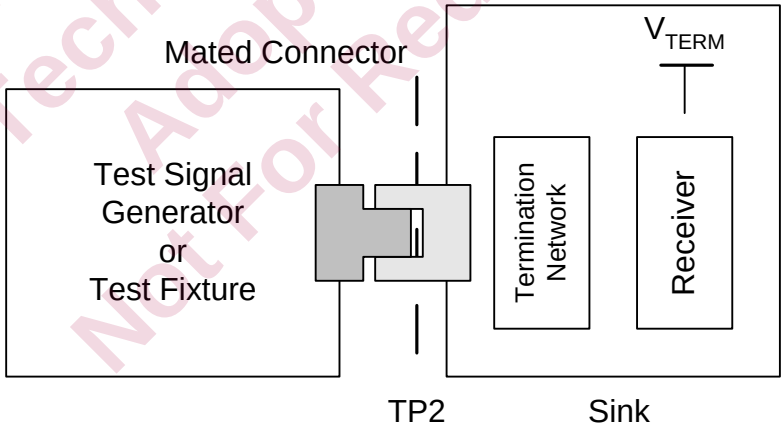


Figure 13-9. Sink Device Test Points

13.3.1.6 Dongle Device Test Points

MHL Dongle Device electrical testing shall be performed using a test signal generator as shown in Figure 13-10. TP3 represents the connection point of the mated connector for both Type M and Type F Dongles.

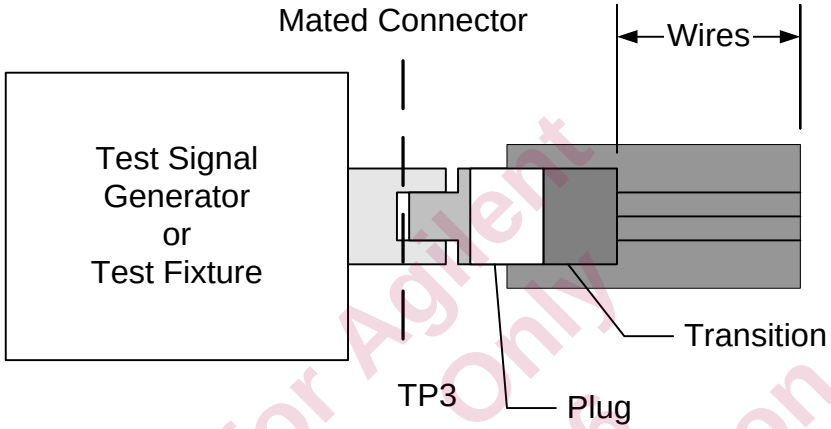


Figure 13-10. Dongle Device Test Points

13.3.1.7 Reference Cable Equalizer

The reference cable equalizer shall meet the transfer function shown in Figure 13-11 and Equation 6, according to the parameters shown in Table 13-2. This cable equalizer shall be used to produce the MHL eye diagram at TP2. See Figure 13-21. The MHL receiver shall recover an input data stream which is compliant to the TP2 eye diagram after application of this reference cable equalizer.

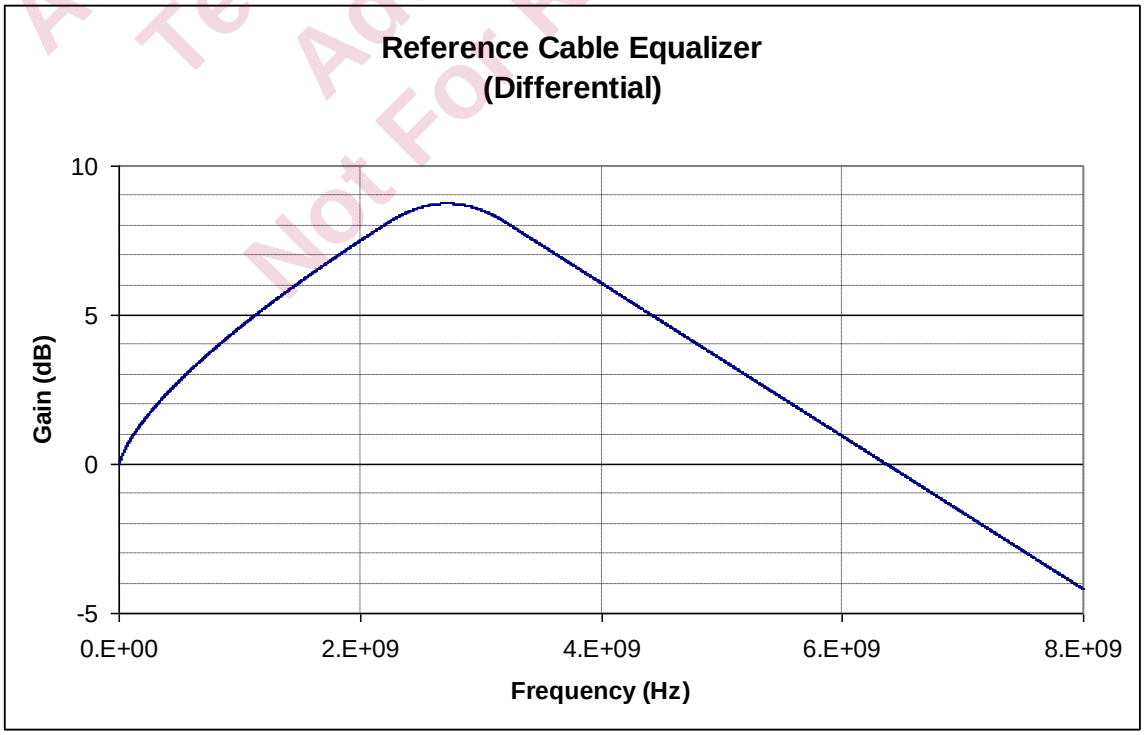


Figure 13-11. Reference Cable Equalizer Gain Curve

Equation 6. Amplitude Transfer function of Reference Cable Equalizer

$$|H(j\omega)| = e^{A\omega^N}, \omega < \omega_0$$

$$|H(j\omega)| = e^{\left(-B(\omega-\omega')^2+C\right)}, \omega_0 < \omega < \omega_1$$

$$|H(j\omega)| = e^{(-D\omega+E)}, \omega_1 < \omega$$

Table 13-2 Amplitude Transfer Function Parameters for Reference Cable Equalizer

Symbol	Value	Symbol	Value
ω_0	$2\pi \times 2.25GHz$	A	6.25×10^{-8}
ω_1	$\sqrt{2} \times \omega_0$	B	$\frac{N \times A \times {\omega_0}^{N-2}}{10 \times (N-0.5)^2}$
ω'	$\frac{\omega_0 + \omega_1}{2}$	C	$\left(1+\frac{N}{10}\right) \times A \times {\omega_0}^N$
N	$\frac{1}{\sqrt{2}}$	D	$N \times A \times {\omega_0}^{(N-1)}$
		E	$\frac{A \times {\omega_0}^N}{N^2}$

13.3.1.8 MHL Clock Recovery Unit

All MHL jitter and eye diagram specifications are specified relative to a Recovered Clock, generated by a Clock Recovery Unit (CRU). The clock to data jitter is not specified numerically, but instead, an MHL system or cable shall adhere to the appropriate eye diagram(s) when the MHL data signals are measured using a compliant Clock Recovery Unit as a trigger source.

The clock signal that is transmitted on the MHL link may contain low-frequency jitter components, which should be tracked by a receiver's clock recovery circuitry, and high-frequency components, which should not be tracked. The purpose of the Clock Recovery Unit is to similarly remove these high frequency jitter components, producing a clock that will give an accurate representation of link performance when used as a trigger for eye diagram and clock jitter measurements.

Figure 13-12 shows a block diagram of a setup for eye diagram measurement. The eye diagram and clock jitter produced by this method will contain only high frequency jitter components that are not tracked by the clock recovery circuit of the receiver.

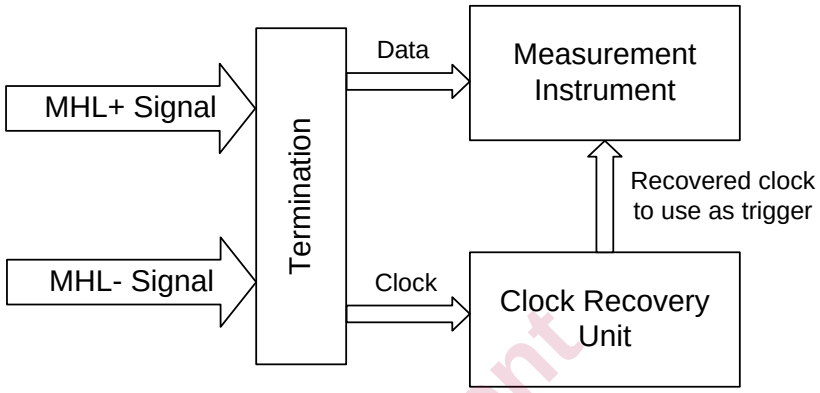


Figure 13-12. Clock Recovery Unit in Eye Diagram Measurement

The Clock Recovery Unit may be any circuit or software algorithm that provides the jitter transfer characteristic shown in Equation 7.

Equation 7. Jitter Transfer Function of Clock Recovery Unit

$$H(jf) = \frac{1}{\left[1 + \frac{jf}{f_0}\right]}$$, where f_0 is 4 MHz

The Clock Recovery Unit may use for its input signal a hardware circuit (buffer or low pass filter) or equivalent software algorithm to remove voltage noise glitches on the MHL clock transition edges. The bandwidth of the circuit or software algorithm shall be at least 3 times as high as the 20dB/decade cut-off frequency, determined by the rise and fall times of the clock edge ($1/\pi t_R$).

13.4 TMDS Protocol Timing Specifications

Table 13-3. TMDS Link Timing Parameters – 24-Bit Mode

Description	Symbol	Min	Max	Unit	Notes
Maximum time between occasional long Control Periods	$T_{CTL\_LONG\_PER}$		50	ms	
Minimum duration of occasional long Control Periods	T_{CTL_LONG}	32		$3*T_{CHAR}$	
Maximum variation in recovery pipeline	Δt_R		1	$3*T_{CHAR}$	
Minimum duration Control Period	T_{SMIN}	12		$3*T_{CHAR}$	

Table 13-4. TMDS Link Timing Parameters – PackedPixel Mode

Description	Symbol	Min	Max	Unit	Notes
Maximum time between occasional long Control Periods	$T_{CTL\_LONG\_PER}$		50	ms	
Minimum duration of occasional long Control Periods	T_{CTL_LONG}	32		$2*T_{CHAR}$	
Maximum variation in recovery pipeline	Δt_R		1	$2*T_{CHAR}$	
Minimum duration Control Period	T_{SMIN}	12		$2*T_{CHAR}$	

13.5 Source Device Electrical and Timing Specs

13.5.1 Source Device DC Characteristics at TP1

The Source device, when measured with the test load shown in Figure 13-7, shall meet the DC specifications in Table 13-5 for the operating conditions specified in Table 13-1. The single-ended high level (V_{SE_HIGH}) and low level (V_{SE_LOW}) output voltages are measured at either V_{D+} or V_{D-} port in the test load using a high impedance single-ended probe. The differential swing voltage ($V_{DFSWING}$) is measured by a high impedance differential probe that is applied between V_{D+} and V_{D-} ports, as in Figure 13-7, or by equivalent method. The common-mode swing voltage ($V_{CMSWING}$) is measured by a high impedance single-ended probe that is applied to the common mode tapping point (V_{CM}) in Figure 13-7, or an equivalent method.

Table 13-5. Source device DC Characteristics at TP1

Item	Symbol	Min	Typ	Max	Unit	Notes
Single-ended standby (off) output voltage	V_{OFF}	V_{TERM}-10	V_{TERM}	V_{TERM}+10	mV	1, 2
Single-ended high level output voltage	V_{SE_HIGH}	V_{TERM}-540		V_{TERM}+10	mV	2, 3
Single-ended low level output voltage	V_{SE_LOW}	V_{TERM}-1760		V_{TERM}-700	mV	2, 3
Differential output swing voltage	$V_{DFSWING}$	600		1000	mV	3
Common-mode output swing voltage	$V_{CMSWING}$	360		Min(720, 0.85$V_{DFSWING}$)	mV	3, 4

Notes on Table 13-5:

1. V_{OFF} is the Source output voltage when terminated to V_{TERM} through the test load in Figure 13-7, and the Source device is in Standby mode or power off mode.
2. V_{TERM} is defined in Table 13-1.
3. Refer to Figure 13-2.
4. Common-mode output swing voltage shall not exceed the smaller of the two terms.

13.5.2 Source Device AC Characteristics at TP1

The Source shall meet the AC specifications in Table 13-6 for the operating conditions specified in Table 13-1. Rise and fall times are defined as the signal transition time between 20% and 80% of the measured differential ($V_{DFSWING}$) and common-mode ($V_{CMSWING}$) swing voltages of the device under test.

The Source intra-pair skew is the maximum allowable time difference (on both low-to-high and high-to-low transitions) as measured at TP1, between the positive and negative signals. This time difference is measured at the midpoint on the single-ended signal swing of the positive and negative signals. The Source intra-pair skew shall be measured for both differential and common-mode transition edges of the two single-ended signals.

