

Universal Serial Bus Power Delivery Specification

Release 3.2, Version 1.2

May 20, 2026

Universal Serial Bus Power Delivery Specification

Release 3.2, Version 1.2

Copyright © 2026 USB 3.0 Promoter Group: Apple Inc., Hewlett-Packard Inc., Intel Corporation, Microsoft Corporation, Renesas, STMicroelectronics, and Texas Instruments.

LIMITED COPYRIGHT LICENSE

The USB 3.0 Promoters grant a conditional copyright license under the copyrights embodied in the USB Power Delivery Specification to use and reproduce the specification for the sole purpose of, and solely to the extent necessary for, evaluating whether to implement the specification in products that would comply with the specification. Without limiting the foregoing, use of the specification for the purpose of filing or modifying any patent application to target the specification or USB-compliant products is not authorized. Except for this express copyright license, no other rights or licenses are granted, including without limitation any patent licenses. In order to obtain any additional intellectual property licenses or licensing commitments associated with the specification, a party must execute the USB 3.0 Adopters Agreement. Note: By using the specification, you accept these license terms on your own behalf and, in the case where you are doing this as an employee, on behalf of your employer.

INTELLECTUAL PROPERTY DISCLAIMER

This specification is provided to you "as is" with no warranties whatsoever, including any warranty of merchantability, non-infringement, or fitness for any particular purpose. The authors of this specification disclaim all liability, including liability for infringement of any proprietary rights, relating to use or implementation of information in this specification. The provision of this specification to you does not provide you with any license, express or implied, by estoppel or otherwise, to any intellectual property rights.

- USB Type-C[®], USB-C[®], and USB4[®] are trademarks of the Universal Serial Bus Implementers Forum (USB-IF). DisplayPort[®] is a registered trademark of VESA. Thunderbolt[™] is a trademark of Intel Corporation. All product names are trademarks, registered trademarks, or service marks of their respective owners.
- You may only use the Thunderbolt[™] trademark or logo in conjunction with products designed to this specification that complete proper certification and execute a Thunderbolt[™] trademark license. See usb.org/compliance for further information.
- All rights reserved.
- Please send comments via electronic mail to techsup@usb.org.
- For industry information, refer to the USB Implementers Forum website at <http://www.usb.org/>.

Table of Contents

1. Introduction	21
1.1. Cable and Connectors	21
1.2. Operational Overview	21
1.3. USB PD Roles	21
1.3.1. Source Port	21
1.3.2. Sink Port	22
1.3.3. Downstream Facing Port (DFP)	22
1.3.4. Upstream Facing Port (UFP)	22
1.3.5. Dual-Role Power (DRP) Ports	22
1.3.6. Dual-Role Data (DRD) Ports	22
1.3.7. VCONN Source	22
1.3.8. Cable Plugs	22
1.4. Power Delivery Operational Contracts	22
1.5. Source and Sink Operation in USB PD	23
1.5.1. General Source-to-Sink Initial Interaction	23
1.6. Common PD Device Types	24
1.7. USB-PD Architectural Overview	24
2. Glossary and Info	26
2.1. Conventions	26
2.1.1. Precedence	26
2.1.2. Keywords	26
2.1.3. Numbering	27
2.2. Related Documents	27
2.3. Terms and Abbreviations	27
3. Port Power Requirements	37
3.1. Introduction	37
3.2. Source Port Power Requirements	37
3.2.1. General Source Port Requirements	37
3.2.2. Guaranteed Capability Port	37
3.2.3. Managed Capability Port	38
3.2.4. Source Voltage and Current Requirements	38
3.2.5. Optional Source Voltage/Currents	41
3.2.5.1. Fixed, Variable and Battery Supply	41
3.2.5.2. SPR Programmable Power Supply (PPS)	42
3.2.5.3. Extended Power Range (EPR)	42
3.3. Dynamic Power Supply (DPS)	45
3.4. Sink Power Requirements	46
3.4.1. Sink Power Rule Considerations	46
3.4.2. Sink Requirements	47
3.5. Sink with Accessory Support	47
3.6. PDUSB Device Requirements	47
4. Power Delivery Electrical Requirements	48
4.1. Source Requirements	48
4.1.1. Behavioral Aspects	48
4.1.2. Source Bulk Capacitance	48
4.1.2.1. Source PDOs or APDOs	49
4.1.3. Source Transitions	50
4.1.3.1. Fixed Voltage Transitions	50
4.1.3.2. AVS/PPS Transitions	52
4.1.4. Non-application of VBUS Slew Rate Limits	58
4.1.5. Robust Source Operation	59
4.1.5.1. Output Over-Current Protection	59

4.1.5.2. Output Over-Voltage Protection	59
4.1.5.3. Over-Temperature Protection	59
4.1.5.4. vSafe5V Externally Applied to Ports Supplying vSafe5V	59
4.1.5.5. Detach	60
4.1.6. Source Peak Current Operation	60
4.1.7. Source Capabilities Extended Parameters	60
4.1.7.1. Load Step Slew Rate	61
4.1.7.2. Load Step Magnitude	61
4.1.7.3. Holdup Time Field	61
4.1.7.4. Compliance Field	61
4.1.7.5. Batteries	61
4.2. Sink Requirements	62
4.2.1. Behavioral Aspects	62
4.2.2. Sink Bulk Capacitance	62
4.2.3. Sink Standby	63
4.2.4. Suspend Power Consumption	63
4.2.5. Zero Negotiated Power	63
4.2.6. Transient Load Behavior	63
4.2.7. Considerations for Current Limit in PPS Mode	63
4.2.8. Sink Peak Current Operation	64
4.2.9. Robust Sink Operation	64
4.2.9.1. Sink Bulk Capacitance Discharge at Detach	64
4.2.9.2. Sink Over-Current Protection	64
4.2.9.3. Sink Over-Voltage Protection	65
4.2.9.4. Sink Over-Temperature Protection	65
4.3. Dual Role Ports	65
4.3.1. Swap Standby for the Initial Source	65
4.3.2. Swap Standby for the Initial Sink	65
4.3.3. Power Role Swap Sequence	66
4.4. Response to Hard Resets	67
4.4.1. Hard Reset Sequence	68
4.5. Transitions	69
4.5.1. Fixed and AVS transitions	69
4.5.1.1. Large-step voltage increases (Fixed or AVS)	69
4.5.1.2. Small-step voltage increases (AVS)	71
4.5.1.3. Unchanged or decreased voltage transitions (current unchanged, increased, or de- creased)	72
4.5.2. PPS voltage or current changes	74
4.5.2.1. PPS voltage transitions (CV Mode)	74
4.5.2.2. PPS current transitions (CL Mode)	76
4.6. VCONN Power Cycle	78
4.7. Electrical Parameters	78
4.7.1. Source Electrical Parameters	78
4.7.2. Sink Electrical Parameters	83
4.7.3. Common Electrical Parameters	84
5. PD Communications Physical Layer	85
5.1. Overview	85
5.2. Signaling Through DC Biasing	85
5.2.1. Power Role Indication	85
5.2.2. Collision Avoidance	85
5.2.2.1. Definition of Idle	86
5.3. BMC Signaling	86
5.3.1. Transmission	86
5.3.1.1. Packet Formation	87
5.3.1.2. Resets	90

5.3.2. Reception	91
5.3.2.1. Preamble	94
5.3.2.2. Ordered Sets (K-codes)	94
5.3.2.3. Payload Processing	94
5.3.3. Electrical Characteristics for Transmission and Reception	95
5.3.4. Encoding and Signaling	95
5.3.4.1. Transmit Mask	98
5.3.4.2. Receive Masks	100
5.3.4.3. Transmit Load Model	105
5.3.4.4. Cable Plug Transceivers	106
5.3.4.5. Transceiver with VCONN Source Capability	107
5.3.4.6. Capacitance when not transmitting	108
5.3.4.7. Source Output Impedance	108
5.3.4.8. Bit Rate Drift	108
5.3.4.9. Inter-Frame Gap	109
5.3.4.10. Shorting of Transmitter Output	109
5.4. Built in Self-Test (BIST)	109
5.4.1. BIST Carrier Mode	109
5.5. PD Communications Physical Layer Parameters	110
5.5.1. BMC Common Parameters	110
5.5.2. BMC Transmitter Parameters	110
5.5.3. BMC Receiver Parameters	110
6. PD Communications Protocol Message Definitions	112
6.1. Overview	112
6.1.1. Message Types	112
6.1.2. Message ID	112
6.1.3. Specification Revision	112
6.1.3.1. SOP Specification Revision Detection Process	113
6.1.3.2. SOP' Specification Revision Detection Process	113
6.1.4. Chunking	114
6.1.5. Vendor and Product IDs	115
6.2. Message Construction	116
6.2.1. Message Header	116
6.2.2. Number of Data Objects	117
6.3. Control Message	118
6.3.1. GoodCRC Message	119
6.3.2. GotoMin Message (Deprecated)	119
6.3.3. Accept Message	119
6.3.4. Reject Message	119
6.3.5. Ping Message (Deprecated)	120
6.3.6. PS_RDY Message	120
6.3.7. Get_Source_Cap Message	120
6.3.8. Get_Sink_Cap Message	121
6.3.9. DR_Swap Message	121
6.3.10. PR_Swap Message	121
6.3.11. VCONN_Swap Message	121
6.3.12. Wait Message	121
6.3.13. Soft_Reset Message	122
6.3.14. Data_Reset Message	122
6.3.15. Data_Reset_Complete Message	122
6.3.16. Not_Supported Message	122
6.3.17. Get_Source_Cap_Extended Message	123
6.3.18. Get_Status Message	123
6.3.19. FR_Swap Message	123
6.3.20. Get_PPS_Status	123

6.3.21. Get_Country_Codes	123
6.3.22. Get_Sink_Cap_Extended Message	124
6.3.23. Get_Source_Info Message	124
6.3.24. Get_Revision Message	124
6.4. Data Message	124
6.4.1. Capabilities Messages	125
6.4.1.1. Source_Capabilities Message	125
6.4.1.2. Sink_Capabilities Message	126
6.4.1.3. Power Data Objects	126
6.4.2. Request Message	136
6.4.2.1. Sink Request Data Objects	137
6.4.3. BIST Message	140
6.4.3.1. BIST Test Data Mode	140
6.4.3.2. BIST Shared Capacity Test Mode	140
6.4.3.3. BIST Shared Test Mode Entry	140
6.4.3.4. BIST Shared Test Mode Exit	141
6.4.4. Battery_Status Message	141
6.4.5. Alert Message	142
6.4.5.1. Power State Change Extended Event	144
6.4.5.2. Power Button Press Extended Event	144
6.4.5.3. Power Button Release Extended Event	144
6.4.5.4. Controller Initiated Wake Extended Event	144
6.4.5.5. Source Reducing Capabilities Extended Event	144
6.4.6. Get_Country_Info Message	144
6.4.7. Enter_USB Message	145
6.4.8. EPR_Request Message	146
6.4.9. EPR_Mode Message	147
6.4.10. Source_Info Message	148
6.4.11. Revision Message	149
6.4.12. Vendor Defined Message	150
6.4.12.1. Unstructured VDM	151
6.4.12.2. Structured VDM	151
6.4.12.3. Discover Identity	154
6.4.12.4. Discover SVIDs	168
6.4.12.5. Discover Mode	170
6.4.12.6. Enter Mode	170
6.4.12.7. Exit Mode	170
6.4.12.8. Attention	171
6.5. Extended Message	171
6.5.1. Extended Message Header	172
6.5.1.1. Chunked	172
6.5.2. Source_Capabilities_Extended Message	173
6.5.2.1. Touch Current Field	175
6.5.2.2. Peak Current Fields	176
6.5.3. Status Message	176
6.5.4. Get_Battery_Cap Message	181
6.5.5. Get_Battery_Status Message	181
6.5.6. Battery_Capabilities Message	182
6.5.7. Get_Manufacturer_Info Message	182
6.5.8. Manufacturer_Info Message	183
6.5.9. Security_Request Message	183
6.5.10. Security_Response Message	183
6.5.11. Firmware_Update_Request Message	184
6.5.12. Firmware_Update_Response Message	184
6.5.13. PPS_Status Message	184

6.5.14. Country_Info Message	185
6.5.15. Country_Codes Message	186
6.5.16. Sink_Capabilities_Extended Message	186
6.5.16.1. Minimum PDP	189
6.5.16.2. Operational PDP	189
6.5.16.3. Maximum PDP	190
6.5.17. Extended_Control Message	190
6.5.17.1. EPR_Get_Source_Cap Message	190
6.5.17.2. EPR_Get_Sink_Cap Message	190
6.5.17.3. EPR_Keep_Alive Message	191
6.5.17.4. EPR_Keep_Alive_Ack Message	191
6.5.18. EPR_Source_Capabilities Message	191
6.5.19. EPR_Sink_Capabilities Message	192
6.5.20. Vendor_Defined_Extended Message	192
6.6. Value Parameters	193
7. PD Communications Protocol Message Usage	194
7.1. Reset	194
7.1.1. Soft Reset and Protocol Error	194
7.1.2. Data Reset	196
7.1.3. Hard Reset	196
7.1.3.1. Cable Plugs and Hard Reset	197
7.1.3.2. Modal Operation and Hard Reset	197
7.1.4. Cable Reset	197
7.2. Collision Avoidance	197
7.3. Message Discarding	198
7.4. Common Messages	198
7.4.1. Accept	198
7.4.2. Reject	198
7.4.3. Wait	199
7.5. Applicability Tables	199
7.6. AMS Usages	204
7.6.1. Basic Message Exchange	204
7.6.2. Possible Points of Failure for Message Delivery	205
7.7. Soft Reset AMS	206
7.8. Data Reset AMS	206
7.9. Power Negotiation (SPR) AMS	207
7.9.1. Source_Capabilities Message	207
7.9.1.1. Power Data Objects	207
7.9.1.2. USB Suspend Supported	207
7.9.1.3. Unconstrained Power	207
7.9.1.4. EPR Mode Capable	208
7.9.1.5. Source Offering No Capabilities	208
7.9.2. Request Message	208
7.9.2.1. Capability Mismatch	208
7.9.2.2. Wait in Response to Request Message	209
7.9.3. PS_Rdy	209
7.10. Power Role Swap AMS	209
7.10.1. Wait in response to a PR_Swap Message	210
7.11. Fast Role Swap AMS	210
7.12. Data Role Swap AMS	210
7.12.1. DR_Swap	210
7.12.2. Wait in response to a DR_Swap Message	211
7.13. VCONN Swap AMS	211
7.13.1. Wait in response to a VCONN_Swap Message	212
7.14. Alert AMS Message	212

7.14.1. Alert Types	212
7.15. Get Status AMS	212
7.16. Get PPS Status AMS	213
7.17. Get Source Capabilities (SPR) AMS	213
7.18. Get Sink Capabilities (SPR) AMS	213
7.19. Extended Capabilities AMS	213
7.19.1. Get_Source_Cap_Extended	213
7.19.2. Source_Capabilities_Extended	213
7.19.2.1. Touch Current Field	213
7.19.2.2. Peak Current Field	213
7.19.2.3. Number of Batteries/Battery Slots Field	214
7.19.3. Get_Sink_Cap_Extended	214
7.19.4. Sink_Capabilities_Extended	214
7.19.4.1. XID Field	214
7.19.4.2. Battery Info	214
7.19.4.3. Sink Modes	214
7.20. Battery Capabilities AMS	214
7.20.1. Get_Battery_Cap Message	214
7.21. Battery Status AMS	214
7.21.1. Get_Battery_Status Message	214
7.21.2. Battery Status Message	214
7.22. Manufacturer Information AMS	215
7.22.1. Get_Manufacturer_Info Message	215
7.22.2. Manufacturer_Info Message	215
7.23. Country Codes AMS	215
7.23.1. Get_Country_Codes Message	215
7.23.2. Country Codes Message	215
7.24. Country Information AMS	215
7.24.1. Country_Info Message	215
7.25. Revision Information AMS	215
7.25.1. Get_Revision Message	215
7.25.2. Revision Message	215
7.26. Source Information AMS	215
7.26.1. Get_Source_Info Message	215
7.26.2. Source_Info Message	215
7.26.2.1. Port Maximum PDP Field	216
7.26.2.2. Port Present PDP Field	216
7.26.2.3. Port Guaranteed PDP Field	216
7.27. Security AMS	216
7.28. Firmware Update AMS	216
7.29. Enter USB AMS	217
7.29.1. Wait in Response to an Enter_USB Message	217
7.30. EPR Mode AMS	217
7.30.1. Process to Enter EPR Mode AMS	217
7.30.2. EPR_Source_Capabilities Message	221
7.30.3. EPR_Request Message	221
7.30.4. EPR_Sink_Capabilities Message	222
7.30.5. Operation in EPR Mode	222
7.30.6. EPR_Keep_Alive Message	222
7.30.7. Exiting EPR Mode	222
7.30.7.1. Commanded Exit	222
7.30.7.2. Implicit Exit	223
7.30.7.3. Exits due to errors	223
7.30.8. EPR_Get_Source_Cap Message	223
7.30.9. EPR_Get_Sink_Cap Message	223

7.31. Timers	224
7.31.1. CRCReceiveTimer	224
7.31.2. SenderResponseTimer	224
7.31.3. Capability Timers	224
7.31.3.1. SourceCapabilityTimer	225
7.31.3.2. SinkWaitCap Timer	225
7.31.3.3. tFirstSourceCap	225
7.31.4. Wait Timers and Times	225
7.31.4.1. SinkRequestTimer	225
7.31.4.2. tPRSwapWait	226
7.31.4.3. tDRSwapWait	226
7.31.4.4. tVconnSwapWait	226
7.31.4.5. tVconnSwapDelayDFP	226
7.31.4.6. tVconnSwapDelayUFP	226
7.31.4.7. tEnterUSBWait	226
7.31.5. Power Supply Timers	226
7.31.5.1. PSTransition Timer	226
7.31.5.2. PSSourceOffTimer	227
7.31.5.3. PSSourceOnTimer	227
7.31.5.4. NoResponseTimer	228
7.31.6. BIST Timers	228
7.31.6.1. tBISTCarrierMode	228
7.31.6.2. BISTContModeTimer	228
7.31.6.3. tBISTSharedTestMode	228
7.31.7. Power Role Swap Timers	229
7.31.7.1. SwapSourceStartTimer	229
7.31.8. Soft Reset Timers	229
7.31.8.1. tSoftReset	229
7.31.8.2. tProtErrSoftReset	229
7.31.9. Data Reset Timers	229
7.31.9.1. VCONNDischargeTimer	229
7.31.9.2. tDataReset	229
7.31.9.3. DataResetFailTimer	229
7.31.9.4. DataResetFailUFPTimer	229
7.31.10. Hard Reset Timers	230
7.31.10.1. HardResetCompleteTimer	230
7.31.10.2. PSHardResetTimer	230
7.31.10.3. tDRSwapHardReset	230
7.31.10.4. tProtErrHardReset	230
7.31.10.5. tSendHardReset	230
7.31.10.6. tInitiateErrorRecovery	230
7.31.11. Structured VDM Timers	230
7.31.11.1. VDMResponseTimer	230
7.31.11.2. VDMModeEntryTimer	231
7.31.11.3. VDMModeExitTimer	231
7.31.11.4. tVDMBusy	231
7.31.12. VCONN Timers	231
7.31.12.1. VconnOnTimer	231
7.31.12.2. tVconnSourceOff	232
7.31.12.3. tCableMessage	232
7.31.12.4. DiscoverIdentityTimer	232
7.31.13. Collision Avoidance Timers	232
7.31.13.1. SinkTxTimer	232
7.31.13.2. tSrcHoldsBus	232
7.31.14. Fast Role Swap Timers	232

7.31.14.1. tFRSwap5V	232
7.31.14.2. tFRSwapComplete	233
7.31.14.3. tFRSwapInit	233
7.31.15. Chunking Timers	233
7.31.15.1. ChunkingNotSupportedTimer	233
7.31.15.2. ChunkSenderRequestTimer	233
7.31.15.3. ChunkSenderResponseTimer	234
7.31.16. Programmable Power Supply Timers	234
7.31.16.1. SinkPPSPeriodicTimer	234
7.31.16.2. SourcePPSCommTimer	234
7.31.17. tEnterUSB	235
7.31.18. EPR Timers	235
7.31.18.1. SinkEPREnterTimer	235
7.31.18.2. SourceEPRKeepAlive Timer	235
7.31.18.3. tEPSSourceCableDiscovery	235
7.31.19. Time Values and Timers	236
7.32. Counters	238
7.32.1. MessageID Counter	238
7.32.2. Transmitter Usage	238
7.32.3. Receiver Usage	239
7.32.4. Retry Counter	239
7.32.5. Hard Reset Counter	239
7.32.6. Capabilities Counter	239
7.32.7. Discover Identity Counter	240
7.32.8. VDMBusyCounter	240
7.32.9. Counter Values and Counters	240
8. Vendor Defined Message Usage and Alternate Modes	241
8.1. Overview	241
8.2. Alternate Modes	241
8.3. General VDM Rules	241
8.4. Unstructured VDM Rules	241
8.5. Structured VDM Rules	242
8.5.1. Structured VDM Version	242
8.5.2. Object Position	242
8.5.3. SVID	242
8.6. Command Usage	242
8.6.1. Discover Identity	244
8.6.1.1. SOP Use	244
8.6.1.2. SOP' Usage	244
8.6.2. Discover SVIDs	245
8.6.2.1. SOP' Usage	246
8.6.3. Discover Modes	246
8.6.4. Enter Mode Command	246
8.6.5. Exit Mode Command	248
8.6.6. Attention	249
8.7. Command Processes	250
8.7.1. Discovery Process	250
8.7.1.1. Port Partner Discovery	250
8.7.1.2. Cable Plug Discovery	251
8.7.2. Entering Alternate Modes	251
8.7.3. Exiting Alternate Modes	251
9. State Machine Diagrams	254
9.1. Protocol Layer State Diagrams	254
9.1.1. Introduction to Protocol Layer State Diagrams	254
9.1.2. State Operation	255

9.1.2.1. Protocol Layer Chunking	255
9.1.2.2. Protocol Layer Message Transmission	264
9.1.2.3. Protocol Layer Message Reception	271
9.1.2.4. Hard Reset operation	273
9.2. Policy Engine Layer State Diagrams	276
9.2.1. Introduction to State diagrams	276
9.2.2. SenderResponseTimer State Diagram	278
9.2.2.1. SRT_Stopped State	279
9.2.2.2. SRT_Running State	279
9.2.2.3. SRT_Expired State	280
9.2.3. Policy Engine Source Port State Diagram	280
9.2.3.1. PE_SRC_Startup State	282
9.2.3.2. PE_SRC_Discovery State	282
9.2.3.3. PE_SRC_Send_Capabilities State	283
9.2.3.4. PE_SRC_Negotiate_Capability State	284
9.2.3.5. PE_SRC_Transition_Supply State	284
9.2.3.6. PE_SRC_Ready State	285
9.2.3.7. PE_SRC_Disabled State	286
9.2.3.8. PE_SRC_Capability_Response State	286
9.2.3.9. PE_SRC_Hard_Reset State	286
9.2.3.10. PE_SRC_Hard_Reset_Received State	287
9.2.3.11. PE_SRC_Transition_to_default State	287
9.2.3.12. PE_SRC_Get_Sink_Cap State	287
9.2.3.13. PE_SRC_Wait_New_Capabilities State	288
9.2.3.14. PE_SRC_EPR_Keep_Alive State	288
9.2.3.15. PE_SRC_Give_Source_Cap State	288
9.2.4. Policy Engine Sink Port State Diagram	288
9.2.4.1. PE_SNK_Startup State	290
9.2.4.2. PE_SNK_Discovery State	290
9.2.4.3. PE_SNK_Wait_for_Capabilities State	290
9.2.4.4. PE_SNK_Evaluate_Capability State	290
9.2.4.5. PE_SNK_Select_Capability State	291
9.2.4.6. PE_SNK_Transition_Sink State	291
9.2.4.7. PE_SNK_Ready State	292
9.2.4.8. PE_SNK_Hard_Reset State	292
9.2.4.9. PE_SNK_Transition_to_default State	292
9.2.4.10. PE_SNK_Give_Sink_Cap State	293
9.2.4.11. PE_SNK_EPR_Keep_Alive	293
9.2.4.12. PE_SNK_Get_Source_Cap State	293
9.2.5. SOP Soft Reset and Protocol Error State Diagrams	294
9.2.5.1. SOP Source Port Soft Reset and Protocol Error State Diagram	294
9.2.5.2. SOP Sink Port Soft Reset and Protocol Error State Diagram	295
9.2.6. Data Reset State Diagrams	297
9.2.6.1. DFP Data_Reset Message State Diagrams	297
9.2.6.2. UFP Data_Reset Message State Diagrams	300
9.2.7. Not Supported Message State Diagrams	302
9.2.7.1. Source Port Not Supported Message State Diagram	302
9.2.7.2. Sink Port Not Supported Message State Diagram	303
9.2.8. Alert State Diagrams	304
9.2.8.1. Source Port Source Alert State Diagram	304
9.2.8.2. Sink Port Source Alert State Diagram	306
9.2.8.3. Sink Port Sink Alert State Diagram	306
9.2.8.4. Source Port Sink Alert State Diagram	308
9.2.9. Source/Sink Capabilities Extended State Diagrams	308
9.2.9.1. Sink Port Get Source Capabilities Extended State Diagram	308

9.2.9.2. Source Give Source Capabilities Extended State Diagram	309
9.2.9.3. Source Port Get Sink Capabilities Extended State Diagram	310
9.2.9.4. Sink Give Sink Capabilities Extended State Diagram	310
9.2.10. Source Information State Diagrams	311
9.2.10.1. Sink Port Get Source Information State Diagram	311
9.2.10.2. Source Give Source Information State Diagram	311
9.2.11. Status State Diagrams	312
9.2.11.1. Get Status State Diagram	312
9.2.11.2. Give Status State Diagram	313
9.2.11.3. Sink Port Get Source PPS Status State Diagram	313
9.2.11.4. Source Give Source PPS Status State Diagram	314
9.2.12. Battery Capabilities State Diagrams	315
9.2.12.1. Get Battery Capabilities State Diagram	315
9.2.12.2. Give Battery Capabilities State Diagram	315
9.2.13. Battery Status State Diagrams	316
9.2.13.1. Get Battery Status State Diagram	316
9.2.13.2. Give Battery Status State Diagram	317
9.2.14. Manufacturer Information State Diagrams	317
9.2.14.1. Get Manufacturer Information State Diagram	317
9.2.14.2. Give Manufacturer Information State Diagram	318
9.2.15. Country Codes and Information State Diagrams	319
9.2.15.1. Get Country Codes State Diagram	319
9.2.15.2. Give Country Codes State Diagram	319
9.2.15.3. Get Country Information State Diagram	320
9.2.15.4. Give Country Information State Diagram	321
9.2.16. Revision State Diagrams	321
9.2.16.1. Get Revision State Diagram	321
9.2.16.2. Give Revision State Diagram	322
9.2.17. Enter_USB Message State Diagrams	323
9.2.17.1. DFP Enter_USB Message State Diagrams	323
9.2.17.2. UFP or Cable Plug Enter_USB Message State Diagrams	323
9.2.18. Security State Diagrams	324
9.2.18.1. Send Security Request State Diagram	324
9.2.18.2. Send Security Response State Diagram	325
9.2.18.3. Security Response Received State Diagram	325
9.2.19. Firmware Update State Diagrams	326
9.2.19.1. Send Firmware Update Request State Diagram	326
9.2.19.2. Send Firmware Update Response State Diagram	326
9.2.19.3. Firmware Update Response Received State Diagram	327
9.2.20. Dual-Role Port State Diagrams	328
9.2.20.1. DFP to UFP Data Role Swap State Diagram	328
9.2.20.2. UFP to DFP Data Role Swap State Diagram	330
9.2.20.3. Policy Engine in Source to Sink Power Role Swap State Diagram	333
9.2.20.4. Policy Engine in Sink to Source Power Role Swap State Diagram	336
9.2.20.5. Policy Engine in Source to Sink Fast Role Swap State Diagram	339
9.2.20.6. Policy Engine in Sink to Source Fast Role Swap State Diagram	342
9.2.20.7. Dual-Role (Source Port) Get Source Capabilities State Diagram	345
9.2.20.8. Dual-Role (Source Port) Give Sink Capabilities State Diagram	346
9.2.20.9. Dual-Role (Sink Port) Get Sink Capabilities State Diagram	346
9.2.20.10. Dual-Role (Sink Port) Give Source Capabilities State Diagram	347
9.2.20.11. Dual-Role (Source Port) Get Source Capabilities Extended State Diagram	348
9.2.20.12. Dual-Role (Sink Port) Give Source Capabilities Extended State Diagram	349
9.2.20.13. Dual-Role (Sink Port) Get Sink Capabilities Extended State Diagram	349
9.2.20.14. Dual-Role (Source Port) Give Sink Capabilities Extended State Diagram	350
9.2.20.15. Dual-Role (Source Port) Get Source Information State Diagram	351

9.2.20.16. Dual-Role (Sink Port) Give Source Information State Diagram	351
9.2.21. VCONN Swap State Diagram	352
9.2.21.1. PE_VCS_Send_Swap State	353
9.2.21.2. PE_VCS_Evaluate_Swap State	354
9.2.21.3. PE_VCS_Accept_Swap State	354
9.2.21.4. PE_VCS_Reject_Swap State	354
9.2.21.5. PE_VCS_Wait_for_VCONN State	355
9.2.21.6. PE_VCS_Turn_Off_VCONN State	355
9.2.21.7. PE_VCS_Turn_On_VCONN State	355
9.2.21.8. PE_VCS_Send_PS_Rdy State	355
9.2.21.9. PE_VCS_Force_VCONN State	355
9.2.22. Initiator Structured VDM State Diagrams	355
9.2.22.1. Initiator Structured VDM Discover Identity State Diagram	355
9.2.22.2. Initiator Structured VDM Discover SVIDs State Diagram	357
9.2.22.3. Initiator Structured VDM Discover Modes State Diagram	359
9.2.22.4. Initiator Structured VDM Attention State Diagram	359
9.2.23. Responder Structured VDM State Diagrams	361
9.2.23.1. Responder Structured VDM Discover Identity State Diagram	361
9.2.23.2. Responder Structured VDM Discover SVIDs State Diagram	362
9.2.23.3. Responder Structured VDM Discover Modes State Diagram	363
9.2.23.4. Receiving a Structured VDM Attention State Diagram	364
9.2.24. DFP Structured VDM State Diagrams	364
9.2.24.1. DFP Structured VDM Mode Entry State Diagram	364
9.2.24.2. DFP Structured VDM Mode Exit State Diagram	366
9.2.25. UFP Structured VDM State Diagrams	367
9.2.25.1. UFP Structured VDM Enter Mode State Diagram	367
9.2.25.2. UFP Structured VDM Exit Mode State Diagram	369
9.2.26. Cable Plug Specific State Diagrams	371
9.2.26.1. Cable Plug Cable Ready State Diagram	371
9.2.26.2. Soft/Hard/Cable Reset	372
9.2.26.3. EPR Mode State Diagrams	381
9.2.26.4. BIST State diagrams	388
9.2.26.5. USB Type-C Referenced States	391
10. Fast Role Swap (FRS)	392
10.1. Introduction	392
10.2. FRS Initialization	392
10.3. FRS Sequence	393
10.4. Initial Source/New Sink	396
10.4.1. Initial Sink/New Source	397
10.4.2. Constraints During FRS	397
10.5. FRS Parameters	398
A. Revision History	400
B. Contributors	405

List of Figures

1.1. USB PD Capable Device High Level Architecture	25
3.1. SPR Required Source Power Rule Illustration for Fixed PDOs	39
3.2. EPR Source Power Rule Illustration for Fixed PDOs	43
3.3. Valid EPR AVS Operating Region	45
3.4. Example of a DPS temperature profile	46
4.1. Placement of Source Bulk Capacitance	49
4.2. Application of vNew and vValid limits after tSrcReady or tSrcTransSmall or tSrcTransLarge	50
4.3. Transition Envelope for Positive Voltage Transitions	51
4.4. Transition Envelope for Negative Voltage Transitions	52
4.5. PPS/AVS Voltage Transitions	53
4.6. PPS Programmable Voltage and Current Limit	55
4.7. PPS Constant Power	56
4.8. Expected PPS Ripple Relative to an LSB	57
4.9. Allowed DNL Errors and Tolerance of Voltage and Current in AVS/PPS Mode	58
4.10. Source Peak Current Overload	60
4.11. Holdup Time Measurement	61
4.12. Placement of Sink Bulk Capacitance	62
4.13. Transition Diagram for a Power Role Swap	67
4.14. Source VBUS and VCONN Response to Hard Reset	68
4.15. Transition diagram for a Hard Reset	69
4.16. Transition Diagram for Large-step voltage increase (Fixed or AVS > vSmallStep)	70
4.17. Transition Diagram for Small-step voltage increase (AVS ≤ vSmallStep), current unchanged or in- creased	71
4.18. Transition Diagram for Small-step voltage increase (AVS ≤ vSmallStep), current decreased	72
4.19. Transition Diagram for voltage unchanged or decreased with current unchanged or increased	73
4.20. Transition Diagram for voltage unchanged or decreased with current decreased	74
4.21. Transition Diagram for PPS voltage increase (CV Mode)	75
4.22. Transition Diagram for PPS voltage decrease (CV Mode)	76
4.23. Transition Diagram for PPS current increase (CL Mode)	77
4.24. Transition Diagram for PPS current decrease (CL Mode)	77
4.25. Data Reset VCONN Power Cycle	78
5.1. Transmit Packet Format	88
5.2. Line Format for Hard Reset and Cable	91
5.3. Receive Packet Interpretation.	93
5.4. BMC Example	95
5.5. BMC Transmitter Block Diagram	95
5.6. BMC Receiver Block Diagram	96
5.7. BMC Encoded Start of Preamble	96
5.8. Transmitting or Receiving BMC Encoded Frame Terminated by Zero with High-to-Low Last Transition	97
5.9. Transmitting or Receiving BMC Encoded Frame Terminated by One with High-to-Low Last Transition	97
5.10. Transmitting or Receiving BMC Encoded Frame Terminated by Zero with Low-to-High Last Transition	97
5.11. Transmitting or Receiving BMC Encoded Frame Terminated by One with High-to-Low Last Transition	98
5.12. BMC Tx 'ONE' Mask.	99
5.13. BMC Tx 'ZERO' Mask.	99
5.14. BMC Rx 'ONE' Mask when Sourcing Power	102
5.15. BMC Rx 'ZERO' Mask when Sourcing Power	102
5.16. BMC Rx 'ONE' Mask when Sinking Power	103
5.17. BMC Rx 'ZERO' Mask when Sinking Power	103
5.18. BMC Rx 'ONE' Mask when Power Neutral	104
5.19. BMC Rx 'ZERO' Mask when Power Neutral	104
5.20. Transmitter Load Model for BMC Tx from a Source	106
5.21. Transmitter Load Model for BMC Tx from a Sink	106

5.22. Example Multi-Drop Configuration showing two DRPs and two Cable Transceivers	107
5.23. Example Multi-Drop Configuration showing a DFP and UFP with a VCONN Source	107
5.24. Transmitter diagram illustrating zDriver	108
5.25. Inter-Frame Gap Timings	109
6.1. Construction of Chunked Extended and Chunk Request Messages	115
6.2. Format of PD Packets for Different Message Types	116
6.3. Figure: EPR_Request Message	147
6.4. Figure: Vendor Defined Message	150
6.5. Discover Identity Response Format	154
6.6. Discover SVIDs Response Format	169
6.7. Example Discover Modes response for a given SVID with 3 Modes	170
6.8. Source Capabilities Message Format	191
6.9. Source Capabilities Message Format (with no EPR PDOs)	192
7.1. Sequence diagram of messaging core between protocol layers	204
7.2. Basic Message flow indicating possible errors	205
7.3. Illustration of process to enter EPR Mode.	219
8.1. Successful Enter Mode sequence	247
8.2. Unsuccessful Enter Mode sequence due to NAK	248
8.3. Exit Mode sequence	249
8.4. Attention Command Request/response sequence	250
8.5. Enter/Exit Mode Process	252
9.1. Outline of States	254
9.2. References to States	254
9.3. Chunking architecture Showing Message and Control Flow	256
9.4. Chunked Rx State Diagram	257
9.5. Chunked Tx State Diagram	260
9.6. Chunked Message Router State Diagram	263
9.7. Common Protocol Layer Message Transmission State Diagram	264
9.8. Source Protocol Layer Message Transmission State Diagram	268
9.9. Sink Protocol Layer Message Transmission State Diagram	270
9.10. Protocol Layer Message reception	271
9.11. Hard/Cable Reset	274
9.12. Outline of States	277
9.13. References to states	277
9.14. Example of State reference with conditions	278
9.15. Example of State reference with the same entry and exit	278
9.16. SenderResponseTimer Policy Engine State Diagram	279
9.17. Source Port State Diagram	281
9.18. Sink Port State Diagram	289
9.19. SOP Source Port Soft Reset and Protocol Error State Diagram	294
9.20. Sink Port Soft Reset and Protocol Error Diagram	296
9.21. DFP Data_Reset Message State Diagram	298
9.22. UFP Data_Reset Message State Diagram	300
9.23. Source Port Not Supported Message State Diagram	302
9.24. Sink Port Not Supported Message State Diagram	303
9.25. Source Port Source Alert State Diagram	305
9.26. Sink Port Source Alert State Diagram	306
9.27. Sink Port Sink Alert State Diagram	307
9.28. Source Port Sink Alert State Diagram	308
9.29. Sink Port Get Source Capabilities Extended State Diagram	309
9.30. Source Give Source Capabilities Extended State Diagram	309
9.31. Source Port Get Sink Capabilities Extended State Diagram	310
9.32. Sink Give Sink Capabilities Extended State Diagram	310
9.33. Sink Port Get Source Information State Diagram	311
9.34. Source Give Source Information State Diagram	312

9.35. Get Status State Diagram	312
9.36. Give Status State Diagram	313
9.37. Sink Port Get Source PPS Status State Diagram	314
9.38. Source Give Source PPS Status State Diagram	314
9.39. Get Battery Capabilities State Diagram	315
9.40. Give Battery Capabilities State Diagram	316
9.41. Get Battery Status State Diagram	316
9.42. Give Battery Status State Diagram	317
9.43. Get Manufacturer Information State Diagram	318
9.44. Give Manufacturer Information State Diagram	318
9.45. Get Country Codes State Diagram	319
9.46. Give Country Codes State Diagram	320
9.47. Get Country Information State Diagram	320
9.48. Give Country Information State Diagram	321
9.49. Get Revision State Diagram	322
9.50. Give Revision State Diagram	322
9.51. DFP Enter_USB Message State Diagram	323
9.52. UFP Enter_USB Message State Diagram	324
9.53. Send security Request State Diagram	324
9.54. Send security response State Diagram	325
9.55. Security response received State Diagram	325
9.56. Send firmware update Request State Diagram	326
9.57. Send firmware update response State Diagram	327
9.58. Firmware update response received State Diagram	327
9.59. DFP to UFP Data Role Swap State Diagram	328
9.60. UFP to DFP Data Role Swap State Diagram	331
9.61. Dual-Role Port in Source to Sink Power Role Swap State Diagram	334
9.62. Dual-role Port in Sink to Source Power Role Swap State Diagram	337
9.63. Dual-Role Port in Source to Sink Power Role Swap State Diagram	340
9.64. Dual-role Port in Sink to Source Fast Role Swap State Diagram	343
9.65. Dual-Role (Source) Get Source Capabilities diagram	345
9.66. Dual-Role (Source) Give Sink Capabilities diagram	346
9.67. Dual-Role (Sink) Get Sink Capabilities State Diagram	347
9.68. Dual-Role (Sink) Give Source Capabilities State Diagram	348
9.69. Dual-Role (Source) Get Source Capabilities Extended State Diagram	348
9.70. Dual-Role (Sink) Give Source Capabilities Extended diagram	349
9.71. Dual-Role (Sink) Get Sink Capabilities Extended State Diagram	350
9.72. Dual-Role (Source) Give Sink Capabilities Extended diagram	350
9.73. Dual-Role (Source) Get Source Information State Diagram	351
9.74. Dual-Role (Source) Give Source Information diagram	352
9.75. VCONN Swap State Diagram	353
9.76. Initiator to Port VDM Discover Identity State Diagram	356
9.77. Initiator VDM Discover SVIDs State Diagram	358
9.78. Initiator VDM Discover Modes State Diagram	359
9.79. Initiator VDM Attention State Diagram	360
9.80. Responder Structured VDM Discover Identity State Diagram	361
9.81. Responder Structured VDM Discover SVIDs State Diagram	362
9.82. Responder Structured VDM Discover Modes State Diagram	363
9.83. Receiving a Structured VDM Attention State Diagram	364
9.84. DFP VDM Mode Entry State Diagram	365
9.85. DFP VDM Mode Exit State Diagram	366
9.86. UFP Structured VDM Enter Mode State Diagram	368
9.87. UFP Structured VDM Exit Mode State Diagram	370
9.88. Cable Ready State Diagram	371
9.89. Cable Plug Soft Reset State Diagram	372

9.90. Cable Plug Hard Reset State Diagram	373
9.91. DFP/VCONN Source Soft Reset or Cable Reset of a Cable Plug or VPD State Diagram	374
9.92. UFP/VCONN Source Soft Reset of a Cable Plug or VPD State Diagram	375
9.93. Source Startup Structured VDM Discover Identity State Diagram	377
9.94. Cable Plug Structured VDM Enter Mode State Diagram	379
9.95. Cable Plug Structured VDM Exit Mode State Diagram	380
9.96. Source EPR Mode Entry State Diagram	382
9.97. Sink EPR Mode Entry State Diagram	384
9.98. Source EPR Mode Exit State Diagram	386
9.99. Sink EPR Mode Exit State Diagram	387
9.100. BIST Carrier Mode State Diagram	388
9.101. BIST Test Data Mode State Diagram	389
9.102. BIST Shared Capacity Test Mode State Diagram	390
10.1. Power Flow Before and After an FRS	392
10.2. FRS Signaling and Receiving Activity Diagrams	394
10.3. FRS Sequence with VBUS Starting at Voltage > vSafe5V (long discharge)	395
10.4. FRS Sequence with VBUS starting at vSafe5V (short or no discharge)	395

List of Tables

2.1. Keywords	26
2.2. Document References	27
2.3. Terms and Abbreviations	27
3.1. SPR Source Voltages and Minimum Currents based on Port Maximum PDP	39
3.2. SPR Voltages and Minimum Currents when Port Present PDP < Port Maximum PDP	40
3.3. Examples of Required SPR Source Capabilities when Port Present PDP < Port Maximum PDP	40
3.4. SPR PPS APDOs based on the Port Maximum PDP	42
3.5. EPR Source Capabilities based on the Port Maximum PDP	43
3.6. EPR Source Capabilities when Port Present PDP < Port Maximum PDP	44
3.7. Examples of EPR Source Capabilities when Port Present PDP <Port Maximum PDP	44
4.1. Universal Aliases for PDO or APDO Variables	48
4.2. SPR Programmable Power Supply (PPS) Voltage Ranges	53
4.3. SPR Adjustable Voltage Supply (AVS) Voltage Ranges	53
4.4. EPR Adjustable Voltage Supply (AVS) Voltage Ranges	53
4.5. PPS Status	54
4.6. Source Electrical Parameters	79
4.7. Sink Electrical Parameters	83
4.8. Common Source/Sink Electrical Parameters	84
5.1. Rp values used for Collision Avoidance	86
5.2. K-codes and 4b5b symbol encoding	89
5.3. K-codes for SOP* Sequences	90
5.4. K-codes for Resets	91
5.5. Validation of Ordered Sets	94
5.6. BMC Tx Mask Definition, X Values	100
5.7. BMC Tx Mask Definition, Y Values	100
5.8. BMC Rx Mask Definition	104
5.9. BMC Common Requirements	110
5.10. BMC Transmitter Parameters	110
5.11. BMC Receiver Parameters	111
6.1. Revision Interoperability during an Explicit Contract	114
6.2. Message Header	117
6.3. Number of Data Objects	118
6.4. Control Message Types	118
6.5. Data Message Types	125
6.6. PDO Format	126
6.7. Augmented PDO Format	127
6.8. Fixed 5V Source PDO Format	127
6.9. Fixed 5V Sink PDO Format	128
6.10. Fixed PDO (>5V)	129
6.11. Battery PDO	129
6.12. Variable PDO	130
6.13. SPR Programmable Power Supply Source APDO	131
6.14. SPR Programmable Power Supply Sink APDO	132
6.15. SPR Adjustable Voltage Supply APDO	133
6.16. EPR Adjustable Voltage Supply Source APDO	134
6.17. EPR Adjustable Voltage Supply Sink APDO	135
6.18. PDO Peak Current	136
6.19. Fixed and Variable RDO	137
6.20. Battery RDO	137
6.21. Programmable Power Supply RDO	138
6.22. Adjustable Voltage Supply RDO	138
6.23. BIST Modes	140

6.24. Battery Status Data Object (BSDO)	142
6.25. Alert Data Object (ADO)	142
6.26. Country Code Data Object (CCDO)	145
6.27. Enter_USB Data Object (EUDO)	145
6.28. EPR Mode Data Object (EPRMDO)	148
6.29. Source_Info Data Object 1 (SIDO1)	149
6.30. Source_Info Data Object 2 (SIDO2)	149
6.31. Revision Message Data Object (RMDO)	150
6.32. Unstructured VDM Header	151
6.33. Structured VDM Header	151
6.34. ID Header VDO	154
6.35. Product Types (UFP)	156
6.36. Product Types (Cable Plug/VPD)	156
6.37. Product Type (DFP)	157
6.38. Cert Stat VDO	157
6.39. Product VDO	157
6.40. UFP VDO	158
6.41. DFP VDO	161
6.42. Passive Cable VDO	161
6.43. Active Cable VDO1	164
6.44. Active Cable VDO2	166
6.45. VCONN Powered USB Device VDO	168
6.46. Discover SVIDs Responder VDO	169
6.47. Extended Message Types	171
6.48. Extended Message Header Fields	172
6.49. Chunked	173
6.50. Source Capabilities Extended Data Block (SCEDB)	173
6.51. SOP Status Data Block (SDB)	176
6.52. SOP"/SOP" Status Message Byte Definitions	181
6.53. Get Battery Cap Data Block (GBCDB)	181
6.54. Get Battery Status Data Block (GBSDB)	182
6.55. Battery Capability Data Block (BCDB)	182
6.56. Get_Manufacturer_Info Message	183
6.57. Manufacturer_Info Message	183
6.58. PPS Status Data Block (PPSSDB)	185
6.59. Country Info Data Block (CIDB)	186
6.60. Country Codes Data Block (CCDB)	186
6.61. Sink Capabilities Extended Data Block (SKEDB)	187
6.62. Extended Control Data Block (ECDB)	190
6.63. Extended Control Message Types	190
6.64. Vendor_Defined_Extended Message+	192
6.65. Value Parameters	193
7.1. Response to an incoming Message (except VDM)	195
7.2. Response to an incoming VDM	195
7.3. Message Discarding	198
7.4. Port Sequence Applicability	199
7.5. Cable Plug Sequence Applicability	202
7.6. Port Message Applicability	203
7.7. Cable Plug Message Applicability	204
7.8. Potential Issues in the Basic Message Flow	206
7.9. Timer Values	236
7.10. Timer Specifications	237
7.11. Counter Parameters	240
7.12. Counters	240
8.1. Commands and Responses	243

10.1. FRS Timers	398
10.2. FRS Parameters	398
A.1. Revision History	400

Chapter 1. Introduction

USB Power Delivery (PD) is a power transfer standard that allows USB cables and connectors to deliver higher power levels (up to 240W) for a wide range of devices, including laptops, tablets, smartphones, and peripherals. In addition to power delivery, USB PD supports data protocol negotiations over USB-C, thus making it a versatile solution for both power and data configurations.