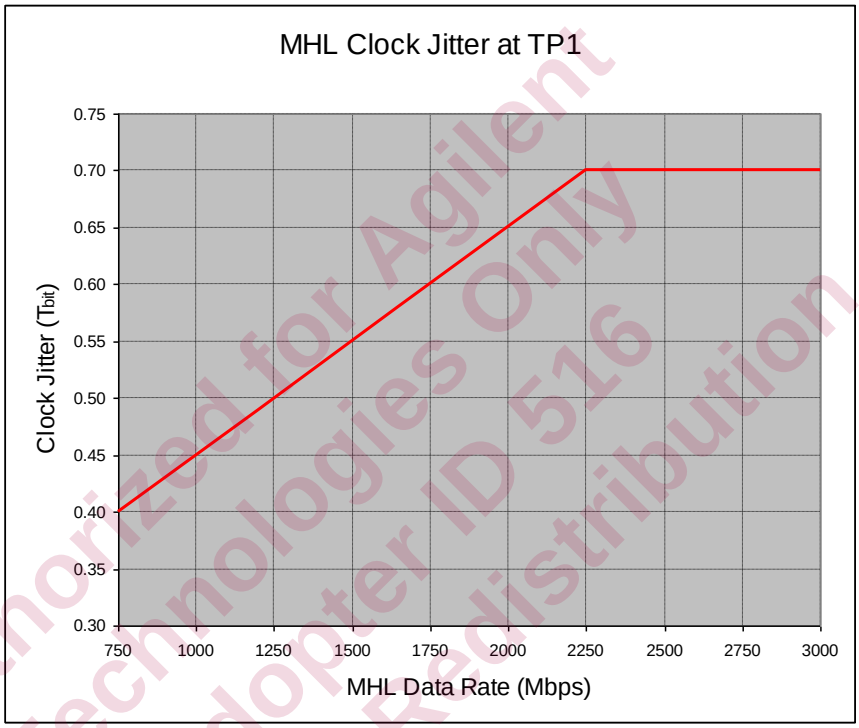
Table 13-6. Source device AC Characteristics at TP1

Item	Symbol	Min	Max	Unit	Notes
Pixel Clock Frequency	F_{PIXEL}	25	150	MHz	1
MHL Link Clock Frequency 24-Bit Mode	F_{MHL}	25	75	MHz	
MHL Link Clock Period 24-Bit Mode	T_{MHL}	13.33	40	nsec	2
MHL Link Clock Frequency PackedPixel Mode	F_{MHL_PP}	25	75	MHz	
MHL Link Clock Period PackedPixel Mode	T_{MHL_PP}	13.33	40	nsec	2
Bit Time on Link	T_{BIT}	333	1333	psec	
Differential Rise Time	T_{R_DF}	75		psec	3, 4
Differential Fall Time	T_{F_DF}	75		psec	3, 4
Common-Mode Rise Time	T_{R_CM}	600	2500	psec	3
Common-Mode Fall Time	T_{F_CM}	600	2500	psec	3
Differential Intra-Pair Skew	T_{SKEW_DF}		Min(0.12T_{BIT}, 50)	psec	5, 7
Common-Mode Intra-Pair Skew	T_{SKEW_CM}		Min(0.12T_{BIT}, 50)	psec	6, 7
Clock Duty Cycle 24-Bit Mode		35	65	%T_{MHL}	
Clock Duty Cycle PackedPixel Mode		35	65	%T_{MHL_PP}	

Notes on Table 13-6:

1. Pixel clock above 75 MHz is allowed only when output is in PackedPixel mode.
2. MHL Clock Period is the inverse of the MHL Clock Frequency (F_{MHL}), and is not a controlling specification.
3. Rise and fall times are measured for 20% to 80% transition.
4. Differential rise and fall times shall be measured for specific patterns defined in MHL Compliance Test Specification.
5. Differential intra-pair skew is measured as the time difference between two mid-points on the same T_{BIT} of the differential transition edges, one on MHL+ and one on MHL-.

6. Common mode intra-pair skew is measured as the time difference between two mid-points of the common-mode transition edges, one on MHL+ and one on MHL-.
7. Differential and Common-Mode Intra-Pair Skew compliance of a Source DUT can be alternatively tested by TP2 clock jitter (Figure 13-20) and eye diagram (Figure 13-21) compliance when the worst case cable emulator is connected to the Source device. The worst case cable emulator is defined by cable differential insertion loss (13.8.2.1), common-mode insertion loss (13.8.2.2), differential intra-pair skew (13.8.1) and common-mode intra-pair skew (13.8.1) specifications.



$T_{CLK\_JITTER\_TP1}$ = 0.25T_{BIT}+200ps up to 2.25Gbps. $T_{CLK\_JITTER\_TP1}$ = 0.7T_{BIT} above 2.25Gbps.

Figure 13-13. MHL Clock Jitter at TP1

The Source device shall meet the MHL clock jitter requirement (T_{CLOCK_JITTER}) in Figure 13-13. The MHL clock jitter at TP1 shall be measured by the MHL Clock Recovery Unit.

The Source device clock jitter compliance may be alternatively tested by TP2 clock jitter compliance (Figure 13-20) when the worst case cable emulator is connected to the Source device.

13.5.3 Source Device Eye Diagram Characteristics at TP1

The Source device shall meet the differential eye diagram requirements of Figure 13-14.

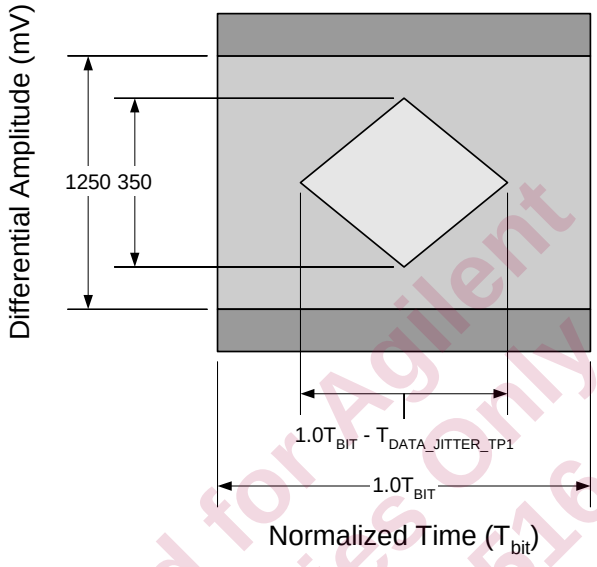
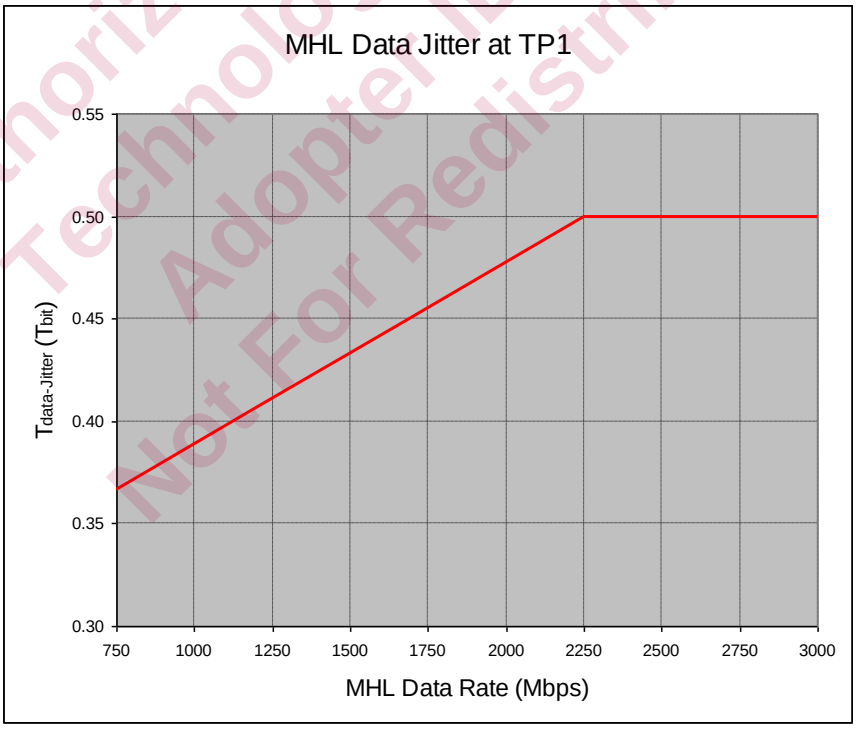


Figure 13-14. Differential Eye Diagram Mask at TP1 for Source Device Requirements



$T_{DATA\_JITTER\_TP1}$ = 0.3T_{BIT}+88.88psec up to 2.25Gbps. $T_{DATA\_JITTER\_TP1}$ = 0.5T_{BIT} above 2.25Gbps.

Figure 13-15. MHL Data Jitter at TP1

The Source device shall meet the MHL data jitter requirement ($T_{DATA\_JITTER\_TP1}$) in Figure 13-15. The differential data eye diagram at TP1 shall be measured by the MHL Clock Recovery Unit. The mask for data eye opening at TP1 may be shifted horizontally from the center of the data bit time. Vertical position of the eye opening mask shall be centered at 0V.

The Source device eye diagram compliance may be alternatively tested by TP2 eye diagram compliance when the worst case cable emulator is connected to the Source device.

13.5.4 Source Device TP2 Clock Jitter and Eye Diagram

The Source device shall meet the MHL clock jitter and eye diagram requirements of Figure 13-20 and Figure 13-21, using the output signal from the Source, processed through an emulation of the worst-case MHL cable performance. The cable emulation tightens the signal's eye diagram by adding timing skew in the TMDS signal pair and cable insertion loss. The required skew shall be achieved by skewing the TMDS signal pair first with a worst-case cable skew in one direction, and then with a worst-case cable skew in the opposite direction for both differential and common-mode signals. Cable insertion losses of differential and common-mode signals shall be added to achieve the worst case insertion loss requirements.

The cable emulation shall be calculated to meet the electrical performance shown in Figure 13-16 through Figure 13-19.

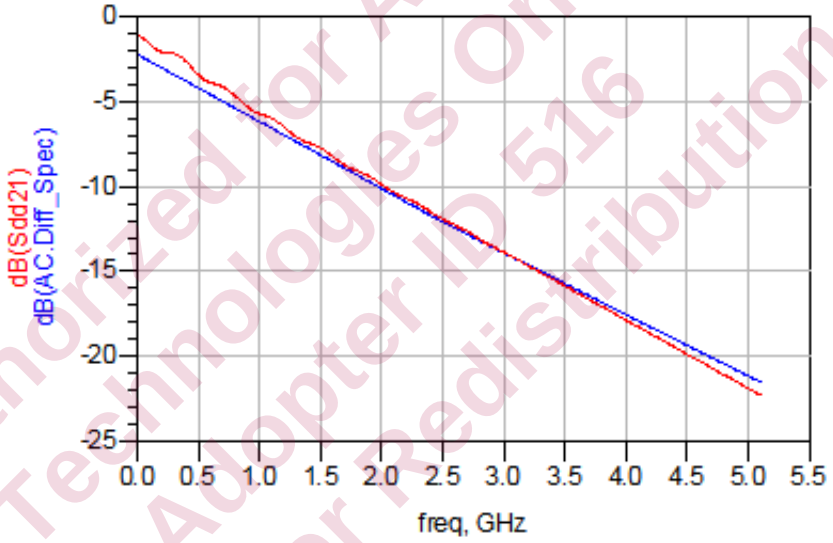


Figure 13-16. Cable Emulation Model –Differential Insertion Loss

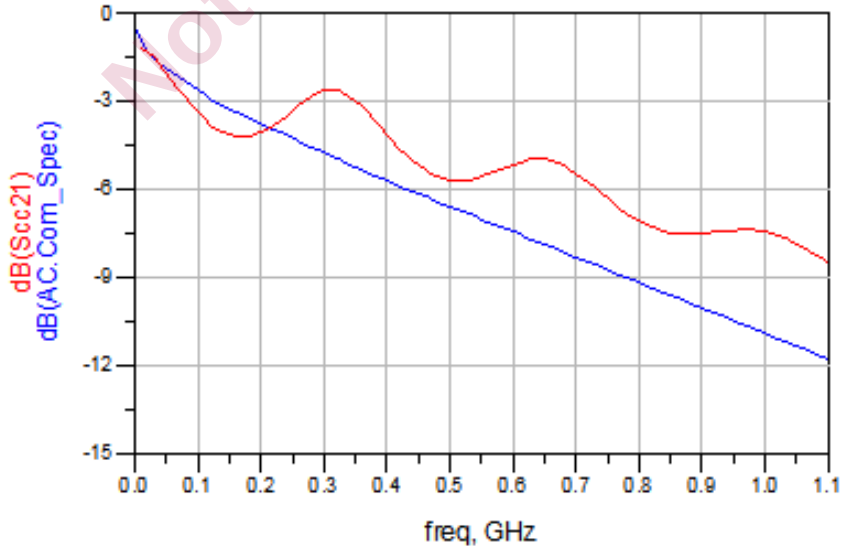


Figure 13-17. Cable Emulation Model –Common Mode Insertion Loss

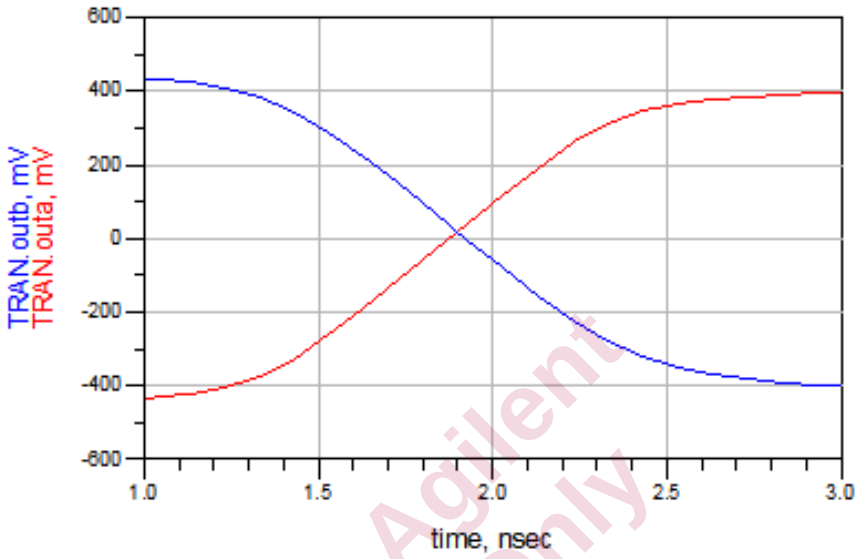


Figure 13-18. Cable Emulation Model –Differential Intra-Pair Skew

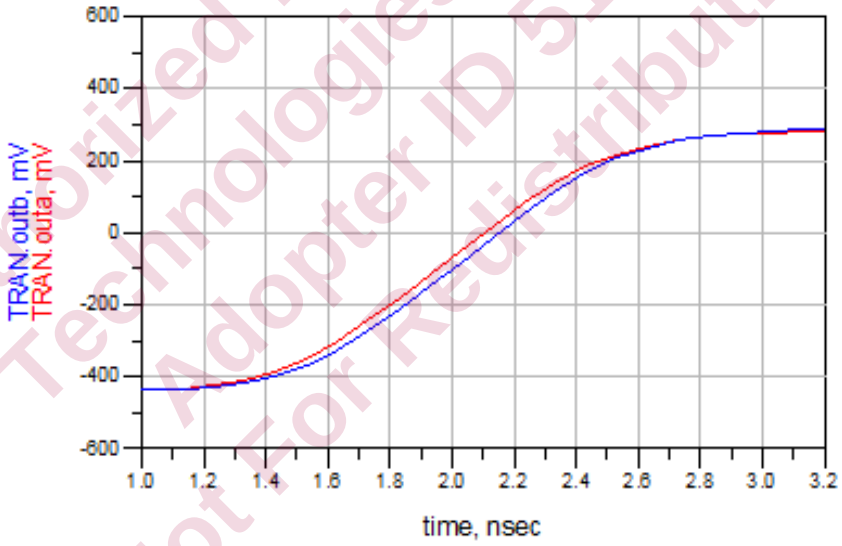


Figure 13-19. Cable Emulation Model –Common Mode Intra-Pair Skew

13.6 Sink Device Electrical and Timing Specs

13.6.1 Sink Device DC Characteristics at TP2

The receiver signal at TP2 shall meet the signal requirements listed in Table 13-7. Single-ended, differential and common-mode voltages shall be calibrated to the required conditions using the test load specified in Figure 13-8 and Table 13-1.

Table 13-7. Receiver DC Characteristics at TP2

Item	Symbol	Min	Max	Unit	Notes
Single-ended input DC Voltage	V_{IDC}	V_{TERM}-1200	V_{TERM}-300	mV	1
Differential input swing voltage	V_{IDF}	200	1000	mV	
Common-mode input swing voltage	V_{ICM}	170	Min(720, 0.85V_{IDF})	mV	2

Notes on Table 13-7:

1. Input DC voltage is the average voltage of a single-ended signal measured over at least one horizontal line time.
2. The maximum common-mode input swing voltage shall not exceed the smaller of the two terms shown.

13.6.2 Sink Device AC Characteristics at TP2

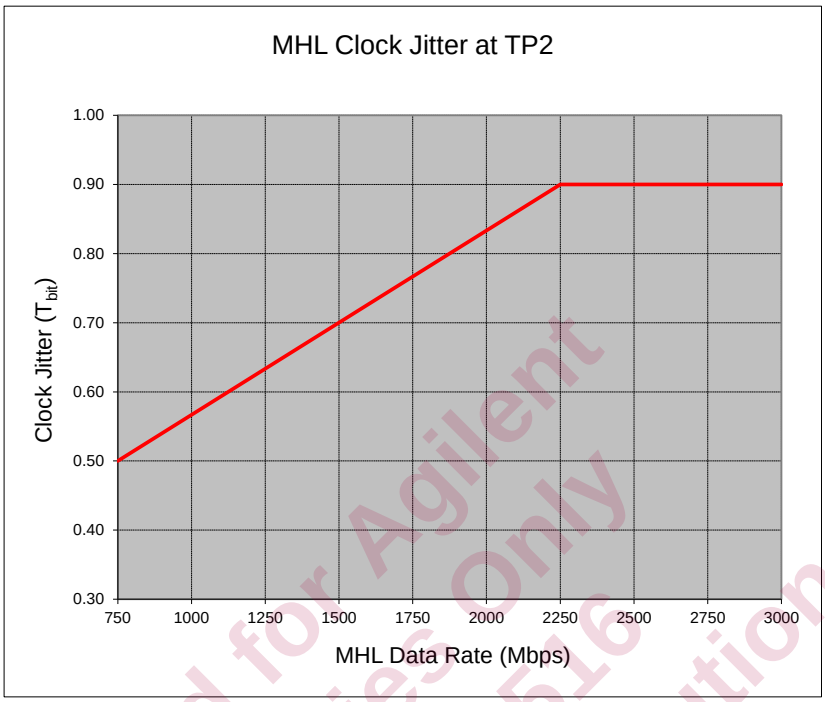
An MHL Sink shall perform properly in response to a signal at TP2 with the characteristics described in Table 13-8. The signal shall be calibrated to these timing conditions with the test load specified in Figure 13-8 and Table 13-1.

Table 13-8. Receiver AC Characteristics at TP2

Item	Symbol	Min	Max	Unit	Notes
Differential Intra-Pair Skew	T_{SKEW_DF}		Min(0.25T_{BIT}, 93)	psec	1, 2
Common-Mode Intra-Pair Skew	T_{SKEW_CM}		Min(0.25T_{BIT}, 93)	psec	1, 3

Notes on Table 13-8:

1. The maximum intra-pair skew shall not exceed the smaller of the two terms shown.
2. Differential intra-pair skew is measured as the time difference between two mid-points on the same T_{BIT} of the differential transition edges, one on MHL+ and one on MHL-.
3. Common mode intra-pair skew is measured as the time difference between two mid-points of the common-mode transition edges, one on MHL+ and one on MHL-.



$T_{CLK\_JITTER\_TP2}$ = 0.3T_{BIT}+266.7psec up to 2.25Gbps. $T_{CLK\_JITTER\_TP2}$ = 0.9T_{BIT} above 2.25Gbps.

Figure 13-20. MHL Clock Jitter at TP2

The maximum MHL input clock jitter shall meet the MHL clock jitter requirement ($T_{CLK\_JITTER\_TP2}$) at TP2 as shown in Figure 13-20. The MHL clock jitter at TP2 shall be calibrated to the required jitter condition using the MHL Clock Recovery Unit.

13.6.3 Sink Device Impedance Characteristics at TP2

An MHL Sink shall meet the impedance requirements in Table 13-9.

Table 13-9. Sink Device Impedance Characteristics at TP2

Item	Symbol	Min	Typ	Max	Unit	Notes
Differential Impedance through Connections		85		115	ohms	1, 3
Differential Impedance at Termination	Z_{DT}	90		110	ohms	1
Common-Mode Impedance through Connections		24		36	ohms	2, 4
Common-Mode Impedance at Termination	Z_{CT}	25		35	ohms	2
RxSense Impedance						
RxSense active	$Z_{RXSENSE\_TERM}$			100k	ohms	7, 8
TMDS termination	$Z_{RXSENSE\_TMDS}$			70	ohms	5, 7
RxSense open (leakage current)	$I_{RXSENSE\_LEAK}$			2	µA	6

Notes on Table 13-9:

1. Measure impedance at TP2 using a TDR with differential rise time (20-80%) of 200 psec.
2. Measure impedance at TP2 using a TDR with common-mode rise time (20-80%) of 600 psec.

3. A single excursion is permitted out to 75 – 125 ohms, no wider than 250 psec.
4. A single excursion is permitted out to 20 – 40 ohms, no wider than 500 psec.
5. Termination on MHL+ and MHL- shall be interpreted by the Source as "TMDS active", for content signaling, when measured in this range.
6. Termination on MHL+ and MHL- shall be interpreted by the Source as "open", non-terminated, when measured with leakage at or below this maximum.
7. Minimum values for $Z_{RXSENSE\_TERM}$ and $Z_{RXSENSE\_TMDS}$ are determined by $Z_{DT}/2$.
8. It is optional for a Sink to implement an $Z_{RXSENSE\_TERM}$ value higher than $Z_{RXSENSE\_TMDS}$. A Sink which implements a higher $Z_{RXSENSE\_TERM}$ shall do so in compliance with the restrictions described in Section 8.1.2. The specified range for $Z_{RXSENSE\_TERM}$ encompasses the specified range for $Z_{RXSENSE\_TMDS}$.

13.6.4 Sink Device Pixel Error Rate Characteristics

Under all operating conditions specified in Section 13.6, the receiver shall reproduce a test data stream, with TMDS character error rate 10^{-9}, when presented with differential input signal illustrated by the eye diagram of Figure 13-21.

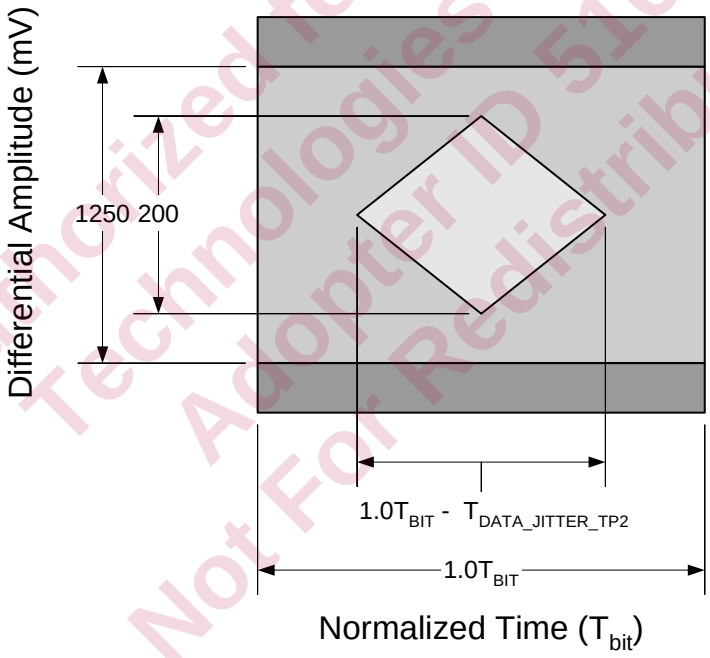
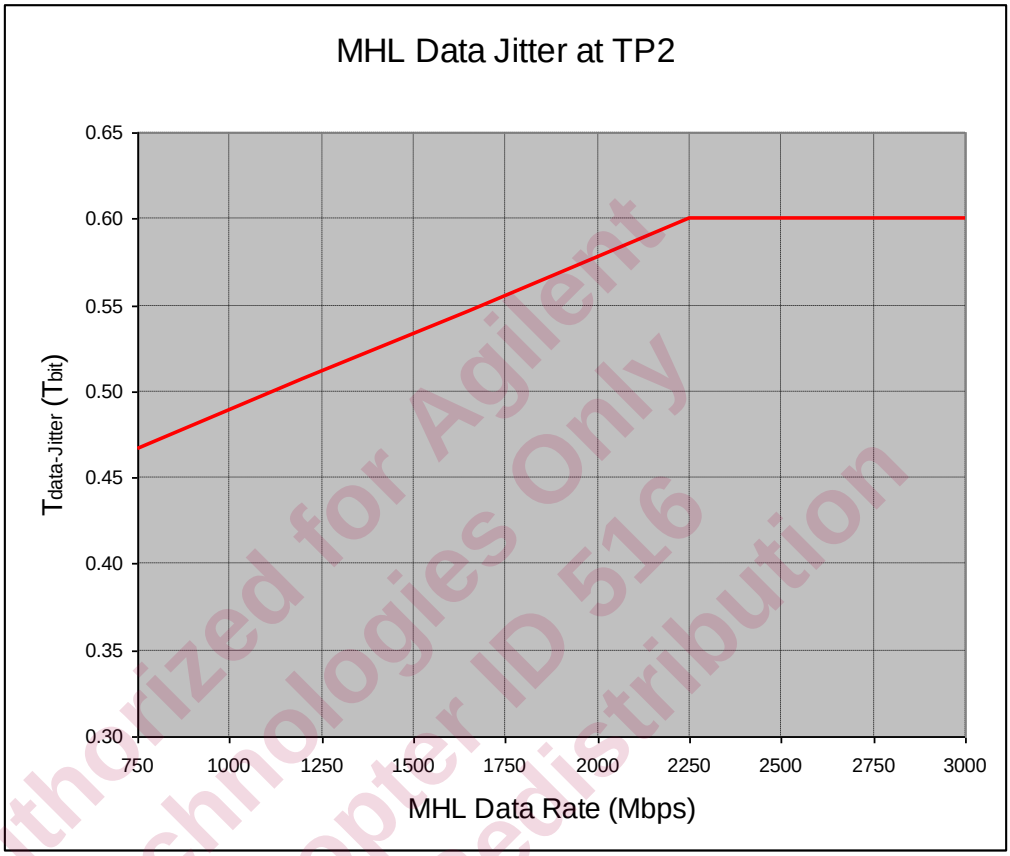


Figure 13-21. MHL Data Eye Mask at TP2

The mask for data eye opening at TP2 may be shifted horizontally from the center of the data bit time. Vertical position of the eye opening mask shall be centered at 0V.

The receiver differential input data jitter and eye diagram shall be calibrated to the required conditions using the reference equalizer and clock recovery unit described in Sections 13.3.1.7 and 13.3.1.8.



$T_{DATA\_JITTER\_TP2}$ = $0.4T_{BIT}$+88.88psec up to 2.25Gbps. $T_{DATA\_JITTER\_TP2}$ = $0.6T_{BIT}$ above 2.25Gbps.

Figure 13-22. MHL Data Jitter at TP2

Figure 13-22 shows the limit on $T_{DATA\_JITTER\_TP2}$.

13.7 Dongle Device Electrical and Timing Specs

13.7.1 Dongle Device DC Characteristics at TP3

The MHL signal at TP3 shall meet the signal requirements listed in Table 13-10. Single-ended, differential and common-mode voltages shall be calibrated to the required conditions using the test load specified in Figure 13-8 and Table 13-1.

Table 13-10. Dongle DC Characteristics at TP3

Item	Symbol	Min	Max	Unit	Notes
Single-ended high level input voltage	V_{SE_HIGH}	V_{TERM}-540	V_{TERM}+10	mV	1, 2
Single-ended low level input voltage	V_{SE_LOW}	V_{TERM}-1760	V_{TERM}-700	mV	1, 2
Differential input swing voltage	$V_{DFSWING}$	600	1000	mV	2
Common-mode input swing	$V_{CMSWING}$	360	Min(720,	mV	2, 3

voltage			$0.85V_{DFSWING}$)		

Notes on Table 13-10:

1. V_{TERM} is defined in Table 13-1.
2. Refer to Figure 13-2.
3. Common-mode input swing voltage shall not exceed the smaller of the two terms.