Key features of USB PD:

- Negotiable power output – Supports negotiating power up to 240W (48V/5A) to power high-demand devices like laptops, monitors, and gaming consoles.
- Renegotiable power – Powered devices and power sources renegotiate power levels to deliver optimal power based on a particular Device's immediate requirements.
- Bidirectional power – Devices can both provide and consume power, enabling features like Host charging (e.g., powering a laptop from an external monitor).
- Unified connector – Uses the USB-C connector that supports both power and data transfer through a single cable.
- Data protocol Negotiation – USB PD allows a Device to Connect using the default USB2/USB3 data connection or switch to alternate data modes, enabling USB-C to support additional protocols such as DisplayPort, USB4, and Thunderbolt 3 for video output and high-speed data transfer.

In summary, USB PD maximizes the potential of USB-C by providing both flexible power delivery and data protocol negotiations, thus making it a universal solution for modern devices.

1.1. Cable and Connectors

The USB Power Delivery specification assumes the use of certified USB cables and associated detection mechanisms as defined in [USB-C].

1.2. Operational Overview

A USB PD connection is based on the underlying USB-C standard and roles as defined in [USB-C]. The Source Port provides power, and the Sink Port consumes power. Each connection has one Source and one Sink, which are established during the initial connection.

At the point of Attachment, the Source Port functions as both the Downstream Facing Port (DFP) and the VCONN Source, while the Sink Port functions as the Upstream Facing Port (UFP).

Power roles (Source/Sink), data roles (DFP/UFP), and the VCONN Source role can be swapped independently during a connection. These role swaps depend on whether the ports support Dual-Role Power (DRP) or Dual-Role Data (DRD) Capabilities. Ports with dual-role Capabilities can switch between providing and consuming power or acting as the data Host or Device, depending on the Negotiation.

1.3. USB PD Roles

1.3.1. Source Port

The Source Port is typically associated with devices like chargers, power banks, notebooks, and docks that power other devices and provide the power in a USB PD connection. The Source defaults to being the DFP at initial connection. It supplies power to a Connected Sink Port and is responsible for initiating power negotiations to determine the appropriate voltage and current levels needed by the connection.

1.3.2. Sink Port

The Sink Port is the power-consuming Port in a USB PD connection and is typically associated with devices like smartphones, tablets, notebooks, and peripherals that rely on an external power Source to charge. The Sink defaults to being the UFP at initial connection. It draws power from a Connected Source Port and participates in power Negotiation to ensure it receives the appropriate voltage and current levels.

1.3.3. Downstream Facing Port (DFP)

The Downstream Facing Port (DFP) is the Port in a USB-C connection associated with the USB Host role. It is the Initiator in Data Role negotiations and handles most downstream data communications. See [USB2] and [USB3] for more information about Downstream ports.

1.3.4. Upstream Facing Port (UFP)

The Upstream Facing Port (UFP) is the Port in a USB-C connection that is associated with the USB Peripheral role. It is the Responder in Data Role Negotiation and handles most upstream data communication. See [USB2] and [USB3] for more information about Upstream ports.

1.3.5. Dual-Role Power (DRP) Ports

Dual-Role Power Ports can operate as either a Source or a Sink and swap between the two Power Roles as needed.

1.3.6. Dual-Role Data (DRD) Ports

Dual-Role Data Ports have the ability to operate as either a DFP or a UFP and to swap between the two Data Roles. Products can be Dual-Role Data Ports without being Dual-Role Power Ports. This means that they can switch logically between DFP and UFP Data Roles, even if they are Source-only or Sink-only for power.

1.3.7. VCONN Source

The VCONN Source is the Port that provides power to Cable Plugs. The VCONN Source is typically the Downstream Facing Port (DFP) and supplies power to enable Active Cable circuitry and support cable identification and management in USB PD connections. To communicate with Cable Plugs, a Port must be the VCONN Source. For more information about a USB Type-C Source Port's requirements regarding VCONN, see [USB-C].

1.3.8. Cable Plugs

In the context of USB PD, Cable Plug refers to embedded USB PD communication-capable circuitry within a cable, plug, or accessory that is typically associated with a single plug end of a cable or a captive plug on a Device. This circuitry consists of the embedded electronic marker (e-marker) chip and/or any active circuitry within the Cable.

Cable Plugs are powered when VCONN is present but are generally not aware of the status of the Contract between the two Connected ports. They do not initiate communication and only respond to messages that are addressed to them.

The Source or Sink communicates with the Cable Plug circuitry to retrieve information about the Attached cable or accessory, such as current-carrying capability or supported data rates, or to direct the plug to enter a specific data Mode.

1.4. Power Delivery Operational Contracts

A PD Source and Sink will be in one of three Contracts:

- **Default Contract** – A non-Negotiated power Contract established at the initial connection, as defined in [USB-C], with a default voltage of 5V. The Sink might draw up to the advertised USB-C current. The Source remains in the Default Contract until the Sink disconnects or both ports Negotiate and establish an Explicit Contract.
- **Explicit Contract** – The State of the Source and Sink after any PD Power Negotiation has taken place. This is the normal operational State for PD and must be entered before other USB PD messaging can occur. A Source offers power contracts to a Sink using USB PD messaging, and the Sink requests a Contract, which is then accepted by the Source. Data connections and modes other than USB2 and USB3 might be established in an Explicit Contract.
- **Implicit Contract** – A transitory power Contract that follows a Power Role Swap or Fast Role Swap. The available voltage, similar to a Default Contract, is 5V, with current defined by the advertised USB-C current. The Source in an Implicit Contract will immediately Negotiate with the Sink to establish an Explicit Contract. All data connections and modes are maintained during an Implicit Contract.

1.5. Source and Sink Operation in USB PD

In a USB PD connection, the Source and Sink engage in a series of Message exchanges to Negotiate power contracts, data roles, and Mode implementation (such as Alternate Mode or USB4 Mode). This process ensures both ports operate with compatible power levels and functional Capabilities.

1.5.1. General Source-to-Sink Initial Interaction

1. **Initial Connection (Attach)** – Defined by [USB-C].
 - The Source asserts R_p (pull-up resistor or pull-up current) on the CC line, and the Sink asserts R_d (pull-down resistor) to establish the connection.
 - The Source provides a Default Contract at the advertised R_p current at 5V upon connection.
2. **Power Discovery.**
 - The Source advertises its available power options through its Source Capabilities. These messages define voltage (e.g., 5V, 9V, 15V, 20V) and current levels the Source can supply.
3. **Power Negotiation and Explicit Contract.** The Sink requests the preferred power option based on its requirements.
 - The Source accepts the Request to establish the Explicit Contract.
 - The Source then indicates that it is supplying the agreed power level.
 - Once the Explicit Contract is established, both ports enter a stable power State and might establish a Data Role change or Mode Entry.
4. **Data Role and Mode Entry.**
 - When a role change is desired, the Source and Sink exchange Data Role messages to determine the DFP (Downstream Facing Port) and UFP (Upstream Facing Port) roles.
 - The Source might Request the Sink to send capability information and to confirm data Mode compatibility.
 - The Source might initiate USB4 Mode Entry, if supported.
 - The Source can initiate Alternate Mode Entry by asking the Sink to Advertise its supported modes.
 - If a compatible Alternate Mode is found (e.g., DisplayPort), the Source requests to enter the Mode.
5. **Ongoing Communication and Role Swaps**

- During the connection, the Source and Sink can continue to exchange Control and Data messages to renegotiate power levels, switch roles, etc.
- Both ports can perform a Power Role Swap, where the Source becomes the Sink, and vice versa.
- A Data Role Swap can also occur, changing which Port acts as the DFP (Host) or UFP (Peripheral) for data transfer.

1.6. Common PD Device Types

USB PD devices fall into a few general categories, such as:

- Dedicated power sources – wall chargers (bricks), power banks, etc.
- Host systems – computers (desktop, portable), mobile phones, tablets, etc.
- Peripherals – mice, keyboards, displays, hard drives, cameras, speakers, headsets, etc.
- Data routers – hubs, docks.

Some devices can be combinations such as a display/dock or cameras that can function as a Host or Device. With USB PD, these can also be power Sources or power Sinks, or both, depending on what they are Attached to. For example, a Host computer can be a power Sink and be charged when Connected to a wall Charger, but it might then be a power Source to a downstream Peripheral such as a hard drive.

1.7. USB-PD Architectural Overview

This section describes the logical architecture of the USB Power Delivery (USB PD) specification. The architecture overview presented here is conceptual and not intended to prescribe a specific implementation. Rather, it provides a high-level framework that is referenced throughout this specification.

USB PD defines a bus protocol specifying voltage levels, current limits, Signaling, timing, and other interface parameters that are Negotiated between two ports. It does not define how these requirements must be implemented in a particular Device. Implementation details are left to the product designer and are outside the scope of this specification.

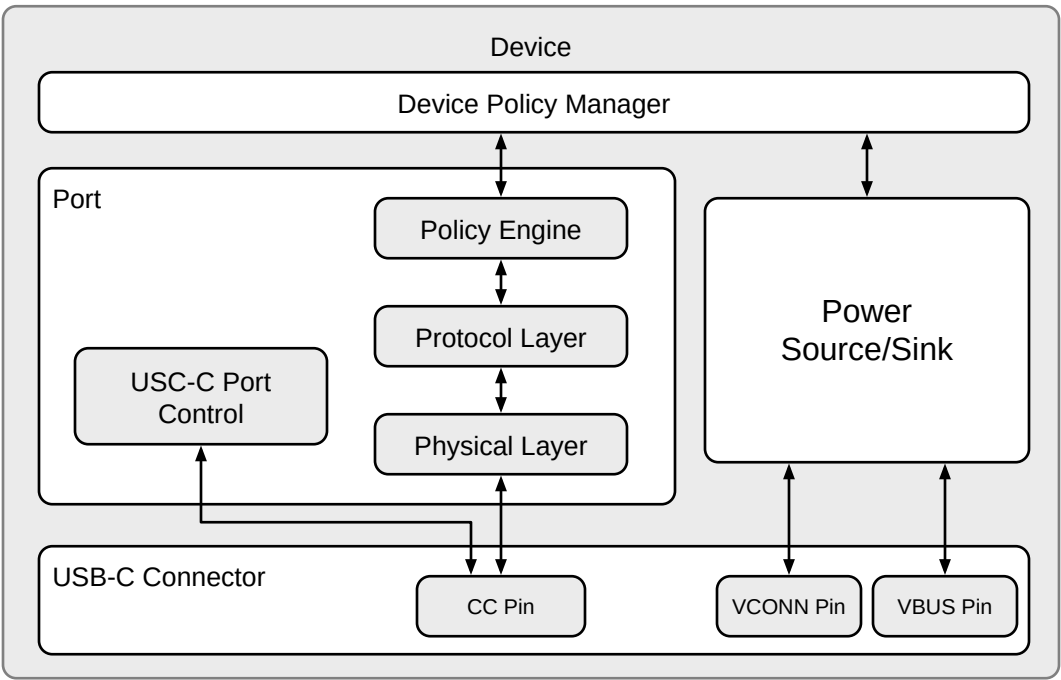
At a high level, USB PD is structured as a hierarchy of control layers, where each layer communicates with its adjacent layers. From lowest to highest, these layers are:

- **Device Policy Manager:** Coordinates USB PD behavior across the Device by managing one or more ports based on the Device's overall power and data Policy.
- **Policy Engine:** Enforces local power and data Policy for an individual Port.
- **Protocol Layer:** Constructs and interprets USB PD messages exchanged between ports.
- **Physical Layer:** Handles bit-level transmission and reception on the physical connector and performs error detection on messages using a CRC mechanism.
- **USB-C Port Control:** Provides mechanisms to execute USB-C State machines and informing the other layers of Attach/Detach events, plug orientation, and set and detect Type-C current advertisements.

Each Port includes its own USB-C Port Control, Physical Layer, Protocol Layer, and Policy Engine. The Device Policy Manager is a single entity that manages all ports within the Device.

USB PD operates over a USB-C connector; as such, the State machines defined in [USB-C] reside within or closely interact with the USB PD control layers. Similarly, voltage and current control for power delivery interface tightly with these layers. [Figure 1.1](#) illustrates these architectural elements.

Figure 1.1. USB PD Capable Device High Level Architecture



Chapter 2. Glossary and Info

2.1. Conventions

2.1.1. Precedence

If there is a conflict between text, figures, and tables, the precedence is tables, figures, and then text.

2.1.2. Keywords

The following keywords differentiate between the various levels of a requirement and the applicable options.

Table 2.1. Keywords

Keyword	Definition
Deprecated	Indicates a feature, supported in previous releases of the specification, which is no longer supported.
Discard / Discards / Discarded	Indicates that a Packet, when received, is thrown away by either the PHY Layer or the Protocol Layer.
Dynamic	Indicates a field or behavior that is not static and is allowed to change.
Ignore / Ignores / Ignored	Indicates a Message or Message field which, when received, results in no special action by the receiver. A Message with an Ignored field is processed normally except for any actions relating to the Ignored field.
Informative	Content provided to improve comprehension; not an implementation requirement.
Invalid	<p>When used in relation to a Packet, indicates that the Packet's usage or fields fall outside of the defined specification usage.</p> <p>When used in relation to an Explicit Contract, indicates that a previously established Explicit Contract can no longer be maintained by the Source.</p> <p>When used in relation to individual K-codes or K-code sequences, indicates that the received Signaling falls outside of the defined specification.</p> <p>When used in relation to a Message Field, indicates that a disallowed value was received.</p> <p>Invalid usages, fields, K-codes, and values might change in future revisions of this specification.</p>
May / May Not	Indicates a choice with no implied preference.
N/A	Indicates that a field or value is not applicable and has no defined value and is not to be checked or used by the recipient.
Optional	Describes features not mandated by this specification. If a PDUSB Device implements an Optional feature, then all Required parts of the feature Shall be implemented.
Reserved	Indicates bits, bytes, words, fields, and code values that might be used by future revisions of this specification. Reserved bits, bytes, words, or fields are set to zero by the sender and are ignored by the receiver.
Required	Describes a feature or behavior mandated by this specification; something that Shall be implemented.
Shall	Equivalent keywords indicating a mandatory requirement. Designers are mandated to implement all such requirements to ensure interoperability with other compliant devices.
Shall Not	Inverse of Shall indicating non-compliant operation.
Should	Indicates a flexibility of choice with a preferred alternative; equivalent to the phrase "it is recommended that...".
Should Not	Inverse of Should; equivalent to the phrase "it is recommended that implementations do not...".
Static	Indicates a field or behavior that never changes.
Valid	Inverse of Invalid indicating either a Packet or Signaling that fall within the defined specification or an Explicit Contract that can be maintained by the Source.

2.1.3. Numbering

Numbers that are immediately followed by a lowercase “b” (e.g., 01b) are binary values. Numbers that are immediately followed by an uppercase “B” are byte values. Numbers that are immediately followed by a lowercase “h” (e.g., 3Ah) or are preceded by “0x” (e.g., 0xFF00) are hexadecimal values. Numbers that do not contain any letters are decimal values.

2.2. Related Documents

Document references listed in [Table 2.2](#) are inclusive of all approved and published ECNs and Errata.

Table 2.2. Document References

Bookmark Reference	Title
[DPTC]	DisplayPort Alt Mode on USB Type-C Standard www.vesa.org .
[IEC 60950-1]	IEC 60950-1 Information technology equipment – Safety – Part 1: General requirements: Amendment 1:2009, Amendment 2:2013. www.iec.ch .
[IEC 60958-1]	IEC 60958-1 Digital Audio Interface Part:1 General. www.iec.ch .
[IEC 62368-1]	IEC 62368-1 Audio/Video, information, and communication technology equipment – Part 1: Safety requirements. www.iec.ch .
[ISO 3166]	ISO 3166 international Standard for country codes and codes for their subdivisions. www.iso.org/iso/home/standards/country_codes.htm .
[TBT3]	See [USB4] for Thunderbolt 3 Device operation www.usb.org/documents .
[UCSI]	USB Type-C Connector System Software Interface (UCSI) Specification www.usb.org/documents .
[USB2]	Universal Serial Bus 2.0 Specification, www.usb.org/documents .
[USB3]	Universal Serial Bus 3.2 Specification www.usb.org/documents .
[USB-C]	Universal Serial Bus Type-C Cable and Connector Specification, www.usb.org/documents .
[USB4]	Universal Serial Bus 4 Specification (USB4), www.usb.org/documents .
[USB BC]	Universal Serial Bus Battery Charging Specification plus Errata (referred to in this document as the Battery Charging specification), www.usb.org/documents .
[PD]	Universal Serial Bus Power Deliver Specification, Revision 3.2 Version 1.2. www.usb.org/documents .
[PD2]	Universal Serial Bus Power Deliver Specification, Revision 2.0 Version 1.3. www.usb.org/documents .
[PDSSR]	PDUSB Status Reporting and Device States, Revision 1.1. www.usb.org/documents .
[PDFU]	Universal Serial Bus Power Delivery Firmware Update Specification, Revision 1.0. www.usb.org/documents .
[USBC Auth]	Universal Serial Bus Type-C Authentication Specification, www.usb.org/documents .
[USBPD Compliance]	USB Power Delivery Compliance Test Specification, www.usb.org/documents .
[PDUSB]	PDUSB Status Reporting and Device States, Revision 1.1. www.usb.org/documents .

2.3. Terms and Abbreviations

This section defines terms used throughout this document. For additional terms that pertain to the Universal Serial Bus, see [Table 2.3](#) in the [USB2], [USB3], [USB-C] and [USB BC] specifications.

Table 2.3. Terms and Abbreviations

Term	Description
AC Supply / AC Supplied	Refers to the main AC power Source typically provided to the wall and often referred to as “mains” or “wall mains”.

Term	Description
Active Cable	A cable with a USB Type-C plug on each end that incorporates data bus signal conditioning circuits. The cable supports the Structured VDM Discover Identity Command to expose its characteristics in addition to other Structured VDM Commands (Electronically Marked Cable see [USB-C]).
Active Cable VDO	VDO defining the Capabilities of an Active Cable.
Active Mode	A Mode which has been through the Mode Entry process but not the Mode Exit process.
Adjustable Voltage Supply (AVS)	A power supply whose output voltage can be adjusted to an operating voltage within its Advertised range inside an APDO.
Advertise	An offer for power made by a Source in the Source_Capabilities/EPR_Source_Capabilities Message (e.g., a PDO or APDO).
Alternate Mode	Operation defined by a Vendor or Standard's organization, which is associated with a SVID. The definition of Alternate Modes is outside the scope of USB-IF specifications. Entry to and exit from the Alternate Mode uses the Mode Entry and Mode Exit processes.
Alternate Mode Adapter (AMA)	A Device which supports Alternate Modes.
Alternate Mode Controller (AMC)	A Host that supports connection to AMAs.
Assured Capacity Port	Assured Capacity Port – As defined in [USB-C]. Assured Capacity Ports can be either Managed Capability Ports or Guaranteed Capability Ports.
Atomic Message Sequence (AMS)	A fixed sequence of Messages that must complete before other messages are allowed to be sent.
Attach / Attached / Attachment	Mechanical joining of the Port Pair by a connector.
Augmented Power Data Object (APDO)	Data Object used to expose a Source Port's or Sink Port's power Capabilities as part of a Source_Capabilities/EPR_Source_Capabilities or Sink_Capabilities/EPR_Sink_Capabilities Message respectively. A PPS Data Object, SPR AVS Data Object and EPR AVS Data Object are defined.
Battery	A power storage Device residing behind a Port that can either be a Source or Sink of power.
Battery Slot	A physical location where a Hot Swappable Battery can be installed. A Battery Slot might or might not have a Hot Swappable Battery present in a Battery Slot at any given time.
Battery Supply	A power supply that directly applies the output of a Battery to VBUS. This is exposed by the Battery Supply PDO.
Bi-phase Mark Coding (BMC)	Modification of Manchester coding where each zero has one transition and a one has two transitions (see [IEC 60958-1]).
Built-In Self-Test (BIST)	Power Delivery testing mechanism for the PHY Layer.
BIST Data Object (BDO)	Data Object used by BIST Messages.
BIST Mode	A BIST receiver or transmitter test Mode enabled by a BIST Message.
BIST Carrier Mode	A BIST Mode in which the PHY Layer sends out a BMC encoded continuous string of alternating "1"s and "0"s.
BIST Test Data Mode	A BIST Mode in which the PHY Layer sends out a GoodCRC Message and then enters a test Mode where it sends no further Messages, except GoodCRC Messages, in response to received Messages.
BIST Shared Capacity Test Mode	A BIST Mode applicable only to a Shared Capacity Group of Ports where the maximum Source Capabilities are always offered on every Port, regardless of the availability of shared power i.e., all shared power management is disabled.
Cable Capabilities	Capabilities offered by a Cable Plug.
Cable Discovered	USB Power Delivery Ports that have exchanged a Message and a GoodCRC Message response with a Cable Plug or a VPD using the USB Power Delivery protocol so that both the Port and the Cable Plug know that each is PD Capable and which Revision they each support.
Cable Discovery	See Cable Discovered.
Cable Plug	Term used to describe a PD Capable element in a Multi-Drop system addressed by SOP' Packets/ SOP" Packets. Logically the Cable Plug is associated with a USB Type-C plug at one end of the cable. In a practical implementation, the electronics might reside anywhere in the cable.
Cable Reset	This is initiated by Cable Reset Signaling from the DFP. It restores the Cable Plugs to their default, power up condition and resets the PD communications engine in the cable to its default State. It does not reset the Port Partners.
Cable VDO	VDO returned by the Cable Plug containing Cable Capabilities.
Capabilities	Features supported by a product. These can include, for example, power levels supplied/ needed, cable type, Battery support or [USB4] support.

Term	Description
Capabilities Mismatch	Indication from the Sink that the Source's Advertised Capabilities don't match the Sink's needs.
Configuration Channel (CC)	Single wire used by the BMC PHY Layer Signaling Scheme (see [USB-C]).
Cert Stat VDO	The Cert Stat VDO contains the XID assigned by USB-IF to the product before certification in binary format.
Charge-Through Accessory	A USB accessory that is designed to allow a Source to be Connected through the accessory to charge a system to which it is Attached. Most common use is to allow a single Port USB Host to support a USB Device while being charged through the accessory from an external Source.
Chunk	A 26 byte or less portion of a Data Block. Data Blocks can be sent either as a single Message or as a series of Chunks.
Chunked Extended Message	Extended Message which has been broken up into Chunks.
Chunking	The process of breaking up a Data Block larger than 26 bytes into two or more Chunks.
Chunking Layer	Part of the Protocol Layer responsible for Chunking.
Cold Socket	A Port that does not apply power on VBUS until a Sink is Attached.
Collision Avoidance	Mechanisms to prevent simultaneous communication by the Source, Sink and Cable Plug on CC.
Command	Request and response pair defined as part of a Structured Vendor Defined Message.
Connected / Connect	USB Power Delivery ports that have exchanged a Message and a GoodCRC Message response using the USB Power Delivery protocol so that both Port Partners know that each is PD Capable.
Constant Voltage (CV)	A Constant Voltage feature of a PPS Source. The PPS Source output voltage remains constant as the load changes up to its Current Limit.
Continuous BIST Mode	The BIST Mode where the Port or Cable Plug being tested sends a continuous stream of test data.
Contract	An agreement on both power level and direction between a Port Pair. A Contract could be explicitly Negotiated between the Port Pair or could be an implicit power level defined by the current State. While operating in Power Delivery Mode there will always be either an Explicit Contract or Implicit Contract in place. The Contract can only be altered in the case of a Negotiation/Re-Negotiation, Power Role Swap, Fast Role Swap, Hard Reset, Error Recovery or failure of the Source.
Control Message	A Control Message is defined as a Message with the Number of Data Objects field in the Message Header is set to zero. The Control Message consists only of a Message Header and a CRC.
Cyclic Redundancy Check (CRC)	An error-detecting code used to determine if a block of data has been corrupted.
Current Limit (CL)	A current limiting feature of a PPS Source. When a Sink operating in PPS Mode attempts to draw more current from the Source than the requested Current Limit value, the Source reduces its output voltage so the current it supplies remains at or below the requested value. Note: Current Limit is not supported by SPR AVS and EPR AVS Sources.
Data Block	An Extended Message Payload data unit. This is distinct from a Data Object used by a Data Message which is always a 32-bit object.
Data Message	A Data Message consists of a Message Header followed by one or more Data Objects. Data Messages are easily identifiable because the Number of Data Objects field in the Message Header is always a non-zero value.
Data Object	A Data Message Payload data unit. This 32-bit object contains information specific to different types of Data Message. For example Power, Request, BIST, and Vendor Data Objects are defined.
Data Reset	Process which resets USB Communication.
Data Role	A Port Partner will be in one of two Data Roles; either DFP (USB Host) or UFP (USB Device).
Data Role Swap	Process of exchanging the Data Roles between Port Partners.
Dead Battery	A Device has a Dead Battery when the Battery in a Device is unable to power its functions.
Default Contract	An agreement on current at 5V is reached between a Port Pair based on Type-C current [USB-C].
Detach / Detached	Mechanical unjoining of the Port Pair by removal of the cable.
Device	When in upper cased (Device), refers to a USB Device (Peripheral or Hub). When lower cased (device), it refers to any USB product, either USB Device or USB Host.
Device Policy	Policy applied across multiple Ports in a Source or Sink.
Device Policy Manager (DPM)	Module running in a Source or Sink that applies Device Policy to each Port in the Device, as Local Policy, via the Policy Engine.
DFP VDO	VDO returned by the DFP containing Capabilities.

Term	Description
Differential Non-Linearity (DNL)	The difference between an ideal LSB step, and the real observable LSB step. A DNL of 0 indicates that the step is ideal.
Discovery Process	Command sequence using Structured Vendor Defined Messages resulting in identification of the Port Partner and Cable Plug, and their supported SVIDs and Alternate Modes.
Downstream Facing Port (DFP)	Indicates the Port's position in the USB topology which typically corresponds to a USB Host root Port or Hub downstream Port as defined in [USB-C]. At connection, the Port defaults to operation as the Source and as a USB Host (when USB Communication is supported).
Dual-Role Data (DRD)	Capability of operating as either a DFP or UFP.
Dual-Role Power (DRP)	Capability of operating as either a Source or Sink.
Dynamic Power Supply (DPS)	A Source that offers a guaranteed power and offers a maximum power when conditions are right. Conditions may include low enough internal temperature, ambient temperature, etc. The maximum power capability is not guaranteed and may be revoked by the Source.
End of Packet (EOP)	K-code marker used to delineate the end of a Packet.
Extended Power Range (EPR)	Extends the power range from a maximum of 100W (SPR) to a maximum of 240W (EPR). When operating in the EPR Mode, only EPR specific Messages (the EPR_Source_Capabilities Message and the EPR_Request Message) are used to Negotiate Explicit Contracts.
EPR AVS	A power supply operating in EPR Mode whose output voltage can be adjusted to an operating voltage within its Advertised range through an APDO.
EPR AVS Mode	A EPR Source, currently operating in an EPR AVS Contract, is said to be operating in EPR AVS Mode.
EPR Cable	A cable which is rated to operate in both SPR Mode and EPR Mode.
EPR Capabilities	The EPR Capabilities Messages (EPR_Source_Capabilities and EPR_Sink_Capabilities) are Extended Messages with the first seven positions filled with the same SPR PDOs and APDOs returned by the SPR Capabilities Messages (Source_Capabilities and Sink_Capabilities) followed by the EPR PDOs and APDOs starting in the eighth position.
EPR Capable	A product which has the ability to operate in EPR Mode.
EPR Mode	<p>A Power Delivery Mode of operation where maximum allowable voltage is 48V. The Sink complies to the requirements of [IEC 62368-1] for operation with a PS3 Source. The Source complies to the requirements of [IEC 62368-1] for operation with a PS3 Sink. The cable complies with [IEC 62368-1]. Entry into the EPR Mode requires that an EPR Source is Attached to an EPR Sink with an EPR Cable. The EPR Source will only enter the EPR Mode when requested to do so by the Sink and it has determined it is Attached to an EPR Sink with an EPR Capable cable.</p> <p>Only the EPR_Source_Capabilities and the EPR_Request Messages are allowed to Negotiate EPR Explicit Contracts. The SPR Mode Messages (Source_Capabilities and Request) are not allowed to be used while in EPR Mode.</p>
EPR PDO / EPR APDO	<p>Fixed Supply PDO that offers either 28V, 36V or 48V.</p> <p>Adjustable Voltage Supply (AVS) APDO whose Maximum voltage is the highest Fixed Supply PDO voltage in the EPR_Source_Capabilities Message and no more than 240W.</p>
EPR Sink	A Sink that supports both SPR Mode and EPR Mode.
EPR Source	A Source that supports both SPR Mode and EPR Mode.
Error Recovery	Port enters the ErrorRecovery State as defined in [USB-C].
Explicit Contract	An agreement reached between a Port Pair as a result of the Power Delivery Negotiation process. An Explicit Contract is established (or continued) when a Source sends an Accept Message in response to a Request Message sent by a Sink followed by a PS_RDY Message sent by the Source to indicate that the power supply is ready. This corresponds to the PE_SRC_Ready State for a Source Policy Engine and the PE_SNK_Ready State for a Source Policy Engine. The Explicit Contract can be altered through the Re-Negotiation process.
Extended Capabilities	An Extended Message containing Capabilities information.
Extended Control Message	An Extended Message containing control information only.
Extended Message	A Message containing Data Blocks. The Extended Message is defined by the Extended field in the Message Header being set to one and contains an Extended Message Header immediately following the Message Header.
Extended Message Header	Every Extended Message contains a 16-bit Extended Message Header immediately following the Message Header containing information about the Data Block and any Chunking being applied.

Term	Description
External Power Supply (EPS)	An AC Supplied Device that is only used to provide power as a Source (e.g. Power Brick).
External Supply	Power supply external to the Device. This could be powered from the wall or from any other power Source.
Fast Role Swap (FRS)	Process of exchanging the Source and Sink Power Roles between Port Partners rapidly due to the disconnection of an External Power Supply.
First Explicit Contract	The Explicit Contract that immediately follows an Attach, power on Hard Reset, Power Role Swap or Fast Role Swap event.
Fixed Battery	A Battery that is not easily removed or replaced by an end user e.g., requires a special tool to access or is soldered in.
Fixed Supply	A well-regulated fixed voltage power supply with a narrow variability as described in a Fixed Supply PDO.
Frame	Generic term referring to an atomic communication transmitted by PD such as a Packet, Test Frame or Signaling.
Guaranteed Capability Port	A Guaranteed Capability Port is always capable of delivering its Port Maximum PDP and indicates this by setting its Port Present PDP to be the same as its Port Maximum PDP except when limited by the cable's Capabilities. This is a Static capability.
Hard Reset	This is initiated by Hard Reset Signaling from either Port Partner. It restores VBUS to USB Default Operation and resets the PD communications engine to its default State in both Port Partners as well as in any Attached Cable Plugs. It restores both Port Partners to their default Data Roles and returns the VCONN Source to the Source Port. A DRP Source Port operating as a Source will continue to operate as a Source.
Host	See USB Host.
Hot Swappable Battery	A Battery that is easily accessible for a user to remove or change for another Battery.
Hub	A USB Device that provides additional connections to the USB.
ID Header VDO	The VDO in a Discover Identity Command immediately following the VDM Header. The ID Header VDO contains information corresponding to the Power Delivery Product.
Idle	Condition on CC where there are no signal transitions within a given time window.
Implicit Contract	An agreement on power levels between a Port Pair which occurs, not because of the Power Delivery Negotiation process, but because of a Power Role Swap or Fast Role Swap. Implicit Contracts are transitory since the Port Pair is required to immediately Negotiate an Explicit Contract after the Power Role Swap. An Implicit Contract Shall be limited to Type-C current [USB-C].
Initial Sink	Sink at the start of a Power Role Swap or Fast Role Swap which transitions to being the New Source.
Initial Source	Source at the start of a Power Role Swap or Fast Role Swap which transitions to being the New Sink.
Initiator	The initial sender of a Command Request in the form of a query.
Invariant PDOs	A Source Port that offers Invariant PDOs will always Advertise the same PDOs except when limited by the cable.
IoC	The Negotiated operating current.
IR Drop	The voltage drop across the cable and connectors between the Source and the Sink as defined in [USB-C]. It is a function of the resistance of the ground and power wire in the cable plus the contact resistance in the connectors times the current flowing over the path.
K-code	Special symbols provided by the 4b5b coding scheme. K-codes are used to signal Hard Reset and Cable Reset and delineate Packet boundaries.
Local Policy	Every PD Capable Device has its own Policy, called the Local Policy that is executed by its Policy Engine to control its power delivery behavior. The Local Policy at any given time might be the default Policy, hard coded or modified by changes in operating parameters or one provided by the system USB Host or some combination of these. The Local Policy Optionally can be changed by a System Policy Manager.
LPS	Limited Power Supply as defined in [IEC 62368-1].
LSB	An abbreviation for Least Significant Bit.
Managed Capability Port	A Managed Capability Port can have its Port Present PDP set to a different value than its Port Maximum PDP. Its Port Present PDP value can be dynamic and change during normal operation.
Message	The Packet Payload consisting of a Message Header for Control Messages and a Message Header and data for Data Messages and Extended Messages.

Term	Description
Message Header	Every Message starts with a 16-bit Message Header containing basic information about the Message and the PD Port's Capabilities.
Modal Operation	Operation where there are one or more Active Data Mode. Modal Operation ends when there are no longer any Active Data Modes.
Mode	Mode is a general term used to describe a particular type of operation of a given Device. Examples of modes are: Alternate Mode, EPR Mode, SPR Mode.
Mode Entry	Process to start operation in a particular Mode.
Mode Exit	Process to end operation in a particular Mode.
Multi-Drop	PD is a Multi-Drop system sharing the Power Delivery communication channel between the Port Partners and the cable.
Negotiate / Negotiated / Negotiation	The PD process whereby the Source Advertises its Capabilities, the Sink requests one of the Advertised Capabilities, and the Source acknowledges the Request, alters its output to satisfy the request and informs the Sink. The result of the Negotiation is an Explicit Contract.
New Sink	Sink at the end of a Power Role Swap or Fast Role Swap which has transition from being the Initial Source.
New Source	Source at the end of a Power Role Swap or Fast Role Swap which has transition from being the Initial Sink.
Packet	One entire unit of PD communication including a Preamble, SOP*, Payload, CRC and EOP.
Passive Cable	Cable with a USB plug on each end at least one of which is a Cable Plug supporting SOP* that does not incorporate data bus signal conditioning circuits. Supports the Structured VDM Discover Identity to determine its characteristics (Electronically Marked Cable see [USB-C]). Note: This specification does not discuss Passive Cables that are not Electronically Marked.
Passive Cable VDO	VDO defining the Capabilities of a Passive Cable.
Payload	Data content of a Packet, provided to/from the Protocol Layer.
PD	USB Power Delivery.
PD Capable	A Port that supports USB Power Delivery.
PD Power (PDP)	The output power of a Source or input power of a Sink.
PD SID	Standard ID allocated to this specification by the USB Implementer's Forum.
PDP Rating	The PDP Rating is the Manufacturer declared PDP. The Source Port PDP Rating is the Port Maximum PDP and the Sink Port PDP Rating is the maximum PDP the Sink can utilize.
PDUSB	USB Device Port or USB Host Port that is both PD Capable and capable of USB Communication. See also PDUSB Host, PDUSB Device and PDUSB Hub.
PDUSB Device	A USB Device with a PD Capable UFP. A PDUSB Device is only addressed by SOP Packets.
PDUSB Host	A USB Host which is PD Capable on at least one of its DFPs.
PDUSB Hub	A Port expander USB Device with a UFP and one or more DFPs which is PD Capable on at least one of its Ports. A self-powered PDUSB Hub is treated as a USB Type-C Multi-Port Charger.
PDUSB Peripheral	A USB Device with a PD Capable UFP which is not a PDUSB Hub. A PDUSB Peripheral is only addressed by SOP Packets.
Peripheral	A physical entity that is Attached to a USB cable and is currently operating as a USB Device.
PHY Layer	The Physical Layer responsible for sending and receiving Messages across the USB Type-C CC wire between a Port Pair.
Policy	Policy is defined by the product maker and sets the behavior of PD Capable parts of the system and defines the Capabilities it Advertises, requests made to (re)Negotiate power and the responses made to requests received.
Policy Engine (PE)	The Policy Engine interprets the Device Policy Manager's input to implement Policy for a given Port and directs the Protocol Layer to send appropriate Messages.
Port	An interface typically exposed through a receptacle, or via a plug on the end of a hard-wired captive cable. USB Power Delivery defines the interaction between a Port Pair.
Port Pair	Two Attached PD Capable Ports.
Port Partner	A Contract is Negotiated between a Port Pair Connected by a USB cable. These ports are known as Port Partners.
Power Conductor	The wire that delivers power from the Source to Sink. For example, USB's VBUS.

Term	Description
Power Data Object (PDO)	Data Object used to expose a Source Port's or Sink Port's power Capabilities as part of a Source_Capabilities / EPR_Source_Capabilities or Sink_Capabilities / EPR_Sink_Capabilities Message respectively. Fixed Supply, Variable Supply and Battery Supply Power Data Objects are defined; SPR Mode uses all four while EPR Mode uses only Fixed Supply and AVS PDOs.
Power Delivery Mode	Operation after a Contract has initially been established between a Port Pair. This Mode persists during normal Power Delivery operation, including after a Power Delivery Mode. Power Delivery Mode can only be exited by Detaching the Ports, applying a Hard Reset or by the Source removing power (except when the Initial Source removes power from VBUS during the Power Role Swap procedure).
Power Role	A Port Partner will be in one of two Power Roles; either Source or Sink.
Power Role Swap	Process of exchanging the Source and Sink Power Roles between Port Partners.
Power Rules	Define voltages and current ranges that are offered by compliant USB Power Delivery Sources and used by a USB Power Delivery Sink for a given value of PDP Rating.
PPS Mode	An SPR Source, currently operating as a Programmable Power Supply, is said to be operating in PPS Mode.
Preamble	Start of a transmission which is used to enable the receiver to lock onto the carrier. The Preamble consists of a 64-bit sequence of alternating 0s and 1s starting with a "0" and ending with a "1" which is not 4b5b encoded.
Product ID (PID)	16-bit unsigned value assigned by the vendor for a product.
Product Type	Product categorization returned as part of the Discover Identity Command.
Product Type VDO	VDO identifying a certain Product Type in the ID Header VDO of a Discover Identity Command.
Product VDO	The Product VDO contains identity information relating to the product.
Programmable Power Supply (PPS)	A power supply, operating in SPR Mode, whose output voltage can be programmatically adjusted in small increments over its Advertised range and has a programmable output current fold back. The Capabilities are exposed by the SPR Programmable Power Supply APDO.
Protocol Error	An Unexpected Message during an Atomic Message Sequence. A Protocol Error during an AMS will result in either a Soft Reset or a Hard Reset.
Protocol Layer	The entity that forms the Messages used to communicate information between Port Partners.
PS1 / PS2 / PS3	Classification of electrical power as defined in [IEC 62368-1].
Power Sinking Device (PSD)	Sink which draws power but has no other USB or Alternate Mode communication function e.g., a power bank.
Ra	Prior to application of VCONN, a powered cable applies a pull-down resistor Ra on its VCONN pin.
Rd	Pull-down resistor on the USB Type-C CC wire used to indicate that the Port is a Sink [USB-C].
Re-Attach	Attach of the Port Pair by a cable after a previous Detach.
Re-Negotiate / Re-Negotiated / Re-Negotiation	A process wherein one of the Port Partners wants to alter the Negotiated Contract.
Request	Message used by a Sink Port to Negotiate a Contract; refers to either a Request/EPR_Request Message.
Request Data Object (RDO)	Data Object used by a Sink Port to Negotiate a Contract as a part of a Request/EPR_Request Message.
Responder	The receiver of a Command Request sent by an Initiator that replies with a Command response.
Revision	Major release of the USB Power Delivery specification. Each Revision will have various Versions associated with it.
Revision 1.0	Deprecated major Revision of the USB Power Delivery Specification.
Revision 2.0	Superseded major Revision of the USB Power Delivery Specification as defined in [PD2], with which this specification is compatible.
Revision 3.x	Current major Revisions of the USB Power Delivery Specification.
Rp	Pull-up resistor on the USB Type-C CC wire used to indicate that the Port is a Source (see [USB-C]).
Shared Capacity Port	As defined in [USB-C]. Shared Capacity Ports can only be Managed Capability Ports
Signaling	Physical mechanism of transmitting bits.
Sink	The Port consuming power from VBUS; most commonly a USB Device.
Sink/Source	A Sink with the additional capability to function as a Source. This corresponds to a Dual-Role Power Port with Rd asserted on its CC wire.