13.7.2 Dongle Device AC Characteristics at TP3

An MHL Dongle shall perform properly in response to a signal at TP3 with the characteristics described in Table 13-11. The signal shall be calibrated to these timing conditions with the test load specified in Figure 13-8 and Table 13-1.

Table 13-11. Dongle AC Characteristics at TP3

Item	Symbol	Min	Max	Unit	Notes
Pixel Clock Frequency	F_{PIXEL}	25	150	MHz	1
MHL Link Clock Frequency 24-Bit Mode	F_{MHL}	25	75	MHz	
MHL Link Clock Period 24-Bit Mode	T_{MHL}	13.33	40	nsec	2
MHL Link Clock Frequency PackedPixel Mode	F_{MHL_PP}	25	75	MHz	
MHL Link Clock Period PackedPixel Mode	T_{MHL_PP}	13.33	40	nsec	2
Bit Time on Link	T_{BIT}	333	1333	psec	
Differential Rise Time	T_{R_DF}	75		psec	3, 4
Differential Fall Time	T_{F_DF}	75		psec	3, 4
Common-Mode Rise Time	T_{R_CM}	600	2500	psec	3
Common-Mode Fall Time	T_{F_CM}	600	2500	psec	3
Differential Intra-Pair Skew	T_{SKEW_DF}		Min($0.12T_{BIT}$, 50)	psec	5
Common-Mode Intra-Pair Skew	T_{SKEW_CM}		Min($0.12T_{BIT}$, 50)	psec	6
Clock Duty Cycle 24-Bit Mode		35	65	$\%T_{MHL}$	
Clock Duty Cycle PackedPixel Mode		35	65	$\%T_{MHL\_PP}$	

Notes on Table 13-11:

1. Pixel clock above 75 MHz is allowed only when output is in PackedPixel mode.
2. MHL Clock Period is the inverse of the MHL Clock Frequency (F_{MHL}), and is not a controlling specification.
3. Rise and fall times are measured for 20% to 80% transition.
4. Differential rise and fall times shall be measured for specific patterns defined in MHL Compliance Test Specification.
5. Differential intra-pair skew is measured as the time difference between two mid-points on the same T_{BIT} of the differential transition edges, one on MHL+ and one on MHL-.

6. Common mode intra-pair skew is measured as the time difference between two mid-points of the common-mode transition edges, one on MHL+ and one on MHL-.

The maximum MHL input clock jitter at TP3 for a Dongle device shall be the same as the maximum MHL clock jitter of a Source device, defined in Figure 13-13. The MHL input clock jitter at TP3 shall be calibrated to the required jitter condition using the MHL Clock Recovery Unit.

13.7.3 Dongle Device Impedance Characteristics at TP3

An MHL Dongle shall meet the impedance requirements in Table 13-12.

Table 13-12. Dongle Device Impedance Characteristics at TP3

Item	Symbol	Min	Typ	Max	Unit	Notes
Differential Impedance through Connections		85		115	ohms	1, 3
Differential Impedance at Termination	Z_{DT}	90		110	ohms	1
Common-Mode Impedance through Connections		24		36	ohms	2, 4, 9
Common-Mode Impedance at Termination	Z_{CT}	25		35	ohms	2
RxSense Impedance						
RxSense active	$Z_{RXSENSE\_TERM}$			100k	ohms	7, 8
TMDS termination	$Z_{RXSENSE\_TMDS}$			70	ohms	5, 7
RxSense open (leakage current)	$I_{RXSENSE\_LEAK}$			2	µA	6

Notes on Table 13-12:

1. Measure impedance at TP3 using a TDR with differential rise time (20-80%) of 200 psec.
2. Measure impedance at TP3 using a TDR with common-mode rise time (20-80%) of 600 psec.
3. Two excursions are permitted out to 65-125 ohms, each excursion no wider than 350 psec.
4. Two excursions are permitted out to 20-40 ohms, each excursion no wider than 500 psec.
5. Termination on MHL+ and MHL- shall be interpreted by the Source as "TMDS active", for content signaling, when measured in this range.
6. Termination on MHL+ and MHL- shall be interpreted by the Source as "open", non-terminated, when measured with leakage at or below this maximum.
7. Minimum values for $Z_{RXSENSE\_TERM}$ and $Z_{RXSENSE\_TMDS}$ are determined by $Z_{DT}/2$.
8. It is optional for a Dongle to implement an $Z_{RXSENSE\_TERM}$ value higher than $Z_{RXSENSE\_TMDS}$. A Dongle which implements a higher $Z_{RXSENSE\_TERM}$ shall do so in compliance with the restrictions described in Section 8.1.2. The specified range for $Z_{RXSENSE\_TERM}$ encompasses the specified range for $Z_{RXSENSE\_TMDS}$.
9. Common-mode impedance is permitted out to 24-140 ohms only for Dongles with input captive cable length not exceeding 30 centimeters.

13.7.4 Dongle Device Pixel Error Rate Characteristics

Under all operating conditions specified in 13.7, the Dongle shall reproduce a test data stream, with TMDS character error rate 10^{-9}, when presented with differential input signal illustrated by

the eye diagram of Figure 13-14. The differential data eye diagram shall be calibrated to the required conditions at TP3 using the MHL Clock Recovery Unit.

13.8 Cable Electrical and Timing Specs

The term "Cable" is the cable assembly, including all five parts listed below;

- Source device side plug
- Source device side transition (from plug to cable)
- Wire (cable itself)
- Sink device side transition
- Sink device side plug

Performance of an MHL cable is measured with respect to the test points TP3 and TP4 shown in Figure 13-23. TP1 and TP2 are not available because connection points between plug and receptacle cannot be accessed during testing. Therefore, TP3 and TP4 are used, even though the effects of receptacles at both ends are included in the test result.

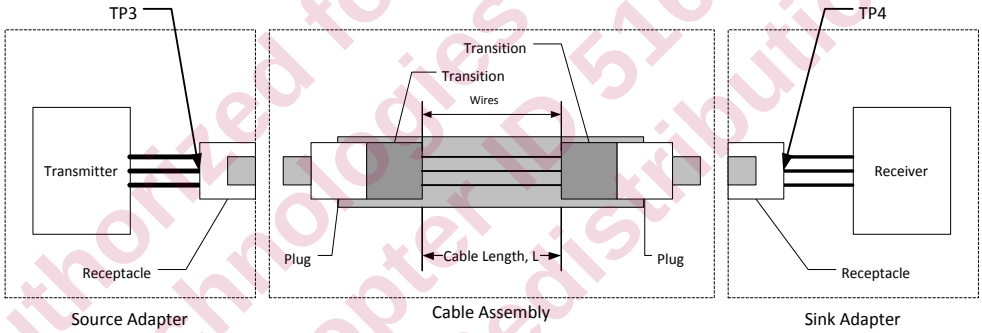


Figure 13-23. Cable Assembly Test Points

13.8.1 Cable DC Electrical Specifications

Table 13-13. Cable Assembly MHL Characteristics

Item		Symbol	Min	Max	Unit	Notes
Ground Resistance		R_{GND}		0.08	ohms	1
Differential Intra-Pair Skew				43	psec	
Common-Mode Intra-Pair Skew				43	psec	
MHL+ and MHL- Differential Characteristic Impedance	Cables longer than 300mm length	Z_{OD}	85	115	ohms	2, 3, 4
			90	110	ohms	2, 3, 5
	Cables up to 300mm length	Z_{OD}	85	115	ohms	2, 3, 4
			85	115	ohms	2, 6, 7
			90	110	ohms	2, 7, 8
MHL Clock Common-Mode Characteristic Impedance	Cables longer than 600mm length	Z_{OC}	24	36	ohms	2, 9, 10
			25	35	ohms	2, 9, 11
	Cables up to 600mm length	Z_{OC}	24	36	ohms	2, 9, 10
			20	40	ohms	2, 12, 13
			24	36	ohms	2, 13, 14
CBUS Line Capacitance		C_{CBUS}	30	350	pF	15, 16
CBUS Cable Delay		$T_{CBUS\_CBL\_DLY}$		35	nsec	
Common-mode Clock Swing Voltage		$V_{CABLE\_CM\_SWING}$	170		mV	17

Notes on Table 13-13:

1. Ground resistance limits ground drop to 40mV when passing 500mA through the maximum allowed resistance.
2. Cable length as shown by dimension L in Figure 13-23, measured from the end of the plug on the source side to the end of the plug on the sink side.
3. Differential impedance measured on TDR with differential rise time (20%-80%) of 200 psec.
4. Connection point and transition area, up to 1 nsec distance. A single excursion is permitted out to 65-125 ohms, no wider than 250 psec.
5. Cable area, from 1 nsec to 2 nsec distance, measured on TDR with differential rise time (20%-80%) of 200 psec.
6. Differential impedance measured on TDR with differential rise time (20%-80%) of 50 psec.
7. Cable area, from 1 nsec to 1.5 nsec distance, measured on TDR with differential rise time (20%-80%) of 50 psec.
8. Arithmetic mean of differential impedance measurements along the cable area from 1 nsec to 1.5 nsec distance, on TDR with differential rise time (20%-80%) of 50 psec.
9. Common mode impedance measured on TDR with common-mode rise time (20%-80%) of 600 psec.
10. Connection point and transition area, up to 3 nsec distance. A single excursion is permitted out to 20-40 ohms, no wider than 500 psec.

11. Cable area, from 3 nsec to 4 nsec distance, measured on TDR with common-mode rise time (20%-80%) of 600 psec.
12. Common mode impedance measured on TDR with common-mode rise time (20%-80%) of 50 psec.
13. Cable area, from 1 nsec to 1.5 nsec distance, measured using TDR with common-mode rise time (20%-80%) of 50 psec.
14. Arithmetic mean of common mode impedance measurements along the cable area from 1 nsec to 1.5 nsec distance, on TDR with common-mode rise time (20%-80%) of 50 psec.
15. Measured with LCR meter and 100kHz signal, or calculated from TDR measurement.
16. Minimum capacitance on CBUS is required to limit rise and fall times when connecting with minimum load, as from a Source to Dongle with minimum cable length.
17. Minimum common-mode clock voltage swing at the cable output (TP2) when input clock voltage swing (TP1) is 360mV. MHL receiver shall recover the data and clock when the common mode clock voltage swing at TP2 is equal to or larger than this value.

An MHL cable shall meet the differential eye diagram requirement specified in Figure 13-21 for all MHL Source conditions specified in Section 13.5. Measurements are made using two procedures:

1 Non-equalized, wherein the Reference Cable Equalizer is not applied, and
2 Equalized, wherein the Reference Cable Equalizer is applied.

When a source signal marginally compliant to the eye diagram mask shown in Figure 13-14 is applied to TP1 on the MHL cable, at the highest MHL data bit rate, the MHL cable shall produce an MHL output waveform that meets the sink eye diagram mask for TP2 shown in Figure 13-21 with or without the application of the Reference Cable Equalizer.

An MHL cable shall meet the non-equalized eye diagram requirements, or the equalized eye diagram requirements, or both, with the following exception. If the MHL cable satisfies all parametric requirements shown in Table 13-14, then the MHL cable eye diagram requirements in Figure 13-21 and Common-mode Swing Voltage requirement in Table 13-13 are not required to be met. However, if the MHL cable does not meet one or more of the parametric requirements listed in Table 13-14, then the MHL cable shall meet the cable eye diagram requirements in Figure 13-21 and shall meet the Common-mode Swing Voltage requirement in Table 13-13.

Table 13-14. Minimal Cable AC Parametric Requirements

Item	Section
Differential Intra-Pair Skew	13.8.1
Common-mode Intra-Pair Skew	13.8.1
Cable Differential Insertion Loss	13.8.2.1
Cable Common-mode Insertion Loss	13.8.2.2
Cable Differential and Common-mode Conversion	13.8.2.3

13.8.2 Cable AC Electrical Specification

13.8.2.1 Cable Differential Insertion Loss

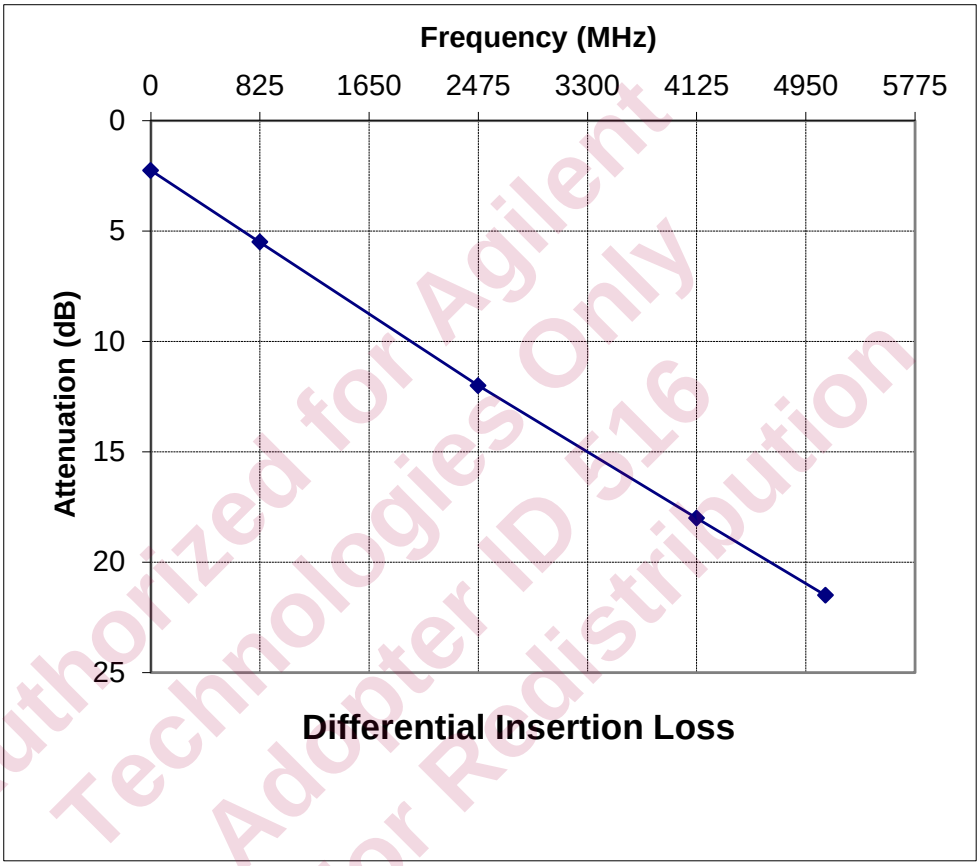


Figure 13-24. MHL Cable Differential Insertion Loss for MHL Data Signals

Table 13-15. MHL Cable Differential Insertion Loss for MHL Data Signals

Frequency (MHz)	Attenuation (dB)
0	2.25
825	5.5
2475	12.0
4125	18.0
5100	21.5

13.8.2.2 Cable Common-mode Insertion Loss

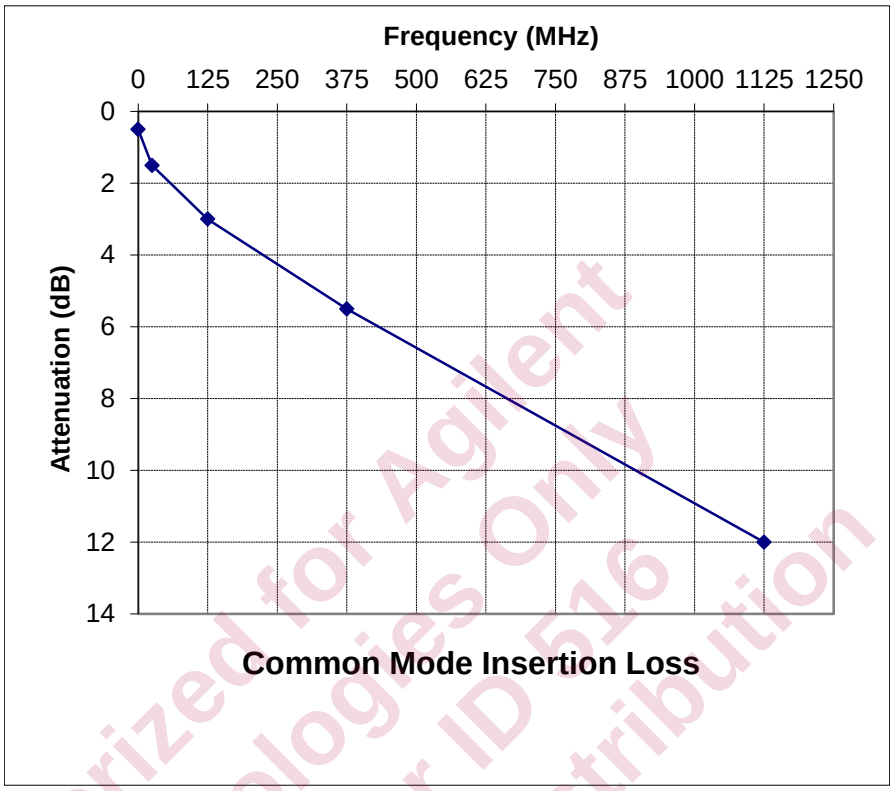


Figure 13-25. MHL Cable Common-mode Insertion Loss for MHL Clock Signals

Table 13-16. MHL Cable Common-mode Insertion Loss for MHL Clock Signals

Frequency (MHz)	Attenuation (dB)
0	0.5
25	1.5
125	3.0
375	5.5
1125	12.0

13.8.2.3 Cable Differential and Common-mode Conversion

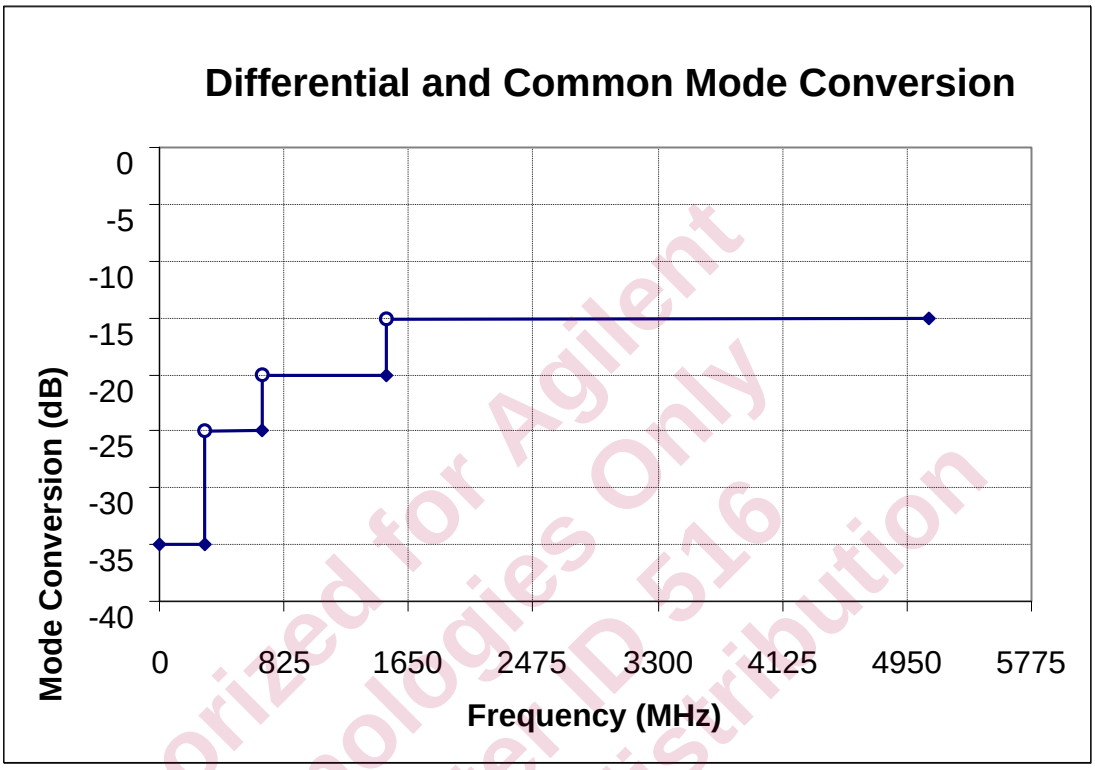


Figure 13-26. MHL Cable Mode Conversion between Differential and Common-mode Signals

Table 13-17. MHL Cable Mode Conversion between Differential and Common-mode Signals

Frequency Range (MHz)	Mode Conversion (dB)
0 to 300	-35
>300 to 675	-25
>675 to 1500	-20
>1500 to 5100	-15

13.8.2.4 Cable CBUS Insertion Loss

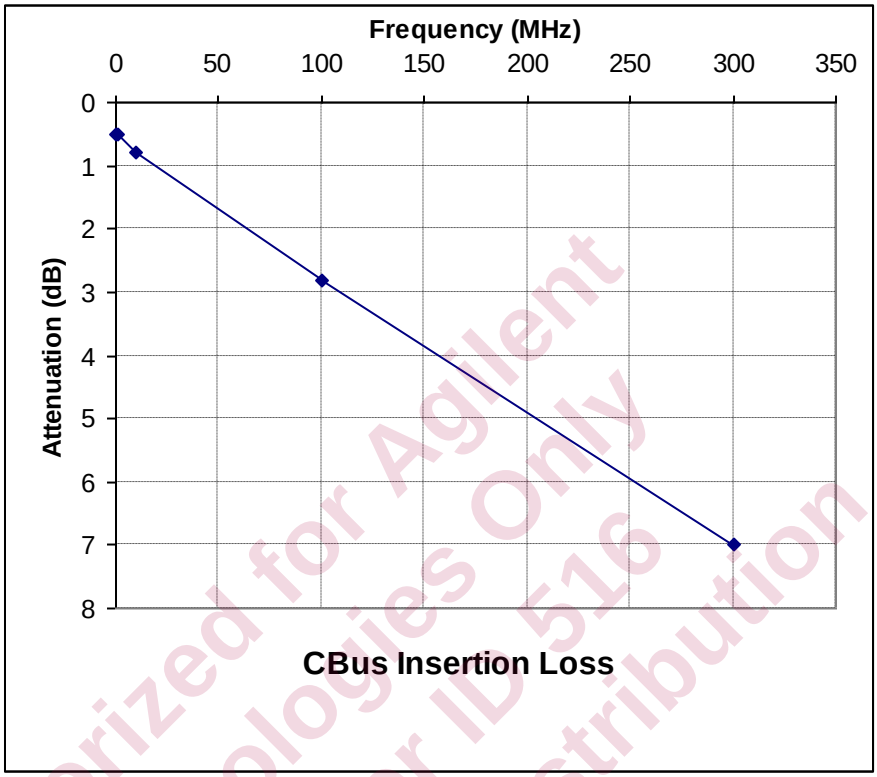


Figure 13-27. MHL CBUS Signal Insertion Loss

Table 13-18. MHL CBUS Signal Insertion Loss

Frequency (MHz)	Attenuation (dB)
0	0.5
1	0.5
10	0.8
100	2.8
300	7.0

13.8.2.5 Cable Far-End Crosstalk Performance

13.8.2.5.1 Cable Far-End Crosstalk between MHL and CBUS

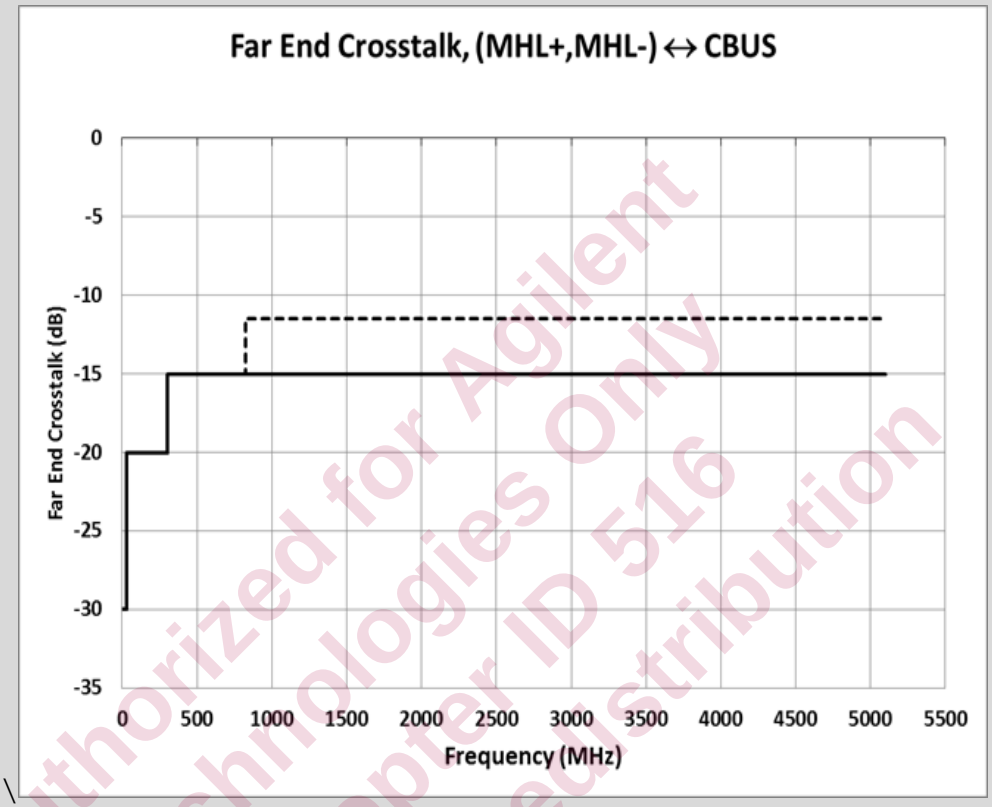


Figure 13-28. MHL Cable Assembly Far-End Crosstalk between MHL and CBUS

Far-end crosstalk limit is defined in Table 13-19 for:

MHL+ to CBUS and CBUS to MHL+

MHL- to CBUS and CBUS to MHL-

Table 13-19. MHL Cable Assembly Far-End Crosstalk between MHL and CBUS

Frequency Range	Far-End Crosstalk (dB)	Notes
0-30MHz	-30	
>30 to 300MHz	-20	
>300MHz to 5.1GHz	-15	1

Note on Table 13-19:

1. Excursions are permitted out to -11.5 dB for the frequency range of 825 MHz to 5100 MHz if the differential insertion loss of the cable meets Table 13-19-1 and Figure 13-28-1 shown below.