Term	Description
Sink Capabilities	Capabilities wanted by a Sink.
Sink Port	Port operating as a Sink.
Sink Standby	During Sink Standby the Sink reduces its current draw to minimum current.
Sink with Accessory Support	A Sink that can source VCONN, but does not source VBUS [USB-C].
Soft Reset	A process that resets the PD communications engine to its default State.
Start of Packet (SOP)	K-code marker used to delineate the start of a Packet.
SOP Communication	Communication using SOP Packets also implies that an AMS is being followed.
SOP Packet	A Power Delivery Packet which starts with an SOP.
SOP' Communication	Communication with a Cable Plug using SOP' Packets, also implies that an AMS is being followed.
SOP' Packet	Any Power Delivery Packet which starts with an SOP' used to communicate with a Cable Plug.
SOP'' Communication	Communication with a Cable Plug using SOP'' Packets, also implies that an AMS is being followed.
SOP'' Packet	Any Power Delivery Packet which starts with an SOP'' used to communicate with a Cable Plug when SOP' Packets are being used to communicate with the other Cable Plug.
SOP' SOP''	K-code marker used for communication between a Port and a Cable Plug. See also SOP.
SOP*	Used to generically refer to K-code markers: SOP, SOP' and SOP''. See also SOP.
SOP* Communication	Communication using SOP* Packets, also implies an AMS is being followed.
SOP* Packet	A term referring to any Power Delivery Packet starting with either SOP, SOP', or SOP''.
Source	The Port supplying power over VBUS.
Source/Sink	A Source with the additional capability to act as a Sink. This corresponds to a Dual-Role Power Port with Rp asserted on its CC wire.
Source Capabilities	Capabilities offered by a Source.
Source Port	Port operating as a Source.
Standard Power Range (SPR)	Only the Source_Capabilities and the Request Messages are allowed to Negotiate SPR Explicit Contracts. The EPR Messages (the EPR_Source_Capabilities Message and the EPR_Request Message) are not allowed to be used while in SPR Mode.
SPR AVS	An SPR Source whose output voltage can be adjusted to an operating voltage within its Advertised range as exposed by the SPR AVS APDO.
SPR AVS Mode	A SPR Source, currently operating in an SPR AVS Contract, is said to be operating in SPR AVS Mode.
SPR Capabilities	An SPR Capabilities Message (Source_Capabilities Message or Sink_Capabilities Message) has at least one Power Data Object for 5V followed by up to 6 additional Power Data Objects.
SPR Contract	Explicit Contract Negotiated, in SPR Mode, based on SPR PDOs and APDOs.
SPR Mode	The classic Mode of PD operation where Explicit Contracts are Negotiated using SPR PDOs and APDOs.
SPR PDO / SPR APDO	A PDO offered in SPR Mode.
PPS Mode	A power supply, currently operating in a PPS Contract, is said to be operating in PPS Mode.
SPR Sink	A Sink which only supports SPR Mode and does not support EPR Mode.
SPR Source	A Source which only supports SPR Mode and does not support EPR Mode.
Standard ID (SID)	16-bit unsigned value assigned by the USB-IF to a given industry standards organization's specification.
Standard or Vendor ID (SVID)	Generic term referring to either a VID or a SID. SVID is used in place of the phrase "Standard or Vendor ID."
State	PD State machine State.
Structured Vendor Defined Message (SVDM)	A Vendor Defined Message where the contents and usage of bits 14...0 of the VDM Header are defined by this specification.
SVDM Header	The VDM Header for a Structured Vendor Defined Message.
Swap Standby	During Swap Standby the Source does not drive VBUS and the Sink's current draw does not exceed a minimum current.

Term	Description
System Policy	Overall System Policy generated by the system, broken up into the policies required by each Port Pair to affect the System Policy. It is programmatically fed to the individual devices for consumption by their Policy Engines.
System Policy Manager (SPM)	Module running on the USB Host. It applies the System Policy through communication with PD Capable Sinks and Sources that are also Connected to the USB Host via USB.
Test Frame	Frame consisting of a Preamble, SOP*, followed by test data.
Test Pattern	Continuous stream of test data in a given sequence .
Tester	The Tester is assumed to be a piece of test equipment that manages the BIST testing process of a PD UUT.
Upstream Facing Port (UFP)	Indicates the Port's position in the USB topology typically a Port on a Device as defined in [USB-C]. At connection, the Port defaults to operation as a USB Device (when USB Communication is supported) and as a Sink.
UFP VDO	VDO returned by the UFP containing Capabilities.
Unit Interval (UI)	The time to transmit a single data bit on the wire.
Unchunked	See Unchunked Extended Message.
Unchunked Extended Message	Extended Message that has been transmitted whole without using Chunking.
Unexpected Message	Message that a Port supports but has been received in an incorrect State.
Unit Under Test (UUT)	The PD Device that is being tested by the Tester and responds to the initiation of a particular BIST test sequence.
Unrecognized Message	Message that a Port does not understand e.g., a Message using a Reserved Message type, a Message defined by a higher specification Revision than the Revision this Port supports, or an Unstructured Vendor Defined Message for which the VID is not recognized.
Unstructured Vendor Defined Message (UVDM)	A Vendor Defined Message where the contents of bits 14...0 of the VDM Header are undefined.
UVDM Header	The VDM Header for an Unstructured Vendor Defined Message.
Unsupported Message	Message that a Port recognizes but does not support. This is a Message defined by the specification, but which is not supported by this Port.
USB Attached State	Synonymous with the [USB2] and [USB3] definition of the Attached State
USB Communication	Transfer of USB data Packets as defined in [USB2] and [USB3].
USB Default Operation	Operation of a Port at Attach or after a Hard Reset where the DFP Source applies 5V on VBUS and the UFP Sink is operating at 5V as defined in [USB2], [USB3], [USB-C] or [USB BC].
USB Device	Either a Hub or a Peripheral Device as defined in [USB2], [USB3] and [USB4].
USB Host	The system where the USB Host controller is installed as defined in [USB2], [USB3] and [USB4].
USB Powered State	Synonymous with the [USB2] and [USB3] definition of the powered State.
USB Safe State	State of the USB Type-C connector when there are pins to be re-purposed (see [USB-C]) so they are not damaged by and do not cause damage to their Port Partner.
USB Type-A	Term used to refer to any A plug or receptacle including USB Micro-A plugs and USB Standard- A plugs and receptacles. USB Micro-AB receptacles are assumed to be a combination of USB Type-A and USB Type-B.
USB Type-B	Terms used to refer to any B-plug or receptacle including USB Mini-B plugs, USB Micro-B plugs, and USB Standard-B plugs and receptacles, including the PD and non-PD versions. USB Micro-AB receptacles are assumed to be a combination of USB Type-A and USB Type-B.
USB Type-C / USB-C	Term used to refer to the USB Type-C connector plug, or receptacle as defined in [USB-C].
USB Type-C Multi-Port Charger	A product that exposes multiple USB Type-C Source Ports for the purpose of charging multiple Connected USB Devices as defined in [USB-C].
USB-C Port Control	Module in a PD Capable Device which controls Attach/Detach and either detects or sets the Rp value and contains the USB Type-C State machines.
USB4 Mode	Device is operating in a Mode as defined in [USB4].
Variable Supply	A power supply that has an output voltage that varies beyond the Fixed Supply range. This is exposed by the Variable Supply PDO.
VBUS	The VBUS wire delivers power from a Source to a Sink.

Term	Description
VCONN	Once the connection between USB Host and Device is established, the CC pin (CC1 or CC2) in the receptacle that is not Connected via the CC wire through the standard cable is re-purposed to Source VCONN to power circuits in a Cable Plug, VCONN Powered Accessory or VCONN Powered USB Device (see [USB-C]).
VCONN Powered Accessory (VPA)	An accessory that is powered from VCONN to operate in an Alternate Mode (see [USB-C]).
VCONN Powered USB Charge Through Device (CT-VPD)	A CT-VPD is a VPD with an additional Port for connecting a Source (e.g., a Charger) as defined in [USB-C].
VCONN Powered USB Device (VPD)	A captive cable USB Device that can be powered by either VCONN or VBUS as defined in [USB-C]. The term VPD refers to either a VPD or a CT-VPD with no Charger Connected.
VCONN Source	The USB Type-C Port responsible for sourcing VCONN.
VCONN Swap	Process of exchanging the VCONN Source between Port Partners.
Vendor Defined Message (VDM)	PD Data Message defined for vendor/standards usage. These are further partitioned into Structured Vendor Defined Messages, where Commands are defined in this specification, and Unstructured Vendor Defined Messages which are entirely vendor defined.
VDM Header	The first Data Object following the Message Header in a Vendor Defined Message.
VDO	See Vendor Data Object.
Vendor Data Object	Data Object used to send Vendor specific information as part of a Message.
Vendor Defined Extended Message (VDEM)	PD Extended Message defined for vendor/standards usage. A VDEM does not define any structure and Messages can be created in any manner that the vendor chooses.
Vendor ID (VID)	16-bit unsigned value assigned by the USB-IF to a given Vendor.
Version	A minor release of the USB Power Delivery specification associated with a particular Revision. Version numbers are also defined in VDMs.
VI	Same as power (i.e., voltage × current = power)

Chapter 3. Port Power Requirements

3.1. Introduction

Power Requirements define voltages and current ranges that are offered by compliant USB Power Delivery Source Ports and used by compliant USB Power Delivery Sink Ports. Power Requirements enable a collection of interoperable power sources to function with a variety of powered devices in a standard and compliant ecosystem. Power Requirements are generally defined based on a Port's PDP Rating and the Source PDPs (Port Guaranteed PDP, Port Maximum PDP, Port Present PDP) described in [Section 6.4.10](#). The Source Port PDP Rating is the Port Maximum PDP described in [Section 6.4.10](#) and the Sink Port PDP Rating is the maximum PDP the Sink can utilize described in [Section 6.5.16](#). The PDP Rating of a product with multiple Source ports is the highest Port Maximum PDP from the individual Source Ports.

Note: All of the Power Requirements defined in this chapter are for each individual Port.

3.2. Source Port Power Requirements

A Source Port is defined as a Guaranteed Capability Port or a Managed Capability Port. For all of the defined requirements, the Capabilities a Source exposes are based on the Port Maximum PDP, or if power constrained, the Port Present PDP.

3.2.1. General Source Port Requirements

The following are general Requirements for every Source Port.

1. The Maximum current for a Fixed or Variable PDO is based on the Port Present PDP (x/voltage). The Maximum current **May** be RoundUp or RoundDown to the nearest 10mA.
2. A Source **Shall Not** Advertise a current for a Fixed PDO that would result in the maximum power for that PDO (Voltage × Maximum Current) > maximum power of a higher voltage Fixed PDO.
3. Maximum power for Battery PDOs **Shall** be ≤ PDP Rating and the value can be rounded down.
4. The Maximum current for the SPR Programmable Power Supply (PPS) APDO **Shall** be as defined in [Table 3.4](#).
 - a. Note: When the PPS Power Limited bit is set in the PPS APDO, the output current is as defined in [Table 3.4](#). However, the Programmable Power Supply **May** limit its output current so that the product of its actual output voltage times the output current does not exceed the Port Present PDP.
 - b. If a 9V Prog, 15V Prog or 20V Prog Programmable Power Supply APDO is advertised, the maximum current **Shall** be RoundDown (x/Prog Voltage) to the nearest 50mA. When the PPS Power Limited bit is clear, the Source **Shall** provide this current at Maximum Voltage.
5. The first Source Capabilities, when in a Default or Implicit Contract, **Shall** set the Maximum Current of the 5V Fixed PDO ≥ the USB Type-C Current advertised by Rp.
6. The special case of a Source offering no capabilities (0mA or 0W) is described in [Section 7.9.1.5](#).

3.2.2. Guaranteed Capability Port

A Guaranteed Capability Port is a Port that, with the exception of a fault condition or a Cable power rating limitation, is always able to provide the Port Maximum PDP. Simple wall chargers, self-powered docks, displays, etc. are typically Guaranteed Capability Ports.

The following Requirements apply to a Guaranteed Capability Port:

1. A Guaranteed Capability Source Port **Shall** set Port Guaranteed PDP = Port Maximum PDP.
2. A Guaranteed Capability Source Port **Shall** set Port Present PDP = Port Maximum PDP except when
 - a. Port Maximum PDP > 60W and the cable VBUS Current Handling Capability = 3A. In this case, the Port **Shall** set Port Present PDP = 60W.
 - b. Port Maximum PDP > 100W (EPR Capable) and the Cable is not EPR Capable. In this case, the Port **Shall** set the Port Present PDP ≤ 100W. Note: Margin to 100W (e.g. 95W) is allowed and may be required to comply with safety standards. The actual Port Present PDP is product specific but should be as close to 100W as possible.

3.2.3. Managed Capability Port

A Managed Capability Port is a Port that may be constrained such that it does not always provide the Port Maximum PDP. Generally, these types of ports are on products where providing power is not the main function or they are Shared Capacity Ports where power is shared across multiple ports. Portable computers, mobile phones, tablets, shared-capacity docks and displays, etc. are typically Managed Capability Ports.

The following Requirements apply to a Managed Capability Port:

1. A Source **Shall** set Port Guaranteed PDP ≤ Port Maximum PDP.
2. A Source **Shall** set Port Present PDP ≤ Port Maximum PDP.
3. A Source **Shall** provide the PDO/APDOs required for its Port Present PDP as shown in [Table 3.2](#).
4. A Managed Capability Source Port **Shall** limit Port Present PDP when
 - a. Port Present PDP would be > 60W but the cable VBUS Current Handling Capability = 3A. In this case, the Port **Shall** set Port Present PDP = 60W.
 - b. Port Present PDP would be > 100W (EPR Capable) and the Cable is not EPR Capable. In this case, the Port **Shall** set the Port Present PDP ≤ 100W.

Note: Margin to 100W is allowed (e.g. 95W) to prevent actual power delivered due to voltage variation to be > 100W. The actual Port Present PDP is product specific but should be as close to 100W as possible.

5. A Source **May** offer Source Capabilities with power less than the Port Present PDP. However, when the Attached Sink has set Mismatch in the [Request](#) RDO, the Source **Shall** provide Source Capabilities equivalent to the minimum of either the Source Port Present PDP or the maximum sinking capacity of the Attached Sink found in either Sink Capabilities (PDO) or Extended Sink Capabilities (PDP).

3.2.4. Source Voltage and Current Requirements

The following are minimum Requirements for Source Port voltages and currents:

1. The voltages and minimum currents an SPR Source **Shall** support based on the Port Maximum PDP are defined in [Table 3.1](#).
2. The voltages and minimum currents an SPR Source **Shall** support based on the Port Present PDP are defined in [Table 3.2](#).
3. Each SPR PDO and APDO **Shall** only be available if it were required for the Port Maximum PDP as shown in [Table 3.1](#).

Figure 3.1 shows the **Required** voltage and minimum current for each SPR Fixed PDO based on Port Maximum PDP that is described in Table 3.1.

Figure 3.1. SPR Required Source Power Rule Illustration for Fixed PDOs

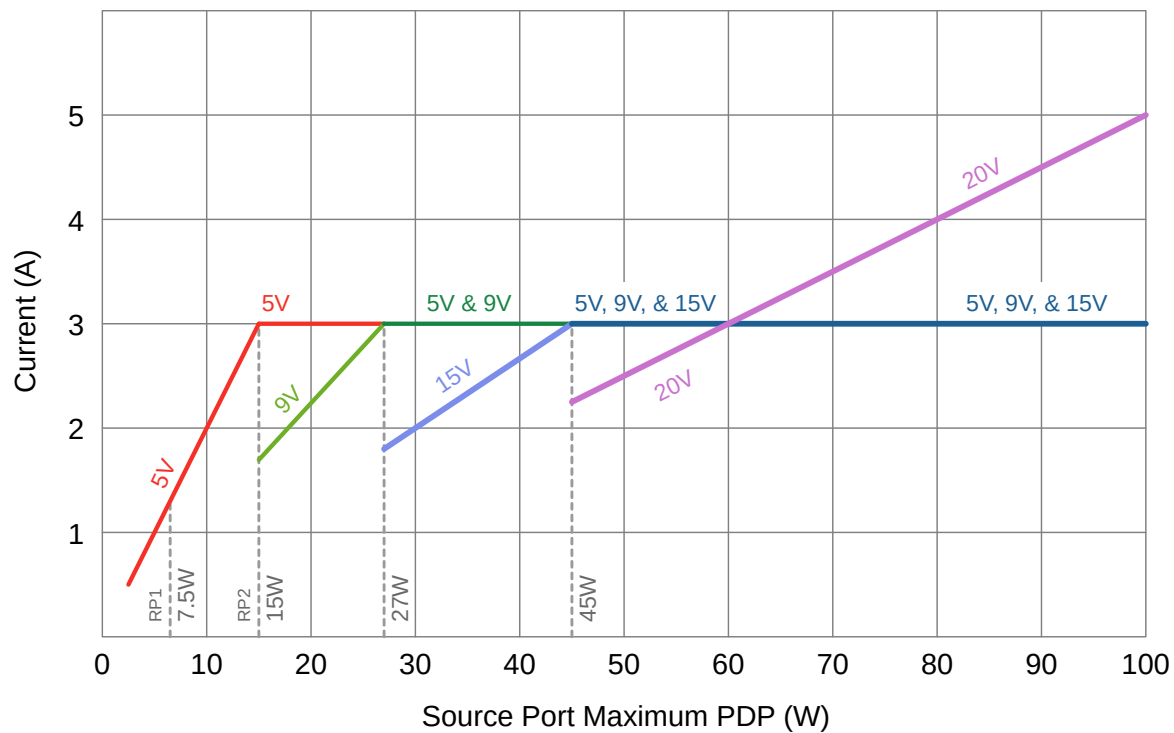


Table 3.1. SPR Source Voltages and Minimum Currents based on Port Maximum PDP

Port Maximum PDP Rating (W)	5V Fixed (A)	9V Fixed (A)	15V Fixed (A)	20V Fixed (A)	SPR AVS ² (A)
$0.5 \leq x \leq 15$	$PDP \div 5$	-	-	-	-
$15 < x \leq 27$	3	$PDP \div 9$	-	-	-
$27 < x \leq 45$	3	3	$PDP \div 15$	-	Max Voltage = 15V Max Current (9-15V) = 15V Fixed PDO Max Current
$45 < x \leq 100$	3	3	3	$PDP \div 20$ ¹	Max Voltage = 20V Max Current (9-15V) = 15V Fixed PDO Max Current Max Current (15-20V) = 20V Fixed PDO Max Current
1. Currents > 3A assume the Cable VBUS Current Carrying Capability is capable of the Port Maximum PDP. When it is not, Port Present PDP is limited by the cable. Refer to Table 3.2 for this case. 2. The Sink is allowed to Request up to the 20V Fixed Supply Max Current when the requested voltage from the AVS APDO is 15.0V. 3. '-' indicates PDO or APDO is not allowed.					

Table 3.2. SPR Voltages and Minimum Currents when Port Present PDP < Port Maximum PDP

Port Present PDP (W)	5V Fixed (A)	9V Fixed (A)	15V Fixed (A)	20V Fixed (A)	SPR AVS ^{3,4} (A)
$0.5 \leq x \leq 15$	$PDP \div 5$	$PDP \div 9^3$	$PDP \div 15^3$	$PDP \div 20^3$	Max Voltage = Max Fixed PDO Voltage ¹
$15 < x \leq 27$	3	$PDP \div 9$	PDP ÷ 15	PDP ÷ 20	Max Current (9-15V) = 15V Fixed PDO Max Current
$27 < x \leq 45$	3	3			
$45 < x \leq 100^3$	3	3	3		Max Current (15-20V) = 20V Fixed PDO Max Current
<ol style="list-style-type: none"> 1. SPR AVS APDO Shall only be available if the 15V Fixed PDO is available. 2. > 60W Port Present PDP requires a 5A cable. 3. This PDO and/or APDO is optional for Port Present PDP ≤ 15W. 4. The Sink is allowed to Request up to the 20V Fixed Supply Max Current when the requested voltage from the AVS APDO is 15.0V 					

The minimum **Required** Source Capabilities described in [Table 3.1](#) and [Table 3.2](#) can be generalized based on the Port Present PDP by:

- 5V, 9V, and 15V SPR Fixed PDOs
 - PDO Maximum Current = Minimum(3A, Port Present PDP ÷ Fixed PDO Voltage)
- 20V SPR Fixed PDO
 - PDO Maximum Current = Minimum(Cable Rating, Port Present PDP ÷ Fixed PDO Voltage)
- SPR AVS
 - 9V to 15V maximum current = 15V Fixed PDO Maximum Current
 - 15V to 20V maximum Current = 20V Fixed PDO Maximum Current

[Table 3.3](#) shows examples of the SPR Source Capabilities = advertised based on Port Present PDP from [Table 3.2](#).

Table 3.3. Examples of Required SPR Source Capabilities when Port Present PDP < Port Maximum PDP

Port Maximum PDP (W)	Cable Rating (A)	Port Present PDP (W)	Offers				
			5V Fixed (A)	9V Fixed (A)	15V Fixed (A)	20V Fixed (A)	SPR AVS (A)
80	5	65	3	3	3	3.25	9V - 15V: 3 15V - 20V: 3.25
80	5	40	3	3	2.67	2	9V - 15V: 2.67 15V - 20V: 2
80	3	40	3	3	2.67	2	9V - 15V: 2.67 15V - 20V: 2
80	5	20	3	2.22	1.33	1	9V - 15V: 1.33 15V - 20V: 1
80	3	20	3	2.22	1.33	1	9V - 15V: 1.33 15V - 20V: 1
40	5	20	3	2.22	1.33	Not offered	9V - 15V: 1.33

Port Maximum PDP (W)	Cable Rating (A)	Port Present PDP (W)	Offers				
			5V Fixed (A)	9V Fixed (A)	15V Fixed (A)	20V Fixed (A)	SPR AVS (A)
40	3	20	3	2.22	1.33	Not offered	9V - 15V: 1.33
80	3	15	3	1.67 ¹	1 ¹	0.75 ¹	9V - 15V: 1 ¹ 15V - 20V: 0.75 ²
40	3	15	3	1.67 ¹	1 ¹	Not offered	9V - 15V: 1 ¹

1. This Capability is **Optional** at this Port Present PDP and **May** be offered at this Port Present PDP if the Fixed 15V PDO is offered.

2. This Capability is **Optional** at this Port Present PDP and **May** be offered at this Port Present PDP if the Fixed 20V PDO is offered.

3.2.5. Optional Source Voltage/Currents

3.2.5.1. Fixed, Variable and Battery Supply

In addition to the voltages and currents specified in [Section 3.2.4](#), an SPR Source **May** Optionally supply additional voltages and increased currents.

While allowed, the use of **Optional** voltages and currents is strongly discouraged to avoid user confusion. **Optional** voltages or currents allow Sinks that might perform differently with a Source with an **Optional** voltage or current than with a Source with an equivalent Port Maximum PDP but without the **Optional** voltage or current.

The following are Requirements apply to **Optional** Fixed, Variable, and Battery PDOs:

1. A Source **Shall Not** Advertise an optional Fixed PDO voltage > 9V.
2. A Source **Shall** only Advertise a Variable or Battery PDO with a Maximum Voltage \leq the highest offered Fixed PDO voltage with allowed variance ($\leq 1.05 \times$ Fixed PDO Voltage)
3. A Source **Shall Not** Advertise an optional Fixed PDO voltage > the highest **Required** Fixed Voltage PDO in [Table 3.1](#).
4. A Source **May** offer a Variable PDO, a Battery PDO, or a higher Maximum Current in a Fixed PDO than that found in [Table 3.1](#) and [Table 3.2](#). The following Requirements apply:
 - a. A Source **Shall** not Advertise a Maximum Current for a Fixed PDO > the current defined by the Port Present PDP in [Table 3.1](#) and [Table 3.2](#).
 - b. A Source **Shall Not** Advertise a Maximum Current for a Fixed PDO that would result in the maximum power for that PDO exceeding the maximum power of the next higher voltage required Fixed PDO (5V, 9V, 15V, or 20V) as specified in [Table 3.1](#) and [Table 3.2](#).
 - c. A Source **Shall Not** Advertise a Maximum Current for a Variable PDO that would result in the maximum power for that PDO (Maximum Voltage \times Maximum Current) exceeding the maximum power ($1.05 \times$ Voltage \times Maximum Current) of the next higher voltage required Fixed PDO (5V, 9V, 15V, or 20V) as specified in [Table 3.1](#) and [Table 3.2](#). The 1.05 factor accounts for the $\pm 5\%$ voltage tolerance of Fixed PDOs.
 - d. A Source **Shall Not** Advertise a Maximum Power for a Battery PDO that would result in the maximum power for that PDO exceeding the maximum power ($1.05 \times$ Voltage \times Maximum Current) of the next higher voltage required Fixed PDO (5V, 9V, 15V, or 20V) as specified in [Table 3.1](#) and [Table 3.2](#). The 1.05 factor accounts for the $\pm 5\%$ voltage tolerance of Fixed PDOs.
5. EPR Capable Sources **Shall Not** implement optional Fixed PDOs.
6. An EPR Capable Source **Shall** only implement a Variable or Battery PDO as an SPR PDO. SPR Variable and Battery PDOs are allowed in [EPR Mode](#).

3.2.5.2. SPR Programmable Power Supply (PPS)

Programmable Power Supply (PPS) is **Optional**. When included, the voltages and currents a PPS **Shall** support are as defined in [Table 3.4](#).

Table 3.4. SPR PPS APDOs based on the Port Maximum PDP

Port Maximum PDP (W)	Prog Voltage = 9V ^{3,4} (A)	Prog Voltage = 15V ^{3,4} (A)	Prog Voltage = 20V ^{3,4} (A)
$x \leq 15$	-	-	-
$15 < x < 27$	$PDP \div 9^1$	-	-
27	3	-	-
$27 < x < 45$	3^2	$PDP \div 15^1$	-
45	-	3	-
$45 < x < 60$	-	3^2	$PDP \div 20^1$
60	-	-	3
$60 < x < 100$	-	-	$PDP \div 20^1$
100	-	-	5

1. The PPS APDOs Maximum Current field **Shall** Advertise RoundDown(PDP ÷ Prog Voltage) to the nearest 50mA.
2. The PPS APDOs Maximum Current field **Shall** Advertise at least 3A. When the cable VBUS Current Handling Capability = 5A, the Maximum Current field **May** Advertise up to RoundDown(PDP ÷ Prog Voltage) to the nearest 50mA.
3. See [Table 4.2](#) for the voltage ranges defined for the PPS Prog APDOs.
4. Applies to PPS APDOs regardless of value of the PPS Power Limited bit.
5. '-' indicates APDO is not allowed.

3.2.5.3. Extended Power Range (EPR)

Support of [EPR Mode](#) is **Optional**. An EPR Capable Port has a Port Maximum PDP > 100W and ≤ 240W.

[Table 3.5](#) and [Table 3.6](#) define the requirements for EPR Source Ports.

The following Requirements apply to an EPR Capable Source Port (EPR Source Port):

1. An EPR Source Port **May** operate in either SPR Mode or [EPR Mode](#) when providing power ≤ 100W.
2. An EPR Source Port **May** be implemented as a Managed Capability Port or Guaranteed Capability Port.
3. An EPR Source Port, operating in SPR Mode, **May** Advertise the SPR 20V Fixed PDO with < 5A (< 100W) to avoid potentially violating external safety regulations. This value should be as close as possible to 100W if not otherwise limited by Port Present PDP.
4. When operating in [EPR Mode](#), an EPR Source Port with a Port Present PDP ≥ 100W **Shall** offer 5A in the 20V Fixed PDO.
5. When the Attached cable is not an EPR Capable cable, an EPR Capable Source **Shall** only operate in SPR Mode.
6. When in [EPR Mode](#), an EPR Source Port that supports PPS **Shall** offer the SPR Fixed 20V PDO and PPS 20V Prog APDO at the Port Present PDP, up to 100W.
7. An EPR Source Port in an EPR AVS Contract **Shall** stay within the power offered in the PDP Field from the EPR AVS APDO defined in [Table 6.16](#) and **Shall Reject** any Request whose operating current exceeds (PDP ÷ requested voltage) or 5A.

[Figure 3.3](#) shows the definition of the valid operating range for an EPR Source operating in an AVS Explicit Contract based on the PDP Field.

Figure 3.2 combined with Figure 3.1 shows the minimum current that an EPR Source **Shall** support at each EPR Fixed PDO for a given Port Maximum PDP.

Figure 3.2. EPR Source Power Rule Illustration for Fixed PDOs

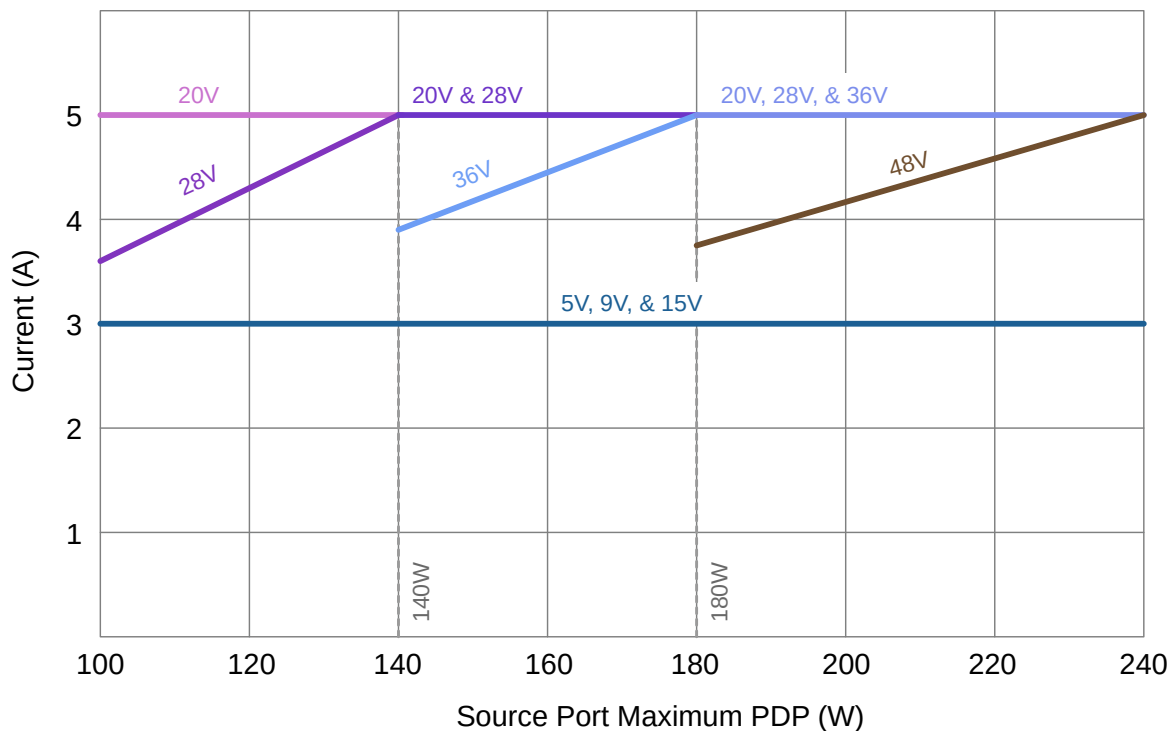


Table 3.5. EPR Source Capabilities based on the Port Maximum PDP

Port Maxi- mum PDP (W)	28V Fixed (A)	36V Fixed (A)	48V Fixed (A)	EPR AVS ² (A)
100 < x ≤ 140	PDP÷28	-	-	Max Voltage = 28V Available Current (15-28V) = PDP ÷ AVS voltage (maximum of 5A)
140 < x ≤ 180	5	PDP÷36	-	Max Voltage = 36V Available Current (15-36V) = PDP ÷ AVS voltage (maximum of 5A)
180 < x ≤ 240	5	5	PDP÷48	Max Voltage = 48V Available Current (15-48V) = PDP ÷ AVS voltage (maximum of 5A)

1. '-' indicates PDO or APDO is not allowed.

2. The current available for a given AVS voltage is as indicated in this column. The current defined here is describing the top edge of the **Valid** Operating Region as illustrated in Figure 3.3. The AVS APDO does not have a Maximum Current field, so the maximum current is calculated from the Port Present PDP.

Table 3.6. EPR Source Capabilities when Port Present PDP < Port Maximum PDP

Port Present PDP (W)	28V Fixed ¹ (A)	36V Fixed ¹ (A)	48V Fixed ¹ (A)	EPR AVS (A)
7.5 < x ≤ 27	PDP÷28 ²	PDP÷36 ²	PDP÷48 ²	Max Voltage = Max Fixed PDO Voltage ³ Available Current = PDP ÷ AVS Voltage (maximum of 5A)
27 < x ≤ 45	PDP÷28			
45 < x ≤ 60		PDP÷36		
60 < x ≤ 75				
75 < x ≤ 100				
100 < x ≤ 140	PDP÷28	PDP÷36	PDP÷48	
140 < x ≤ 180	5			
180 < x ≤ 240				
1. Each EPR Fixed PDO is only available if it were required for the Port Maximum PDP as shown in Table 3.5 .				
2. This PDO is Optional for this Port Present PDP.				
3. This APDO is Required when an EPR Fixed PDO is offered.				

EPR Source Capabilities described in [Table 3.5](#) and [Table 3.6](#) can be generalized based on the Port Present PDP by:

- For each Fixed PDO
 - PDO Maximum Current = Minimum(5A, $PDP \div \text{Fixed PDO Voltage}$)
 - 48V Fixed PDO is not allowed if Port Maximum PDP $\leq 180W$ and otherwise optional if Port Present PDP $\leq 60W$.
 - 36V Fixed PDO is not allowed if Port Maximum PDP $\leq 140W$ and otherwise optional if Port Present PDP $\leq 45W$.
 - 28V Fixed PDO is optional if Port Present PDP $\leq 27W$.
- EPR AVS
 - PDO Maximum Voltage = Maximum offered Fixed PDO Voltage
 - PDO Minimum Voltage = 15V
 - PDO PDP = Port Present PDP

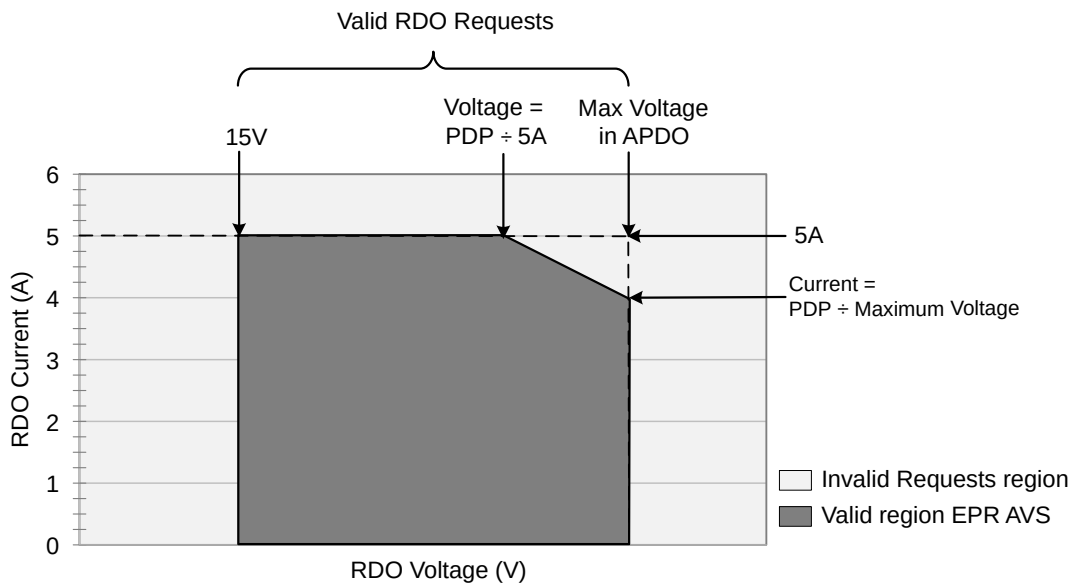
[Table 3.7](#) shows examples of the SPR Source Capabilities = advertised based on Port Present PDP from [Table 3.2](#).

Table 3.7. Examples of EPR Source Capabilities when Port Present PDP <Port Maximum PDP

Port Maximum PDP (W)	Port Present PDP (W)	Offers			
		28V Fixed(A)	36V Fixed(A)	48V Fixed(A)	EPR AVS max Voltage (V) and Power (W)
200	108	3.86	3	2.25	48V@108W
160	108	3.86	3	Not offered	36V@108W
120	108	3.86	Not offered	Not offered	28V@108W
200	72	2.57	2	1.5	48V@72W
160	72	2.57	2	Not offered	36V@72W
120	72	2.57	Not offered	Not offered	28V@72W
200	36	1.29	1 ¹	0.75 ¹	48V@36W ¹
160	36	1.29	1 ¹	Not offered	36V@36W ¹

Port Maximum PDP (W)	Port Present PDP (W)	Offers			
		28V Fixed(A)	36V Fixed(A)	48V Fixed(A)	EPR AVS max Voltage (V) and Power (W)
120	36	1.29	Not offered	Not offered	28V@36W
1. These Capabilities are not required but may be offered at this Port Present PDP.					

Figure 3.3. Valid EPR AVS Operating Region



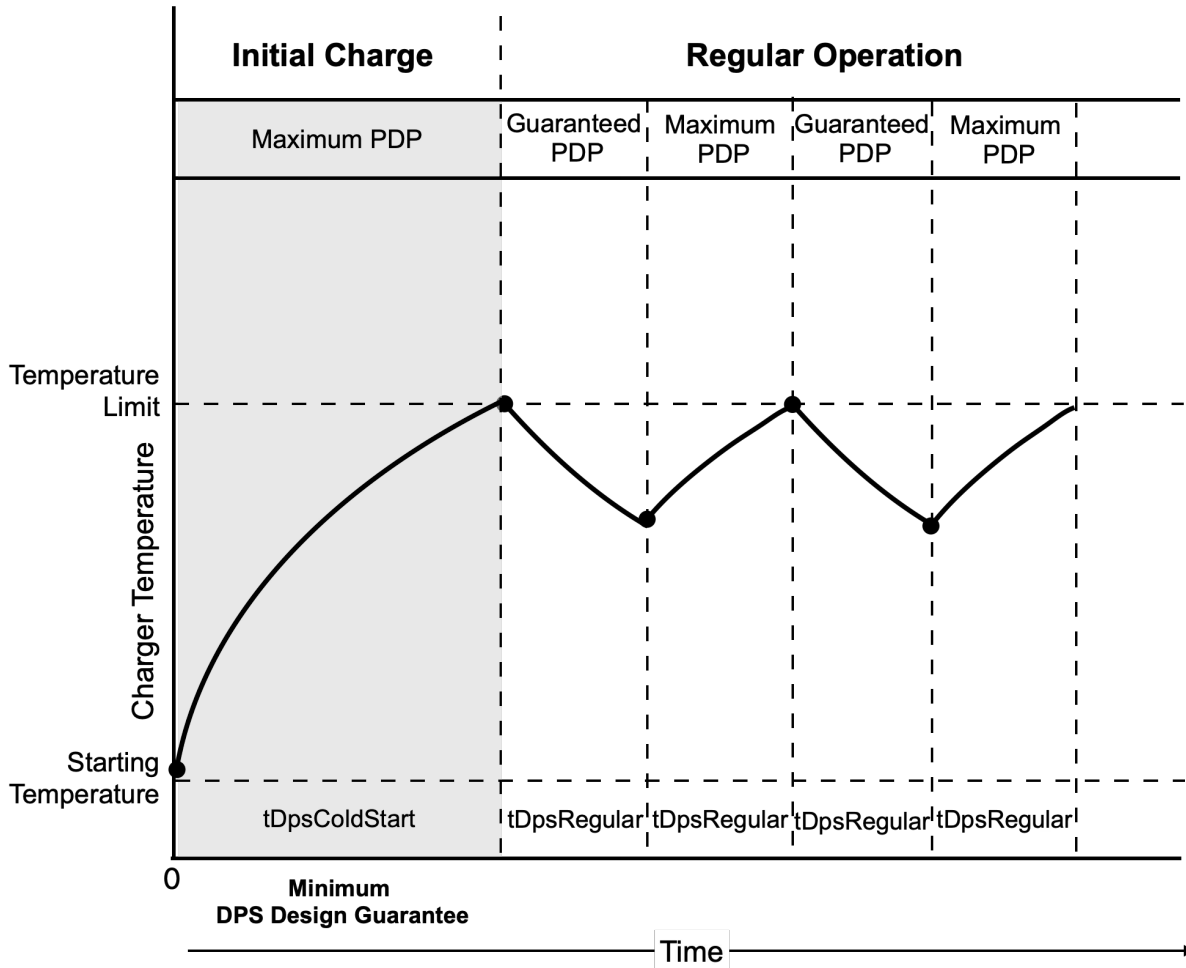
3.3. Dynamic Power Supply (DPS)

The Source power Requirements defined in [Section 3.2.4](#) and [Section 3.2.5](#) **Shall** apply to a DPS Port, together with the following additional Requirements:

- A DPS Port **Shall** guarantee `tDpsColdStart` at Maximum PDP provided the following conditions are met:
 - The unit is starting from a cold start. This means the components inside the unit are at ambient temperature, and the ambient temperature **Should** be around 25°C.
 - The Source Port is not sourced by an internal Battery, per the Source Inputs Field of the [Source Capabilities Extended Message](#).
 - No other events arise that may require the Source to send New Source Capabilities (e.g. a new [Request](#) from the Sink).
- The Source **Should** not send updated Source Capabilities within `tDpsRegular` or `tDpsColdStart`, with the following exceptions:
 - If the Source just entered its First Explicit Contract with the Sink, it **May** send an updated [Source Capabilities Message](#), provided the Sink sets the Capability Mismatch bit, and the Source can increase the power offered.
 - The updated Source Capabilities are due to sharing power with another Port when the Source is a shared capability Port.

- c. Events arise that may require the Source to send New Source Capabilities (e.g. a new [Request](#) from the Sink).
- d. Parameters other than the temperature affect the Source Capabilities.

Figure 3.4. Example of a DPS temperature profile



A DPS Source uses [Alert Message](#), [Status Message](#), and [Source Info Message](#) fields to inform a Sink about changes in its Capabilities. See [Section 7.14](#) and [Section 7.26](#) for additional details on requirements for these fields.

3.4. Sink Power Requirements

3.4.1. Sink Power Rule Considerations

The Sink Power Requirements are designed to ensure the best possible user experience when a given Sink is powered by a compliant Source that only supplies the **Required** voltages and currents.

Sinks are designed to use Sources with a matching or higher PDP Rating. The Sink Power Requirements are based on the following considerations:

- Low power Sources (e.g., 5V) are expected to be very common and will be used with Sinks designed for a higher PDP.
- Optimizing the user experience when Sources with a higher PDP Rating are used with low power Sinks.

- Preventing Sinks that only function well (or at all) when using **Optional** voltages and currents.

3.4.2. Sink Requirements

The following Requirements apply to Sink RDO:

1. A Sink **Shall** set RDO Operational Current or RDO Maximum Power to result in power \leq Sink Maximum PDP.
2. A Sink **Should** set Capabilities Mismatch when the voltage, current, or power being offered in Source Capabilities is insufficient for optimal operation or charging. A Sink **Shall** not set Capabilities Mismatch otherwise.
3. A Sink **Shall** set RDO Operational Current or Maximum Current as RoundDown to the nearest 10mA for Fixed/Variable/AVS PDO/APDO Supplies and 50mA for PPS APDO.
4. A Sink without a Battery **Shall** set Maximum Power for a Battery Supply RDO to \leq Sink Operational PDP.
5. A Sink requiring no power from the Source, **Shall** send the 5V RDO with Operational Current set to 0mA.