13.8.2.5.2 Cable Far-End Crosstalk between MHL and VBUS

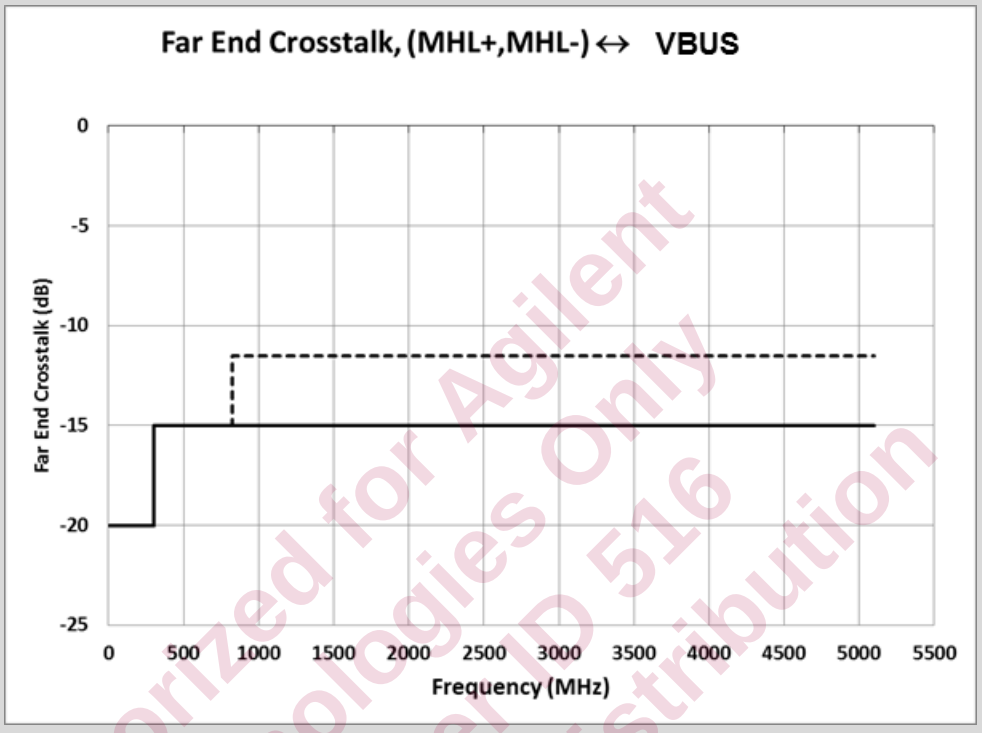


Figure 13-29. MHL Cable Assembly Far-End Crosstalk between MHL and VBUS

Far-end crosstalk limit is defined in Table 13-20 for:

MHL+ to VBUS and VBUS to MHL+

MHL- to VBUS and VBUS to MHL-

Table 13-20. MHL Cable Assembly Far-End Crosstalk between MHL and VBUS

Frequency Range	Far-End Crosstalk (dB)	Notes
0-300MHz	-20	
>300MHz to 5.1GHz	-15	1

Note on Table 13-19:

1. Excursions are permitted out to -11.5 dB for the frequency range of 825 MHz to 5100 MHz if the differential insertion loss of the cable meets Table 13-19-1 and Figure 13-28-1 shown below.

13.8.2.5.3 Cable Far-End Crosstalk between CBUS and VBUS

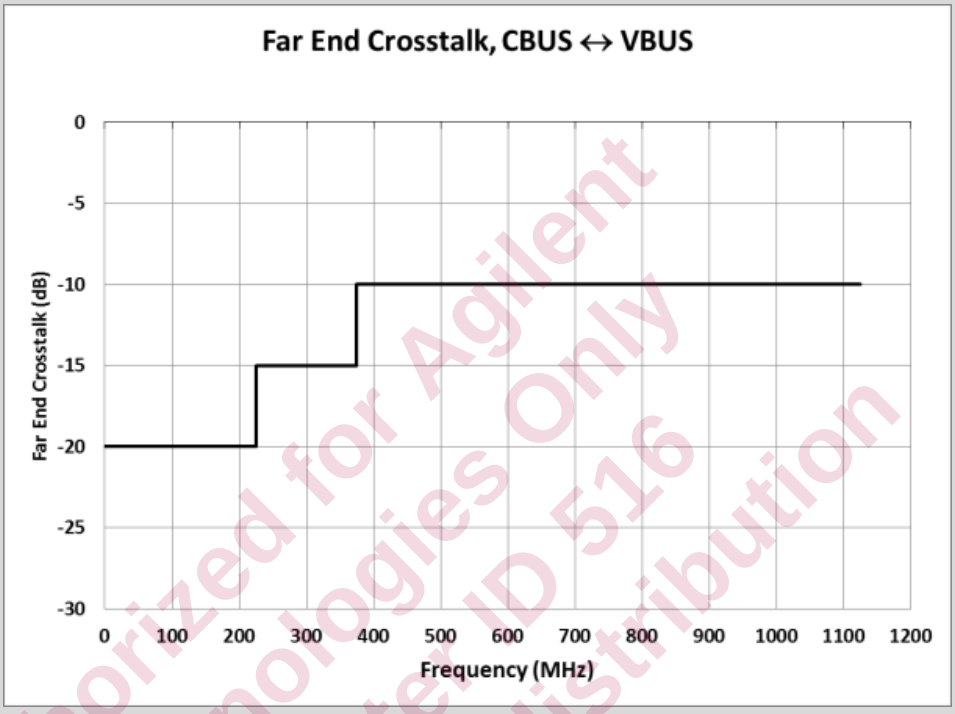


Figure 13-30. MHL Cable Assembly Far-End Crosstalk between CBUS and VBUS

Far-end crosstalk limit is defined in Table 13-21 for CBUS-to-VBUS and VBUS-to-CBUS.

Table 13-21. MHL Cable Assembly Far-End Crosstalk between CBUS and VBUS

Frequency Range	Far-End Crosstalk (dB)
0 to 225MHz	-20
>225 to 375MHz	-15
>375MHz to 1.125GHz	-10

13.8.2.5.4 Cable Differential Insertion Loss Limit for Far-End Crosstalk

Table 13-19-1. MHL Cable Differential Insertion Loss Limit for Far-End Crosstalk Excursion

Frequency (MHz)	Attenuation (dB)
0	2.0
825	2.0
2475	8.0
4125	14.5
5100	18.0

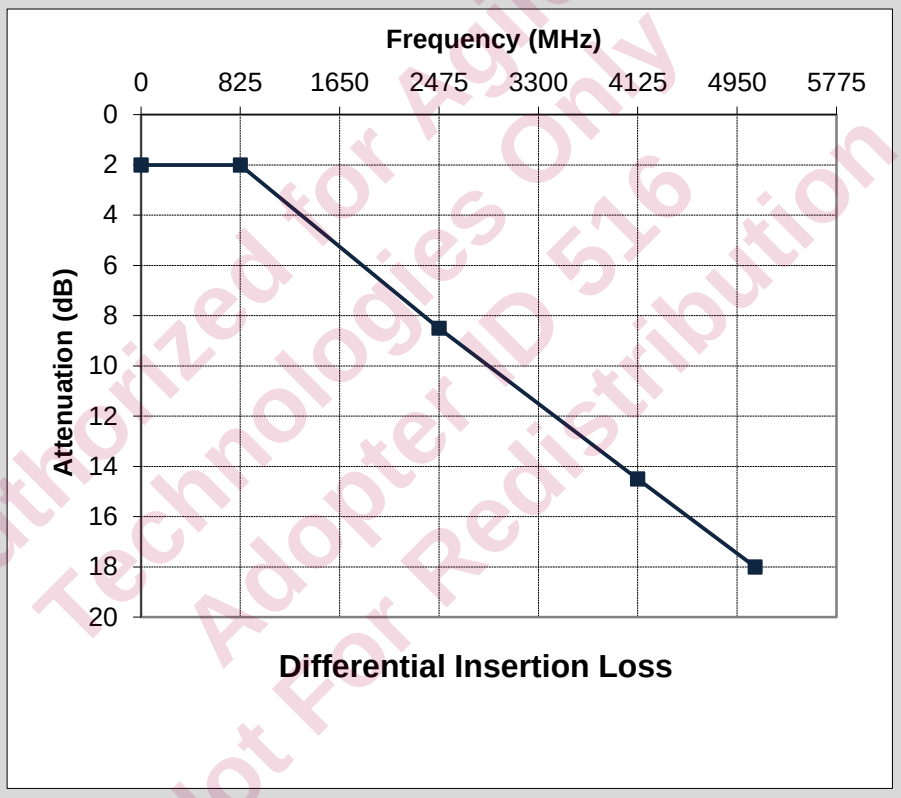


Figure 13-28-1. MHL Cable Differential Insertion Loss Limit for Far-End Crosstalk Excursion

13.8.3 Cable Power Signal (VBUS)

The cable shall not cause a voltage drop larger than the V_{VBUS_DROP} in Table 13-22 nor a ground shift larger than V_{GND_SHIFT}, measured between TP3 and TP4, when drawing I_{VBUS_Ref}..

Table 13-22. Cable +5V Power Pin (VBUS) Voltage

Item	Symbol	Min	Max	Unit	Notes
Cable VBUS Drop (TP4 to TP3)	V_{VBUS_DROP}		160	mV	1
Cable Ground Shift (TP4 to TP3)	V_{GND_SHIFT}		40	mV	1, 2
Absolute Maximum Cable Current	$I_{CABLE\_VBUS\_MAX}$	1.8		A	3

Notes on Table 13-22:

1. I_{VBUS_Ref} is the current through VBUS and GND, used for measurement of voltage drops across the cable. I_{VBUS_Ref} is 500mA.
2. Ground shift is measured across the GND and Shield wire in the MHL cable.
3. All cables shall be able to carry at least $I_{CABLE\_VBUS\_MAX}${min} without damage to the cable. Functional performance is not guaranteed at this level of current. Refer also to Section 12 for possible restrictions on specific connections.

13.9 +5V Power Signal (VBUS)

The MHL connector provides a VBUS pin allowing the Sink or Self-Powered Dongle to supply a nominal +5 Volts to the cable and Source, or in some instances allowing a Source to supply a nominal +5 Volts to the cable and unpowered Dongle.

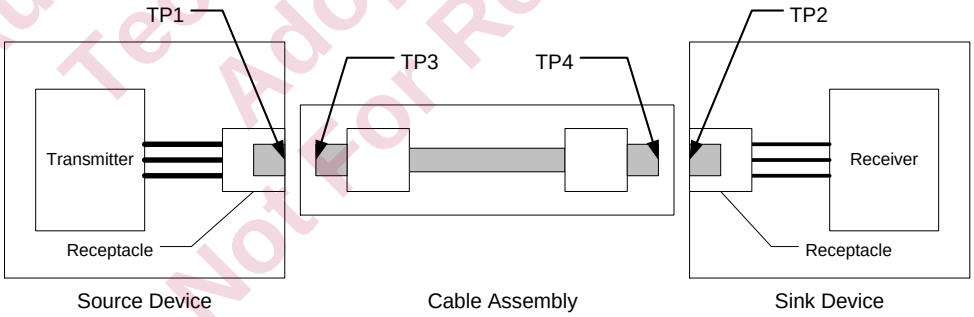


Figure 13-31. Cable Assembly Voltage Drop

13.9.1.1 VBUS Voltage Control

An MHL Sink shall assert VBUS when connected to an MHL Source device. The voltage driven by the Sink shall be within the limits specified for TP2 voltage in Table 13-23.

In the case when the Source is connected to an Unpowered Dongle, and the Source outputs VBUS, the Source shall meet the $V_{VBUS\_TP1\_SRC\_DRV}$ requirement.

Table 13-23. +5V Power Pin (VBUS) Voltage

Item	Symbol	Min	Max	Unit	Notes
TP2 Voltage, Sink asserting VBUS	$V_{VBUS\_TP2\_SINK\_DRV}$	4.75	5.25	V	
TP1 Voltage, Sink asserting VBUS	$V_{VBUS\_TP1\_SINK\_DRV}$	4.51	5.25	V	1, 3
TP1 Voltage, Dongle asserting VBUS	$V_{VBUS\_TP1\_DNGL\_DRV}$	4.51	5.25	V	2, 4
TP1 Voltage, Source asserting VBUS	$V_{VBUS\_TP1\_SRC\_DRV}$	4.75	5.25	V	

Notes on Table 13-23:

1. The VBUS minimum voltage measured at the Source, when driven through the cable from the Sink.
2. The VBUS minimum voltage measured at the Source, when driven through the cable or interconnect from the Dongle is "Dongle voltage".
3. Drawing more than $I_{VBUS\_Post\_Discovery}${max} may cause TP1 voltage to be below $V_{VBUS\_TP1\_SINK\_DRV}${min}.
4. According to Section 13.9.1.2, TP1 voltage for MHL 2 Dongle should remain at or above $V_{VBUS\_TP1\_DNGL\_DRV}${min}, at any valid current drawn by Source.

13.9.1.2 VBUS Current Control

MHL Sinks, Sources and Dongles shall supply and draw current on the VBUS pin according to the limits shown in Table 13-24.

- Prior to successful CBUS discovery:
  - Source shall limit its current draw to $I_{VBUS\_Prediscovery}${max}, irrespective of the voltage at its VBUS input, or
  - Source shall monitor the voltage at its VBUS input and shall limit its current draw so that it does not pull this input voltage lower than $V_{VBUS\_TP1\_SINK\_DRV}${min} while also limiting its current draw to $I_{VBUS\_PostDiscovery}${max}.
- After successful CBUS discovery:
  - An MHL 2 Source, when connected to an MHL 1 Sink (as determined by the Sink's Device Capability registers), shall limit its current draw to $I_{VBUS\_Sink\_to\_Source}${min} for MHL 1 Sink. The current is limited because the Source has no information about the current-carrying capacity of the cable between Source and Sink.
  - An MHL 2 Source, when connected to an MHL 1 Dongle (as determined by the Dongle's Device Capability registers):
    - shall limit its current draw to $I_{VBUS\_Prediscovery}${max}, or
    - may draw more than $I_{VBUS\_Prediscovery}${max} but shall monitor the voltage at its VBUS input and shall limit its current draw to not pull this input voltage lower than $V_{VBUS\_TP1\_DNGL\_DRV}${min}.
  - An MHL 2 Source, when connected to an MHL 2 Sink or Dongle (as determined by the device's Device Capability registers, including DEV_CAT), shall limit its current draw to the value indicated by the device's PLIM[1:0] register setting. See Table 7-26.

An unpowered Dongle drawing current from VBUS shall limit its current draw to the minimum guaranteed current from the Source ($I_{VBUS\_Source\_to\_Dongle}$), as specified in Table 13-24.

An MHL Device shall maintain correct values in its DEV_CAT register at all times.

The return for the VBUS signal is the ground signal.

Hex value shown in Table 13-24 is the setting for DEV_CAT for this type of Sink or Dongle.

Table 13-24. +5V Power Pin (VBUS) Current

Item			Symbol	Min	Max	Unit	Notes
Sink/Dongle providing Current							
Prior to Discovery			$I_{VBUS\_Prediscovery}$		100	mA	
After Discovery			$I_{VBUS\_PostDiscovery}$		500	mA	
0x11	MHL 2 Sink		$I_{VBUS\_Sink\_to\_Source\_MHL2\_Sink}$	900		mA	1
	MHL 1 Sink		$I_{VBUS\_Sink\_to\_Source\_MHL1\_Sink}$	500		mA	
0x13	Self-Powered Dongle		$I_{VBUS\_Dongle\_to\_Source}$	100		mA	
0x03	Unpowered Dongle				0	mA	2
Source providing Current							
0x12	Source into Dongle		$I_{VBUS\_Source\_to\_Dongle}$	200		mA	

Notes on Table 13-24:

1. MHL 2 Sink shall be designed to output lesser of MHL 2 Sink's designed functional max current on VBUS (which is shown here as $I_{VBUS\_Sink\_to\_Source\_MHL2\_Sink}\{min\}$), and $I_{CABLE\_VBUS\_MAX}$ (Table 13-22).
2. Unpowered Dongle relies on power from the Source.