The following Requirements apply to Sink Capabilities:

1. A Sink **Shall Not** Advertise Fixed PDO maximum voltages and currents that exceed the [Sink_Capabilities_Extended_Message](#) SPR Sink Maximum PDP when in SPR Mode and EPR Sink Maximum PDP when in EPR Mode.
2. A Sink **Shall Not** Advertise Variable PDO maximum voltages and currents that exceed the [Sink_Capabilities_Extended_Message](#) SPR Sink Maximum PDP when in SPR Mode and EPR Sink Maximum PDP when in EPR Mode.
3. A Sink **Shall Not** Advertise a Battery PDO maximum allowable power that exceeds the [Sink_Capabilities_Extended_Message](#) SPR Sink Maximum PDP when in SPR Mode and EPR Sink Maximum PDP when in EPR Mode.
4. A Sink **Shall Not** Advertise a PPS APDO maximum allowable power that exceeds the [Sink_Capabilities_Extended_Message](#) SPR Sink Maximum PDP when in SPR Mode and EPR Sink Maximum PDP when in EPR Mode.
5. A Sink **Shall Not** Advertise an AVS APDO maximum allowable power that exceeds the [Sink_Capabilities_Extended_Message](#) SPR Sink Maximum PDP when in SPR Mode and EPR Sink Maximum PDP when in EPR Mode.

3.5. Sink with Accessory Support

The following Requirement applies to a Sink with Accessory Support:

1. When sourcing VCONN, a Sink with Accessory Support is considered a Source even though VBUS is not supplied. The Sink with Accessory Support **Shall** offer a single 5V PDO with the Maximum Current field set to 0mA for the purpose of allowing a VPA to enter an Alternate Mode.

3.6. PDUSB Device Requirements

A PDUSB Device **Shall** comply with the requirements defined in [\[PDUSB\]](#).

Chapter 4. Power Delivery Electrical Requirements

Throughout this chapter, certain functionally equivalent electrical parameters are identified by different names depending on the Contract type (Fixed, AVS, PPS etc.), though their behavior is consistent across Contract type. To standardize terminology and reduce redundancy, the following table defines a set of aliases mapped to specific variables. The aliases will be used throughout this chapter and will collectively refer to each of the parameters it is aliasing. For example, any figure using the [vNew](#) alias represents an equivalent figure for [vSrcNew](#), [vAvsNew](#), and [vPpsNew](#). The alias will not be used where the behavior does not apply to all aliased parameters.

Table 4.1. Universal Aliases for PDO or APDO Variables

Alias	Fixed PDO Parameter	AVS APDO Parameter	PPS APDO Parameter
iSlewNeg	iLoadReleaseRate	iLoadReleaseRate	iLoadReleaseRate or iPpsCLLoadReleaseRate
iSlewPos	iLoadStepRate	iLoadStepRate	iLoadStepRate or iPpsCLLoadStepRate
tSrcTransLarge	--	tAvsSrcTransLarge	tPpsSrcTransLarge
tSrcTransSmall	--	tAvsSrcTransSmall	tPpsSrcTransSmall
tTransient	tSrcTransient	tAvsTransient	tPpsTransient
vMaxVoltage	--	vAvsMaxVoltage	vPpsMaxVoltage
vMinVoltage	--	vAvsMinVoltage	vPpsMinVoltage
vNew	vSrcNew	vAvsNew	vPpsNew
vSlewNeg	vSrcSlewNeg	vAvsSlewNeg	vPpsSlewNeg
vSlewPos	vSrcSlewPos	vAvsSlewPos	vPpsSlewPos
vSmallStep		vAvsSmallStep	vPpsSmallStep
vStep	--	vAvsStep	vPpsStep
vValid	vSrcValid	vAvsValid	vPpsValid

4.1. Source Requirements

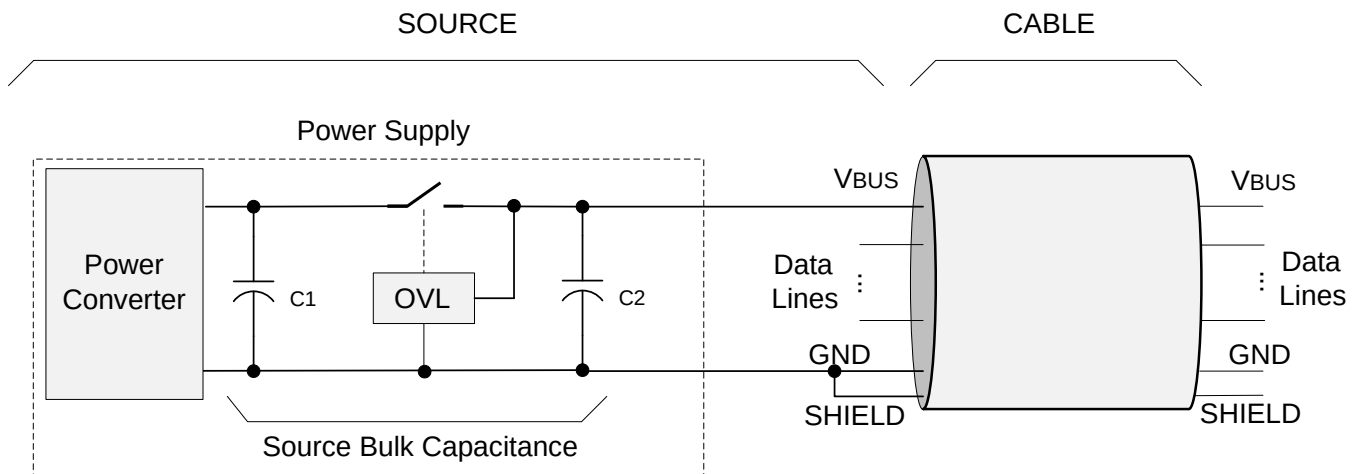
4.1.1. Behavioral Aspects

A USB PD Source is a device that supplies power over a USB connection using the USB Power Delivery protocol. It advertises available voltage and current levels, negotiates power contracts with Sinks, and adjusts its output accordingly. In addition to being a USB Type-C Source and following all [USB-C] defined behaviors, a USB PD Source must meet all requirements defined in this chapter.

4.1.2. Source Bulk Capacitance

The Source bulk capacitance consists of C1 and C2 as shown in. The switch (or Ohmic Interconnect) usually consists of one or two MOSFETs and **May** be part of the circuit implemented by the Source to control its VBUS Output Voltage Limit (OVL) as described in. The capacitance might be a single capacitor, a capacitor bank or distributed capacitance. If the power supply is dedicated to a single Port, the minimum bulk capacitance is defined as [cSrcBulk](#). If the power supply is shared across multiple ports, the bulk capacitance is defined as [cSrcBulkShared](#).

The Source bulk capacitance is allowed to change for a newly Negotiated power level. The capacitance change **Shall** occur before the Source is ready to operate at the new power level. During a [Power Role Swap](#), the Initial Source **Shall** transition to Swap Standby before operating as the New Sink. Any change in bulk capacitance required to complete the [Power Role Swap](#) **Shall** occur during Swap Standby.

Figure 4.1. Placement of Source Bulk Capacitance

4.1.2.1. Source PDOs or APDOs

Consistent with the Power Data Objects discussed in [Section 6.4.1.3](#), the power supply PDO or APDO that are available as a Source in a USB Power Delivery System are:

- The Fixed Supply PDO exposes well-regulated fixed voltage power supplies. A Source **Shall** support at least one Fixed Supply PDO providing [vSafe5V](#). The output voltage of a Fixed Supply **Shall** remain within the range defined by the relative tolerance [vSrcNew](#) and the absolute band [vSrcValid](#) as listed in [Table 4.6](#) and described in [Section 4.1.2.1.1](#)
- The Variable Supply (non-Battery) PDO exposes less well-regulated Sources. The output voltage of a Variable Supply (non-Battery) **Shall** remain within the absolute maximum output voltage and the absolute minimum output voltage exposed in the Variable Supply PDO.
- The Battery Supply PDO exposes Batteries that can be Connected directly as a Source to VBUS. The output voltage of a Battery Supply **Shall** remain within the absolute maximum output voltage and the absolute minimum output exposed in the Battery Supply PDO.
- The Programmable Power Supply (PPS) and Adjustable Voltage Source (AVS) Augmented Power Data Object (APDO) expose a Source with an output voltage that can be adjusted programmatically over a defined range.
 - For AVS, the output voltage **Shall** remain within a range defined by the relative tolerance [vAvsNew](#), and the absolute band [vAvsValid](#).
 - For PPS, the output voltage **Shall** remain within a range defined by the relative tolerance [vPpsNew](#) and the absolute band [vPpsValid](#). These limits are applicable as long as Operating Mode Flag (OMF) is 0. See [Section 4.1.3.2.2](#) for OMF information.

4.1.2.1.1. Voltage Regulation

This section applies to the Source while operating in any of the PDOs discussed on [Section 4.1.2.1](#).

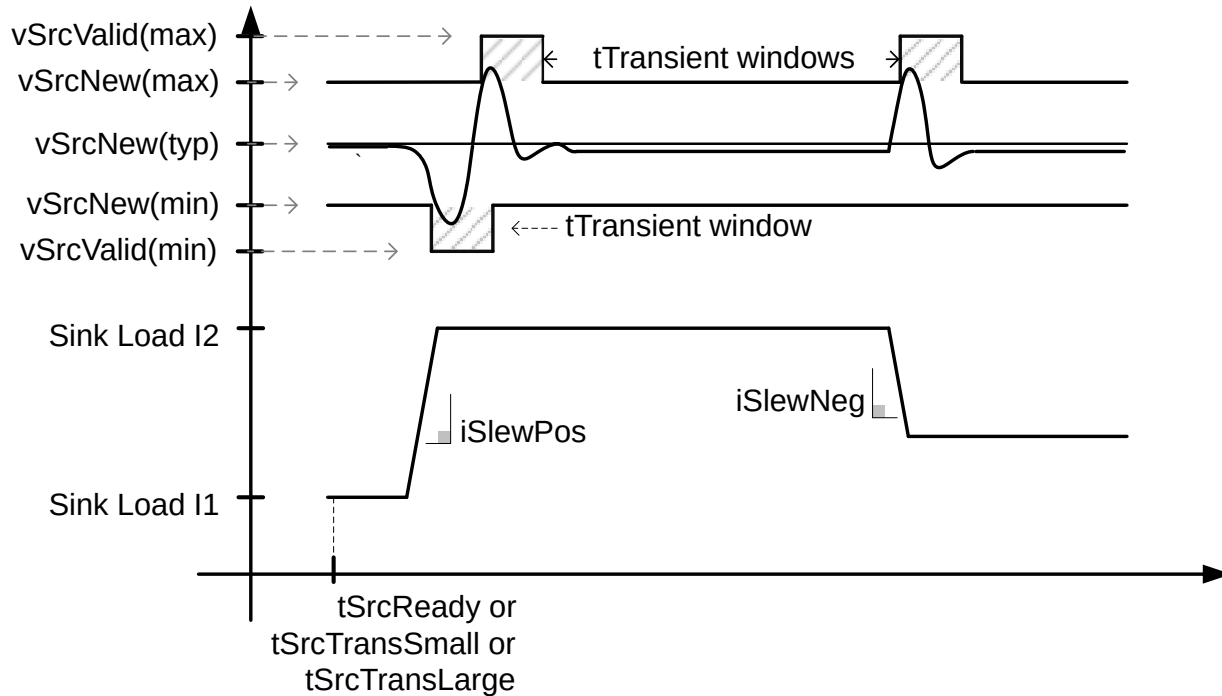
After a voltage transition is complete (i.e., after [tSrcReady](#), [tSrcTransSmall](#) or [tSrcTransLarge](#)) and during static load conditions, the Source output voltage **Shall** remain within the [vNew](#) or [vSafe5V](#) limits as applicable. The ranges defined by [vNew](#) and [vSafe5V](#) account for DC regulation accuracy, line regulation, load regulation, and output ripple.

During transient load conditions (defined by [iSlewPos](#) or [iSlewNeg](#)), the Source output voltage **Shall Not** go beyond the range specified by [vValid](#). The amount of time the Source output voltage can be in the band between either [vNew](#)

or [vSafe5V](#) and [vValid](#) **Shall Not** exceed [tTransient](#). Refer to [Table 4.6](#) for the output voltage tolerance specifications. [Figure 4.2](#) illustrates the application of [vNew](#) and [vValid](#) after the voltage transition is complete.

The [vNew](#) and [vValid](#) limits **Shall Not** apply to VBUS during the VBUS discharge and switchover that occurs during a [Fast Role Swap](#) as described in [Chapter 10](#).

Figure 4.2. Application of vNew and vValid limits after tSrcReady or tSrcTransSmall or tSrcTransLarge



The Source output voltage **Shall** be measured at the connector receptacle. The stability of the Source is left to the discretion of the implementer; however, the design **Shall** account for the possibility that the Sink **May** introduce any load transient inside the constraints of the Contract, provided that the slew rate is between [iSlewPos](#) and [iSlewNeg](#). The transient behavior of the load current is defined in [Section 4.2.6](#).

In some systems it might be necessary to design the Source to compensate for the voltage drop between the output stage of the power supply electronics and the receptacle contact. The determination of whether compensation is necessary is left to the discretion of the Source implementation.

4.1.3. Source Transitions

4.1.3.1. Fixed Voltage Transitions

The Source **Shall** transition VBUS from the starting voltage to the new voltage in a controlled manner. [Figure 4.3](#) and [Figure 4.4](#) illustrate the transition process for positive and negative fixed voltage transitions respectively. The following statements **Shall** be observed for fixed voltage transitions:

- The Negotiated new voltage (e.g., 5V, 9V, 15V, ...) defines the nominal value for [vSrcNew](#).
- During the positive transition the Source **Shall** be able to supply the Sink Standby current and the transient current to charge the total bulk capacitance on VBUS.

- The slew rate of the positive transition **Shall Not** exceed [vSrcSlewPos](#) and the slew rate for a negative transition **Shall Not** exceed [vSrcSlewNeg](#).
- The transitioning Source output voltage **Shall** settle within [vSrcNew](#) by [tSrcSettle](#).
- The Source **Shall** be able to supply the Negotiated power level at the new voltage by [tSrcReady](#).
- Voltage transitions follow these rules:
 - The positive voltage transition **Shall** remain above [vSrcValid](#) (min) of the previous Explicit Contract and below [vSrcValid](#) (max) of the new Explicit Contract.
 - The negative voltage transition **Shall** remain below [vSrcValid](#) (max) of the previous Contract and above [vSrcValid](#) (min) of the new Explicit Contract.
- The starting time, t_0 , in [Figure 4.3](#) starts [tSrcTransition](#) after the last bit of the EOP of the [GoodCRC Message](#) has been received by the Source.
- Negative transitions to [vSafe0V](#) do not have the same requirements illustrated in [Figure 4.4](#). See [Section 4.1.5.5](#).

Figure 4.3. Transition Envelope for Positive Voltage Transitions

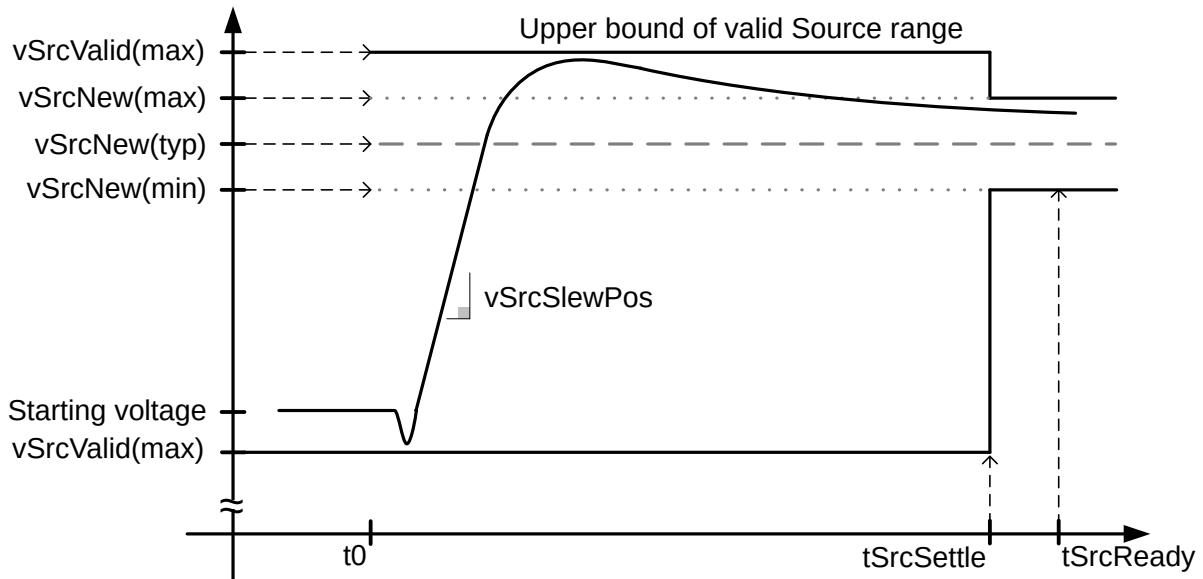
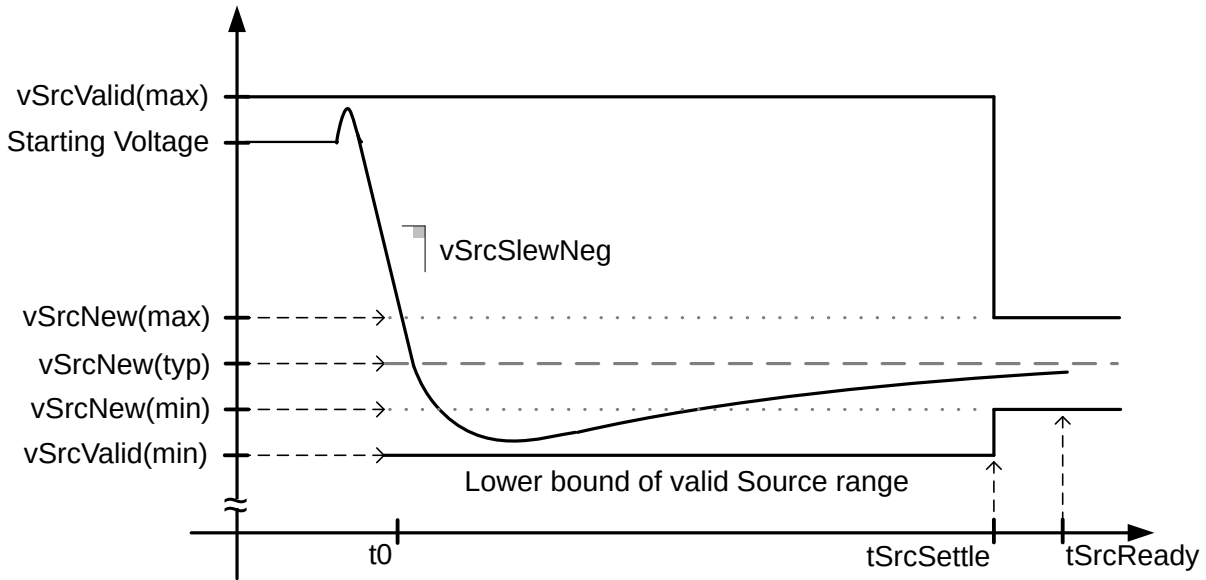


Figure 4.4. Transition Envelope for Negative Voltage Transitions

4.1.3.2. AVS/PPS Transitions

4.1.3.2.1. Voltage Transitions

When in AVS or PPS Mode, the Source **Shall** transition VBUS over the defined voltage range in a controlled manner. The output voltage in the RDO defines the nominal value of the output voltage after completing a voltage transition. The following statements **Shall** be observed by an AVS Source and by a PPS Source when not in Current Limit (CL) Mode (see [Section 4.1.3.2.2](#)):

- Settle within the limits defined by [vNew](#) by [tSrcTransSmall](#) for steps smaller than or equal to [vSmallStep](#).
- Settle within limits defined by [vNew](#) by [tSrcTransLarge](#) for steps larger than [vSmallStep](#).
- Overshoot and undershoot **Shall** not exceed [vValid](#).
- The voltage **May** change in a step-wise or linear manner and the slew rate of either type of change **Shall Not** exceed [vSlewPos](#) for voltage increases or [vSlewNeg](#) for voltage decreases.
- The nominal requested voltage equates to an integer number of LSB changes, which are defined as [vStep](#).
- If an increase in voltage is requested, then the new voltage **Shall** be greater than or equal to the current voltage. See [Section 4.1.3.2.6](#) for more details.
- If a decrease in voltage is requested, then the new voltage **Shall** be less than or equal to the current voltage. See [Section 4.1.3.2.6](#) for more details.
- The voltage range is defined by the parameters [vMinVoltage](#) and [vMaxVoltage](#), corresponding to the Minimum Voltage and Maximum Voltage fields of the respective APDOs. [Table 4.2](#) and [Table 4.3](#) show the valid voltage ranges for PPS and SPR AVS.

Table 4.2. SPR Programmable Power Supply (PPS) Voltage Ranges

Fixed Nominal Voltage	9V Prog	15V Prog	20V Prog
Maximum Voltage	11V	16V	21V
Minimum Voltage	5V	5V	5V

Table 4.3. SPR Adjustable Voltage Supply (AVS) Voltage Ranges

AVS Voltage Range	15V	20V
Maximum Voltage	15V	20V
Minimum Voltage	9V	9V

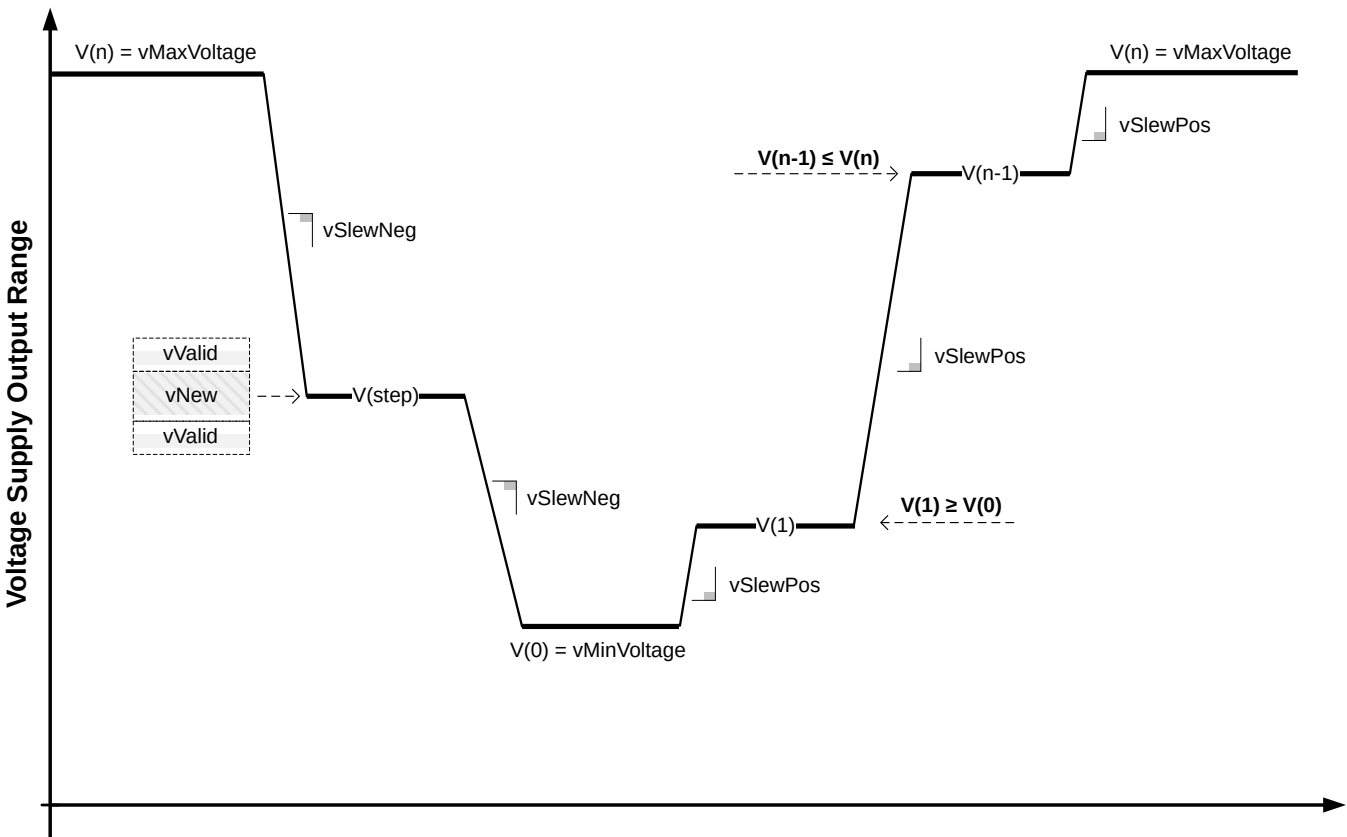
Table 4.4 show the valid voltage ranges for EPR AVS.

Table 4.4. EPR Adjustable Voltage Supply (AVS) Voltage Ranges

AVS Voltage Range	28V	36V	48V
Maximum Voltage	28V	36V	48V
Minimum Voltage	15V	15V	15V

Figure 4.5 illustrates the output voltage behavior of PPS and an AVS Source in response to positive and negative voltage change requests.

Figure 4.5. PPS/AVS Voltage Transitions



While in PPS Mode and in the specific case of CL operation (see [Section 4.1.3.2.2](#)), the voltage might not change to a new requested level, since in this Mode it is the current that is being controlled by the Source (e.g., when the

Sink is directly charging a Battery from VBUS). Note: the Source cannot rely on checking the voltage on VBUS to determine when its power supply is ready to send a [PS_RDY Message](#).

If the Sink negotiates for a new PDO or APDO, then the transition between the current PDO or APDO and the new PDO or APDO **Shall** occur as described in [Section 4.5](#).

[Section 4.1.4](#) lists transitions that are exempt from the [vSlewNeg](#) and [vSlewPos](#) limits.

See [Section 4.1.3.2.5](#) for output voltage ripple limits.

See [Section 4.1.3.2.6](#) for output voltage and current [DNL](#) step adjustments.

4.1.3.2.2. PPS Operation in Current Limit (CL) Mode

The Programmable Power Supply operating in PPS Mode Current Limit (CL) Mode **Shall** observe the following:

- If the Sink attempts to draw current exceeding the Operating Current specified in the RDO, the Source **Shall** limit its output to a level within the tolerance defined by [iPpsCLNew](#).
- The programming step size for the Operating Current is [iPpsCLStep](#), and whenever a new current is requested, the settling time is [tPpsCLProgramSettle](#).
- The Current Limit programmability ranges from [iPpsCLMin](#) and the Maximum Current value in the PPS APDO.
 - A Source which receives a Request for current below [iPpsCLMin](#) **Should** reject the Request.
 - A Source that accepts a Request for current below [iPpsCLMin](#) **Shall** set its Current Limit at 1A.
- A PPS Source that is operating in Current Limit **Shall Not** change its set-point in a manner that exceeds [iPpsCLLoadStepRate](#) or [iPpsCLLoadReleaseRate](#).
- If during CL Mode, the output voltage drops below [vPpsShutdown](#) the Source **May** send a [Hard Reset](#) and **Shall** discharge VBUS to [vSafe0V](#), then resume USB Default Operation at [vSafe5V](#).
- The Source **Shall Not** shut down or otherwise disrupt the available output power while in Current Limit Mode unless another protection mechanism as outlined in [Section 4.1.5](#) is engaged to protect the Source from damage.

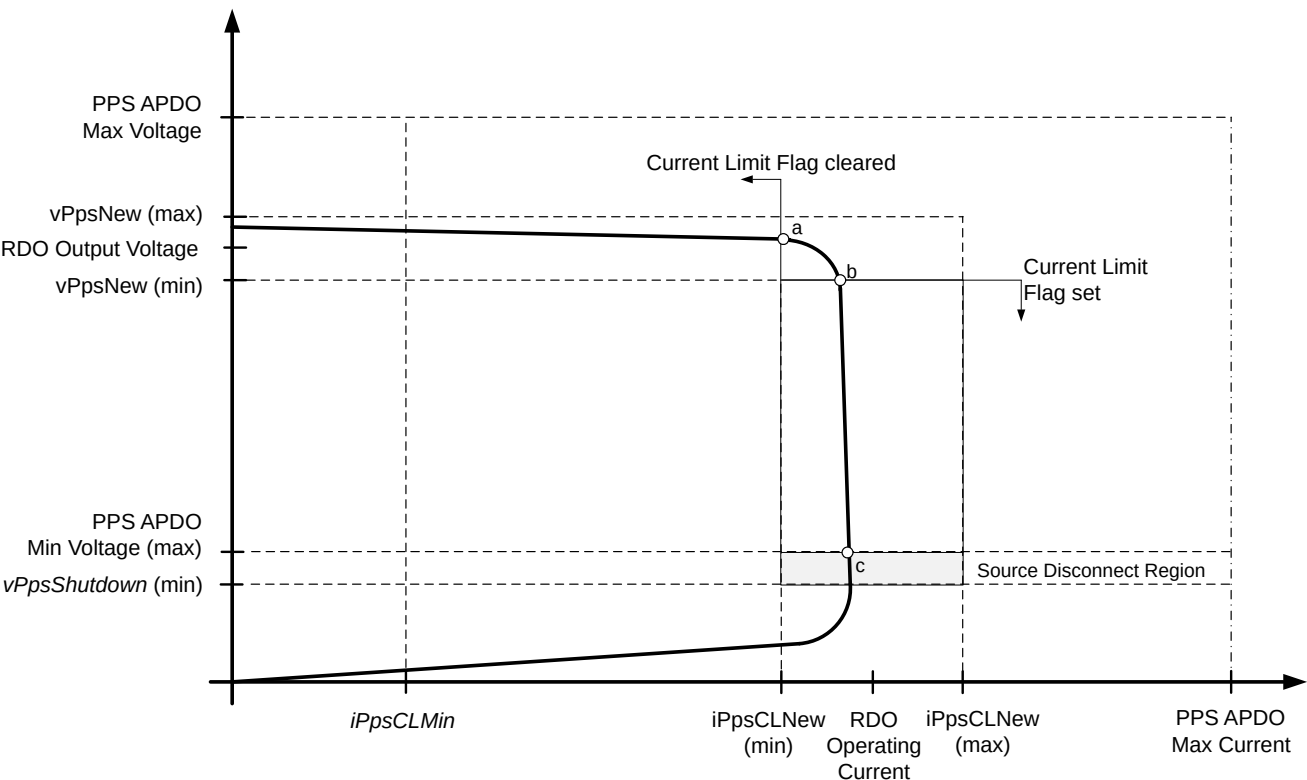
The response of a PPS to a load change depends on the Operating Mode of the PPS and the magnitude of the load change. These dependencies lead to one of four possible responses of a PPS to any load change as shown in. They are differentiated by the value of the PPS Status OMF (see [Section 6.5.13](#)), in the initial and final State of the load change.

Table 4.5. PPS Status

OMF Initial	OMF Final	Operation
0	0	Output Voltage maintained, transient range allowed as shown in insert Figure 4.6
0	1	At the end of the transient, the feedback loop of the Source controls the VBUS Current, as opposed to the voltage, which is determined by the Sink (usually by the Sink's Battery). The Current Shall settle within iPpsCLNew by tPpsCVCLTransient . During the transient itself the Current is allowed to be within iPpsCVCLTransient .
1	0	At the end of the transient, the feedback loop of the Source controls the VBUS Voltage, as opposed to the current. During the transient the voltage Shall stay within vPpsCLCVTransient . After tPpsCLCVTransient the VBUS voltage Shall be within vPpsNew .
1	1	Output Current maintained. The transient range allowed is extended by iPpsCLTransient , and the settling time is tPpsCLSettle . After this period the allowed range is iPpsCLNew .

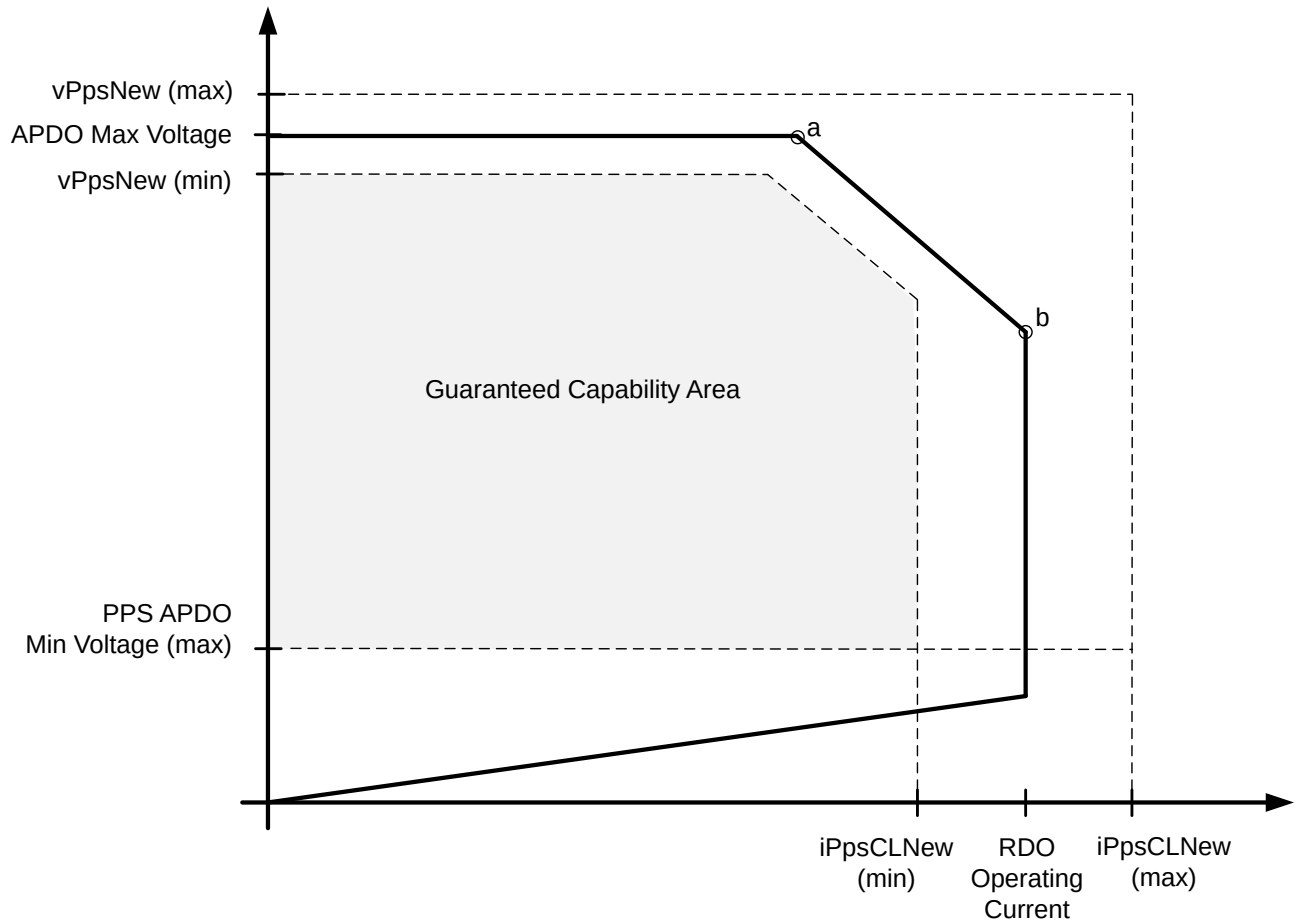
The relationship between PPS programmable output voltage and PPS programmable Current Limit is as shown in. The transition between the Constant Voltage Mode and the Current Limit Mode occurs between points a and b. The PPS Status OMF **Shall** be set or cleared within this region. When VBUS falls below the APDO Minimum Voltage (point c), the Source is allowed to disconnect.

Figure 4.6. PPS Programmable Voltage and Current Limit



4.1.3.2.3. PPS Operation in Constant Power

The tolerances along the Constant Power Curve **Shall Not** extend into the Guaranteed Capability Area shown in [Figure 4.7](#) as the region inside the dashed line defined by PPS APDO Minimum Voltage (max), [vPpsNew](#) (min), and [iPpsCLNew](#) (min).

Figure 4.7. PPS Constant Power

In the example above, the section between points a and b represents the constant power area of the CV/CL curve.

4.1.3.2.4. Adjustable Voltage Supply Source Current

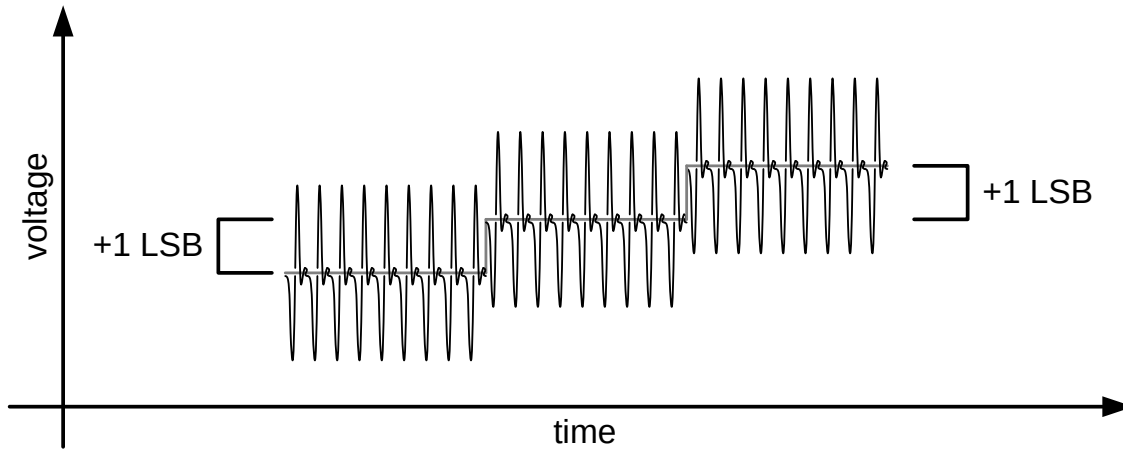
Unless otherwise noted, AVS **Shall** follow the same requirements as a Fixed Supply.

The maximum operating current the AVS shall supply is determined as follows:

- For SPR AVS APDOs, the maximum operating current is defined in the Maximum Current field of the 15V AVS APDO and the 20V AVS APDO in the SPR [Source_Capabilities Message](#).
- For EPR AVS APDOs, the maximum operating current is calculated as the lower of the ((PDP field value) ÷ (Output Voltage)) or 5A whichever is lower. See [Table 3.5](#).

4.1.3.2.5. Source AVS/PPS Ripple

The AVS/PPS output voltage ripple is expected to exceed the magnitude of one or more LSB as shown in the [Figure 4.8](#).

Figure 4.8. Expected PPS Ripple Relative to an LSB

4.1.3.2.6. Source DNL Tolerance

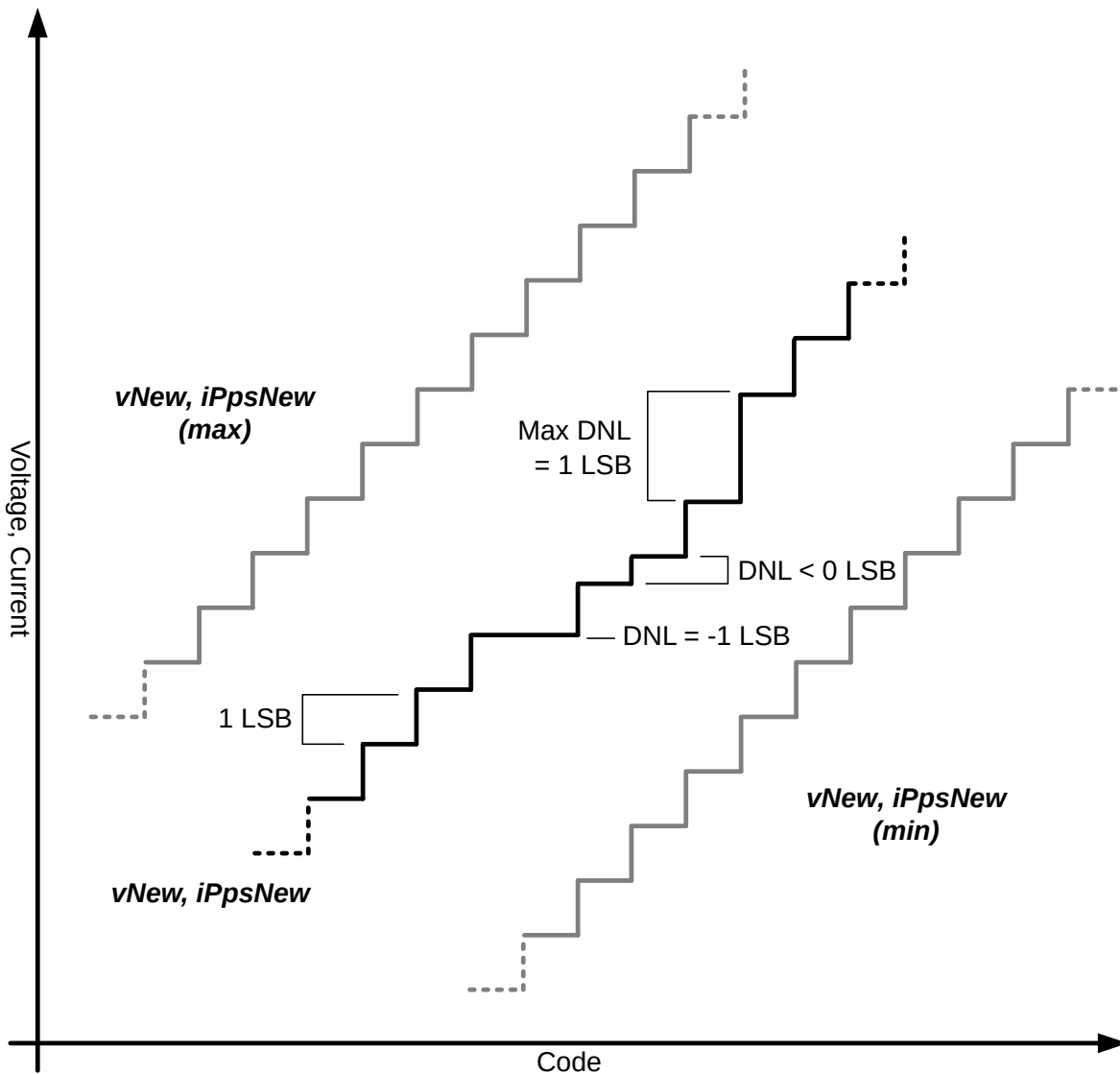
The PPS voltage and current discrete LSB steps have a [DNL](#) tolerance as shown in [Figure 4.9](#). In absolute terms the step size of the LSB for both voltage and current is defined by [vPpsStep](#) / [vAvsStep](#) for voltage and [iPpsCLStep](#) for current. Several examples of **Valid** LSB steps are shown in [Figure 4.9](#):

- The upper end of the [DNL](#) error (+1 LSB) shows the case where one step is effectively skipped.
- The lower end of the [DNL](#) error (-1 LSB) shows the case where the voltage or current set-point remained the same.

The ideal [DNL](#) is 0 LSB where the voltage or current step is exactly equal to the defined LSB step size.

The intent of [DNL](#) is to guarantee that changes to the voltage/current have the correct directionality, and that the maximum step size is clearly defined.

Note: The Source **Should** avoid scenarios where multiple consecutive steps have errors close to the Maximum and Minimum [DNL](#).

Figure 4.9. Allowed DNL Errors and Tolerance of Voltage and Current in AVS/PPS Mode

4.1.4. Non-application of VBUS Slew Rate Limits

Scenarios where [vSrcSlewPos](#) and [vPpsSlewPos](#) VBUS slew rate limits do not apply and VBUS **May** transition faster than specified are as follows:

- When first applying VBUS after an Attach.
- When applying VBUS as part of a [Power Role Swap](#) to Source Power Role.
- When increasing VBUS from [vSafe0V](#) to [vSafe5V](#) during a [Hard Reset](#).
- During a [Fast Role Swap](#) when the Initial Sink applies VBUS.
- When discharging VBUS to [vSafe0V](#) during a [Hard Reset](#).

- When discharging VBUS to [vSafe0V](#) as part of a [Power Role Swap](#) to Sink Power Role.
- When discharging VBUS to [vSafe0V](#) after a Detach.
- During a [Fast Role Swap](#) when the VBUS power Source Connected to the Hub UFP stops sourcing power.

4.1.5. Robust Source Operation

4.1.5.1. Output Over-Current Protection

A Source **Shall** implement Over-Current Protection (OCP) in accordance with the applicable safety standards. The over-current protection **May** be dynamic depending on the Contract established with the Sink, but it **Shall Not** activate if the current being drawn is valid per the current Contract.

A Source **Should** attempt to send [Hard Reset](#) Signaling when OCP engages followed by an [Alert Message](#) indicating an OCP event when an Explicit Contract has been established. The over-current protection response **May** engage at either the Port or system level. Systems or ports that have engaged over-current protection **Should** attempt to resume USB Default Operation.

The Source **Shall** renegotiate with the Sink after choosing to resume USB Default Operation. The decision of how to renegotiate after an over-current event is left to the discretion of the Source implementation.

During the over-current response and subsequent system or Port shutdown, all affected Source ports operating with VBUS greater than [vSafe5V](#) **Shall** discharge VBUS to [vSafe5V](#) within [tSafe5V](#) and [vSafe0V](#) within [tSafe0V](#).

4.1.5.2. Output Over-Voltage Protection

Over-voltage protection (OVP) is left to the Source implementation to meet any applicable safety or reliability standards. Any OVP **Shall** account for [vNew](#) and [vValid](#) when establishing the threshold for OVP, irrespective of the type of Contract that is established with the Sink.

4.1.5.3. Over-Temperature Protection

A Source **Should** implement Over-Temperature Protection (OTP) to prevent damage from temperature that exceeds the thermal capability of the Source. The definition of thermal capability and the monitoring locations used to trigger the OTP are left to the discretion of the Source implementation.

In order to avoid reaching an OTP event, a Source **May** proactively reduce the available power being offered to the Sink, even though this might be lower than the Source would be expected to offer during normal thermal operating conditions. Prior to reducing power, the Source **Should** generate an [Alert Message](#) indicating an Operating Condition Change and set the Temperature Status bit in the SOP [Status Message](#) to Warning (10b).

A Source **Should** attempt to send [Hard Reset](#) Signaling when OTP engages followed by an [Alert Message](#) indicating an OTP event once an Explicit Contract has been established. The OTP response **May** engage at either the Port or system level. Systems or ports that have engaged OTP **Should** attempt to resume USB Default Operation and **May** latch off to protect the Port or system.

The Source **Shall** renegotiate with the Sink after choosing to resume USB Default Operation. The decision of how to renegotiate after an over-temperature event is left to the discretion of the Source implementation.

During the OTP and subsequent system or Port shutdown, all affected Source ports operating with VBUS greater than [vSafe5V](#) **Shall** discharge VBUS to [vSafe5V](#) within [tSafe5V](#) and [vSafe0V](#) within [tSafe0V](#).

4.1.5.4. vSafe5V Externally Applied to Ports Supplying vSafe5V

A Power Delivery Source **Shall** be tolerant of [vSafe5V](#) being present on VBUS when simultaneously applying power to VBUS. Normal USB PD communication **Shall** be supported when this [vSafe5V](#) to [vSafe5V](#) connection exists.

4.1.5.5. Detach

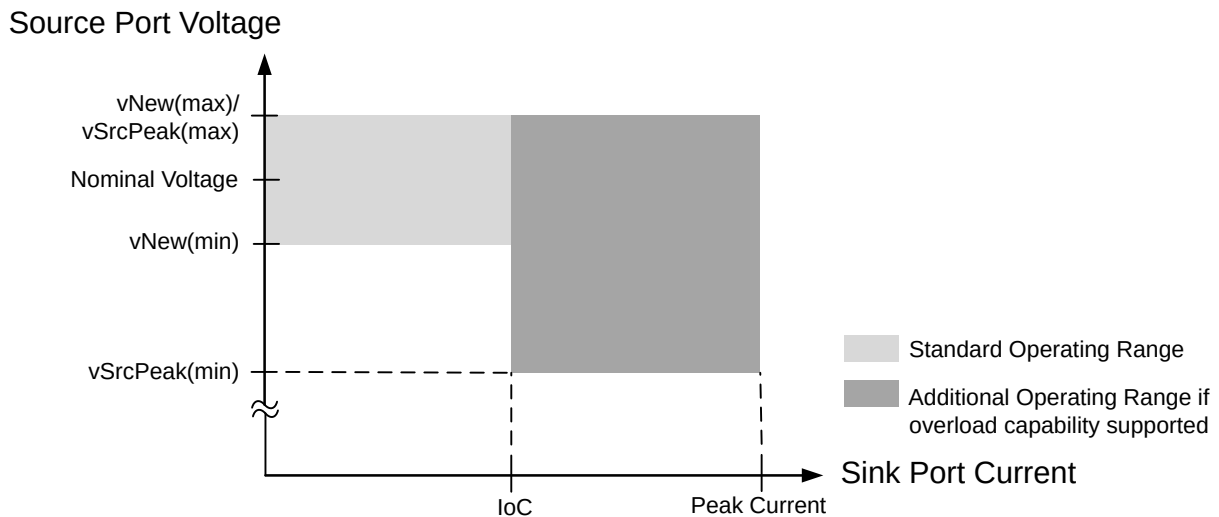
A USB Detach is detected electrically using CC detection on the USB Type-C connector. When the Source is Detached, the Source **Shall** transition to [vSafe0V](#) within [tSafe0V](#) from when the Detach event occurred. During the transition to [vSafe0V](#), the VBUS voltage **Shall** be below [vSafe5V](#) (max) within [tSafe5V](#) from when the Detach event occurred and **Shall Not** exceed [vSafe5V](#) max after this time.

Note: A USB-PD transmission by the Source during a disconnect event will delay disconnect detection by the Source.

4.1.6. Source Peak Current Operation

A Source that has the Fixed Supply PDO or AVS APDO Peak Current bits set **Shall** be designed to support one of the overload Capabilities defined in [Table 6.18](#) respectively. The overload conditions are bound in magnitude, duration and duty cycle. When overload conditions occur, the Source is allowed the range of [vSrcPeak](#) (instead of [vNew](#)) relative to the nominal value. See [Figure 4.10](#). Note: A Source operating in PPS Mode cannot support peak currents, due to the CL requirements.

Figure 4.10. Source Peak Current Overload



Each overload period **Shall** be followed by a period of reduced current draw such that the rolling average current over the Overload Period field value with the specified Duty Cycle field value (see [Section 6.4.1.3.13](#)) **Shall Not** exceed the Negotiated current. This is calculated as:

$$\text{Period of reduced current} = (1 - (\text{value in Duty Cycle field} \div 100)) \times \text{value in Overload Period field}$$

The Source **May** send a New [Source_Capabilities Message](#) with the Fixed Supply PDO or AVS APDO Peak Current bits set to 00b to prohibit overload operation even if an overload capability was previously Negotiated with the Sink.

4.1.7. Source Capabilities Extended Parameters

Implementers can choose to make available certain characteristics of a Source Port as a set of **Static** and/or dynamic parameters to improve interoperability between external power sources and portable computing devices. The complete list of reportable **Static** parameters is described in full in. The following section offers additional details on some specific electrical parameters.

4.1.7.1. Load Step Slew Rate

The default load step slew rate is established at 150mA/μs. A Source **Shall** meet the following requirements under the load step reported in the [Source_Capabilities_Extended Message](#):

- The Source **Shall** maintain VBUS regulation within the [vValid](#) range.
- The noise on the CC line **Shall** remain below [vNoiseIdle](#) and [vNoiseActive](#).

Test conditions require a change in both positive and negative load steps from 1Hz to 5000Hz, up to the Advertised Load Step Magnitude of the full load output including from both 10 mA and 10% initial load. The Source **Shall** ensure that PD Communications meet the transmit and receive masks as specified in [Section 5.3.4.1](#) and [Section 5.3.4.2](#) under all load conditions.

4.1.7.2. Load Step Magnitude

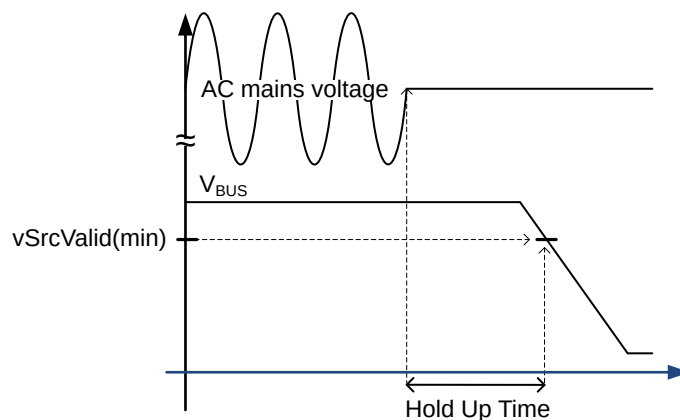
The default load step magnitude rate **Shall** be 25% of IoC. The Source **May** report higher capability, tolerating a load step of 90% of IoC.

4.1.7.3. Holdup Time Field

The Holdup Time field **Shall** return a numeric value of the number of milliseconds the output voltage stays in regulation upon a short interruption of the AC Supply.

An AC Supplied Source **Shall** report its holdup time in this field. The holdup time is measured with the load at rated maximum, with the AC Supply at 115VAC rms and 60Hz (or at 230VAC rms and 50Hz for a Source that does not support 115VAC AC Supply). The reported time describes the minimum length of time from the last completed AC Supply input cycle (zero-degree phase angle) until when the output voltage decays below [vSrcValid](#) (min). A Source is recommended to support a minimum of 3ms and is preferred to support over 10 milliseconds holdup time (equivalent to a half cycle drop from the AC Supply). See [Figure 4.11](#).

Figure 4.11. Holdup Time Measurement



4.1.7.4. Compliance Field

An SPR Source claiming LPS, PS1 or PS2 compliance (see [IEC 60950-1] and [IEC 62368-1]) **Shall** report its Capabilities in the Compliance field.

4.1.7.5. Batteries

The Number of Batteries/Battery Slots field **Shall** report the number of Batteries the Source supports. The Source **Shall** independently report the number of Hot Swappable Batteries and the number of Fixed Batteries.

4.2. Sink Requirements

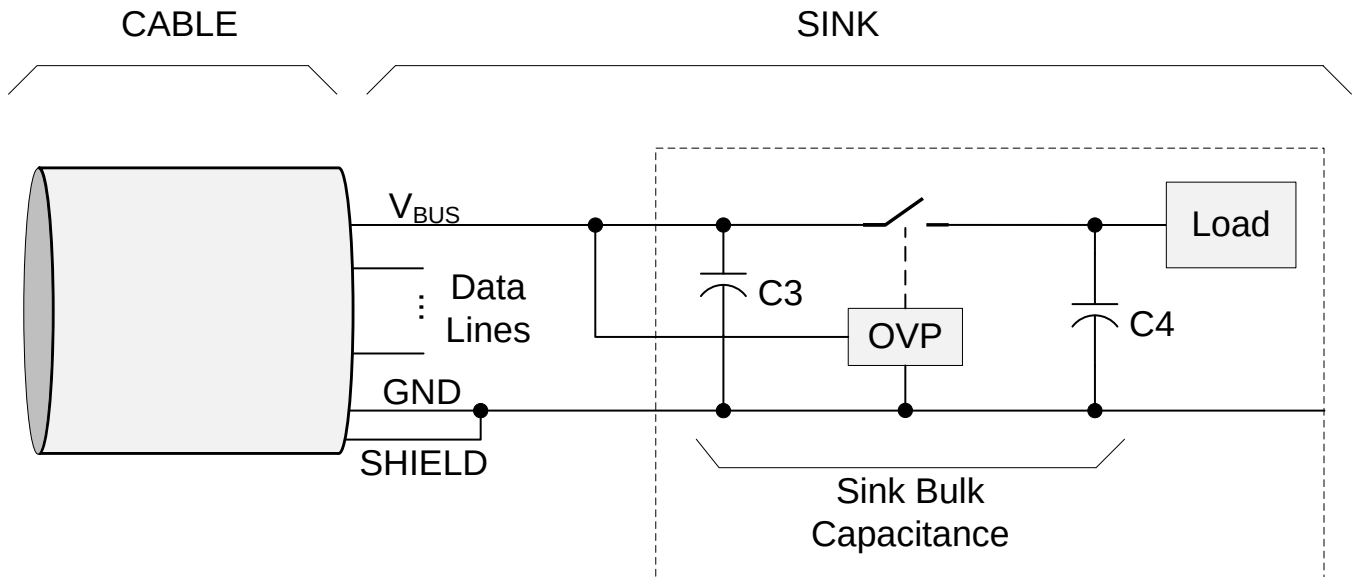
4.2.1. Behavioral Aspects

A USB PD Sink is a Device that receives power over a USB connection using the USB Power Delivery protocol. It monitors available power from the Source, requests specific voltage and current levels, and manages its power consumption based on the Negotiated Contract. In addition to being a USB Type-C Sink and following all [USB-C] defined behaviors, a USB PD Sink must meet all requirements defined in this chapter.

4.2.2. Sink Bulk Capacitance

The Sink bulk capacitance consists of C3 and C4 as shown in [Figure 4.12](#). The Ohmic Interconnect might consist of PCB traces for power distribution or power switching devices. The Ohmic Interconnect is expected to be part of an input Over-Voltage Protection (Sink OVP) circuit implemented by the Sink as described in [Section 4.1.5.2](#) to protect against excessive VBUS input voltage. A Sink **shall** implement OVP and **shall not** rely on the Source output voltage limit for its input OVP. The capacitance might be a single capacitor, a capacitor bank or distributed capacitance.

Figure 4.12. Placement of Sink Bulk Capacitance



The total Sink bulk capacitance Connected to VBUS **shall** not exceed [cSnkBulk](#) or [cSnkBulkPd](#) limits. The capacitance value **may** be changed at any time, provided the change:

- does not cause a transient current on VBUS that violates the maximum allowed current of the present Contract;
- does not cause [iSnkStdby](#) to be exceeded during positive load transitions;
- Guarantees the rate of change of currents stays below [iLoadStepRate](#).

Capacitance beyond the [cSnkBulk](#) or [cSnkBulkPd](#) limit is permitted provided it is isolated from VBUS through current-limiting circuitry as described in [USB3]. The Sink **shall** discharge this additional capacitance and **shall** remove it in the event of a disconnect, [Hard Reset](#), [Fast Role Swap](#), or [Power Role Swap](#). Since the additional capacitance might create a slow VBUS discharge on a disconnect, care must be taken to meet the requirements detailed in [Section 4.2.9.1](#) and in [USB-C] to detect a disconnect event.

During a [Power Role Swap](#), the Default Sink **shall** transition to Swap Standby before operating as the New Source.

Any change in bulk capacitance required to complete the [Power Role Swap](#) **Shall** occur during Swap Standby.

4.2.3. Sink Standby

The Sink **Shall** transition to Sink Standby before these transitions:

- a positive voltage transition of any Fixed Supply PDO
- a positive voltage transition larger than [vSmallStep](#) while in AVS Mode
- a positive voltage transition larger than [vSmallStep](#) while in PPS Mode.

A Sink is not required to transition to Sink Standby for these transitions:

- when operating within the Negotiated PPS APDO
- negative voltage transitions (ramp rate limited by [vSlewNeg](#))
- PPS voltage transition smaller or equal to [vSmallStep](#)
- AVS voltage transition smaller or equal to [vSmallStep](#).

During Sink Standby, the Sink **Shall** reduce the current drawn to [iSnkStdbY](#). This allows the Source to manage the voltage transition as well as supply sufficient operating current to the Sink to maintain PD operation during the transition. The Sink **Shall** complete this transition to Sink Standby within [tSnkStdbY](#) after evaluating the [Accept Message](#) from the Source. The transition when returning to Sink operation from Sink Standby **Shall** be completed within [tSnkNewPower](#). See [Section 4.5](#) for details.

4.2.4. Suspend Power Consumption

When the Source has set its USB Suspend Supported flag (see [Section 6.4.2.1.8](#)), a Sink **Shall** go to the lowest power State during USB suspend. The lowest power State **Shall** be [pSnkSusp](#) or lower for a PDUSB Peripheral and [pHubSusp](#) or lower for a PDUSB Hub. There is no requirement for the Source voltage to be changed during USB suspend.

4.2.5. Zero Negotiated Power

When a Sink requests zero current as part of a power Negotiation with a Source, the Sink **Shall** go to the lowest power State ([pSnkSusp](#) or lower) where it can still communicate using PD Signaling.

4.2.6. Transient Load Behavior

The rate of change of any shift in Sink load current during normal operation **Shall Not** exceed [iSlewPos](#) (for load steps) and [iSlewNeg](#) (for load releases) as measured at the Sink receptacle.

The Sink's operating current **Shall Not** change faster than the value reported in the Source's Voltage Regulation bit field (see [Table 6.50](#)) and **Shall** ensure that PD Communications meet the transmit and receive masks as specified in [Section 5.3.4.1](#) and.

4.2.7. Considerations for Current Limit in PPS Mode

Prior to operating the PPS in Current Limit, the Sink **Shall** program the PPS Operating Voltage to the lowest practical level that satisfies the Sink load requirement. Doing so will minimize the inrush current that occurs when the transition to Current Limit occurs.

When requesting a new Current Limit, the Sink **Shall Not** Request a magnitude change that exceeds [iPpsCLLoad-Step](#).

4.2.8. Sink Peak Current Operation

Sinks **Shall** only make use of a Source overload capability as indicated in the Fixed Supply PDO or AVS APDO Peak Current (see [Section 6.4.1.3.13](#)). Sinks **Shall** manage thermal aspects of the overload event by not exceeding the average Negotiated output of a Fixed Supply or AVS that supports Peak Current operation.

Sinks that depend on the Peak Current capability for enhanced system performance **Shall** also function correctly when Attached to a Source that does not offer the Peak Current capability or when the Peak Current capability has been inhibited by the Source.

4.2.9. Robust Sink Operation

4.2.9.1. Sink Bulk Capacitance Discharge at Detach

When a Sink is Detached from a Source, the Sink **Shall** continue to draw power from its input bulk capacitance. VBUS **Shall** discharge to [vSafe5V](#) or below within [tSafe5V](#) of the Detach event. This safe Sink requirement **Shall** apply to all Sinks operating with a Negotiated VBUS level greater than [vSafe5V](#) and **Shall** apply during all low power and high-power operating modes of the Sink.

If the Detach is detected during a Sink low power State, such as USB Suspend, the Sink can then draw as much power as needed from its bulk capacitance since a Source is no longer Attached. In order to achieve a successful Detach detect based on VBUS voltage level droop, the Sink power consumption **Shall** be high enough so that VBUS will decay below [vSrcValid](#) (min) such that the disconnect is detected and the Sink VBUS pin is discharged to [vSafe5V](#) within [tSafe5V](#) after the Source bulk capacitance is removed due to the Detach. Once adequate VBUS droop has been achieved, a discharge circuit can be enabled to meet the safe Sink requirement.

To illustrate the point, the following set of Sink conditions will not meet the safe Sink requirement without additional discharge circuitry:

- Negotiated VBUS = 20V.
- Maximum allowable supplied VBUS voltage = 21.55V.
- Maximum bulk capacitance = 30μF.
- Power consumption at Detach = 12.5mW.

When the Detach occurs (hence removal of the Source bulk capacitance), the 12.5mW power consumption will draw down the VBUS voltage from the worst-case maximum level of 21.55V to 17V in approximately 205ms. At this point, with VBUS well below [vSrcValid](#) (min) an approximate 100mW discharge circuit can be enabled to increase the rate of Sink bulk capacitance discharge and meet the safe Sink requirement. The power level of the discharge circuit is dependent on how much time is left to discharge the remaining voltage on the Sink bulk capacitance. If a Sink has the ability to detect the Detach in a different manner and in much less time than [tSafe5V](#), then this different manner of detection can be used to enable a discharge circuit, allowing even lower power dissipation during low power modes such as USB Suspend.

In most applications, the safe Sink requirement will limit the maximum Sink bulk capacitance well below the [cSnkBulkPd](#) limit. A Detach occurring during Sink high power operating modes must quickly discharge the Sink bulk capacitance to [vSafe5V](#) or lower as long as the Sink continues to draw adequate power until VBUS has decayed to [vSafe5V](#) or lower.

4.2.9.2. Sink Over-Current Protection

A Sink **May** be required to implement its own internal current protection mechanism to protect against internal and external VBUS current faults in accordance with the applicable safety standards.

4.2.9.3. Sink Over-Voltage Protection

A Sink **Shall** implement input Over-Voltage Protection (OVP) to prevent damage from input voltage that exceeds the voltage handling capability of the Sink, in accordance with applicable safety standards. The definition of voltage handling capability is left to the discretion of the Sink implementation. The over-voltage response of a Sink **Shall Not** interfere with normal PD operation and **Shall** account for [vNew](#), or [vValid](#) as determined by the Negotiated VBUS value. An SPR Sink **Should** tolerate input voltages as high as [vSprMax](#). All Sinks **Shall** meet applicable safety requirements.

A Sink **Should** attempt to send [Hard Reset](#) Signaling when OVP engages followed by an [Alert Message](#) indicating an OVP event once an Explicit Contract has been established. The OVP response **May** engage at either the Port or system level. Systems or ports that have engaged OVP **Shall** resume USB Default Operation when the Source has re-established [vSafe5V](#) on VBUS. The Sink **Shall** be able to renegotiate with the Source after resuming USB Default Operation.

The Sink **Should** prevent continual system or Port cycling if OVP continues to engage after initially resuming either USB Default Operation or renegotiation. Latching off the Port or system is an acceptable response to recurring over-voltage.

4.2.9.4. Sink Over-Temperature Protection

A Sink **Shall** implement Over-Temperature Protection (OTP) in order to comply with applicable safety standards. The definition of thermal capability and the monitoring locations used to trigger the over-temperature protection are left to the discretion of the Sink implementation.

4.3. Dual Role Ports

A Dual-Role Power Port in either the Source role or Sink role **Shall** support Swap Standby, which is required for a [Power Role Swap](#). Note: During a [Power Role Swap](#), the USB connection **Shall Not** reset even though [vSafe5V](#) is no longer present on VBUS.

4.3.1. Swap Standby for the Initial Source

Swap Standby starts for the Source after the Source power supply has discharged the bulk capacitance on VBUS to [vSafe0V](#) as part of the [Power Role Swap](#) transition. It ends after the New Source sends a [PS_RDY Message](#).

While in Swap Standby:

- The Source **Shall Not** drive VBUS, which is therefore expected to remain at [vSafe0V](#).
- The New Sink **Shall Not** draw more than [iSnkSwapStdby](#) as the New Source turns on VBUS.
- The Dual-Role Power Port **Shall** be configured as a Sink.
- The [PS_RDY Message](#) associated with the Source being in Swap Standby **Shall** be sent after the VBUS drive is removed.
- The transition time from Swap Standby to being the New Sink **Shall** be no more than [tNewSnk](#).

4.3.2. Swap Standby for the Initial Sink

Swap Standby starts for the Sink after evaluating the [Accept Message](#) from the Source or sending an [Accept Message](#) to the Source during a [Power Role Swap](#). It ends once the New Source reaches [vSafe5V](#). While in Swap Standby:

- The Sink's current draw **Shall Not** exceed [iSnkSwapStdby](#) from VBUS.

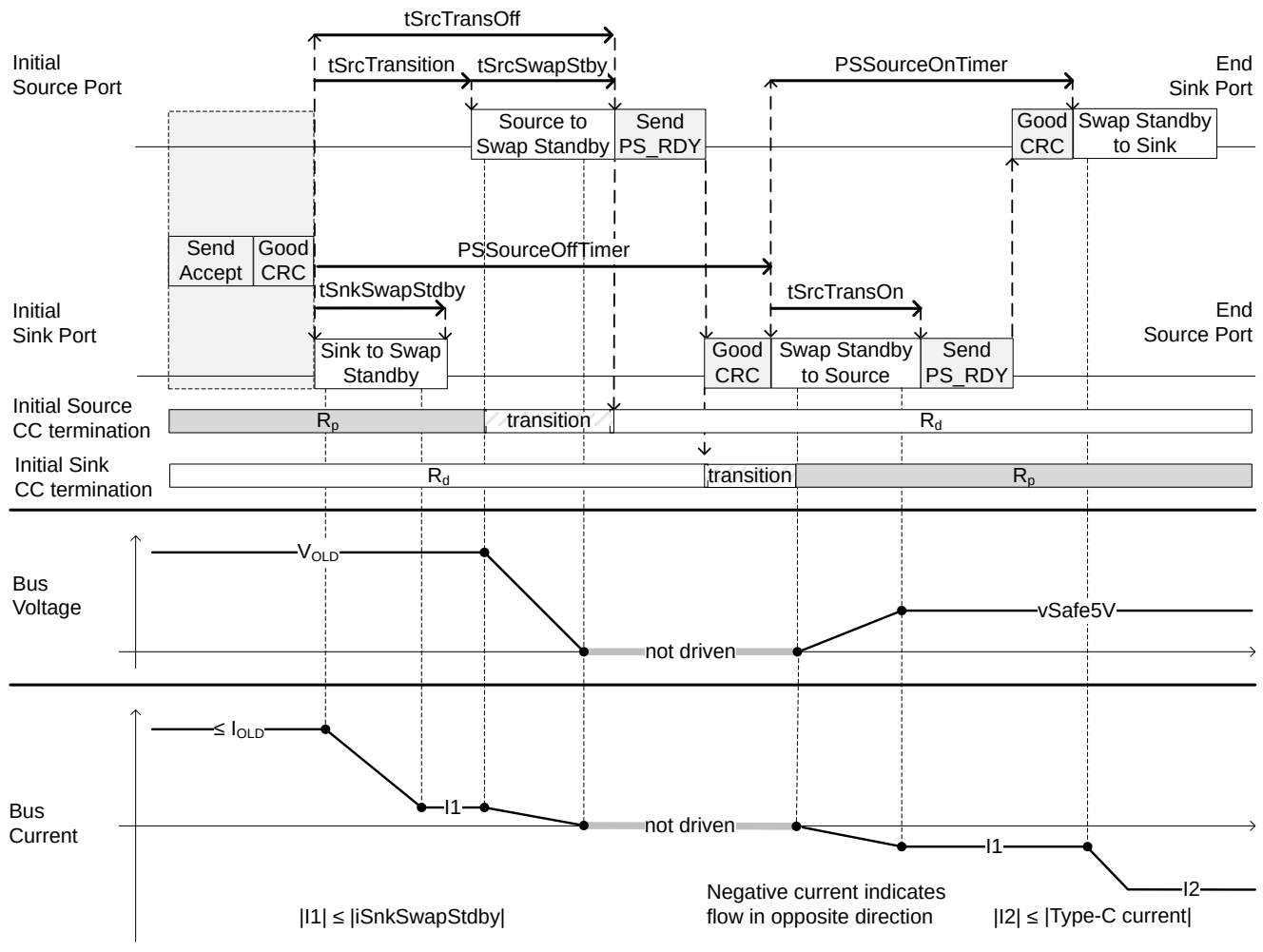
- The Dual-Role Power Port **Shall** be configured as a Source after VBUS has been discharged to [vSafe0V](#) by the existing Initial Source.
- The time for the Sink to transition to Swap Standby **Shall** be no more than [tSnkSwapStdby](#).
- When in Swap Standby, the Sink has relinquished its Power Role as Sink and will prepare to become the New Source. The transition time from Swap Standby to New Source **Shall** be no more than [tNewSrc](#).

4.3.3. Power Role Swap Sequence

In [Figure 4.13](#), a [PR_Swap Message](#) has been sent by either the Sink or the Source. See [Chapter 10](#) for a [Fast Role Swap](#) (FRS) Sequence.

The [Power Role Swap](#) will follow the same sequence irrespective of which Port partner makes the Request:

- Once the Port Partner accepts the Request, and a [GoodCRC Message](#) is sent in return, the PSSourceOff Timer is started.
- The Sink has [tSnkStdby](#) to reduce the current consumption to [iSnkSwapStdby](#).
- Once [tSrcTransition](#) elapses, the Source has [tSrcSwapStdby](#) to discharge VBUS.
- The Initial Source **Shall** transition the termination from Rp to Rd before sending [PS_RDY Message](#).
- After the Initial Source sends a [PS_RDY Message](#) it initiates the PSSourceOn Timer.
- The New Source has [tSrcTransOn](#) to turn on VBUS and transition to [vSafe5V](#). During this time the New Sink **Shall** limit its current to [iSnkSwapStdby](#).
- Once the [PS_RDY Message](#) is sent by the New Source, the Sink can start taking the Type-C Current defined by Rp. New Contract negotiations can now take place.

Figure 4.13. Transition Diagram for a Power Role Swap

4.4. Response to Hard Resets

Hard Reset Signaling indicates a communication failure has occurred. The Source is subject to the following requirements:

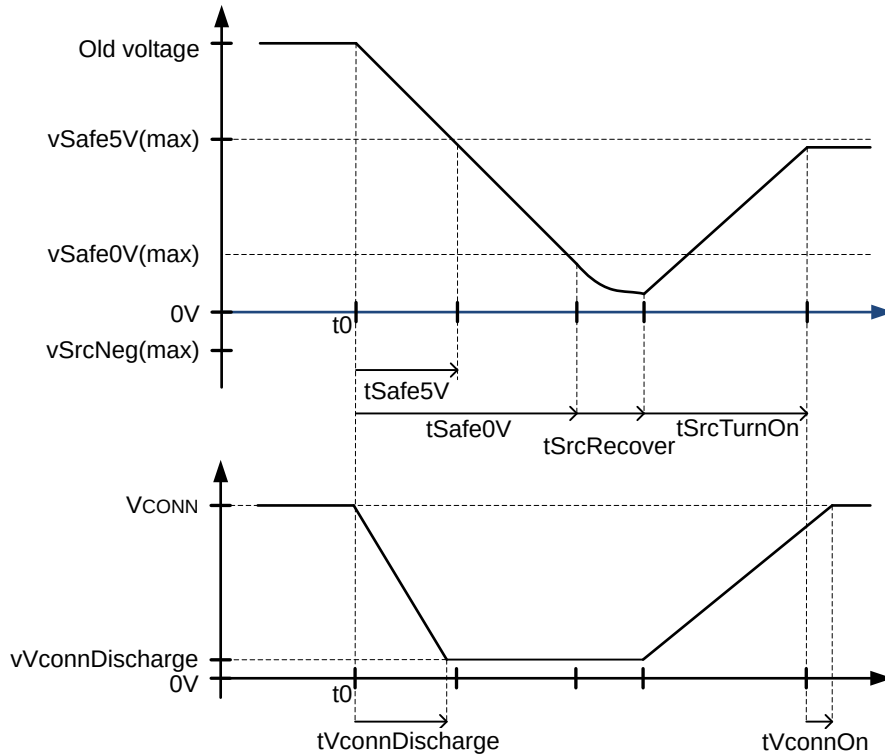
- **Shall** drive VBUS to [vSafe0V](#).
- These actions start [tPSHardReset](#) after the last bit of the [Hard Reset](#) Signaling has been received from the Sink or sent by the Source. The Source **Shall** take into consideration that the Sink **May** have [cSnkBulkPd](#) present until VBUS drops below [vSafe0V](#).
- The Source **Shall** meet both [tSafe5V](#) and [tSafe0V](#) relative to the start of the voltage transition.
- After establishing the [vSafe0V](#) voltage condition on VBUS, the Source **Shall** wait [tSrcRecover](#) before re-applying VCONN and restoring VBUS to [vSafe5V](#).

The Port Partner that drives VCONN is subject to the following requirements:

- **Shall** stop driving VCONN.

- If the Source was driving VCONN, then R_p **shall** be removed from the CC line.
- If the Sink was driving VCONN, R_d **shall** be added to the CC line.
- VCONN will meet t_{VCONNDischarge} relative to the start of the voltage transition as shown in [Figure 4.14](#) due to the discharge circuitry in the CablePlug. VCONN **shall** meet t_{VCONNOn} relative to VBUS reaching [vSafe5V](#).
- t_{VCONNOn} and t_{VCONNDischarge} are defined in [USB-C].

Figure 4.14. Source VBUS and VCONN Response to Hard Reset



The USB connection **May** reset during a [Hard Reset](#) since the VBUS voltage will be less than [vSafe5V](#) for an extended period of time.

Device operation during and after a [Hard Reset](#) is defined as follows:

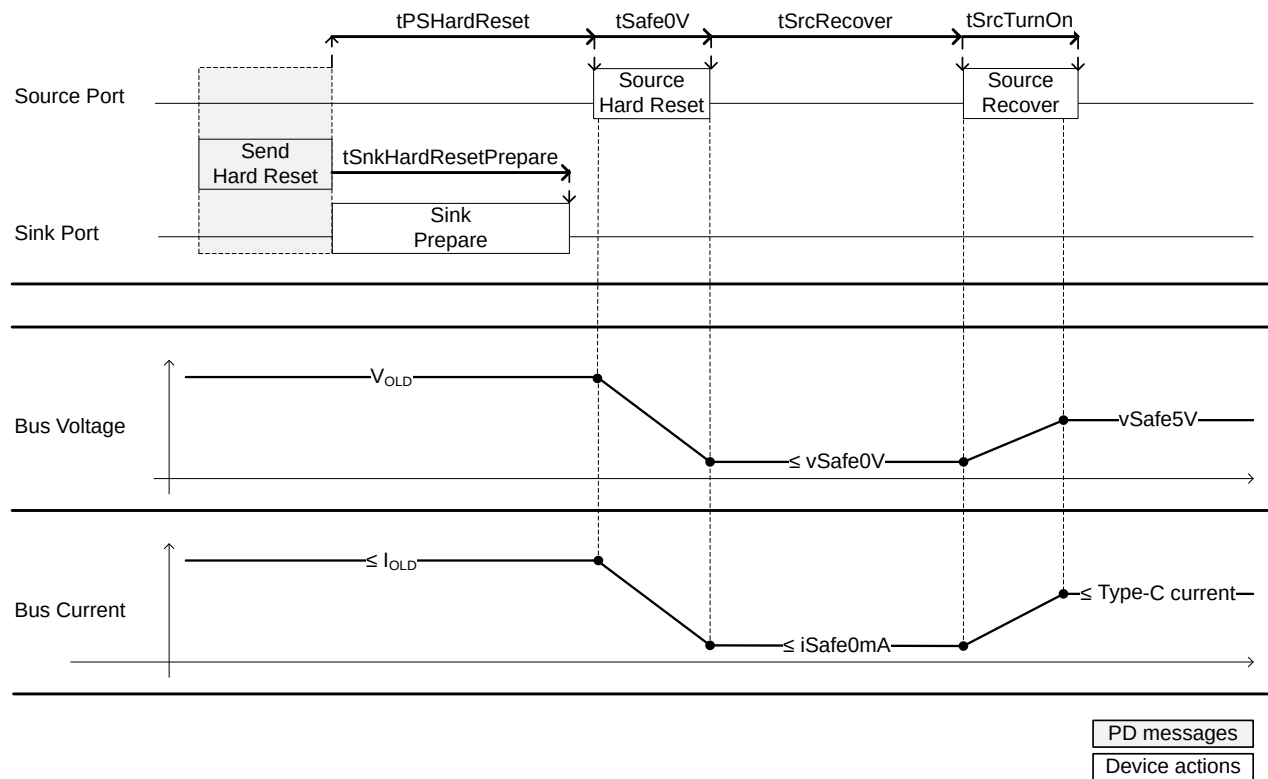
- Self-powered devices **Should Not** disconnect from USB during a Hard Reset.
- Self-powered devices operating at more than [vSafe5V](#) **May Not** maintain full functionality after a [Hard Reset](#).
- Bus powered devices will disconnect from USB during a [Hard Reset](#) due to the loss of their power Source.

4.4.1. Hard Reset Sequence

The [Hard Reset](#) transition diagram is shown in. The sequence is the same irrespective of which Port Partner started it.

- Once the [Hard Reset](#) is initiated t_{PSHardReset} starts. The Sink has t_{SnkHardResetPrepare}, where the current Contract is still guaranteed.

- Once [tPSHardReset](#) elapses, the Source has [tSafe0V](#) to drive the VBUS voltage to [vSafe0V](#) or lower. At this point the Sink **Shall Not** draw more than [iSafe0mA](#).
- After [tSrcRecover](#) the Source applies power to VBUS. The transition to [vSafe5V](#) **Shall** occur within [tSrcTurnOn](#), and initially the current will be limited to Type-C Current until a new explicit Contract is established.

Figure 4.15. Transition diagram for a Hard Reset

4.5. Transitions

The following sections illustrate the power supply's response to common Negotiation events, including transitions initiated by a [Request Message](#), as well as Power Role Swaps and [Hard Reset](#) scenarios.

4.5.1. Fixed and AVS transitions

The examples of PDO or APDO changes below illustrate the most common voltage and current transition requests. In all the following examples, the Sink has previously sent a [Request Message](#) to the Source. The figures group the common cases; each title states the applicable voltage/current change. These examples illustrate typical successful transitions but are not to scale.

The timing parameters that **Shall** be followed are listed in [Table 4.6](#), [Table 4.7](#), and [Table 4.8](#).

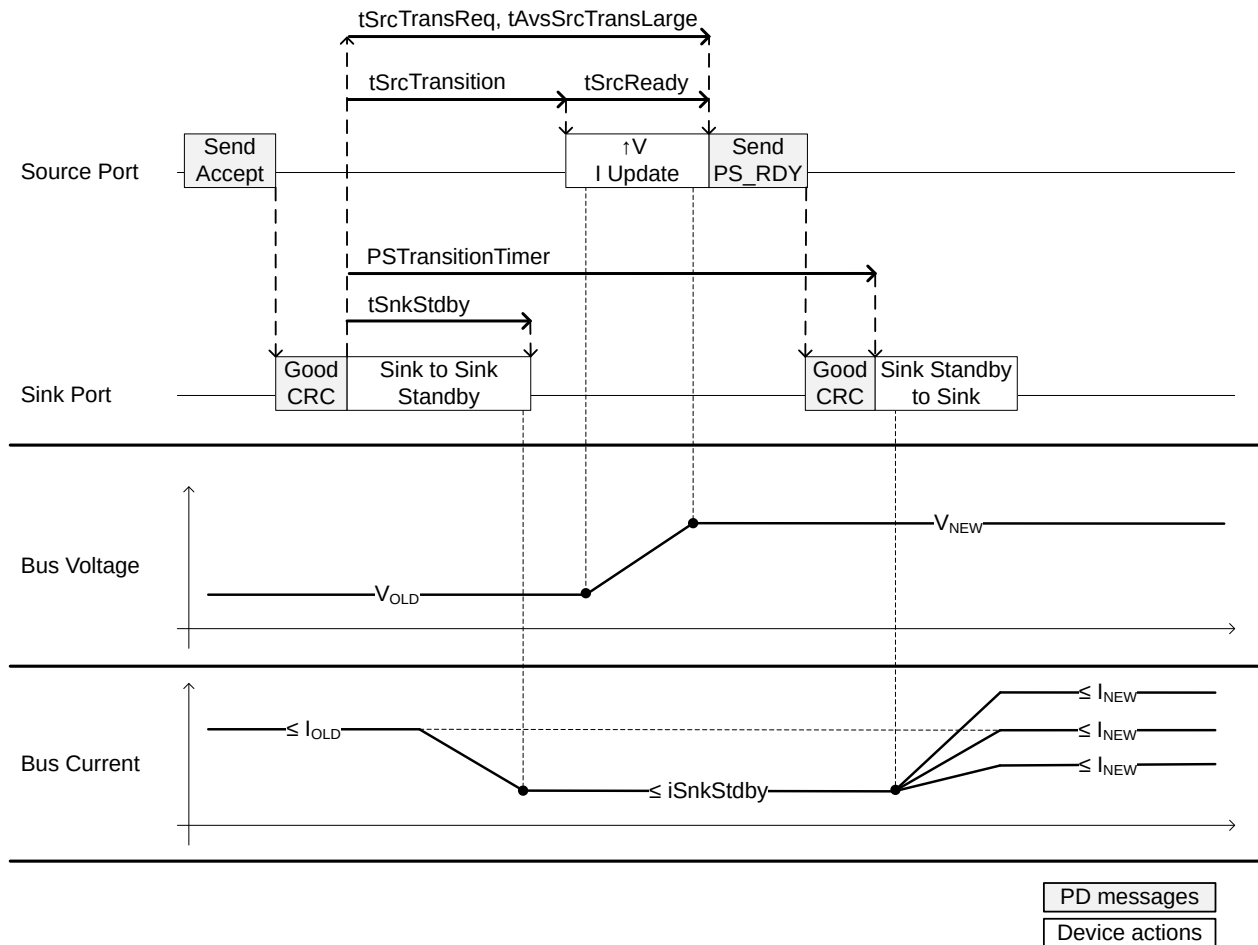
4.5.1.1. Large-step voltage increases (Fixed or AVS)

This diagram applies to fixed voltage transitions and AVS transitions where the voltage is increasing and the voltage change $> v_{SmallStep}$. In these cases, the Sink **Shall** reduce the VBUS current to [iSnkStdby](#) as specified below.

The sequence for these types of transitions is as follows:

- The Source Policy Engine sends an [Accept Message](#).
- The [PSTransitionTimer](#) starts immediately after the Sink completes sending the [GoodCRC Message](#).
- [tSrcTransition](#), [tSnkStdby](#), and [tSrcTransReq](#) / [tAvsSrcTransLarge](#) also start at this point.
- The Sink **Should** reduce the current to no more than [iSnkStdby](#) within [tSnkStdby](#). It **Shall** be reduced to [iSnkStdby](#) before the end of the minimum value of [tSrcTransition](#) expires.
- Once [tSrcTransition](#) elapses, the Source **May** start increasing the voltage. It **Shall** reach [vNew](#) and begin sending a [PS_RDY Message](#) before:
 - [tSrcTransReq](#) expires for transitions to a PDO.
 - [tAvsSrcTransLarge](#) expires for transitions to an AVS APDO.
- The Sink acknowledges the [PS_RDY Message](#) with a [GoodCRC Message](#) and **May** increase its current up to the newly Negotiated value at any time, provided the required slew rate is not exceeded.

Figure 4.16. Transition Diagram for Large-step voltage increase (Fixed or AVS > vSmallStep)



4.5.1.2. Small-step voltage increases (AVS)

Figure 4.17 applies to AVS transitions where the voltage is increasing and the voltage change $\leq v_{\text{SmallStep}}$. In these cases, the Sink can maintain the maximum current through the transition.

The sequence for these types of transitions is as follows:

- The Source Policy Engine sends an [Accept Message](#).
- The [PSTransitionTimer](#) starts immediately after the Sink completes sending the [GoodCRC Message](#).
- There is no requirement for the Sink to reduce the current during the transition period.
- The Source **May** immediately start increasing the voltage. It **Shall** reach [vNew](#) and send a [PS_RDY Message](#) before [tSrcTransSmall](#) expires.
- The Sink acknowledges the [PS_RDY Message](#) with a [GoodCRC Message](#) and **May** increase its current up to the newly Negotiated value at any time, provided the required slew rate is not exceeded.