13.10 CBUS Electrical and Timing Specifications

The CBUS signal shall meet the electrical and timing specifications shown in this section.

13.10.1 CBUS Signaling

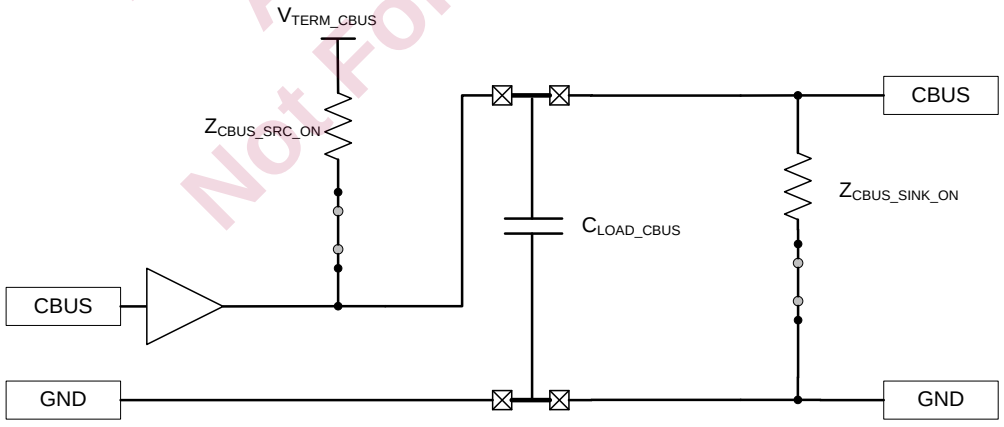


Figure 13-32. CBUS Loading Schematic View

Table 13-25. Required Output Characteristics of CBUS Signal

Item	Symbol	Min	Max	Unit	Notes
CBUS Absolute Max Voltage			5.25	V	
CBUS Termination Voltage	V_{TERM_CBUS}	1.7	1.9	V	
Output High voltage level	V_{OH_CBUS}	1.5	1.9	V	1
Output Low voltage level	V_{OL_CBUS}		0.2	V	1
Output High voltage level during Wake and Discovery	$V_{OH\_CBUS\_DISC}$	1.4	1.9	V	2
Rise / Fall time in Connected State	T_{R_CBUS} T_{F_CBUS}	5 5	200 200	nsec	3
Rise to Fall time Difference	ΔT_{RF}		100	nsec	4
Bit time (1 MHz clock)	T_{BIT_CBUS}	0.80	1.20	µsec	5, 16
Bit Time Variation in Packet	$T_{BIT\_VARY\_PACKET}$	-1%	+1%	T_{BIT_CBUS}	6
Bit Time Variation between Packets	$T_{BIT\_VARY\_LONG}$	-4%	+4%	T_{BIT_CBUS}	7, 15
SYNC Bit Period	T_{SYNC_TOTAL}	2	2	T_{BIT_CBUS}	8
ACK Bit Period	T_{CBUS_ACK}	2	2	T_{BIT_CBUS}	8
ACK Bit Initial Falling Edge	$T_{CBUS\_ACK\_FALL}$		530	nsec	10
ACK Bit Drive low time	$T_{CBUS\_ACK\_0}$	400	600	nsec	9
Drive high time for CBUS	$T_{DRV\_HI\_CBUS}$	100	400	nsec	11
Duty Cycle of '1' Bit	T_{CBUS_DUTY}	0.4	0.6	T_{BIT_CBUS}	12, 13, 16
Duty Cycle of SYNC Bit	T_{SYNC_DUTY}	1.4	1.6	T_{BIT_CBUS}	12, 14, 16

Notes on Table 13-25:

1. Initiator shall drive these voltage levels when sending packets on CBUS. V_{OH_CBUS} maximum is set by V_{TERM_CBUS}{max}.
2. Initiator shall drive these voltage levels when sending wake pulses or discovery pulses on CBUS. V_{OH_CBUS} maximum is set by V_{TERM_CBUS}{max}.
3. Rise and fall times measured from V_{OL_CBUS}{max} to V_{OH_CBUS}{min}, or V_{OH_CBUS}{min} to V_{OL_CBUS}{max}.
4. After a Source and Sink connect, the rise and fall times on CBUS shall not differ by more than this limit.
5. Bit time is measured from initial rising edge in a '1' bit to the initial rising edge in the next consecutive '1' bit, or from the initial falling edge in a '1' bit to the initial falling edge in the next consecutive '1' bit.. Bit time for a '0' bit is measured by dividing the time of two '0' bits by 2.
6. Bits driven by the initiator within one packet shall match the mean bit time in the packet by this limit.
7. Bits driven in one packet shall have bit times which vary from the mean bit time of the preceding packet (without intervening arbitration) by no more than these limits. Bit times of a packet after an intervening arbitration need only fit within the minimum and maximum limits for T_{BIT_CBUS}.

8. The ACK bit period and SYNC bit period are each nominally two bit times.
9. A positive ACK shall drive CBUS low for nominally one-half T_{BIT_CBUS}. Measured in absolute time, since the time is measured by the follower, not based on initiator's bit time.
10. The falling edge for a positive ACK, driven by the follower, shall occur no later than this limit after the end of the preceding Parity bit time, as measured at the follower end of the CBUS connection.
11. Drive high time measured from the start of the rising edge. The rising edge may be at the beginning of the bit time (for '0') or in the middle of the bit time (for '1').
12. T_{CBUS_DUTY} and T_{SYNC_DUTY} are meant to allow a reasonable difference between the rise and fall times on CBUS. Hence it is expected that these two parameters will track each other in their deviation from nominal.
13. The middle edge in a '1' bit shall be no earlier than 0.4 bit time from the leading edge of the '1' bit, and no later than 0.6 bit time from the leading edge of the '1' bit.
14. The middle edge in a SYNC bit shall be no earlier than 1.4 bit time from the leading edge of the SYNC bit, and no later than 1.6 bit time from the leading edge of the SYNC bit.
15. The 4% packet-to-packet variation is provided to allow for a "slow drift" of the bit times over short periods, while allowing for a long term drift to the minimum or maximum limits of valid bit times.
16. Timing measurement is made at the 50% point, as defined in Figure 13-33.

Table 13-26. Required Detection Levels of CBUS Signal

Item	Symbol	Min	Max	Unit	Notes
Input High voltage level	V_{IH_CBUS}	1.0		V	1
Input Low voltage level	V_{IL_CBUS}		0.6	V	1

Notes on Table 13-26:

1. The input buffer of the follower shall safely detect these voltage levels as '0' and '1'.

Table 13-27. Source Side CBUS Line Characteristics

Item	Symbol	Min	Typ	Max	Unit	Notes
Capacitance	$C_{SRC\_ON\_CBUS}$			130	pF	1, 2
CBUS Leakage when Tri-stated	$I_{CBUS\_LEAK\_SRC}$			3	µA	
Pull up resistance – discovery	$Z_{CBUS\_SRC\_DISCOVER}$	9k	10k	11k	ohms	
Pull up resistance – on	$Z_{CBUS\_SRC\_ON}$	4k	5k	6k	ohms	

Notes on Table 13-27:

1. Measured with DUT power on and after discovery process is complete and CBUS is connected.
2. Measured using relaxation method as described in the MHL CTS.

Table 13-28. Sink Side CBUS Line Characteristics

Item	Symbol	Min	Typ	Max	Unit	Notes
Capacitance	C_{SINK_CBUS}			80	pF	1, 2
Pull down resistance – discovery	$Z_{CBUS\_SINK\_DISCOVER}$	800	1000	1200	ohms	
Pull down resistance – active	$Z_{CBUS\_SINK\_ON}$	90k	100k	110k	ohms	
CBUS Leakage when Tri-stated	$I_{CBUS\_LEAK\_SINK}$			1	µA	

Notes on Table 13-28:

1. Measured with DUT power on and after discovery process is complete and CBUS is connected.
2. Measured using relaxation method as described in MHL CTS.

Table 13-29. Dongle Side CBUS Line Characteristics

Item	Symbol	Min	Typ	Max	Unit	Notes
Capacitance	C_{DONGLE_CBUS}	30		430	pF	1, 2
Pull down resistance – discovery	$Z_{CBUS\_DONGLE\_DISCOVER}$	800	1000	1200	ohms	
Pull down resistance – active	$Z_{CBUS\_DONGLE\_ON}$	90k	100k	110k	ohms	
CBUS Leakage when Tri-stated	$I_{CBUS\_LEAK\_SINK}$			1	µA	

Notes on Table 13-29:

1. Measured with DUT power on and after discovery process is complete and CBUS is connected.
2. Measured using relaxation method described in MHL CTS.

13.10.1.1 CBUS Connect and Bit Timings

Wake pulse and discovery pulse timings defined in Table 13-30 and Table 13-32 shall be measured at the 50% points on the waveform shown in Figure 13-33. CBUS packet bits defined in

Table 13-25 shall be measured at the 50% points on the waveform shown in Figure 13-33. The 50% point is the midpoint between $V_{IH\_CBUS}\{min\}$ and $V_{IL\_CBUS}\{max\}$.

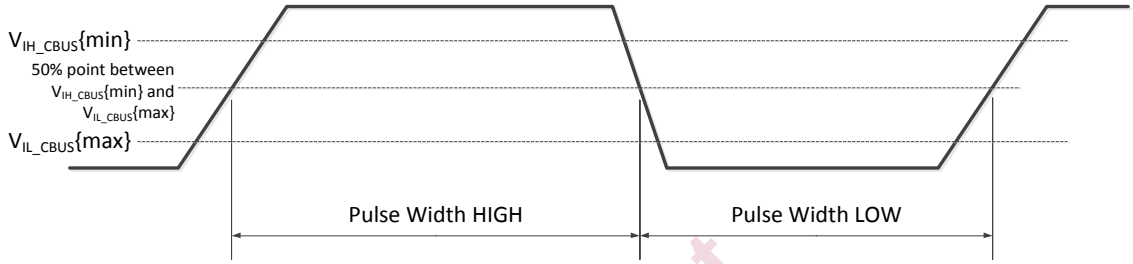


Figure 13-33. CBUS Wake and Discovery Pulse Width Measurement

Table 13-30. Wake Pulse Sequence Timing Parameters

Item	Symbol	Timing			Unit	Notes
		Min	Typ	Max		
Source: Duration for shorter CBUS high and low periods during wake sequence	$T_{SRC\_WAKE\_PULSE\_WIDTH\_1}$	18	20	22	msec	1
Source: Duration of long middle CBUS low period during wake sequence	$T_{SRC\_WAKE\_PULSE\_WIDTH\_2}$	54	60	66	msec	1
Source: Time from end of wake sequence to start of first discovery pulse rising edge	$T_{SRC\_WAKE\_TO\_DISCOVER}$	100		1000	msec	

Notes on Table 13-30:

1. Pulse width shall be measured according to the 50% points shown in Figure 13-33.

Table 13-31. Connection Parameters (part one)

Item	Symbol	Timing			Unit	Notes
		Min	Typ	Max		
Sink: On transition to SINK1 (or DONGLE5) state, time to $Z_{CBUS\_SINK\_DISCOVER}$ and Sink (or Dongle) ready to be discovered	$T_{SINK\_READY\_TO\_DISCOVER}$			100	msec	1
Sink: From Sink VBUS output disabled to Sink VBUS power output stable	$T_{SINK\_VBUS\_EN}$	-	-	100	msec	1, 2
Source: $Z_{CBUS\_SINK\_DISCOVER}$ detected to VBUS and CBUS stable	$T_{SRC\_VBUS\_CBUS\_STABLE}$	100		1000	msec	3
Source: From Source VBUS output disabled to Source VBUS power output stable	$T_{SRC\_VBUS\_EN}$			100	msec	4
Source: From Source VBUS output stable, wait for Dongle stable	$T_{SRC\_VBUS\_OUT\_TO\_STABLE}$	100		1000	msec	5
Source: In Disconnected State: CBUS impedance change detected to impedance measurement.	$T_{SRC\_START\_IMP\_MEAS}$		80		msec	6

Notes on Table 13-31:

1. If necessary, Sink shall float CBUS (state SINK5) until such time as it is capable to meet these parameters.
2. Maximum time for Sink to stabilize its VBUS power output on switch from a VBUS disabled state to a VBUS enabled state.
3. The Source waits at least $T_{SRC\_VBUS\_CBUS\_STABLE}$ (min) to be sure that VBUS from the Dongle or Sink is stable. If Source fails to detect stable VBUS from the Dongle or Sink before $T_{SRC\_VBUS\_CBUS\_STABLE}$(max), then the Source moves to state SRC8.
4. Maximum time for Source to stabilize its VBUS power output on a switch from VBUS disabled state to a VBUS enabled state.
5. $T_{SRC\_VBUS\_OUT\_TO\_STABLE}${min} provides the means for the Source to move from SRC8 to SRC8a or SRC9, thereby limiting the power contention time with an attached Dongle or Sink. See Section 8.2.1.
6. Recommended value to ensure a stable impedance on CBUS after physical cable attachment before impedance measurement.