Figure 4.17. Transition Diagram for Small-step voltage increase ($AVS \leq v_{\text{SmallStep}}$), current unchanged or increased

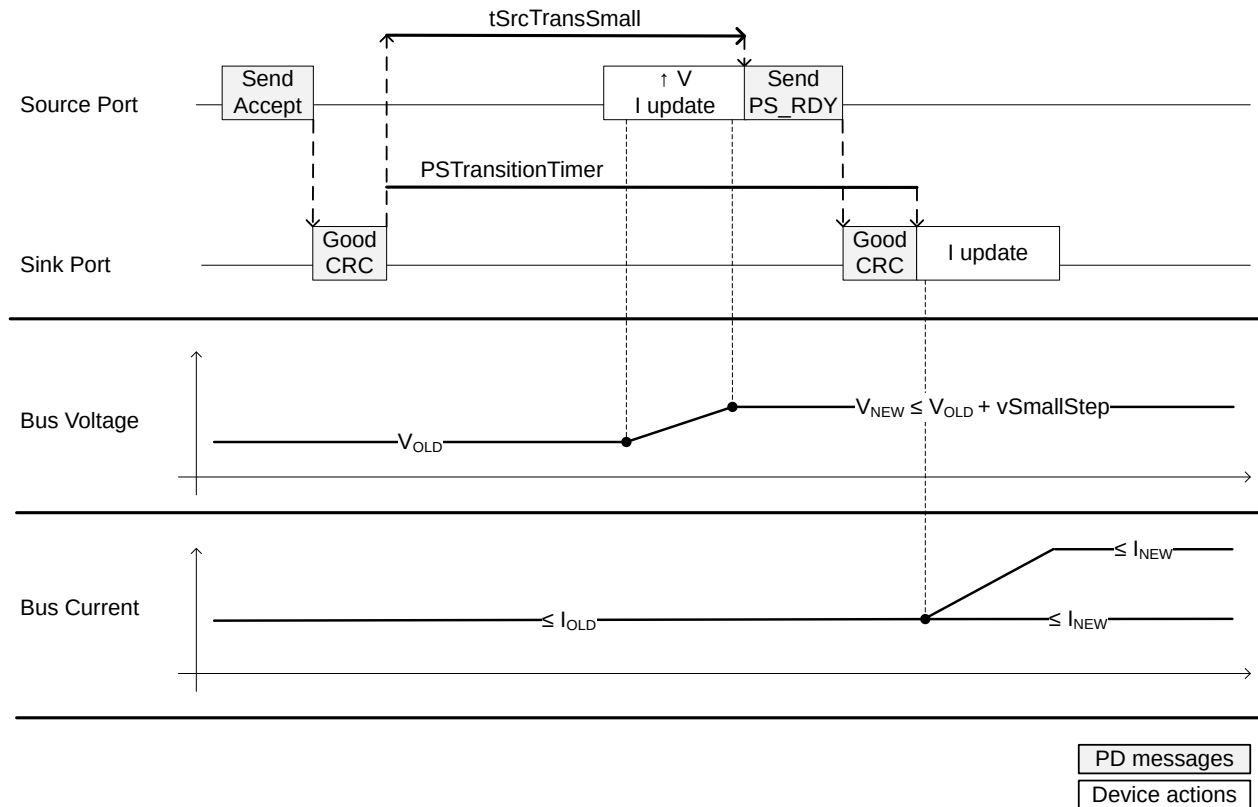
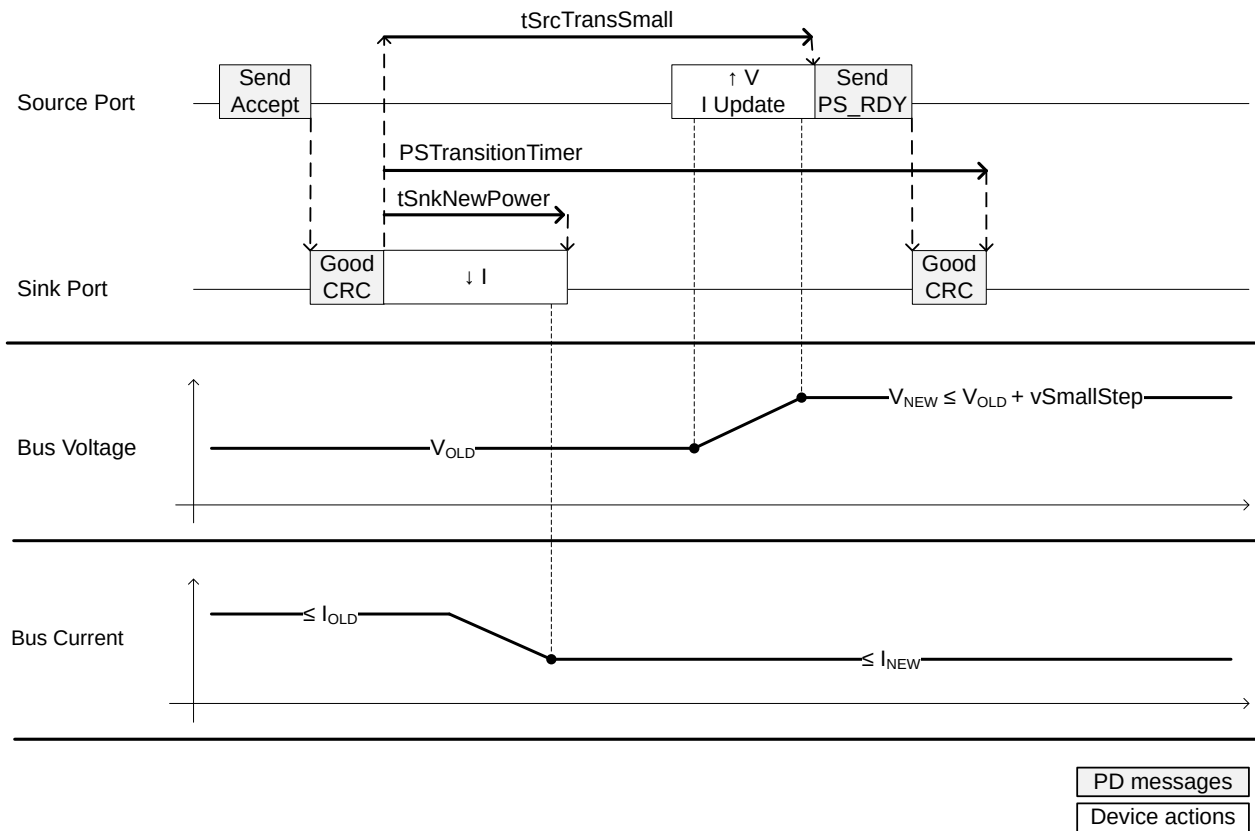


Figure 4.18 applies to AVS transitions where the voltage is increasing and the voltage change $\leq v_{\text{SmallStep}}$ with a decrease in current.

The sequence for these types of transitions is as follows:

- The Source Policy Engine sends an [Accept Message](#).
- The [PSTransitionTimer](#) starts immediately after the Sink completes sending the [GoodCRC Message](#).
- The Sink has [tSnkNewPower](#) to reduce the VBUS current to a maximum of I_{NEW} .
- The Source **shall** reach [vNew](#) and send a [PS_RDY Message](#) before [tSrcTransSmall](#) expires.
- The Source **shall** wait [tSnkNewPower](#) before changing any current limit protection related to I_{NEW} .
- The Sink acknowledges the [PS_RDY Message](#) with a [GoodCRC Message](#).

Figure 4.18. Transition Diagram for Small-step voltage increase ($AVS \leq v_{SmallStep}$), current decreased



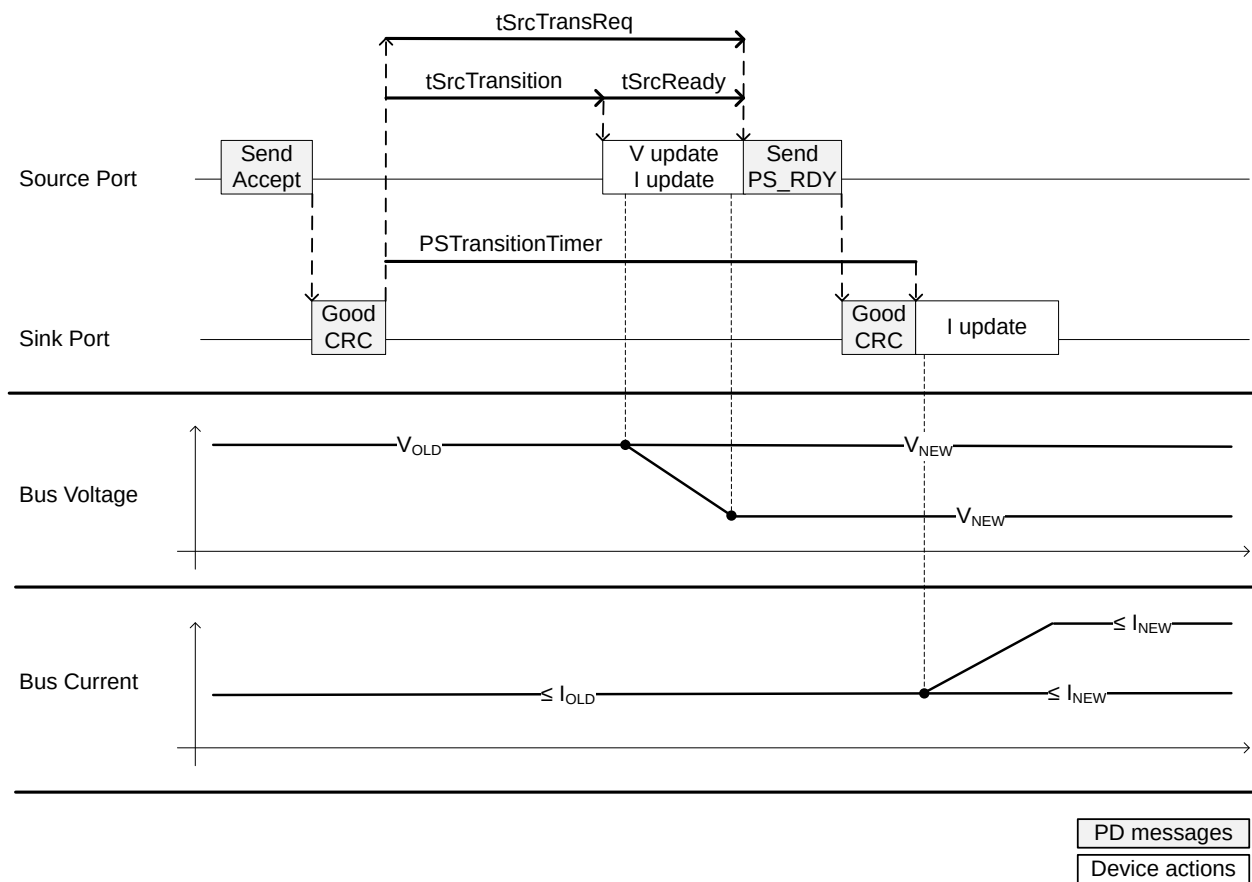
4.5.1.3. Unchanged or decreased voltage transitions (current unchanged, increased, or decreased)

The process for these transitions depends on whether the voltage is unchanged or decreased, and whether the current is increased, decreased, or unchanged.

- The Source Policy Engine sends an [Accept Message](#).
- The [PSTransitionTimer](#) starts immediately after the Sink completes sending the [GoodCRC Message](#).
- There is no requirement for the Sink to reduce the current during the transition period.

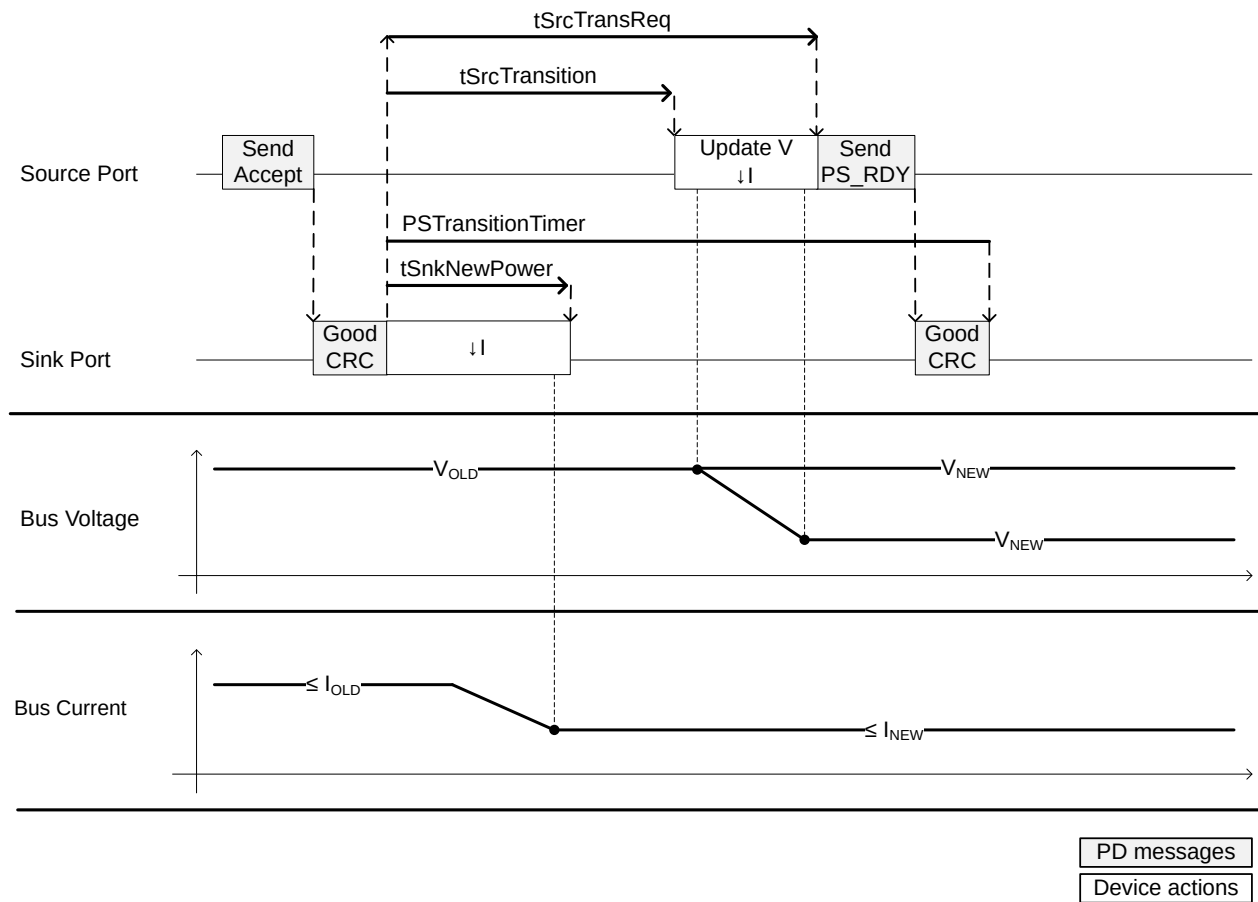
- The Source **Shall** wait [tSrcTransition](#) before starting to decrease the voltage or updating any current protection thresholds as needed.
- Once [tSrcTransition](#) elapses, the Source **May** start decreasing the voltage. It **Shall** reach [vNew](#) and begin sending a [PS_RDY Message](#) before:
 - [tSrcTransReq](#) expires for transitions to a PDO.
 - [tAvsSrcTransLarge](#) expires for transition to an AVS APDO and the voltage change is $> v_{SmallStep}$.
- The Sink acknowledges the [PS_RDY Message](#) with a [GoodCRC Message](#) and **May** increase its current up to the newly Negotiated value at any time, as long as the required slew rate is not exceeded.

Figure 4.19. Transition Diagram for voltage unchanged or decreased with current unchanged or increased



If the current needs to be decreased, the previous sequence shall be followed with one difference as shown in [Figure 4.20](#):

- The Sink **Should** reduce the current to a value that is less than or equal to the newly Negotiated value (I_{NEW}) within [tSnkNewPower](#). It **Shall** reach I_{NEW} before the minimum value of [tSrcTransition](#) expires.

Figure 4.20. Transition Diagram for voltage unchanged or decreased with current decreased

4.5.2. PPS voltage or current changes

4.5.2.1. PPS voltage transitions (CV Mode)

This sequence is valid for any PPS voltage transition (while in CV Mode), and any small AVS transition that is [vAvsSmallStep](#) or smaller.

Compared with transitions that use fixed voltage contracts, the process of starting the voltage transition can begin immediately after the Sink acknowledges the [Accept Message](#) with a [GoodCRC Message](#). Additionally, the Sink can operate at any current level, up to the maximum value defined by the Contract. [Figure 4.21](#) shows the sequence when the voltage is increased and [Figure 4.22](#) shows the sequence when it is decreased.

Figure 4.21. Transition Diagram for PPS voltage increase (CV Mode)

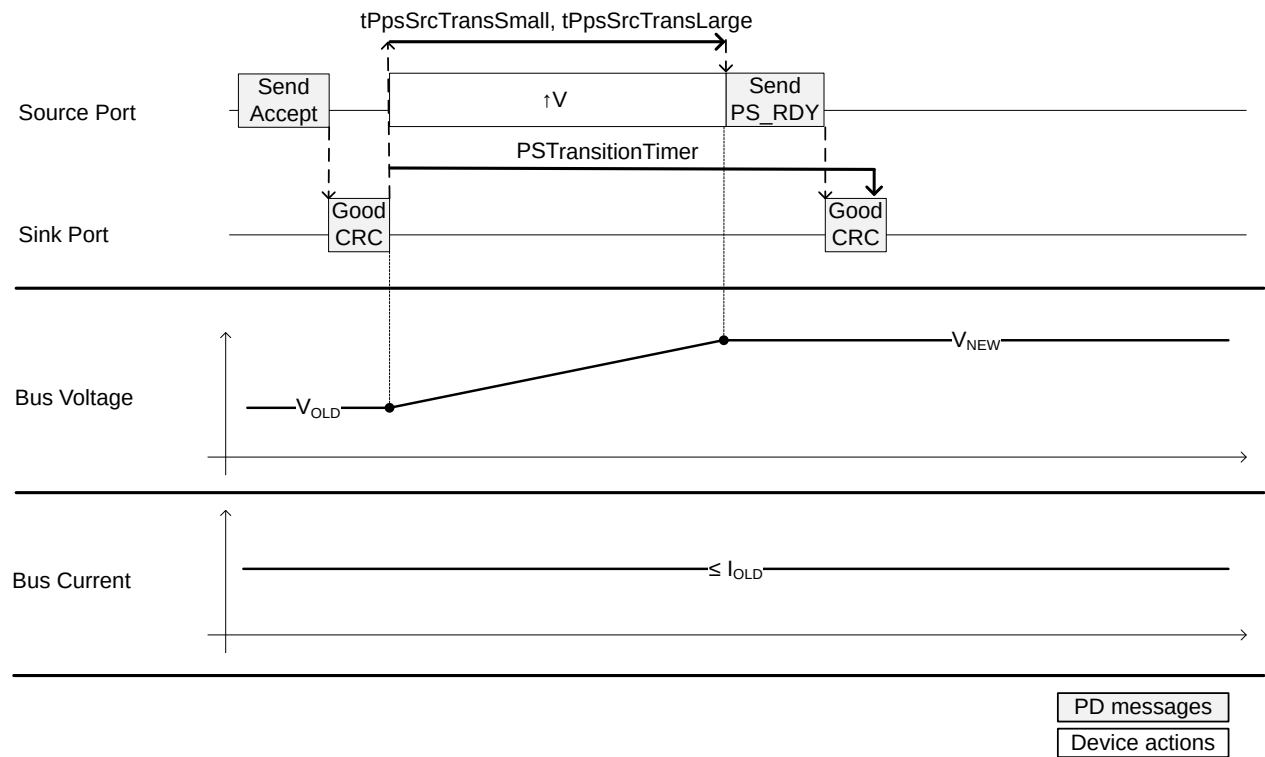
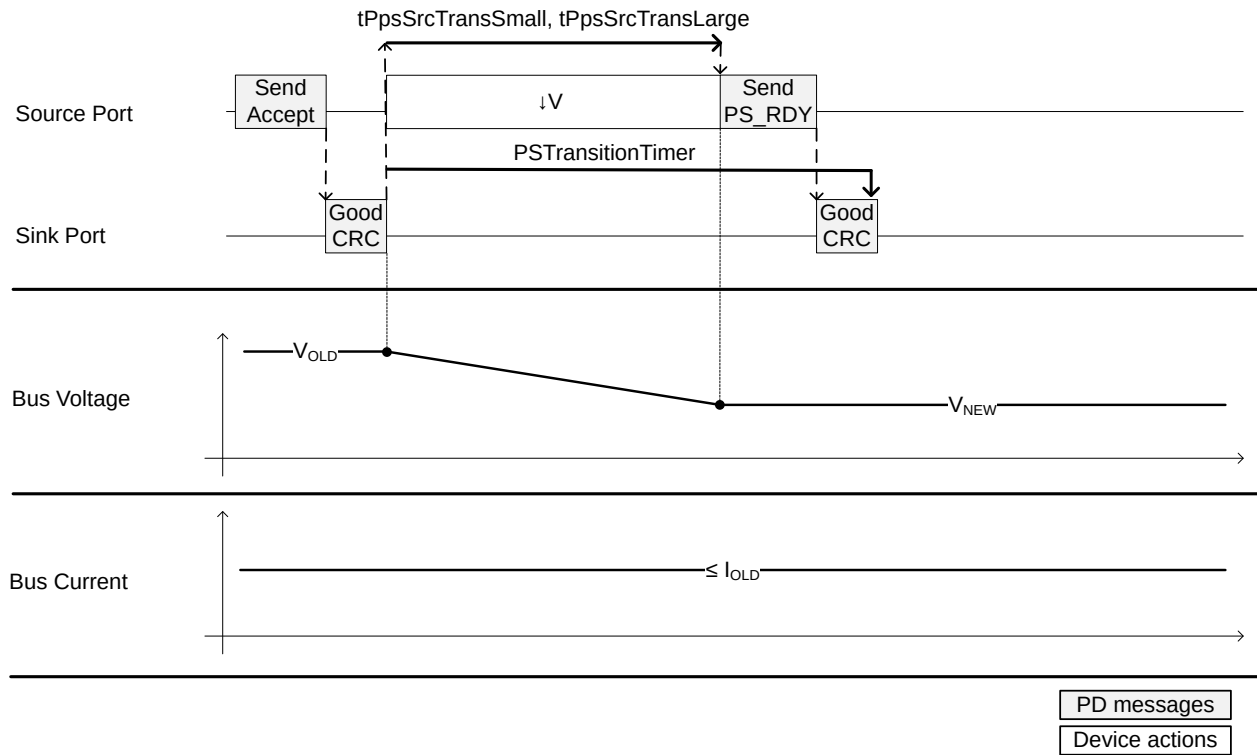


Figure 4.22. Transition Diagram for PPS voltage decrease (CV Mode)

4.5.2.2. PPS current transitions (CL Mode)

In PPS, the Sink can Request a change in current while operating in Current Limit (CL) Mode. In this scenario, the Source is controlling the VBUS current instead of the VBUS voltage. The VBUS voltage can take any value between [vPpsMinVoltage](#), and the Output Voltage value used in the PPS Request Data Object. [Figure 4.23](#) shows the transition for changing current in PPS.

Figure 4.23. Transition Diagram for PPS current increase (CL Mode)

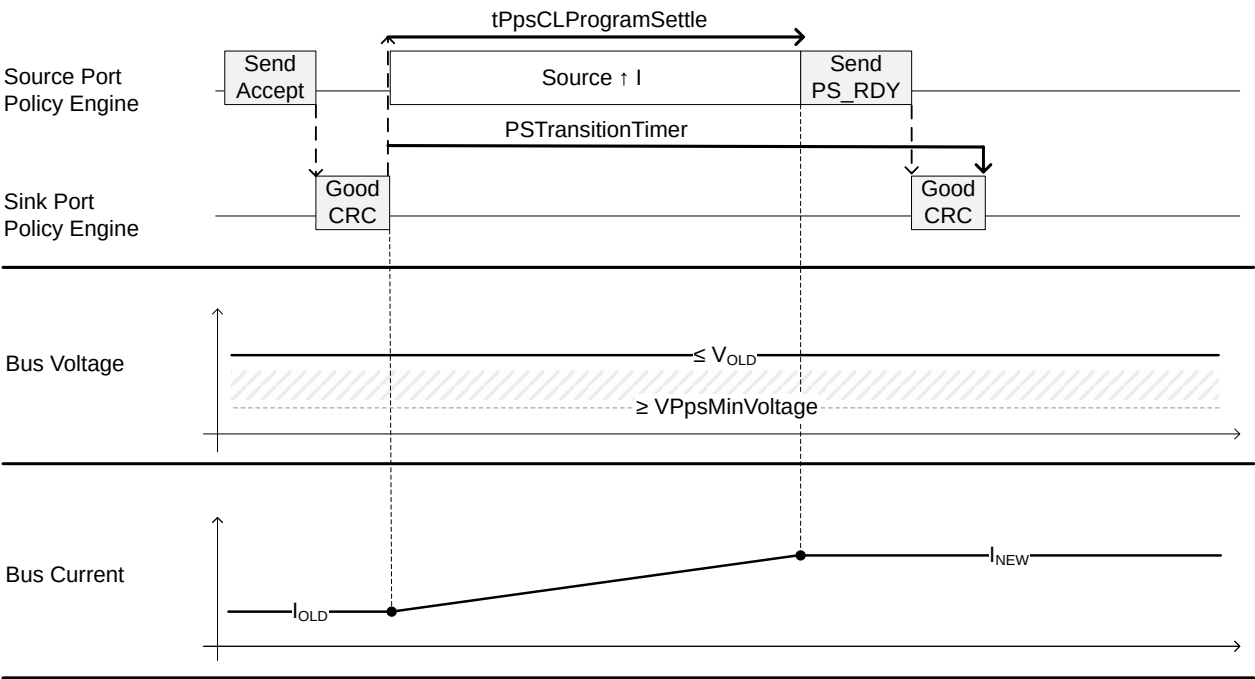
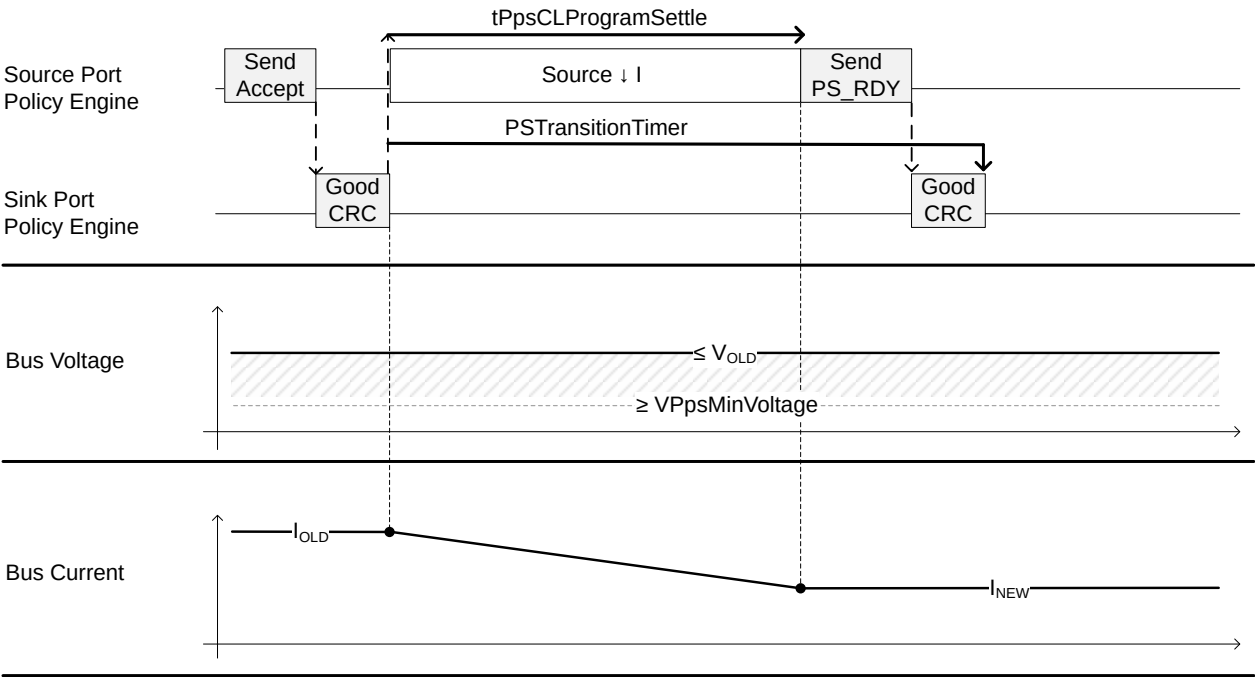


Figure 4.24. Transition Diagram for PPS current decrease (CL Mode)



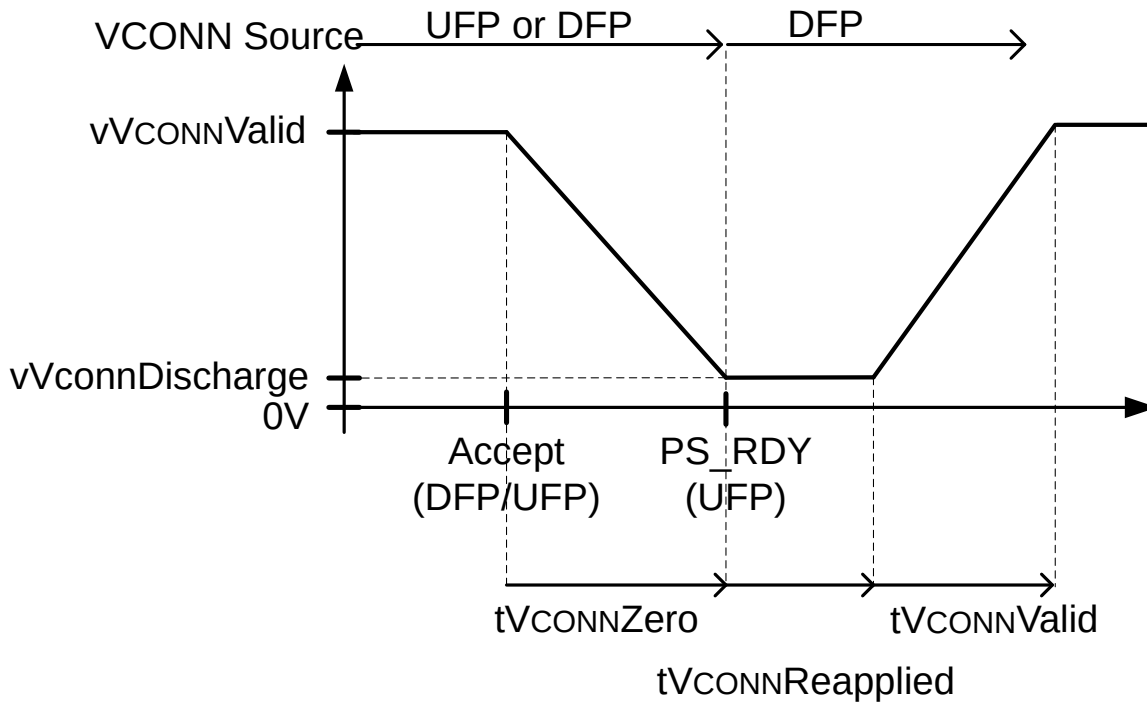
4.6. VCONN Power Cycle

When performing a VCONN Power Cycle during a [Data Reset](#), the following steps **Shall** be followed:

- Following the last bit of the [GoodCRC Message](#) acknowledging the [Accept Message](#), the initial VCONN Source **Shall** turn off VCONN and ensure it is below [USB-C] vRaReconnect within [tVCONNZero](#).
- When VCONN is below vRaReconnect, the initial VCONN Source **Shall** send a [PS_RDY Message](#). Note: for a [Data Reset](#) requested by the UFP, if the UFP was not sourcing VCONN, it **Shall** send the [PS_RDY Message](#).
- The new VCONN Source **Shall** wait [tVCONNReapplied](#) following the last bit of the [GoodCRC Message](#) acknowledging the [PS_RDY Message](#) before sourcing VCONN. The new VCONN **Shall** ensure VCONN is within vVCONNValid (see [USB-C]) within [tVCONNValid](#).

[Figure 4.25](#) illustrates the VCONN Power Cycle process.

Figure 4.25. Data Reset VCONN Power Cycle



4.7. Electrical Parameters

4.7.1. Source Electrical Parameters

The Source Electrical Parameters that **Shall** be followed are specified in [Table 4.6](#).

Table 4.6. Source Electrical Parameters

Parameter Name		Min Value	Nom Value	Max Value	Unit	Description
cSrcBulk		10			μF	Source bulk capacitance when a Port is powered from a dedicated supply.
cSrcBulkShared		120			μF	Source bulk capacitance when a Port is powered from a shared supply. ¹
DNL		-1	0	+1	LSB	Differential Non-Linearity Deviation between ideal analog values corresponding to adjacent input digital values
iPpsCLMin		1			A	PPS Minimum Current Limit setting.
iPpsCLNew	≤ 3A	-150		150	mA	Current Limit Accuracy 1A ≤ Operating Current ≤ 3A
	> 3A	-5		5	%	Current Limit Accuracy Operating current > 3A
iPpsCLStep			50		mA	PPS Current Limit programming step size (1 LSB).
iPpsCLLoadReleaseRate		-150			mA/μs	Maximum load decrease slew rate during Current Limit set-point changes.
iPpsCLLoadStepRate				150	mA/μs	Maximum load increase slew rate during Current Limit set-point changes.
iPpsCLTransient	Load increase			New load + 100	mA	Allowed output current overshoot when a load increase occurs while in CL Mode.
	Load decrease	New load – 100			mA	Allowed output current undershoot when a load decrease occurs while in CL Mode.
iPpsCVCLTransient		iPpsCLNew - 100		New load + 500	mA	CV to CL transient current bounds. See Section 4.1.3.2.2 .
tAvsTransient				5	ms	The maximum time for the AVS to be between vAvsNew and vAvsValid in response to a load transient.
tAvsSrcTransLarge		0		700	ms	The time the AVS set-point Shall transition between requested voltages for steps larger than vAvsSmallStep .
tAvsSrcTransSmall		0		50	ms	The time the AVS set-point Shall transition between requested voltages for steps smaller than vAvsSmallStep .
tNewSnk				15	ms	Time allowed for an Initial Source in Swap Standby to transition to New Sink operation.
tPpsCLCVTransient				275	ms	CL to CV transient voltage settling time.
tPpsCLProgramSettle				250	ms	PPS Current Limit programming settling time.

Parameter Name		Min Value	Nom Value	Max Value	Unit	Description
tPpsCLSettle				250	ms	CL load transient current settling time.
tPpsCVCLTransient				250	ms	CV to CL transient settling time.
tPpsSrcTransLarge		0		275	ms	The time the Programmable Power Supply's set-point Shall transition between requested voltages for steps larger than vPpsSmallStep .
tPpsSrcTransSmall		0		25	ms	The time the Programmable Power Supply's set-point Shall transition between requested voltages for steps less than or equal to vPpsSmallStep .
tPpsTransient	Target load \geq 60mA			5	ms	The maximum time for the Programmable Power Supply to be between vPpsNew and vPpsValid in response to a load transient when target load is greater than or equal to 60mA.
	Target load < 60mA			150	ms	The maximum time for the Programmable Power Supply to be between vPpsNew and vPpsValid in response to a load transient when target load is less than 60mA.
tSrcReady	SPR Mode			285	ms	Time from positive/negative transition start (t0) to when the Source is ready to provide the newly Negotiated power level. Applies only to SPR Mode voltage transitions.
	EPR Mode			720		Time from positive/negative transition start (t0) to when the Source is ready to provide the newly Negotiated power level. Applies to EPR Mode voltage transitions and any voltage transition that either begins or ends in EPR Mode .
tSrcRecover	SPR Mode	0.66		1.0	s	Time allotted for the Source to recover.
	EPR Mode	1.085		1.425		
tSrcSettle	SPR Mode			275	ms	Time from positive/negative transition start (t0) to when the transitioning voltage is within the range vSrcNew . Applies only to SPR Mode voltage transitions.
	EPR Mode			700		Time from positive/negative transition start (t0) to when the transitioning voltage is within the range vAvsNew . Applies to EPR Mode voltage transitions and any voltage transition that either begins or ends in EPR Mode .
tSrcSwapStdby				650	ms	The maximum time for the Source to transition to Swap Standby.

Parameter Name		Min Value	Nom Value	Max Value	Unit	Description
tSrcTransient				5	ms	The maximum time for the Source output voltage to be between vSrcNew and vSrcValid in response to a load transient when target load is greater than or equal to 60mA.
				150	ms	The maximum time for the Source output voltage to be between vSrcNew and vSrcValid in response to a load transient when target load is less than 60mA.
tSrcTransition		25		35	ms	The time the Source Shall wait before transitioning the power supply to ensure that the Sink has sufficient time to prepare (does not apply to transitions within the same PPS or AVS AP-DO).
tSrcTransOff				690	ms	Time from the last bit of the GoodCRC Message acknowledging the Accept Message in response to the PR_Swap Message until the PS_RDY Message must be started. Applies only to SPR Mode voltage transitions.
tSrcTransOn				280	ms	Time from the last bit of the GoodCRC Message acknowledging the PS_RDY Message sent by the New Source, in response to the PR_Swap Message until the PS_RDY Message must be started.
tSrcTransReq	SPR Mode			325	ms	Time from the last bit of the GoodCRC Message acknowledging the Accept Message in response to the Request Message until the PS_RDY Message must be started. Applies only to SPR Mode voltage transitions.
	EPR Mode			760	ms	Time from the last bit of the GoodCRC Message acknowledging the Accept Message in response to the Request Message until the PS_RDY Message must be started. Applies to EPR Mode voltage transitions and any voltage transition that either begins or ends in EPR Mode .
tSrcTurnOn				275	ms	Transition time from vSafe0V to vSafe5V .
tReducePowerAlert		2000		2400	ms	Time between an alert and the New Source Capabilities when reducing Port Present PDP.
tDpsColdStart		15			minute	Time for a DPS Source to maintain Port Present PDP = Port Maximum PDP after a cold start before thermal limiting.
tDpsRegular		1			minute	Time between New Source Capabilities for a DPS Source due to thermal limiting.

Parameter Name		Min Value	Nom Value	Max Value	Unit	Description
vAvsMaxVoltage		APDO Max Voltage $\times 0.95$		APDO Max Voltage $\times 1.05$	V	Maximum Voltage Field in the AVS APDO.
vAvsMinVoltage		APDO Min Voltage $\times 0.95$		APDO Min Voltage $\times 1.05$	V	Minimum Voltage Field in the AVS APDO.
vAvsNew		RDO Output Voltage $\times 0.95$	RDO Output Voltage	RDO Output Voltage $\times 1.05$	V	Adjustable RDO Output Voltage measured at the Source receptacle.
vAvsSlewNeg				-30	mV/ μ s	AVS maximum slew rate for negative voltage changes.
vAvsSlewPos				30	mV/ μ s	AVS maximum slew rate for positive voltage changes.
vAvsSmallStep		-1.0		1.0	V	AVS step size defined as a small step relative to the previous vAvsNew .
vAvsStep			100		mV	AVS voltage programming step size.
vAvsValid		-0.5		0.5	V	The range in addition to vAvsNew which the AVS output is considered Valid during and after a transition as well as in response to a transient load condition.
vPpsCLCVTransient		Operating Voltage $\times 0.95 - 0.1V$		Operating Voltage $\times 1.05 + 0.1V$	V	CL to CV load transient voltage bounds.
vPpsMaxVoltage		APDO Max Voltage $\times 0.95$		APDO Max Voltage $\times 1.05$	V	Maximum Voltage Field in the Programmable Power Supply APDO.
vPpsMinVoltage		APDO Min Voltage $\times 0.95$		APDO Min Voltage $\times 1.05$	V	Minimum Voltage Field in the Programmable Power Supply APDO.
vPpsNew		RDO Output Voltage $\times 0.95$	RDO Output Voltage	RDO Output Voltage $\times 1.05$	V	Programmable RDO Output Voltage measured at the Source receptacle.
vPpsShutdown		APDO Minimum Voltage $\times 0.85$		APDO Minimum Voltage $\times 0.95$	V	The voltage at which the PPS shuts down when operating in CL.
vPpsSlewNeg				-30	mV/ μ s	Programmable Power Supply maximum slew rate for negative voltage changes
vPpsSlewPos				30	mV/ μ s	Programmable Power Supply maximum slew rate for positive voltage changes
vPpsSmallStep		-500		500	mV	PPS Step size defined as a small step relative to the previous vPpsNew .
vPpsStep			20		mV	PPS voltage programming step size (1 LSB).
vPpsValid		-0.1		0.1	V	The range in addition to vPpsNew which the Programmable Power Supply output is considered Valid in response to a load step.
vSrcNeg				-0.3	V	Most negative voltage allowed during transition.
vSrcNew	Fixed Supply	PDO Voltage $\times 0.95$	PDO Voltage	PDO Voltage $\times 1.05$	V	Fixed Supply output measured at the Source receptacle.

Parameter Name		Min Value	Nom Value	Max Value	Unit	Description
	Variable Supply	PDO Minimum Voltage		PDO Maximum Voltage	V	Variable Supply output measured at the Source receptacle.
	Battery Supply	PDO Minimum Voltage		PDO Maximum Voltage	V	Battery Supply output measured at the Source receptacle.
vSrcPeak		PDO Voltage $\times 0.90$		PDO Voltage $\times 1.05$	V	The range that a Fixed Supply or EPR AVS in Peak Current operation is allowed when overload conditions occur.
vSrcSlewNeg				-30	mV/ μ s	Maximum slew rate allowed for negative voltage transitions. Limits current based on a 3 A connector rating and maximum Sink bulk capacitance of 100 μ F.
vSrcSlewPos				30	mV/ μ s	Maximum slew rate allowed for positive voltage transitions. Limits current based on a 3 A connector rating and maximum Sink bulk capacitance of 100 μ F.
vSrcValid		-0.5		0.5	V	The range in addition to vSrcNew which a newly Negotiated voltage is considered Valid during and after a transition as well as in response to a transient load condition. This range also applies to vSafe5V .

4.7.2. Sink Electrical Parameters

The Sink Electrical Parameters that **Shall** be followed are specified in [Table 4.7](#).

Table 4.7. Sink Electrical Parameters

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
cSnkBulk			See [USB3]		Sink bulk capacitance on VBUS at Attach and during FRS after the Initial Source stops sourcing and prior to establishing the First Explicit Contract.
cSnkBulkPd			100	μ F	Bulk Capacitance that a Sink is allowed to present on VBUS after a successful Negotiation.
iLoadReleaseRate	-150			mA/ μ s	Load release di/dt.
iLoadStepRate			150	mA/ μ s	Load step di/dt.
iPpsCLLoadStep	-500		500	mA	Maximum Current set-point change while operating in CL Mode.
iSafe0mA			1.0	mA	Maximum current a Sink is allowed to draw when VBUS is driven to vSafe0V .
iSnkStdbby			500	mA	Maximum current during voltage transition.
iSnkSwapStdbby			2.5	mA	Maximum current a Sink can draw during Swap Standby. Ideally this current is very near to 0 mA largely influenced by Port leakage current.
pHubSusp			125	mW	Suspend power consumption for a Hub. 25mW + 25mW per downstream Port for up to 4 ports.
pSnkSusp			25	mW	Suspend power consumption for a Peripheral Device.
tNewSrc			275	ms	Maximum time allowed for an Initial Sink in Swap Standby to transition to New Source operation.

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
tSnkHardResetPrepare			15	ms	Time allotted for the Sink power electronics to prepare for a Hard Reset .
tSnkNewPower			15	ms	Maximum transition time between power levels.
tSnkRecover			150	ms	Time for the Sink to resume USB Default Operation.
tSnkStdbby			15	ms	Time to transition to Sink Standby from Sink.
tSnkSwapStdbby			15	ms	Maximum time for the Sink to transition to Swap Standby.
vSprMax			24	V	A Sink Should tolerate this VBUS voltage without damage.
1. If more bypass capacitance than cSnkBulk (max) or cSnkBulkPd (max) is required in the Device, then the Device Shall incorporate some form of VBUS surge current limiting as described in [USB3].					

4.7.3. Common Electrical Parameters

Electrical Parameters that are common to both the Source and the Sink that **Shall** be followed are specified in [Table 4.8](#).

Table 4.8. Common Source/Sink Electrical Parameters

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
tSafe0V			650	ms	Time to reach vSafe0V (max).
tSafe5V			275	ms	Time to reach vSafe5V (max).
tVCONNReapplied	10		20	ms	When the UFP is the VCONN Source: time from the last bit of the GoodCRC Message acknowledging the PS_RDY Message until VCONN is within vVCONNValid (see [USB-C]). When the DFP is the VCONN Source: time from when VCONN drops below vRaReconnect.
tVCONNValid	0		5	ms	Time from tVCONNReapplied until VCONN is within vVCONNValid (see [USB-C]).
tVCONNZero			125	ms	Time from the last bit of the GoodCRC acknowledging the Accept Message in response to the Data_Reset Message until VCONN is below vRaReconnect (see [USB-C]).
vSafe0V	0		0.8	V	Safe operating voltage at “zero volts”.
vSafe5V	4.75		5.5	V	Safe operating voltage at 5V. See [USB2] and [USB3] for allowable VBUS voltage range.
1. tVCONNStable (See [USB-C]) still applies.					

Chapter 5. PD Communications Physical Layer

5.1. Overview

The Physical Layer (PHY Layer) defines the Signaling technology for USB Power Delivery. This chapter defines the electrical requirements and parameters of the PHY Layer to ensure interoperability between PDUSB Devices.

The USB PD PHY Layer consists of a BMC transmitter and receiver that communicate across a single signal wire (CC) along with terminations for DC biasing. All BMC communication is half-duplex. The PHY Layer uses DC biasing for Collision Avoidance to minimize communication errors on the channel, indicate Power Role, and establish Implicit contracts.

The transmitter performs the following functions:

- Receive Packet data from the Protocol Layer.
- Calculate and append a CRC.
- Encode the Packet data including the CRC (i.e., the Payload).
- Transmit the Packet (Preamble, SOP*, Payload, CRC and EOP) across the channel using Bi-phase Mark Coding (BMC) over CC.

The receiver performs the following functions:

- Recover the clock and signal.
- Detect the SOP*.
- Decode the received data including the CRC.
- Detect the EOP and validate the CRC:
 - If the CRC is **Valid**, deliver the Packet data to the Protocol Layer.
 - If the CRC is **Invalid**, flush the received data.

5.2. Signaling Through DC Biasing

The DC biasing Signaling is implemented using Rp pullups and/or Rd pulldowns as defined in [USB-C].

5.2.1. Power Role Indication

[USB-C] defines the usage of Rp and Rd as the Power Role indicator. Rp indicates the Source and Rd indicates the Sink. USB PD allows the roles to be swapped between the Source and Sink. This is done by changing from Rp to Rd when changing from a Source to a Sink and changing from Rd to Rp when changing from a Sink to a Source. The Device Policy Manager (DPM) informs the PHY Layer when the CC termination needs to be changed from an Rp pullup to an Rd pulldown or vice versa. The PHY simply applies the requested CC termination.

5.2.2. Collision Avoidance

When a USB PD Explicit Contract is established, the Rp value presented by the Source is no longer needed to determine [USB-C] Type-C current. The value of Rp is then repurposed to indicate CC bus communications avail-

ability. To avoid collisions, the Source controls who may communicate on CC. CC Collision Avoidance is achieved by the following:

1. During the Default or Implicit Contract, the Rp resistor value is used to specify [USB-C] Type-C Current and **Shall** not be used for Collision Avoidance.
2. The Protocol Layer of a Sink **Shall** Request from the PHY Layer the value of Rp the Source Port Partner is presenting to determine if it may initiate a transmission.
3. The Source **Shall** present Rp=[SinkTxNG](#) to claim control of the CC bus, and the Source **Shall** present Rp=[SinkTxOk](#) to release control of the CC bus. [Table 5.1](#) shows the Rp values the Source **Shall** use. See [Section 7.2](#) for Protocol Layer use of [SinkTxNG](#) and [SinkTxOk](#) for Collision Avoidance.
4. The PHY Layer **Shall** monitor the channel for data transmission and only initiate transmissions when the CC bus is Idle (see [Section 5.2.2.1](#)).
5. The CC bus Idle condition **Shall** be checked immediately prior to transmission.
6. Transmission **Shall** only start if the CC bus Idle condition is present, and if the conditions detailed in [Section 5.3.4.9](#) are met.
7. If transmission cannot be initiated due to the CC bus not being Idle, the Packet **Shall** be discarded and the PHY Layer **Shall** signal to the Protocol Layer as soon as CC becomes Idle that the Message has been discarded.

Table 5.1. Rp values used for Collision Avoidance

Parameter Name	[USB-C] Source Rp	Description ¹
SinkTxNG	1.5A@5V	Sink Transmit "No Go" <ul style="list-style-type: none"> • The Sink is not allowed to start an AMS. • The Source is allowed to start an AMS.
SinkTxOK	3A@5V	Sink Transmit "OK" <ul style="list-style-type: none"> • The Sink is allowed to start an AMS. • The Source is not allowed to start an AMS.
1. See Section 7.2 .		

5.2.2.1. Definition of Idle

BMC Collision Avoidance is performed by the detection of signal transitions at the receiver. Detection is active when [nTransitionCount](#) transitions occur at the receiver within a time window of [tTransitionWindow](#). After waiting [tTransitionWindow](#) without detecting [nTransitionCount](#) transitions the CC bus **Shall** be declared Idle.

Refer to [Section 5.3.4.9](#) for details of when transmissions **May** start.

5.3. BMC Signaling

This section describes how Message packets are formed and sent by the transmitter, and decoded by the receiver.

5.3.1. Transmission

The PHY Layer will make a transmission upon Request from the Protocol Layer. The Protocol Layer **May** Request several types of transmissions:

- Start of Packet Sequences (see [Section 5.3.1.1.3](#)). Note: the alias, SOP*, refers to any of these.

- SOP Message
- SOP' Message
- SOP" Message
- SOP'_Debug Message
- SOP"_Debug Message
- [Hard Reset](#)
- [Cable Reset](#)

5.3.1.1. Packet Formation

The following rules apply to Packet formation.

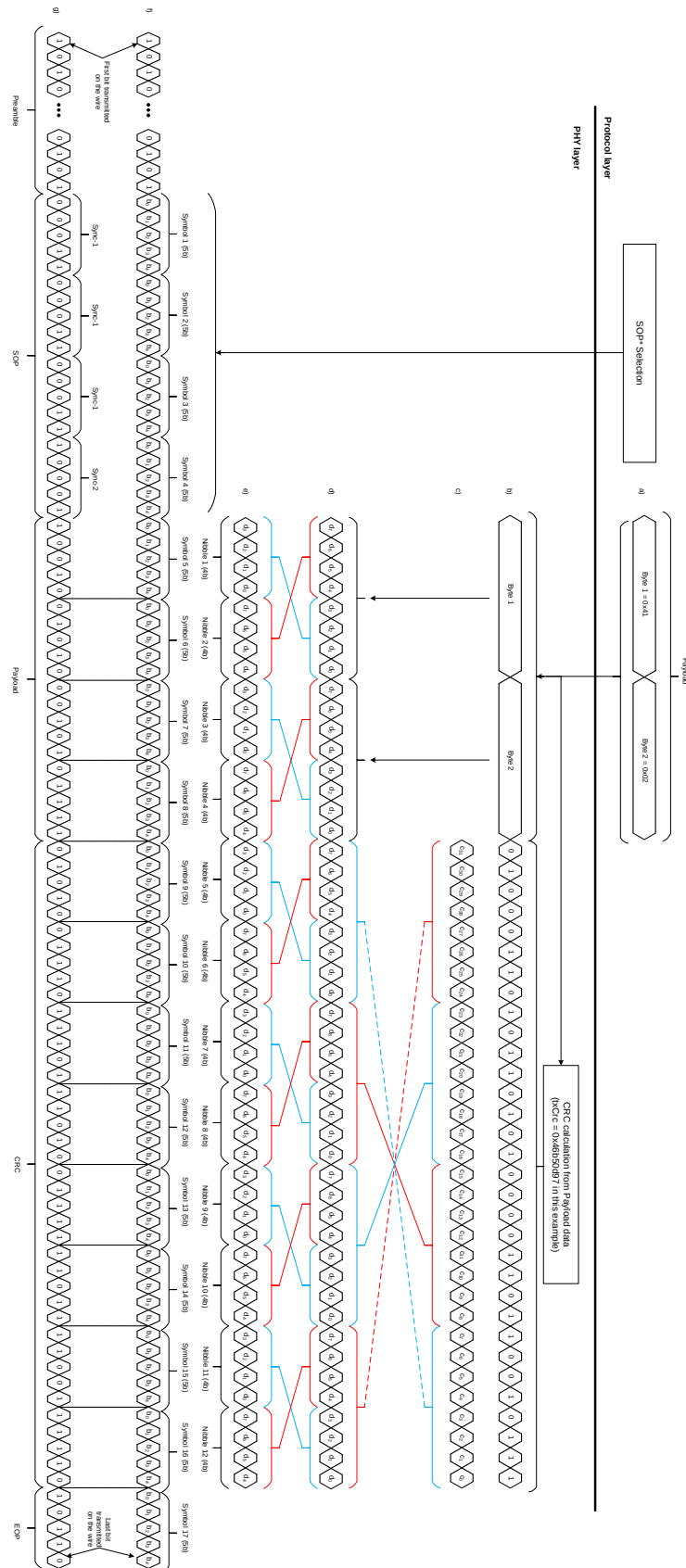
1. The Packet format is shown in [Figure 5.1](#). The Packet format **Shall** consist of
 - a. a Preamble,
 - b. an SOP* (see [Section 5.3.1.1.3](#)),
 - c. Packet data including the Message Header (4b5b encoded),
 - d. a CRC (4b5b encoded) (see [Section 5.3.1.1.4](#)),
 - e. and an EOP (see [Section 5.3.1.1.5](#)).
2. Once 4b/5b encoded, the entire Packet **Shall** be transmitted using BMC over CC.

The Protocol Layer delivers the Payload data bytes, and specifies which SOP* sequence to use.

[Figure 6.2](#) shows how Message packets are constructed. The PHY layer forms the Packet from the given Payload as shown in the [Figure 5.1](#) and outlined in following steps:

1. Example [GoodCRC Message](#) data from the Protocol Layer.
2. Calculates and adds the CRC value to the Message.
3. The calculated CRC value is reformatted into LSB first order.
4. Bytes are reformatted into nibble order.
5. The data is encoded (4b5b) and then packetized with the Preamble and the selected SOP*.

Figure 5.1. Transmit Packet Format



5.3.1.1.1. Preamble

The Preamble is used to alert the receiver to wake and begin receiving. Preamble formation is defined by the following:

1. The Preamble **Shall** consist of a 64-bit sequence of alternating 0s and 1s, starting with a "0" and ending with a "1".
2. The Preamble **Shall Not** be 4b/5b encoded.
3. The Preamble **Shall** be BMC encoded.

5.3.1.1.2. Symbol Encoding

Except for the Preamble, all data in the Packet is encoded with a 4b5b line code. This encodes 4-bit data to 5-bit symbols for transmission and decodes 5-bit symbols to 4-bit data for consumption by the receiver. The following rules apply to encoding.

1. Any K-code is already a 5b value and **Shall** not be encoded further.
2. The header and data **Shall** be 4b5b encoded.
3. The 5b code values in [Table 5.2](#) **Shall** be used for K-codes and 4b data.

Table 5.2. K-codes and 4b5b symbol encoding

Symbol Name	4b Upper or Lower Nibble $d_7d_6d_5d_4$ or $d_3d_2d_1d_0$	5b Symbol $b_4b_3b_2b_1b_0$	Description
0	0000	11110	hex data 0
1	0001	01001	hex data 1
2	0010	10100	hex data 2
3	0011	10101	hex data 3
4	0100	01010	hex data 4
5	0101	01011	hex data 5
6	0110	01110	hex data 6
7	0111	01111	hex data 7
8	1000	10010	hex data 8
9	1001	10011	hex data 9
A	1010	10110	hex data A
B	1011	10111	hex data B
C	1100	11010	hex data C
D	1101	11011	hex data D
E	1110	11100	hex data E
F	1111	11101	hex data F
Sync-1	K-code	11000	Start synch #1
Sync-2	K-code	10001	Start synch #2
RST-1	K-code	00111	Hard Reset #1
RST-2	K-code	11001	Hard Reset #2
EOP	K-code	01101	EOP End of Packet
	Error	00000	Shall Not be used
	Error	00001	Shall Not be used
	Error	00010	Shall Not be used
	Error	00011	Shall Not be used

Symbol Name	4b Upper or Lower Nibble $d_7d_6d_5d_4$ or $d_3d_2d_1d_0$	5b Symbol $b_4b_3b_2b_1b_0$	Description
	Error	00100	Shall Not be used
	Error	00101	Shall Not be used
Sync-3	K-code	00110	Start synch #3
	Error	01000	Shall Not be used
	Error	01100	Shall Not be used
	Error	10000	Shall Not be used
	Error	11111	Shall Not be used

5.3.1.1.3. Start of Packet Sequences (SOP*)

Different SOP* sequences are used depending on the target receiver. A receiver will ignore some SOP* sequences. For example, a Cable Plug will ignore an incoming Message that uses the SOP sequence.

When the Protocol Layer requests a transmission it also specifies which SOP* sequence to use.

Table 5.3. K-codes for SOP* Sequences

SOP* Sequence	K-code1	K-code2	K-code3	K-code4
SOP	Sync-1	Sync-1	Sync-1	Sync-2
SOP'	Sync-1	Sync-1	Sync-3	Sync-3
SOP"	Sync-1	Sync-3	Sync-1	Sync-3
SOP'_Debug	Sync-1	RST-2	RST-2	Sync-3
SOP''_Debug	Sync-1	RST-2	Sync-3	Sync-2

5.3.1.1.4. CRC

A CRC-32 (32-bit) calculated value is used to validate a Message. CRC-32 protects the data integrity of the Message. The CRC-32 requirements as defined as follows:

1. The CRC-32 **Shall** be calculated for all bytes of the Payload except for the Preamble, SOP*, and EOP.
2. The CRC-32 polynomial **Shall** be = 04C1_1DB7h.
3. The CRC-32 Initial value **Shall** be = FFFF_FFFFh.
4. The CRC-32 calculation **Shall** begin at byte 0, bit 0 and continue to bit 7 of each byte of the Packet.
5. The remainder of CRC-32 **Shall** be complemented.
6. The residual of CRC-32 **Shall** be C704 DD7Bh.

Note: The CRC implementation is identical to the one used in [USB3].

5.3.1.1.5. End-Of-Packet (EOP)

1. The EOP **Shall** be a single EOP K-code as defined in [Figure 5.1](#).
2. The EOP **Shall** mark the end of the CRC calculation and **Shall** not be included in the CRC calculation.

5.3.1.2. Resets

A [Hard Reset](#) or [Cable Reset](#) transmission is sent upon Request from the Protocol Layer and does not include any Payload data. The format of a [Hard Reset](#) or a [Cable Reset](#) transmission is defined in [Figure 5.2](#). The K-codes transmitted for each is shown in [Table 5.4](#).

Figure 5.2. Line Format for Hard Reset and Cable

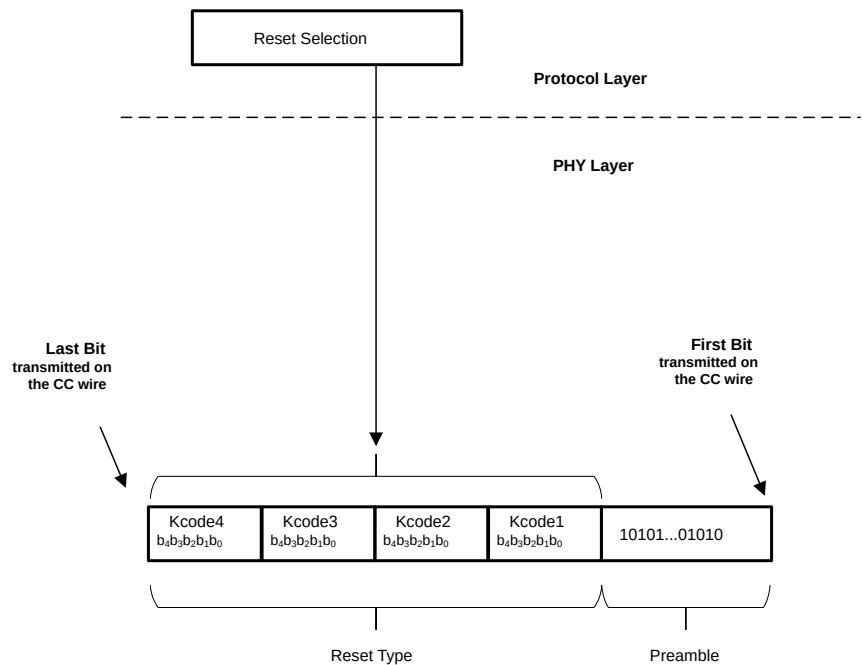


Table 5.4. K-codes for Resets

Name	K-code1	K-code2	K-code3	K-code4
Hard Reset	RST-1	RST-1	RST-1	RST-2
Cable Reset	RST-1	Sync-1	RST-1	Sync-3

5.3.1.2.1. Hard Reset

When the Protocol Layer requests a [Hard Reset](#), the procedure for sending a Hard Reset is as follows:

- If the PHY Layer is currently sending a Message, the Message **Shall** be interrupted by sending an EOP K-code and discarding the rest of the Message.
- If CC is not Idle, the PHY **Shall** wait for it to become Idle (see [Section 5.2.2.1](#)).
- The PHY **Shall** wait [tInterFrameGap](#) after the CC bus becomes Idle and send the [Hard Reset](#) Signaling.
- The transmitter **Shall** disable the channel (i.e., stop sending and receiving), reset the PHY Layer and inform the Protocol Layer that the PHY Layer has been reset.
- The PHY shall re-enable the channel when requested by the Protocol Layer.

5.3.2. Reception

The following describes the Message reception:

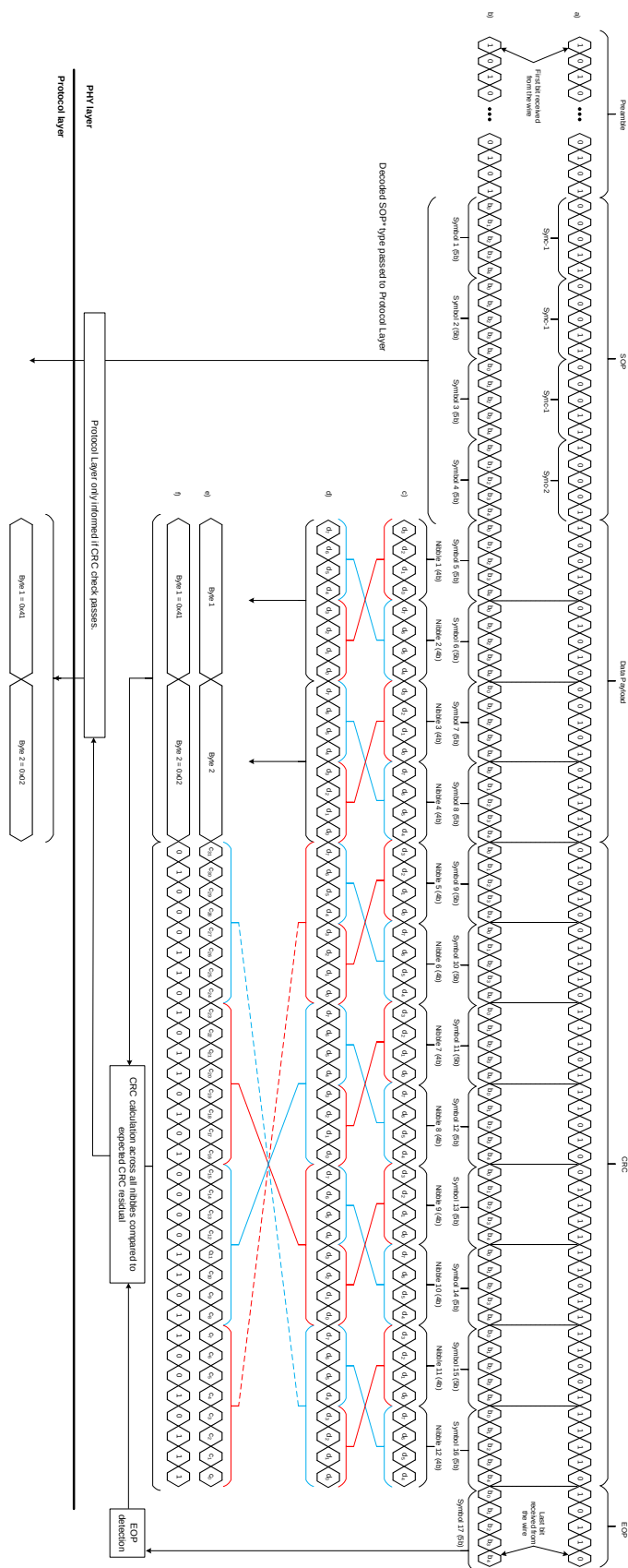
1. When enabled, the PHY Layer **Shall** be able to receive [Hard Resets](#).
2. The Protocol Layer **Shall** inform the PHY Layer which SOP* sequences to receive, and whether or not to receive Cable Resets.
3. The PHY Layer **Shall** detect an incoming transmission and decode the symbols starting with K-code1

4. The PHY shall report what is received to the Protocol Layer as necessary.
5. If the Protocol Layer has not configured the PHY Layer to process the SOP* type for the incoming transmission, then the PHY Layer **Shall** ignore the incoming transmission.
6. Otherwise, any time the PHY Layer receives a transmission it **Shall** inform the Protocol Layer of the type of Frame received ([Hard Reset](#), [Cable Reset](#), or SOP* type) and provide the Payload data (if any).

The PHY Layer decodes the Packet as shown in [Figure 5.3](#) and outlined in following steps:

1. The incoming transmission arrives as Symbols.
2. The data is 4b5b decoded into nibbles.
3. The nibble data is formatted into bytes.
4. Arriving bytes are passed through the CRC calculation (until an EOP Idle condition is detected).
5. If CRC is correct, the SOP* type and Payload data are passed to the Protocol Layer .

Figure 5.3. Receive Packet Interpretation.



5.3.2.1. Preamble

The Preamble may be used to wake the receiver. The receiver may process as much or as little of the Preamble as necessary to allow it to decode the received K codes.

5.3.2.2. Ordered Sets (K-codes)

The K-codes following the Preamble form an Ordered Set. The following rules are used:

1. The PHY Layer **Shall** decode the K-codes and compare to a list of ordered sets to determine whether to continue processing or ignore the rest of the transmission. The two kinds of ordered sets are SOP* sequences listed in [Table 5.3](#) and Reset sequences listed in [Table 5.4](#).
2. The PHY Layer **Shall** report the detected ordered set to the Protocol Layer, unless it has been configured to ignore that ordered set.
3. The receiver **Shall** search for all four K-codes.
4. When the receiver finds all four K-codes in the correct place, it **Shall** interpret this as a **Valid** ordered set.
 - a. When the receiver finds three out of four K-codes in the correct place, it **May** interpret this as a **Valid** ordered set.
 - b. The receiver **Should** ensure that all four K-codes are **Valid** to avoid ambiguity in detection (see [Table 5.5](#)).

Table 5.5. Validation of Ordered Sets

	1st code	2nd code	3rd code	4th code
Valid ¹	Corrupt	K-code	K-code	K-code
Valid ¹	K-code	Corrupt	K-code	K-code
Valid ¹	K-code	K-code	Corrupt	K-code
Valid ¹	K-code	K-code	K-code	Corrupt
Valid ² (perfect)	K-code	K-code	K-code	K-code
Invalid (example)	K-code	Corrupt	K-code	Corrupt
1. May be interpreted as a Valid ordered set. 2. Shall be interpreted as a Valid ordered set.				

5.3.2.3. Payload Processing

The following requirements apply to processing the Payload:

1. After detecting an SOP* sequence which has been configured to process, the PHY Layer **Shall** decode the trailing 5b symbols until it detects the EOP K-code or the CC-line becomes Idle (see [Section 5.2.2.1](#)). The PHY Layer **Shall** monitor for invalid 5b symbols during this step.
2. After the EOP is detected, the PHY Layer **Shall** decode each 5b symbol of the Payload into a 4b nibble, and then form the nibbles into bytes.
3. The CRC-residual **Shall** be checked with the Payload data. The EOP is excluded from the CRC check.
4. If the CRC is not good or any error was detected while decoding the 5b symbols or the CC-line became Idle before an EOP was detected, the whole transmission **Shall** be **Discarded**.
5. If the CRC is good, the SOP* and the decoded data are passed to the Protocol Layer.

5.3.3. Electrical Characteristics for Transmission and Reception

This section describes how the bit values are translated into electrical signals on the CC wire.

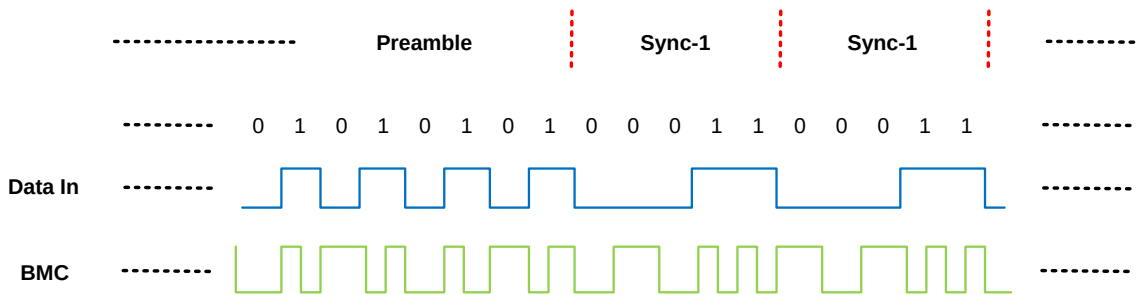
Bi-phase Mark Coding (BMC) is the PHY Layer Signaling Scheme for carrying USB Power Delivery Messages.

Bi-phase Mark Coding is a Version of Manchester coding (see [IEC 60958-1]). In BMC, there is a transition at the start of every bit time (UI) and there is a second transition in the middle of the UI when a 1 is transmitted. BMC is effectively DC balanced, (each 1 is DC balanced and two successive zeros are DC balanced, regardless of the number of intervening 1's). It has bounded disparity (limited to 1 bit over an arbitrary Packet, so a very low DC level).

Figure 5.4 illustrates Bi-phase Mark Coding. This example shows the transition from a Preamble to the Sync-1 K-codes of the SOP Ordered Set at the start of a Message.

Note: Other K-codes can occur after the Preamble for Signaling such as [Hard Reset](#) and [Cable Reset](#).

Figure 5.4. BMC Example



5.3.4. Encoding and Signaling

BMC uses DC coupled baseband Signaling on CC. Figure 5.5 shows a block diagram for a Transmitter and Figure 5.6 shows a block diagram for the corresponding Receiver.

Figure 5.5. BMC Transmitter Block Diagram

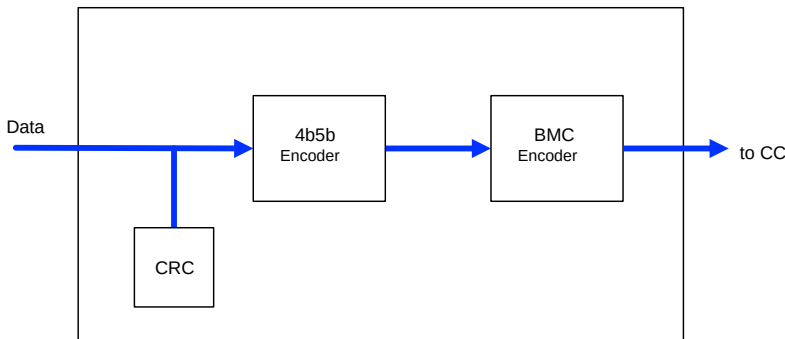
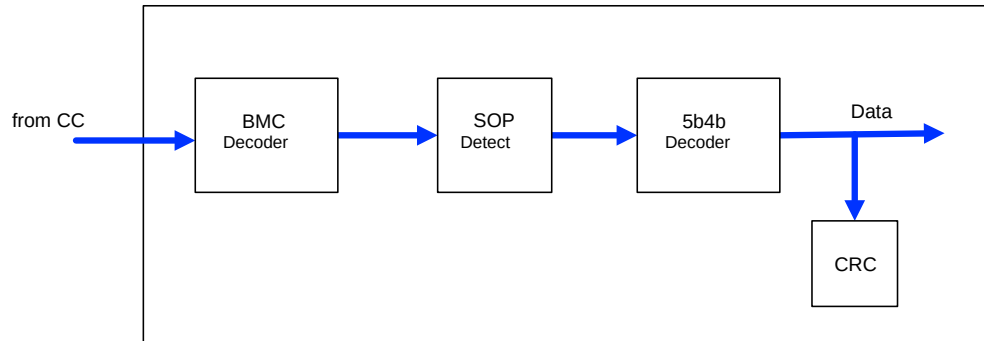


Figure 5.6. BMC Receiver Block Diagram

The following requirements apply to the transmitting and receiving of the signal:

1. The USB PD baseband signal **Shall** be driven on the CC wire with a level [vSwing](#) by a slew-rate limited tristate driver (see min rise/fall time in [Section 5.5.2](#)). This slow rate limiting reduces coupling to adjacent signal wires and can be performed with driver design or an RC filter at the driver output. See [Figure 5.7](#).
2. When sending the Preamble, the transmitter **Shall** start by transmitting a low level. The transmitter **May** vary the start of the Preamble by [tStartDrive](#)_{min}.
3. The receiver **Shall** tolerate the loss of the first edge.
4. The transmitter **Shall** terminate the final bit of the Frame by a trailing edge to help ensure that the receiver clocks the final bit. If the trailing edge results in the transmitter driving CC low. See [Figure 5.8](#) and [Figure 5.8](#).
5. The transmitter **Shall** continue to drive CC low for [tHoldLowBMC](#) and **Should** release CC to high impedance as soon as possible after min [tHoldLowBMC](#) and **Shall** release CC by max [tEndDriveBMC](#).
6. If the trailing edge results in the transmitter driving CC high (i.e., the final half-UI of the Frame is low, see [Figure 5.10](#) and [Figure 5.11](#)):
 - a. The transmitter **Shall** continue to drive CC high for 1 UI.
 - b. Then the transmitter **Shall** drive CC low for [tHoldLowBMC](#) and **Should** release CC to high impedance as soon as possible after min [tHoldLowBMC](#) and **Shall** release CC by max [tEndDriveBMC](#).

[Figure 5.9](#), [Figure 5.10](#), and [Figure 5.11](#) also show the [tInterFrameGap](#) before the next Packet. Beyond the [tInterFrameGap](#), there is no requirement to maintain a timing phase relationship between back-to-back Packets.

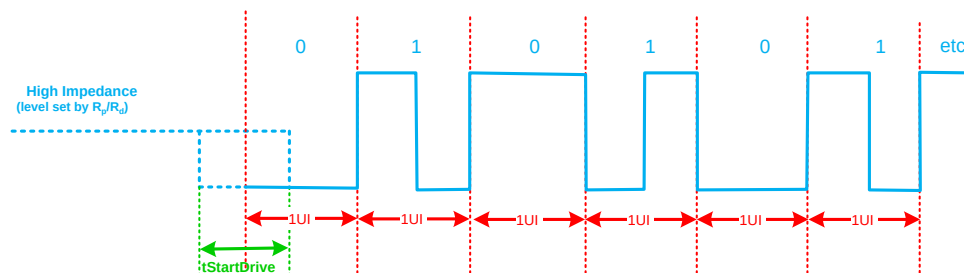
Figure 5.7. BMC Encoded Start of Preamble

Figure 5.8. Transmitting or Receiving BMC Encoded Frame Terminated by Zero with High-to-Low Last Transition

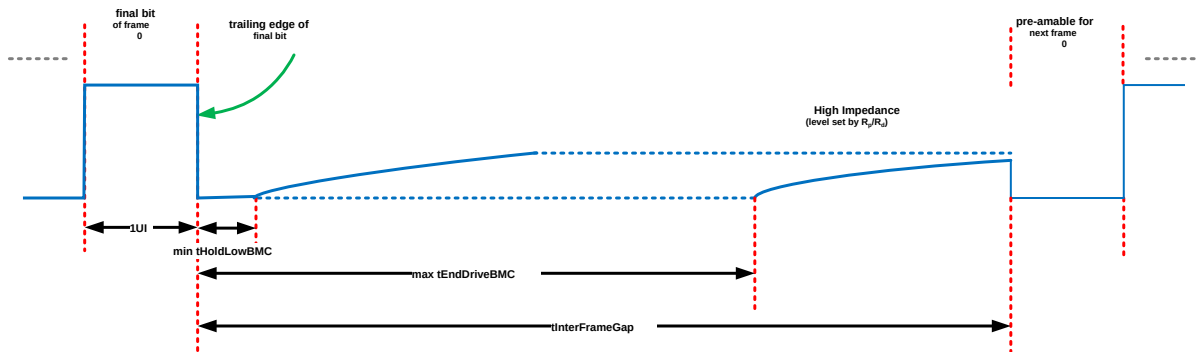


Figure 5.9. Transmitting or Receiving BMC Encoded Frame Terminated by One with High-to-Low Last Transition

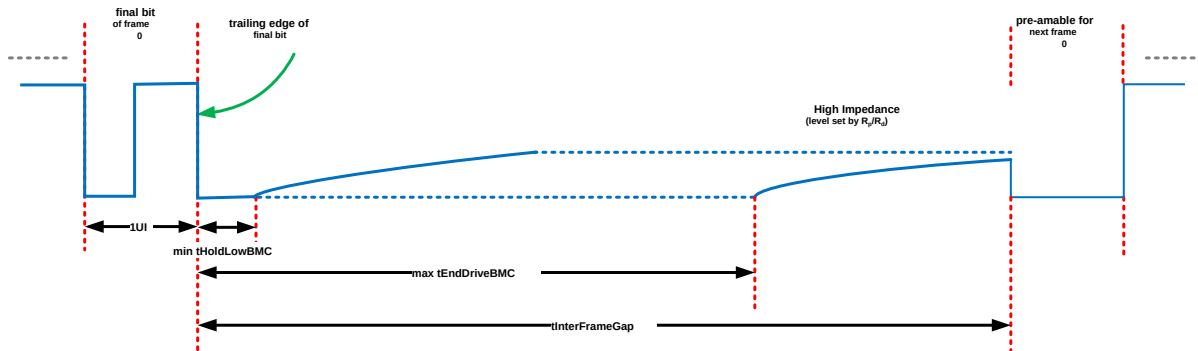


Figure 5.10. Transmitting or Receiving BMC Encoded Frame Terminated by Zero with Low-to-High Last Transition

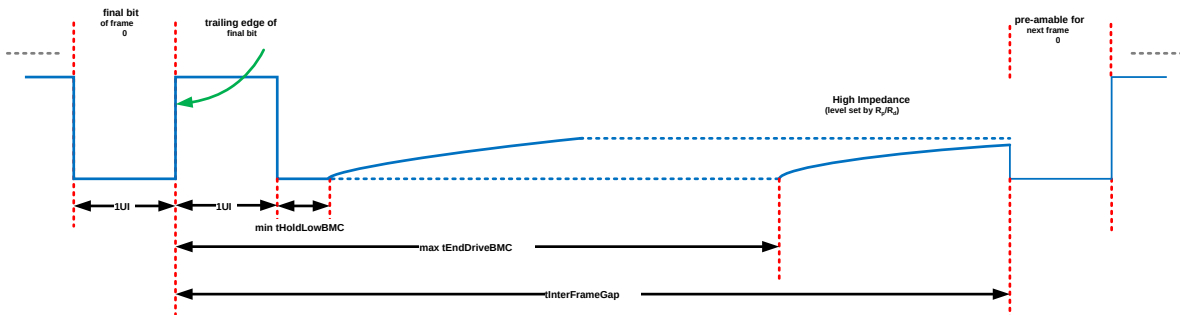
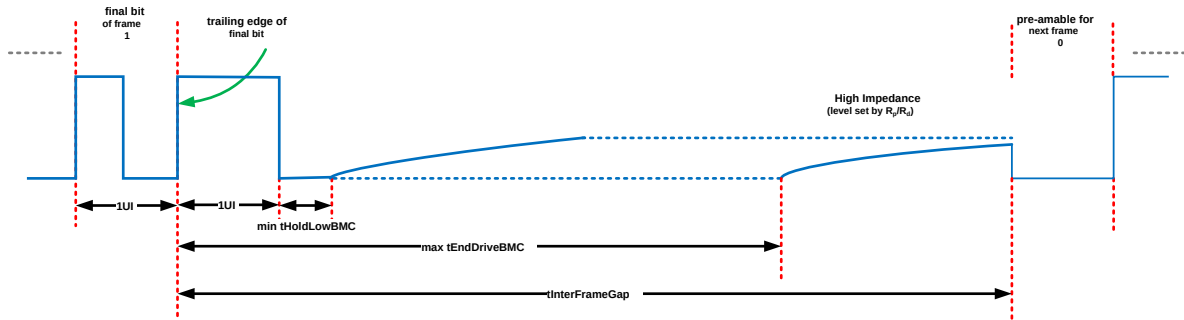


Figure 5.11. Transmitting or Receiving BMC Encoded Frame Terminated by One with High-to-Low Last Transition



5.3.4.1. Transmit Mask

The transmit mask rules are defined by the following:

1. The transmitted signal **Shall Not** violate the masks defined in [Figure 5.12](#), [Figure 5.13](#), [Table 5.6](#) and [Table 5.7](#) at the output of a load equivalent to the cable model and receiver load model described in [Section 5.3.4.3](#).
2. The masks apply to the full range of R_p/R_d values as defined in [USB-C].
3. The transmitted signal **Shall** have a rise time $\leq t_{Rise}$ and is enforced by the Tx inner masks.
4. The transmitted signal **Shall** have a fall time $\leq t_{Fall}$ and is enforced by the Tx inner masks.
5. The measurement of the transmit mask **Shall** not include ground offset when current is flowing in the cable and is measured at the connector.

Figure 5.12. BMC Tx 'ONE' Mask.

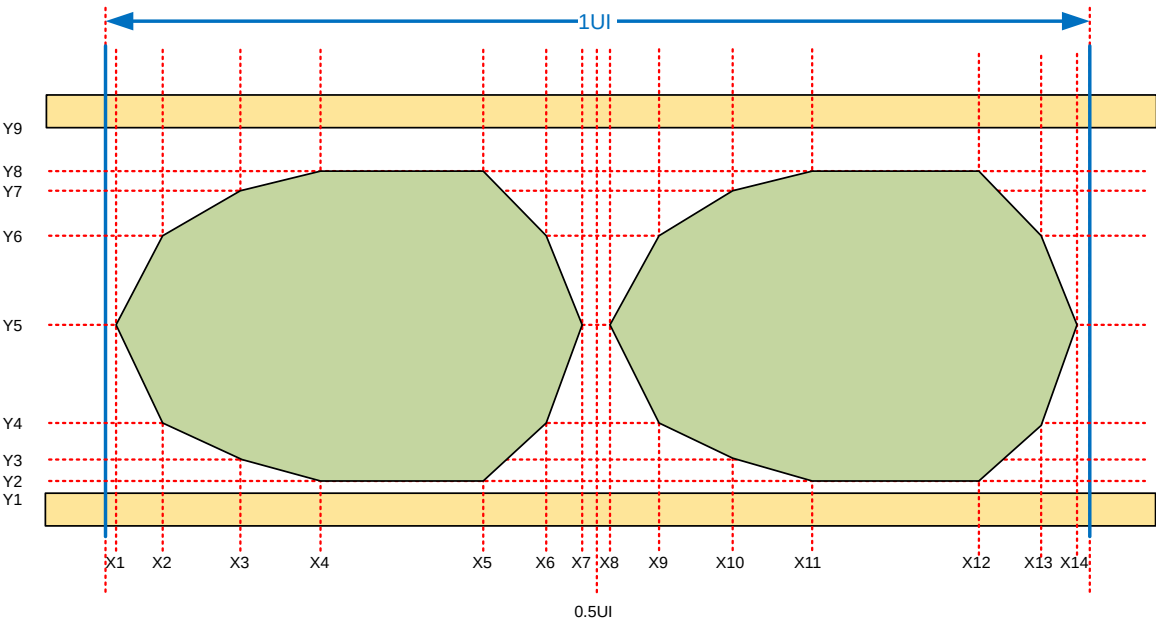


Figure 5.13. BMC Tx 'ZERO' Mask.

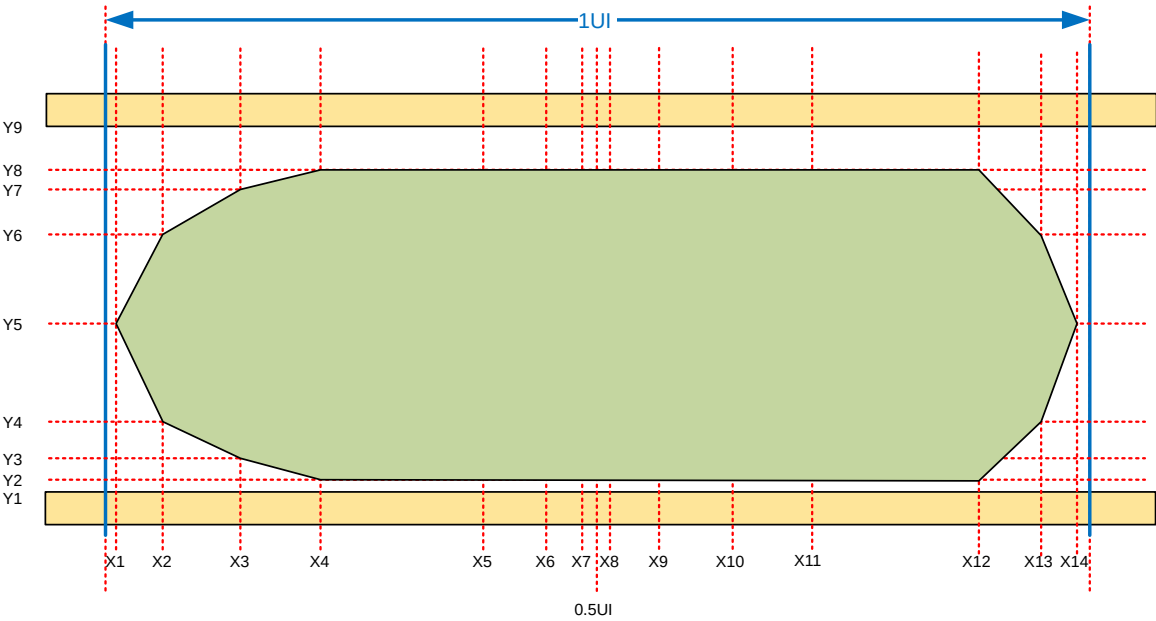


Table 5.6. BMC Tx Mask Definition, X Values

Parameter Name	Value	Unit	Description
X1Tx	0.015	UI	Left Edge of Mask see Figure 5.12 and Figure 5.13
X2Tx	0.07	UI	see Figure 5.12 and Figure 5.13
X3Tx	0.15	UI	see Figure 5.12 and Figure 5.13
X4Tx	0.25	UI	see Figure 5.12 and Figure 5.13
X5Tx	0.35	UI	see Figure 5.12
X6Tx	0.43	UI	see Figure 5.12
X7Tx	0.485	UI	see Figure 5.12
X8Tx	0.515	UI	see Figure 5.12
X9Tx	0.57	UI	see Figure 5.12
X10Tx	0.65	UI	see Figure 5.12
X11Tx	0.75	UI	see Figure 5.12
X12Tx	0.85	UI	see Figure 5.12 and Figure 5.13
X13Tx	0.93	UI	see Figure 5.12 and Figure 5.13
X14Tx	0.985	UI	Right Edge of Mask see Figure 5.12 and Figure 5.13

Table 5.7. BMC Tx Mask Definition, Y Values

Parameter Name	Value	Unit	Description
Y1Tx	-0.075	V	Lower bound of outer mask see Figure 5.12 and Figure 5.13
Y2Tx	0.075	V	Lower bound of inner mask see Figure 5.12 and Figure 5.13
Y3Tx	0.15	V	see Figure 5.12 and Figure 5.13
Y4Tx	0.325	V	see Figure 5.12 and Figure 5.13
Y5Tx	0.5625	V	Inner mask vertical midpoint see Figure 5.12 and Figure 5.13
Y6Tx	0.8	V	see Figure 5.12 and Figure 5.13
Y7Tx	0.975	V	see Figure 5.12 and Figure 5.13
Y8Tx	1.04	V	see Figure 5.12 and Figure 5.13
Y9Tx	1.2	V	Upper bound of outer mask see Figure 5.12 and Figure 5.13

5.3.4.2. Receive Masks

The receive mask requirements are defined by the following:

1. When acting as a Source, a Port **Shall** be capable of receiving a signal that complies with the receive mask defined in [vNoiseActive](#) between power neutral and Source offsets. [Figure 5.14](#), [Figure 5.15](#) and [Table 5.8](#). The Source Rx mask is bounded by sweeping a Tx mask compliant signal, with added [vNoiseActive](#) between power neutral and Source offsets.
2. When acting as a Sink, a Port **Shall** be capable of receiving a signal that complies with the mask defined in [Figure 5.16](#), [Figure 5.17](#), [Table 5.8](#), and [vNoiseActive](#) between power neutral and Sink offsets. The Sink

Rx mask is bounded by sweeping a Tx mask compliant signal, with added [vNoiseActive](#) between power neutral and Sink offsets.