Table 13-32. Connection Parameters (part two)

Item	Symbol	Timing			Unit	Notes
		Min	Typ	Max		
Source: CBUS pulse width for discovery (high time, float time)	$T_{SRC\_PULSE\_WIDTH}$	80	100	120	µsec	6
Sink: CBUS pulse width (high time, float time) for valid MHL discovery pulse.	$T_{SINK\_PULSE\_WIDTH}$	70		130	µsec	1, 6
Sink: Reject CBUS discovery pulses below this limit.	$T_{SINK\_PULSE\_REJECT\_MIN}$			40	µsec	2, 6
Sink: Reject CBUS discovery pulses above this limit.	$T_{SINK\_PULSE\_REJECT\_MAX}$	200			µsec	2, 6
Source: CBUS high detect during discovery pulse float to Connected State	T_{SRC_CONN}			240	µsec	3
Sink: CBUS discovered to Connected State	T_{SINK_CONN}			40	µsec	4
Sink: CBUS high impedance to CBUS $Z_{CBUS\_SINK\_DISCOVER}$.	$T_{SINK\_CBUS\_FLOAT}$	50			msec	
Source: CBUS high impedance during disconnect	$T_{SRC\_CBUS\_FLOAT}$	50	-		msec	5

Notes on Table 13-32:

1. Sink shall accept pulses that are within these limits.
2. Sink shall reject pulses outside these limits: up to $T_{SINK\_PULSE\_REJECT\_MIN}$ or above $T_{SINK\_PULSE\_REJECT\_MAX}$. (See Figure 8-12-1.)
3. This time is the time from the Source detecting a HIGH on CBUS during its discovery pulse float (due to the Sink assertion of $Z_{CBUS\_SINK\_ON}$), to both (a) when the Source has completed switch of CBUS impedance to $Z_{CBUS\_SRC\_ON}$, and (b) when the Source is able to recognize CBUS packets.
4. This time is the time from the Sink recognizing $N_{SINK\_PULSE\_COUNT}${min} discovery drive-and-float pulses, to both (a) when it has completed the switch of CBUS impedance to $Z_{CBUS\_SINK\_ON}$, and (b) when the Sink is able to recognize CBUS packets.
5. Parameter (float time) is used to allow the Sink pull-down to make CBUS LOW long enough for the Sink to realize that there is a disconnect.
6. Pulse width shall be measured according to the 50% points shown in Figure 13-33.

Table 13-33. Connection Parameters (part three)

Item	Symbol	Timing			Unit	Notes
		Min	Typ	Max		
Sink: Connected State to RxSense enable	$T_{SINK\_RXSENSE\_EN}$			200	msec	1
Source: RxSense deglitch window	$T_{SRC\_RXSENSE\_DEGLITCH}$	100		200	msec	2
Source: Connected State to RxSense detection	$T_{SRC\_RXSENSE\_CHK}$	300		500	msec	3
Sink: Connected State to First CBUS Packet Allowed	$T_{SINK\_ARBITRATE}$	500			µsec	4
Source: Connected State to First CBUS Packet Allowed	$T_{SRC\_ARBITRATE}$	500			µsec	5
Sink: Discovery pulse count to assume Connected State	$N_{SINK\_PULSE\_COUNT}$	5		5	pulses	6
Source: Discovery pulses attempted to conclude no MHL device is attached	$N_{SRC\_PULSE\_COUNT}$	6		20	pulses	7

Notes on Table 13-33:

1. Sink shall assert termination on MHL+ and MHL- to $Z_{RXSENSE\_TERM}$ or $Z_{RXSENSE\_TMDS}$.
2. Glitches on RxSense refer to momentary transitions in measured impedance between its three defined ranges.Transitions on RxSense shall be deglitched according to $T_{SRC\_RXSENSE\_DEGLITCH}$:
    a. A Source shall not disconnect due to a glitch on RxSense with a duration of less than $T_{SRC\_RXSENSE\_DEGLITCH}\{min\}$.
    b. A Source shall disconnect due to a glitch on RxSense with a duration of more than $T_{SRC\_RXSENSE\_DEGLITCH}\{max\}$.
3. Source shall wait this time before concluding Sink will not assert RxSense.
4. Sink shall wait at least this time before beginning first arbitration on CBUS. Sink shall be able to receive packets during this period. Once Sink has observed arbitration from the Source, Sink may arbitrate for the CBUS even before $T_{SINK\_ARBITRATE}$ completes. See Figure 8-8.
5. Source shall wait at least this time before beginning first arbitration on CBUS. Source shall be able to receive packets during this period. Once Source has observed arbitration from the Sink, Source may arbitrate for the CBUS even before $T_{SRC\_ARBITRATE}$ completes. See Figure 8-8.
6. A Sink which detects $N_{SINK\_PULSE\_COUNT}\{min\}$ shall change its pull-down resistance on CBUS from $Z_{CBUS\_SINK\_DISCOVER}$ to $Z_{CBUS\_SINK\_ON}$.
7. A Source which detects $N_{SRC\_PULSE\_COUNT}\{max\}$ discovery pulses shall assume that the attached device is not an MHL Sink. See Section 8.2.

13.10.1.2 CBUS Disconnect Timings

Table 13-34. Disconnection Parameters

Item	Symbol	Timing			Unit	Notes
		Min	Typ	Max		
Sink: In Connected State, CBUS lowtime detected as Disconnected State	$T_{SINK\_CBUS\_DISCONN}$	150		240	µsec	1, 2
Sink (or Dongle): After discovery pulse recognition, set of DCAP_RDY status.	$T_{SINK\_DCAP\_RDY}$			500	msec	
Sink (or Dongle): Time in SINK5 from arrival from SINK1.	$T_{SINK\_SINK1\_SINK5}$			60	sec	3
Dongle: In Connected State, detects its own power on to asserting DCAP_CHG interrupt.	$T_{DONGLE\_DCAP\_CHG}$			500	msec	
Source: In Connected State, recognizes DCAP_CHG or DCAP_RDY to read of Device Capability registers.	$T_{SRC\_READ\_DCAP}$			500	msec	
Source: From VBUS enable to disable state	$T_{SRC\_VBUS\_DIS}$			200	msec	4
Source: Disconnected state to TMDS driver disabled and to a CBUS impedance meeting $I_{CBUS\_LEAK\_SRC}$	$T_{SRC\_CBUS\_TMDS\_DIS}$			150	msec	
Sink: Disconnected State to $Z_{CBUS\_SINK\_DISCOVER}$	$T_{SINK\_CBUS\_SINK\_CONN}$			5	msec	5
Sink: Disconnected State to RxSense inactive (high impedance)	$T_{SINK\_RXSENSE\_DIS}$	-	-	5	msec	

Notes on Table 13-34:

1. This includes the case where the Sink keeps repeating a transaction on the bus because the Source does not ACK – in this case the Sink should still time out after 240 µsec.
2. Sink shall not disconnect if CBUS is low for less than $T_{SINK\_CBUS\_DISCONN}\{min\}$, and shall disconnect if CBUS is low for at least $T_{SINK\_CBUS\_DISCONN}\{max\}$. Sink may disconnect if low time is between $T_{SINK\_CBUS\_DISCONN}\{min\}$ and $T_{SINK\_CBUS\_DISCONN}\{max\}$.
3. Sink shall not remain in SINK5 state for longer than $T_{SINK\_SINK1\_SINK5}\{max\}$ after transitioning there from SINK1.
4. Source shall meet this parameter when Source sees the Sink's POW bit changing from '0' to '1' while in a CBUS connected state, or any time it switches to a state where its VBUS output is deasserted.
5. When a Sink exits a connected state (leaves SINK2 or SINK3), the Sink shall switch its CBUS impedance to $Z_{CBUS\_SINK\_DISCOVER}$ within $T_{SINK\_CBUS\_SINK\_CONN}$.

13.10.2 CBUS Link Layer Timing Specifications

An MHL Source or Sink shall operate its CBUS with a bit time, T_{CBIT}. Sources and Sinks are allowed to maintain their own bit times. The follower shall adjust its bit time to the initiator by the duration of the initiator's SYNC pulse. This is considered to be the packet bit time. The bit time of subsequent bits within the same packet shall be within $T_{BIT\_VARY\_PACKET}$ of this packet bit time. The follower shall recover the packet using the packet bit time of the initiator.

A timing parameter in Table 13-35 allows timings with any value within the min/max range. Signaling is not limited to integral numbers of bit times.

Table 13-35. CBUS Link Layer Timing Parameters

Item	Symbol	Timing		Unit	Notes
		Min	Max		
CBUS total arbitration time	T_{ARB_TOT}	4	4	T_{BIT_CBUS}	
CBUS Source arbitration LOW time	T_{ARB_SRC}	4	4	T_{BIT_CBUS}	
CBUS Sink arbitration LOW time	T_{ARB_SINK}	1	1	T_{BIT_CBUS}	
CBUS requester transmit opportunity after arbitration	T_{REQ_OPP}	1	2	T_{BIT_CBUS}	
CBUS wait time for arbitration	T_{WAIT}	12	-	T_{BIT_CBUS}	1
CBUS requester continue after ack	T_{REQ_CONT}	0	1	T_{BIT_CBUS}	
CBUS requester hold-off after nack	T_{REQ_HOLD}	8		T_{BIT_CBUS}	
CBUS responder hold-off after end of packet	T_{RESP_HOLD}	4		T_{BIT_CBUS}	2
Maximum packets without re-arbitration	N_{MAX}	-	24	packets	
Re-try attempts after link layer nack	N_{RETRY}	32			3

Notes on Table 13-35:

1. In case the device last had control of the bus (as initiator), T_{WAIT} is measured from the end of the acknowledge bit time. In every other case, T_{WAIT} is measured from the last rising edge on CBUS.
2. T_{RESP_HOLD} is measured from the last rising edge on CBUS.
3. A Sink or Dongle should provide a pause of at least 350 microseconds between series of N_{RETRY} attempts, to enable the Sink or Dongle to detect a disconnected CBUS state which is signaled by the Source by holding CBUS low for $T_{SINK\_CBUS\_DISCONN}$ (see Table 13-34).

13.10.3 CBUS Translation Layer Timing Specifications

The timing specifications in Table 13-36 should be met by the Source or Sink, whether the timings are implemented in hardware or firmware.

Table 13-36. CBUS Translation Layer Timing Parameters

Item	Symbol	Timing		Unit	Notes
		Min	Max		
Delay from ABORT to next MSC command	$T_{ABORT\_NEXT\_MSC}$	2		sec	1
Timeout for a device waiting for a packet within a command	$T_{PKT\_RECEIVER\_TIMEOUT}$	100		msec	2, 4, 5
Timeout (budget) for a device sending a packet within a command	$T_{PKT\_SENDER\_TIMEOUT}$		100	msec	3, 4, 5
Timeout for a device waiting for a command	$T_{CMD\_RECEIVER\_TIMEOUT}$	320		msec	6, 7
Timeout (budget) for a device sending a command	$T_{CMD\_SENDER\_TIMEOUT}$		320	msec	6, 8
Delay from NACK to MSC_MSG to sending next MSC_MSG.	$T_{NACK\_RETRY\_NEXT}$	1000		msec	9
Delay from CLR_HPD to SET_HPD	T_{HPD_WIDTH}	100		msec	
Wait time for RCPK or RCPE	T_{RCP_WAIT}		1000	msec	
Wait time for RAPK	T_{RAP_WAIT}		1000	msec	
Time to Notify Change in Link Mode	$T_{MODE\_CHG\_DLY}$		100	msec	
Time to initiate RCP "press & hold"	T_{PRESS_MODE}	300		msec	
Time to maintain RCP "press & hold"	$T_{HOLD\_MAINTAIN}$		2	sec	
Time for GRT_WRT in reply to REQ_WRT for WRITE_BURST	T_{BURST_WAIT}		1000	msec	

Notes on Table 13-36:

1. A device shall wait $T_{ABORT\_NEXT\_MSC}$ before starting a new MSC command after receiving an MSC ABORT during the completion of an MSC command.
2. Requester shall wait at least $T_{PKT\_RECEIVER\_TIMEOUT}${min} before taking action. See Sections 7.4.1 and 7.5.2.
3. Responder shall wait at most $T_{PKT\_SENDER\_TIMEOUT}${max} before taking action. See Sections 7.4.1 and 7.5.2.
4. Packet sender and packet receiver: A requester or a responder may be a packet receiver or packet sender at different points in a transaction.
5. These timeouts apply to both the DDC and MSC sub-channels.
6. These timeouts apply only to the MSC sub-channel.
7. Responder shall wait at least $T_{CMD\_\_RECEIVER\_TIMEOUT}${min} before taking action. See Sections 7.4.1.
8. A Requester shall send all bytes of a command within this time (starting with the control packet and including all packets up to the responder's first packet).See Sections 7.4.1.

9. Requester is recommended to hold-off this minimum time. An MSC_MSG may be, for example, an RAP sub-command, from which a translation layer NACK is received from the responder. See Section 7.4.3.11.

13.11 Data Island Timing Specifications

13.11.1 Audio InfoFrame Timing Specifications

Table 13-37. Audio InfoFrame Timing Parameters

Item	Symbol	Min	Max	Unit	Notes
Time between Audio InfoFrames	T_{AIF}		2	Video Fields	1
Time from audio stream mode change to first Audio InfoFrame	T_{AIF0}		33	msec	
Time delay between audio and video streams through MHL link	T_{AV_SKEW}	-2	+2	msec	2

Notes on Table 13-37:

1. An accurate Audio InfoFrame shall be transmitted at least every 2 Video Fields.
2. This skew is the result of the TMDS encoding and decoding, and is averaged across at least 3 consecutive frames. The skew is not the result of intentional adjustment of audio to video delay.

13.12 Robustness Requirements

No damage to the MHL Source or Sink shall result from the shorting of any combination of signals on any connector. If two MHL Sources are connected together with a single cable, no damage shall occur to either of the Sources. If two MHL Sinks are connected together with a single cable, no damage shall occur to either of the Sinks.

14 Informative Appendices

14.1 Physical Devices and Logical Devices

MHL provides a means for each device to identify itself as one or more logical devices on the connection, in order for the opposite device to understand the RCP and RAP commands supported. The following sub-sections are provides to more clearly explain examples in which the LOG_DEV_MAP and other registers are used.

14.1.1 DVD Player with One Disk

A media player with only one disk should identify itself as LD_VIDEO (since it supports output of video content to the downstream MHL device), LD_AUDIO (since it supports output of audio content to the downstream MHL device), LD_MEDIA (since it supports content media), and LD_GUI (since it is able to output its user interface to the downstream display and respond to user menu selection commands).

END OF DOCUMENT