3. When power neutral (neither sinking nor sourcing), a Port **Shall** be capable of receiving a signal that complies with the mask defined in [Figure 5.18](#), [Figure 5.19](#) and [Table 5.8](#).
4. Cable Plugs **Shall** meet the receiver requirements for both a Source and a Sink during any transmission using the BMC Signaling Scheme.
5. The receiver sensitivity **Shall** be set such that the receiver does not treat noise on an un-driven signal path as an incoming signal. Signal amplitudes below [vNoiseIdle](#) max **Shall** be treated as noise when BMC is Idle.
6. The receiver **Shall** tolerate the loss of the first edge

The parameters used in the masks are specified to be appropriate to either edge triggered or oversampling receiver implementations.

The masks are defined for 'ONE' and 'ZERO' separately as BMC enforces a transition at the midpoint of the Unit Interval while a 'ONE' is transmitted.

The Rx masks are defined to bound the Rx noise after the Rx bandwidth limiting filter with the time constant [tRxFilter](#) has been applied.

The boundaries of Rx outer mask, [Y1Rx](#) and [Y5Rx](#), are specified according to [vSwing](#) max and accommodate half of [vNoiseActive](#) from cable noise coupling and the signal offset [vIRDropGNDC](#) due to the ground offset when current is flowing in the cable.

The vertical dimension of the Rx inner mask, [Y4Rx](#) - [Y2Rx](#), for power neutral is derived by reducing the vertical dimension of the Tx inner mask, [Y7Tx](#) - [Y3Tx](#), at time location [X3Tx](#) by [vNoiseActive](#) to account for cable noise coupling. The received signal is composed of a waveform compliant to the Tx mask plus [vNoiseActive](#).

The vertical dimension of the Rx inner mask for sourcing power is derived by reducing the vertical dimension of the Tx inner mask by [vNoiseActive](#) and [vIRDropGNDC](#) to account for both cable noise coupling and signal DC offset.

The received signal is composed of a waveform compliant to the Tx mask plus the maximum value of [vNoiseActive](#) plus [vIRDropGNDC](#) where the [vIRDropGNDC](#) value transitions between the minimum and the maximum values as allowed in this spec.

The vertical dimension of the Rx inner mask for sinking power is derived by reducing the vertical dimension of the Tx inner mask by [vNoiseActive](#) max and [vIRDropGNDC](#) max for account for both cable noise coupling and signal DC offset. The received signal is composed of a waveform compliant to the Tx mask plus the maximum value of [vNoiseActive](#) plus [vIRDropGNDC](#) where the [vIRDropGNDC](#) value transitions between the minimum and the maximum values as allowed in this spec.

The center line of the Rx inner mask, [Y3Rx](#), is at half of the nominal [vSwing](#) for power neutral, and is shifted up by half of [vIRDropGNDC](#) max for sourcing power and is shifted down by half of [vIRDropGNDC](#) max for sinking power.

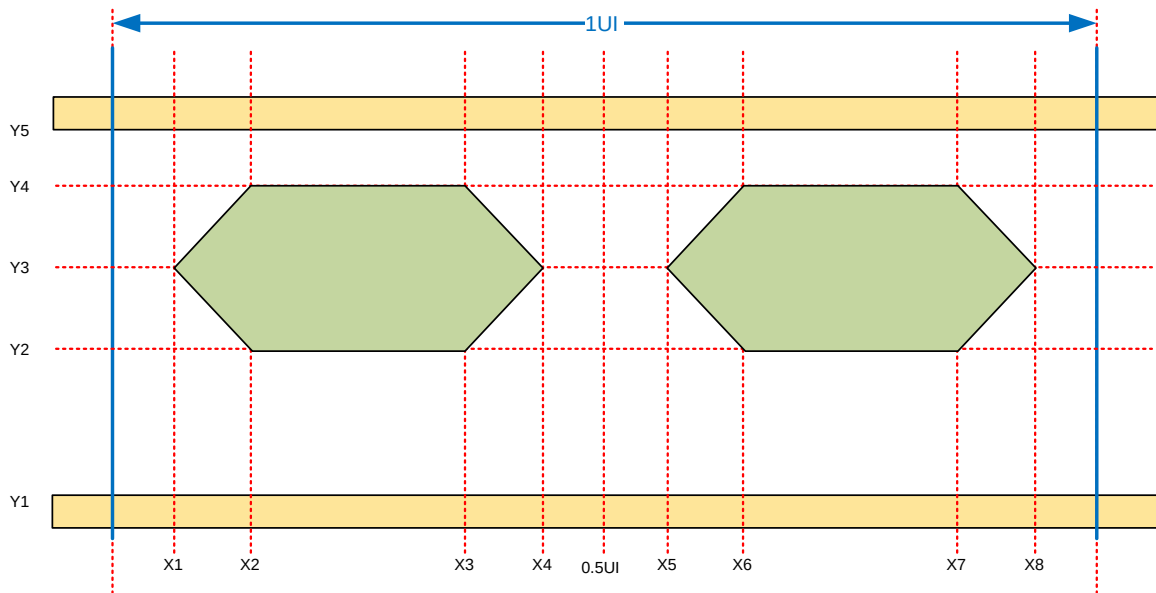
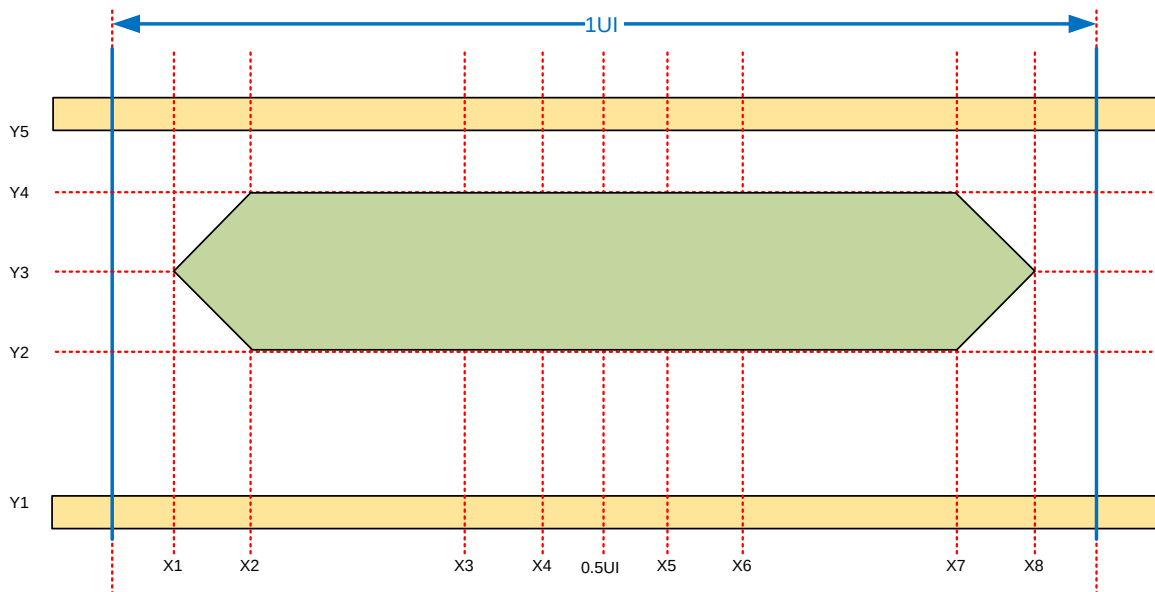
Figure 5.14. BMC Rx 'ONE' Mask when Sourcing Power**Figure 5.15. BMC Rx 'ZERO' Mask when Sourcing Power**

Figure 5.16. BMC Rx 'ONE' Mask when Sinking Power

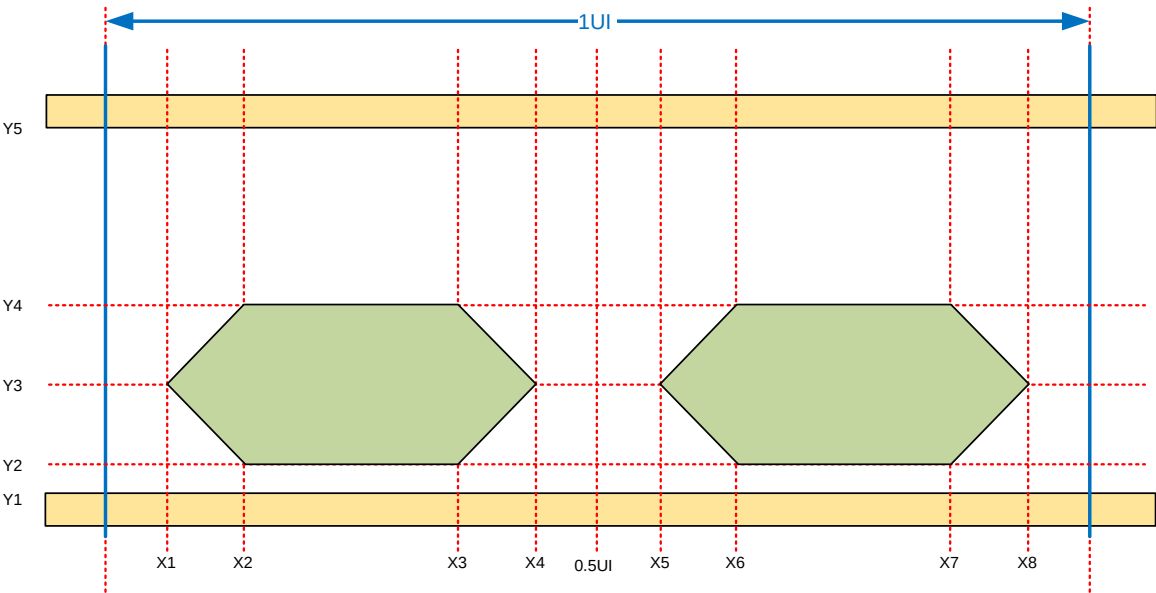


Figure 5.17. BMC Rx 'ZERO' Mask when Sinking Power

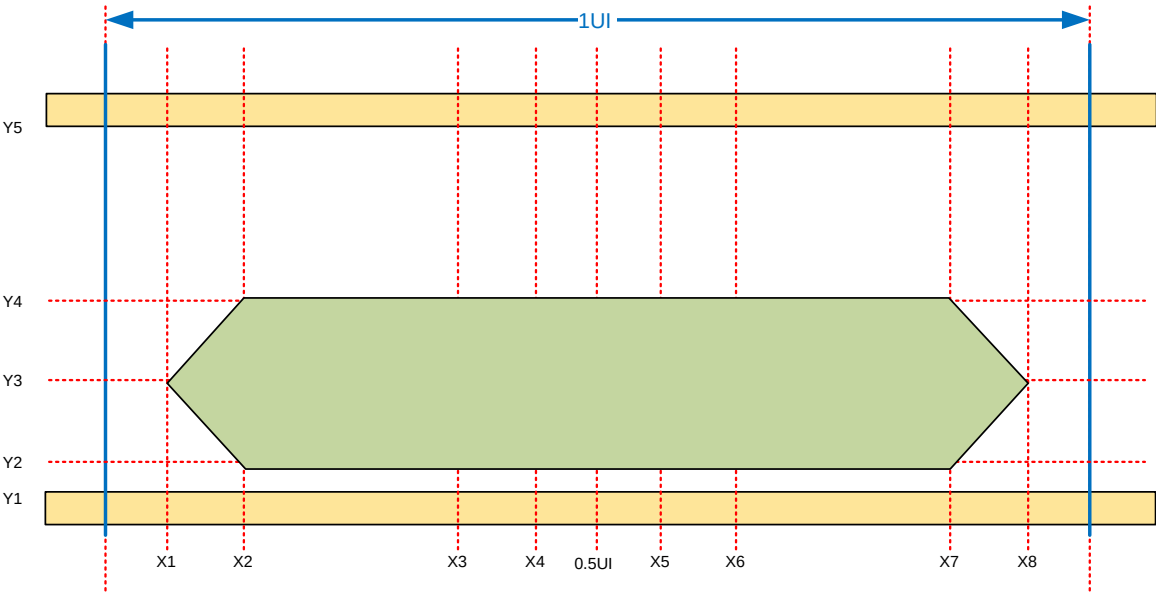
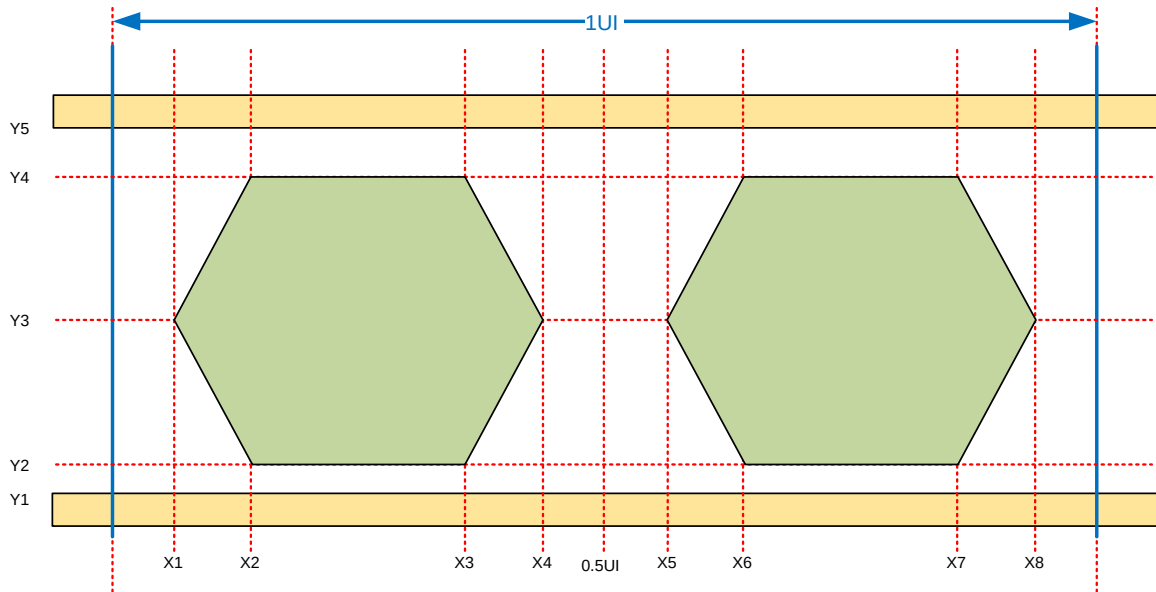
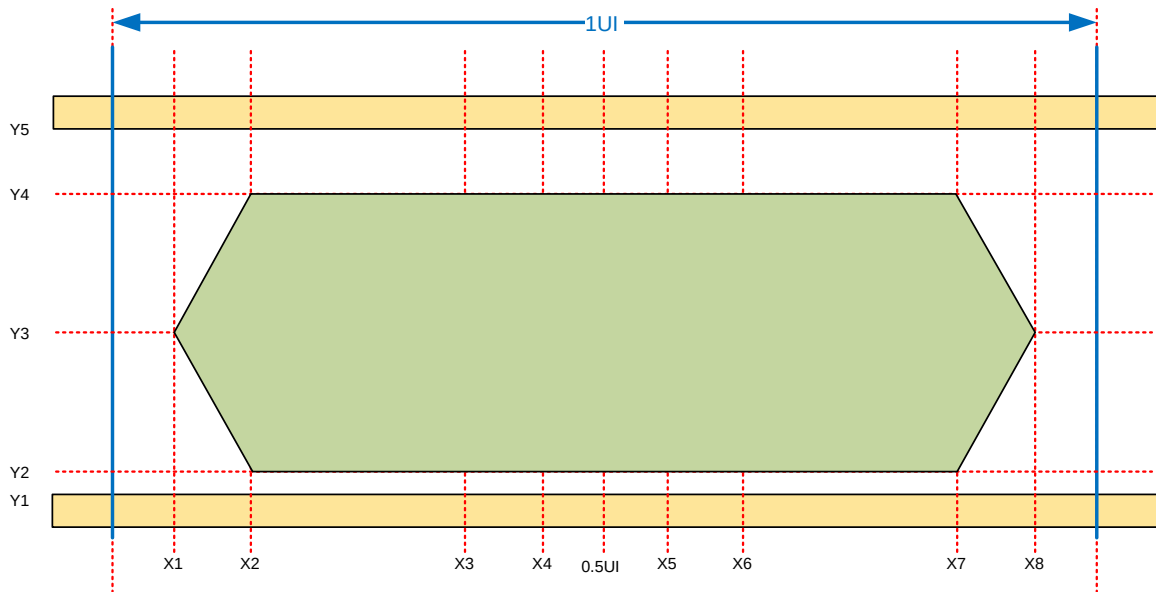


Figure 5.18. BMC Rx 'ONE' Mask when Power Neutral**Figure 5.19. BMC Rx 'ZERO' Mask when Power Neutral****Table 5.8. BMC Rx Mask Definition**

Parameter Name	Value	Unit	Description
X1Rx	0.07	UI	Left Edge of Mask see Figure 5.14 through Figure 5.19

Parameter Name	Value	Unit	Description
X2Rx	0.15	UI	Top Edge of Mask see Figure 5.14 through Figure 5.19
X3Rx	0.35	UI	see Figure 5.14 through Figure 5.19
X4Rx	0.43	UI	see Figure 5.14 through Figure 5.19
X5Rx	0.57	UI	see Figure 5.14 through Figure 5.19
X6Rx	0.65	UI	see Figure 5.14 through Figure 5.19
X7Rx	0.85	UI	see Figure 5.14 through Figure 5.19
X8Rx	0.93	UI	see Figure 5.14 through Figure 5.19
Y1Rx	-0.3325	V	Lower bound of Outer Mask see Figure 5.14 through Figure 5.19
Y2Rx	$Y3Rx - 0.205$ when sourcing power ¹ or sinking power ¹ . $Y3Rx - 0.33$ when power neutral ¹ .	V	Lower Bound of Inner Mask see Figure 5.14 through Figure 5.19
Y3Rx	0.6875 Sourcing Power ¹ . 0.5625 Power Neutral ¹ . 0.4375 Sinking Power ¹ .	V	Center line of Inner Mask see Figure 5.14 through Figure 5.19
Y4Rx	$Y3Rx + 0.205$ when sourcing power ¹ or sinking power ¹ . $Y3Rx + 0.33$ when power neutral ¹ .	V	Upper bound of Inner mask see Figure 5.14 through Figure 5.19
Y5Rx	1.5325	V	Upper bound of the Outer mask see Figure 5.14 through Figure 5.19
1. The position of the center line of the Inner Mask is dependent on whether the receiver is Sourcing or Sinking power or is Power Neutral (see earlier in this section).			

5.3.4.3. Transmit Load Model

The transmit load requirements are defined by the following:

1. The transmitter load model **shall** be equivalent to the circuit outlined in [Figure 5.20](#) for a Source and [Figure 5.21](#) for a Sink. It is formed by the concatenation of a cable load model and a receiver load model. See [USB-C] for details of the Rp and Rd resistors.

The parameters zCable_CC, tCableDelay_CC and cCablePlug_CC are defined in [USB-C].

The transmitter system components rOutput and cShunt are illustrated for **Informative** purposes, and do not form part of the transmitter load model. See [Section 5.5.2](#) for a description of the transmitter system design.

The transmitter load model assumes that there are no other return currents on the ground path.

2. The value of the modeled cable inductance, La, (in nH) **shall** be calculated from the following formula:

$$La = tCableDelay_CC_{max} \times zCable_CC_{min}$$

where tCableDelay_CC is the modeled signal propagation delay through the cable, and zCable_CC is the modeled cable impedance.

The modeled cable inductance is 640nH for a cable with zCable_CCmin = 32Ω and tCableDelay_CCmax = 20ns.

3. The value of the modeled cable capacitance, Ca, (in pF) **shall** be calculated from the following formula:

$$Ca = tCableDelay_CC_{max} \div zCable_CC_{min}$$

The modeled cable capacitance is $C_a = 625\text{pF}$ for a cable with $z_{\text{Cable_CCmin}} = 32\Omega$ and $t_{\text{CableDelay_CCmax}} = 20\text{ns}$. Therefore, $C_a \div 2 = 312.5\text{pF}$.

4. `cCablePlug_CC` models the capacitance of the plug at each end of the cable. `cReceiver` models the capacitance of the receiver and `rBmcRx` models the receiver input impedance.

. The maximum values **Shall** be used in each case.

Figure 5.20. Transmitter Load Model for BMC Tx from a Source

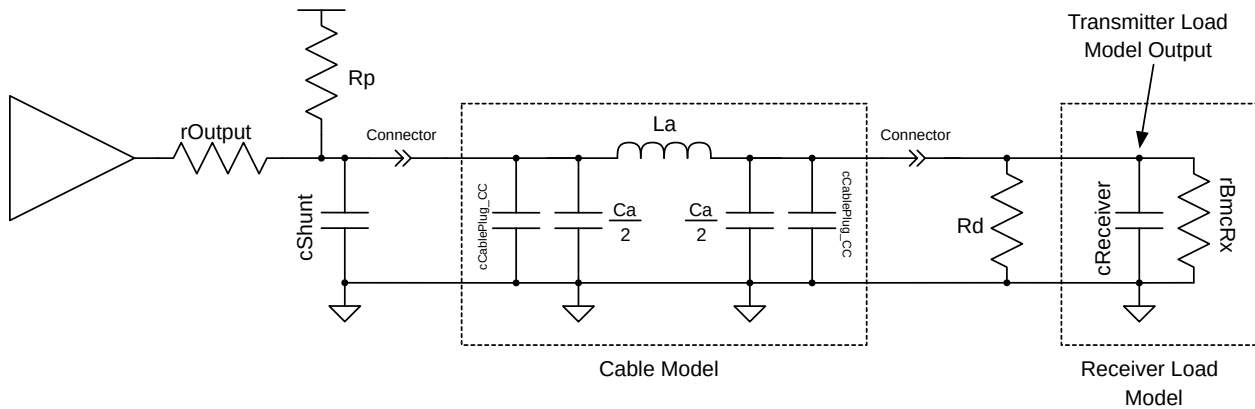
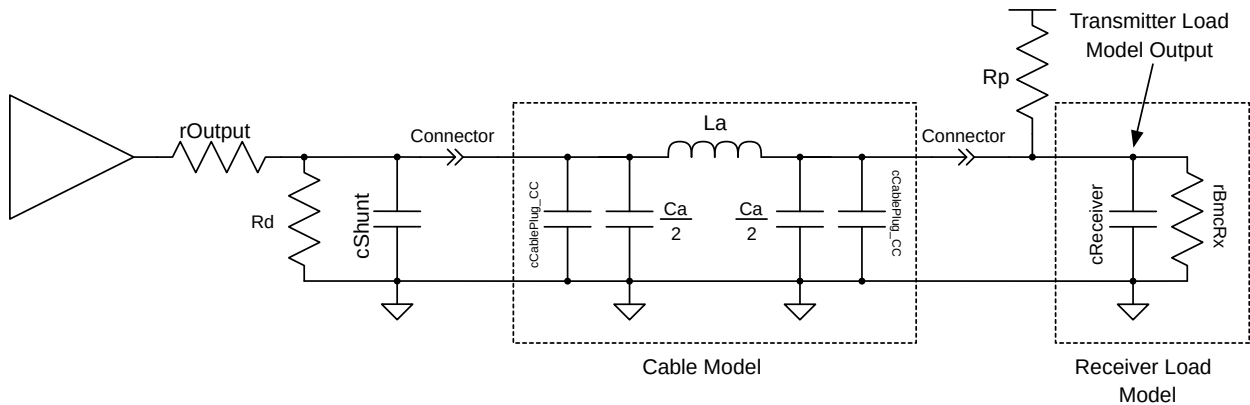


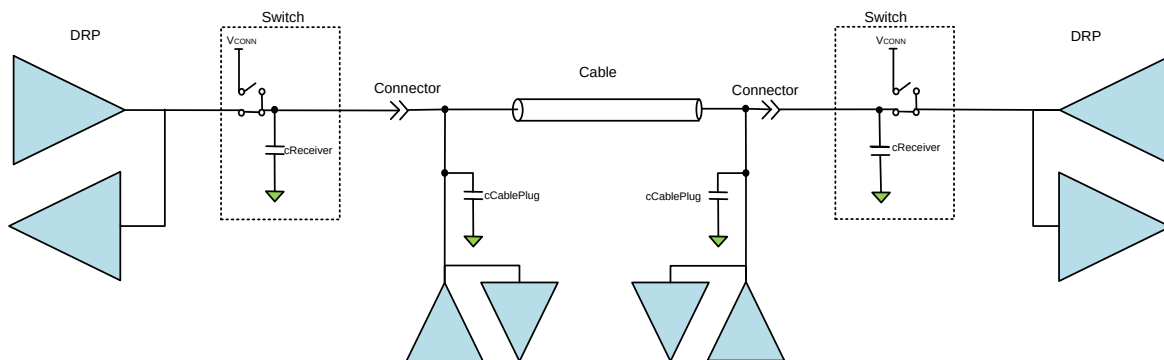
Figure 5.21. Transmitter Load Model for BMC Tx from a Sink



5.3.4.4. Cable Plug Transceivers.

The BMC Signaling Scheme is suitable for use in Multi-Drop configurations containing multiple BMC transceivers Connected to the CC wire. Cable Plugs with transceivers on the CC are such a case.

[Figure 5.22](#) illustrates a typical Multi-Drop configuration with two DRPs and cable transceivers.

Figure 5.22. Example Multi-Drop Configuration showing two DRPs and two Cable Transceivers

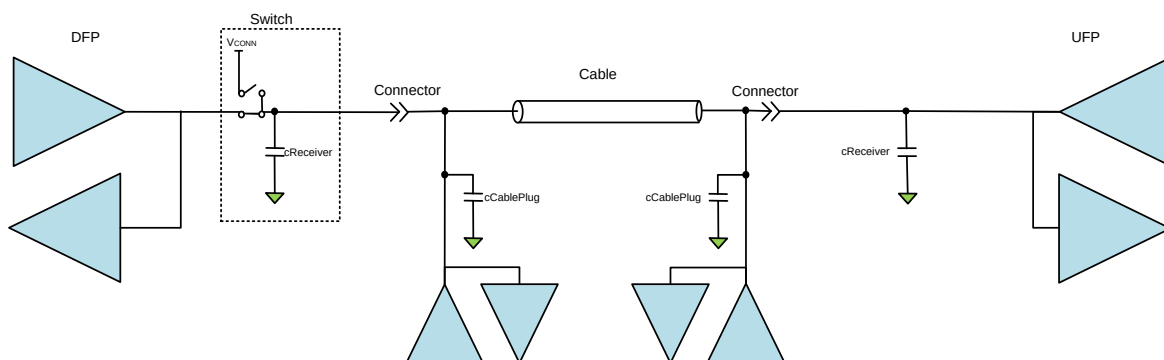
The following requirements apply to Cable Plug transceivers:

1. The Cable Plug transceiver **Shall** obey all the electrical characteristics specified in this section except for those relating to capacitance.
2. The maximum capacitance allowed for the Cable Plug transceiver on the CC wire, when not driving the line, **Shall** be cCablePlug_CC as defined in [USB-C].
3. There are no constraints as to the distance of the Cable Plug transceiver from the end of the wire. The Cable Plug transceiver(s) **May** be located anywhere along the cable including the plugs.
4. The Cable Plug transceiver **Shall** account for any ground offset based on its location.
5. The Cable Plug transceiver shall not add significant reflections.

5.3.4.5. Transceiver with VCONN Source Capability

It is possible to have a configuration at Attach where one or both Ports can be a VCONN Source. An example of a VCONN sourcing capable DFP Attached to a UFP without VCONN sourcing capability is shown in [Figure 5.23](#).

1. The capacitance on the CC pin for a Port that is able to supply VCONN but is not actively supplying VCONN **Shall** still meet the [cReceiver](#) requirements when not transmitting.

Figure 5.23. Example Multi-Drop Configuration showing a DFP and UFP with a VCONN Source

5.3.4.6. Capacitance when not transmitting

The following rules apply for capacitance when not transmitting:

1. [cReceiver](#) is the capacitance that a DFP or UFP **Shall** present on the CC line when the DFP or UFP's receiver is not transmitting on the line.
2. The transmitter **May** have more capacitance than [cReceiver](#) while driving the CC line, but **Shall** meet the waveform mask requirements.
3. Once transmission is complete, the transmitter **Shall** disengage capacitance in excess of [cReceiver](#) from the CC wire within [tInterFrameGap](#).

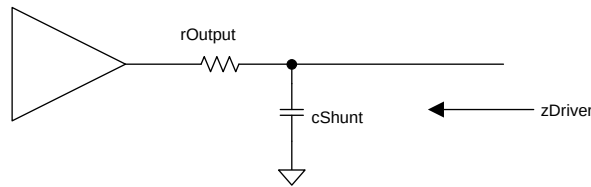
5.3.4.7. Source Output Impedance

Source output impedance [zDriver](#) is determined by the driver resistance and the shunt capacitance of the Source and is hence a frequency dependent term. [zDriver](#) impacts the noise ingress in the cable. It is specified such that the noise at the Receiver is bounded.

[zDriver](#) is defined by the following equation:

$$\text{zDriver} = r_{\text{Output}} \div (1 + s \times r_{\text{Output}} \times c_{\text{Shunt}})$$

Figure 5.24. Transmitter diagram illustrating zDriver



1. c_{Shunt} **Shall Not** cause a violation of [cReceiver](#) when not transmitting.

5.3.4.8. Bit Rate Drift

Limits on the drift in [fBitRate](#) are set to help low-complexity receiver implementations.

[fBitRate](#) is the reciprocal of the average bit duration from the previous 32 bits at a given portion of the Packet.

1. The change in [fBitRate](#) during a Packet **Shall** be less than [pBitRate](#).
2. The reference bit rate (refBitRate) is the average [fBitRate](#) over the last 32 bits of the *Preamble*. [fBitRate](#) throughout the Packet, including the EOP, **Shall** be within [pBitRate](#) of refBitRate . [pBitRate](#) is expressed as a percentage:

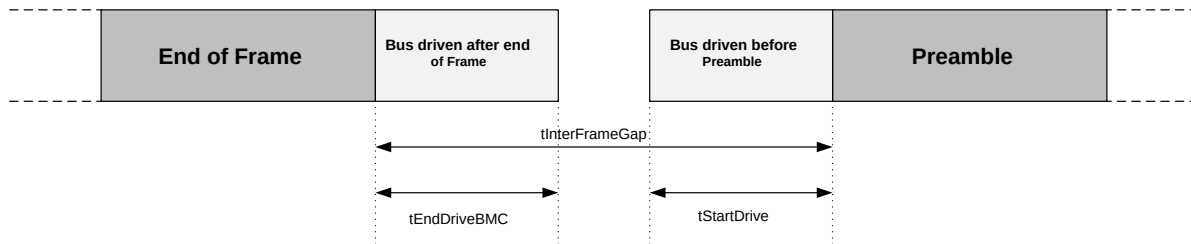
$$\text{pBitRate} = |\text{fBitRate} - \text{refBitRate}| \div \text{refBitRate} \times 100\%$$

3. The transmitter **Shall** have the same [pBitRate](#) for all Packet types. The BIST Carrier Mode and Bit Stream signals are continuous signals without a Payload.
4. When checking [pBitRate](#) any set of 1044 bits (20 bit SOP followed by 1024 PRBS bits) within a continuous signal **May** be considered as the part of the Packet following the Preamble and the 32 preceding bits considered to be the last 32 bits of the Preamble used to compute refBitRate .

5.3.4.9. Inter-Frame Gap

Figure 5.26, "Inter-Frame Gap Timings" illustrates the inter-Frame gap timings.

Figure 5.25. Inter-Frame Gap Timings



Inter-Frame Gap timings are governed by:

1. The transmitter **Shall** drive the CC bus for no longer than [tEndDriveBMC](#) after transmitting the final bit of the Frame.
2. Before starting to transmit the next Frame's Preamble the transmitter of the next Frame **Shall** ensure that it waits for [tInterFrameGap](#) after either:
 - a. Transmitting the previous Frame, for example sending the next Message in an AMS immediately after having sent a [GoodCRC Message](#), or
 - b. Receiving the previous Frame, for example when responding to a received Message with a [GoodCRC Message](#), or
 - c. Observing an Idle condition on CC (see [Section 5.2.2](#)). In this case the Port is waiting to initiate an AMS observes Idle (see [Section 5.2.2.1](#)) and then waits [tInterFrameGap](#) before transmitting the Frame. See also [Section 5.2.2](#) for details on when an AMS can be initiated.
 - d. The transmitter of the next Frame **May** vary the start of the Preamble by [tStartDrive](#) (see [Section 5.3.4](#)).

Also see [Section 5.3.4](#) for figures detailing the timings relating to transmitting, receiving, and observing Idle in relating to Frames.

5.3.4.10. Shorting of Transmitter Output

A Transmitter in a Port or Cable Plug **Shall** tolerate having its output be shorted to ground for [tFRSwapTx](#) (max). This is due to the potential for [Fast Role Swap](#) to be signaled by the Receiver while the Transmitter is in the process of transmitting (see: Chapter 10, [Fast Role Swap](#) (FRS) for details).

5.4. Built in Self-Test (BIST)

The following sections define BIST functionality which **Shall** be supported.

5.4.1. BIST Carrier Mode

In BIST Carrier Mode, the PHY Layer **Shall** send out a BMC encoded continuous string of alternating "1"s and "0"s. This enables the measurement of power supply noise and frequency drift. The Protocol Layer tells the PHY when to enter and exit this Mode.

Note: This transmission is a purely a sequence of alternating bits and **Shall Not** be formatted as a Packet. See also [Section 6.4.3](#).

5.5. PD Communications Physical Layer Parameters

5.5.1. BMC Common Parameters

The electrical requirements specified in [Table 5.9](#) **Shall** apply to both the transmitter and receiver.

Table 5.9. BMC Common Requirements

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
fBitRate	270	300	330	Kbps	Bit Rate
tUnitInterval	3.03		3.70	μs	Unit Interval ¹ (= $1 \div f\text{BitRate}$)
1. Denotes the time to transmit an unencoded data bit, not the shortest high or low times on the wire after encoding with BMC. A single data bit cell has duration of 1UI, but a data bit cell with value 1 will contain a centrally placed 01 or 10 transition in addition to the transition at the start of the cell.					

5.5.2. BMC Transmitter Parameters

The transmitter **Shall** meet the specifications defined in [Table 5.10](#).

Table 5.10. BMC Transmitter Parameters

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
pBitRate			0.25	%	Maximum difference between the bit-rate during the part of the Packet following the Preamble and the reference bit-rate. The reference bit rate is the average bit rate of the last 32 bits of the Preamble.
tEndDriveBMC			23	μs	Time to cease driving the line after the end of the last bit of the Frame. Min value is limited by tHoldLowBMC .
tFall	300			ns	Fall Time. 10% and 90% amplitude points when unloaded.
tHoldLowBMC	1			μs	Time to cease driving the line after the final high-to-low transition. Max value is limited by tEndDriveBMC .
tInterFrameGap	25			μs	Time from the end of last bit of a Frame until the start of the first bit of the next Preamble.
tRise	300			ns	Rise time. 10% and 90% amplitude points when unloaded.
tStartDrive	-1		1	μs	Time before the start of the first bit of the Preamble when the transmitter Shall start driving the line.
vSwing	1.05	1.125	1.2	V	Voltage Swing. Applies to both no load condition and loaded condition specified in Section 5.3.4.3 .
zDriver	33		75	Ω	Transmitter output impedance at the Nyquist frequency of [USB2] low speed (750 kHz) while the Source is driving the CC line.

5.5.3. BMC Receiver Parameters

The receiver **Shall** meet the specifications defined in [Table 5.11](#).

Table 5.11. BMC Receiver Parameters

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
cReceiver	200		600	pF	CC receiver capacitance. The DFP or UFP system Shall have capacitance within this range when not transmitting on the line.
nBER			10 ⁻⁶		Bit error rate, S/N = 25 dB
nTransitionCount	3				Number of transitions to be detected to declare CC bus non-Idle.
tRxFilter	100			ns	Rx bandwidth limiting filter (digital or analog). Time constant of a single pole filter to limit broad-band noise ingress ¹ .
tTransitionWindow	12		20	μs	Time window for detecting non-Idle
vIRDropGNDC			250	mV	Cable Ground IR Drop as specified in [USB-C].
vNoiseActive			165	mV	Peak-to-peak noise from VBUS, [USB2] and SBU lines after the Rx band- width limiting filter with the time constant tRx-Filter has been applied.
vNoiseIdle			300	mV	Peak-to-peak noise from VBUS, [USB2] and SBU lines after the Rx band- width limiting filter with the time constant tRxFilter has been applied.
rBmcRx	1			MΩ	Receiver Input Resistance
1. Broad-band noise ingress is due to coupling in the cable interconnect.					

Chapter 6. PD Communications Protocol Message Definitions

6.1. Overview

This chapter defines the different types of PD Messages and provides details on how they are constructed.

Refer to [Chapter 7](#) for details on Message usage.

6.1.1. Message Types

This specification defines three types of Messages:

- Control Messages that contain the 16-bit Message Header.
- [Data Messages](#) that contain a 16-bit Message Header, followed by between one and seven 32-bit data objects.
- [Extended Messages](#) that contain the 16-bit Message Header, followed by the 16-bit [Extended Message Header](#) and between 1 and 260 bytes of data.

Note: Data messages are word-oriented while extended messages are byte-oriented.

6.1.2. Message ID

All messages are sent with a Message Header that contains the 3-bit MessageID field. The MessageID is a 3-bit counter that is used to ensure duplicate messages are handled properly. Ports **Shall** keep a different MessageID counter for each supported SOP*.

Instead of using the MessageID counter, the MessageID field for [GoodCRC Message](#) **Shall** match the MessageID field of the received Message that the [GoodCRC Message](#) is acknowledging.

6.1.3. Specification Revision

To ensure interoperability with existing PDUSB products, all PDUSB products **Shall** support every PD Specification Revision starting from [PD2] for SOP*; the only exception to this is a VPD which **Shall** ignore Messages sent with PD Specification Revision 2.0 and earlier. In order to facilitate compatibility, the Message Header of all messages contains a field describing the Revision of the USBPD Specification it supports.

After an Attach, a Port discovers the lowest common Specification Revision level between itself and its Port Partner and/or the Cable Plug(s), and **Shall** use this Specification Revision level until a Detach, [Hard Reset](#), or Error Recovery occurs.

After determining the Specification Revision to be used, all PD communications **Shall** comply completely with the relevant Revision of the PD specification.

The 2-bit Specification Revision field of a [GoodCRC Message](#) does not carry any meaning. The value received could be 00b, 01b, 10b, or 11b and **Shall** be ignored by the recipient of the Message. The sender of a [GoodCRC Message](#) **Shall** set the Specification Revision field to 01b (Revision 2.0) when responding to a Message that contains 01b in the Specification Revision field of the Message Header. The sender of a [GoodCRC Message](#) **May** set the Specification Revision field to 01b or 10b when responding to a Message that contains 10b (Revision 3.x) in the Specification Revision field of the Message Header.

All data in all Messages **Shall** be consistent with the definition from the USB-PD spec Version indicated in the Specification Revision field in the Message Header for that particular Message.

6.1.3.1. SOP Specification Revision Detection Process

An Attach event (or a [Hard Reset](#)) **Shall** cause the detection of the applicable Specification Revision to be implemented for Ports according to the rules stated below:

1. The Source Port sends a [Source Capabilities Message](#) to the Sink Port setting the Specification Revision field to the highest Revision of the Power Delivery Specification the Source Port supports.
2. The Sink Port responds with a [Request Message](#) that sets the Specification Revision field to the highest Revision of the Power Delivery Specification the Sink Port supports that is equal to or lower than the Specification Revision received from the Source Port.
3. The Source and Sink Ports **Shall** use the Specification Revision in the [Request Message](#) from the Sink in step 2 in all subsequent communications until a Detach, [Hard Reset](#), or Error Recovery occurs.

6.1.3.2. SOP' Specification Revision Detection Process

After Attach, and prior to a [VCONN Swap](#), the initial VCONN Source **Shall** use the following steps to establish a Specification Revision level for Cable Plugs according to the rules stated below:

1. The VCONN Source sends a [Discover Identity](#) REQ to the Cable Plug (SOP') that sets the Specification Revision field in the Message to the highest Revision of the Power Delivery Specification the VCONN Source supports.
2. The Cable Plug responds with a [Discover Identity](#) ACK that sets the Specification Revision field in the Message to the highest Revision of the Power Delivery Specification it supports that is equal to or lower than the Specification Revision it received from the Source Port.
3. The Cable Plug and VCONN Source **Shall** communicate using the lower of the two revisions until an Explicit Contract has been established.
4. [Table 6.1](#) shows the Specification Revision that **Shall** be used between the Port Partners and the Cable Plugs when the Specification Revision has been discovered and an Explicit Contract is in place.

After a [VCONN Swap](#), the new VCONN Source, that intends to communicate directly with the Cable Plug(s), **Shall** use the following steps to establish a Specification Revision level:

1. The new VCONN Source sends the required [Soft Reset Message](#) to synchronize its SOP' State machine with the Cable Plug, setting the Specification Revision field in the Message Header to the highest Revision of the Power Delivery specification it supports.
2. The Cable Plug responds with an [Accept Message](#), setting the Specification Revision field in the Message to the highest Revision of the Power Delivery Specification it supports that is equal to or lower than the Specification Revision it received from the Source Port.

Notes:

- A VCONN Source that does not communicate directly with the Cable Plug(s) may skip the above procedure.
- When a Cable Plug does not respond to a Revision 3.x [Discover Identity](#) REQ with a [Discover Identity](#) ACK or BUSY, the VCONN Source may repeat steps 1–4 using a Revision 2.0 [Discover Identity](#) REQ in step 1 before establishing that there is no Cable Plug to communicate with.
- This process is closely linked to PD Capability discovery as described in [Section 8.6.1.2.1](#).

A VCONN Source that supports Revision 3.x of the Power Delivery Specification **May** communicate with a Cable Plug also supporting Revision 3.x using Revision 3.x Compliant Communications regardless of the Specification Revision of its Port Partner while no Explicit Contract exists. After an Explicit Contract has been established the Port Partners and Cable Plug(s) **Shall** use [Table 6.1](#) to determine the Revision to be used.

A Cable Plug **Shall Not** save the State of the agreed Specification Revision. A Cable Plug **Shall** respond with the highest Specification Revision it supports that is equal to or lower than the Specification Revision contained in the Message received from the VCONN Source.

Cable Plugs **Shall** operate using the same Specification Revision for both SOP' and SOP". Cable assemblies with two Cable Plugs **Shall** operate using the same Specification Revision for both Cable Plugs.

See [Table 6.1](#) for details of how various Revisions **Shall** inter-operate.

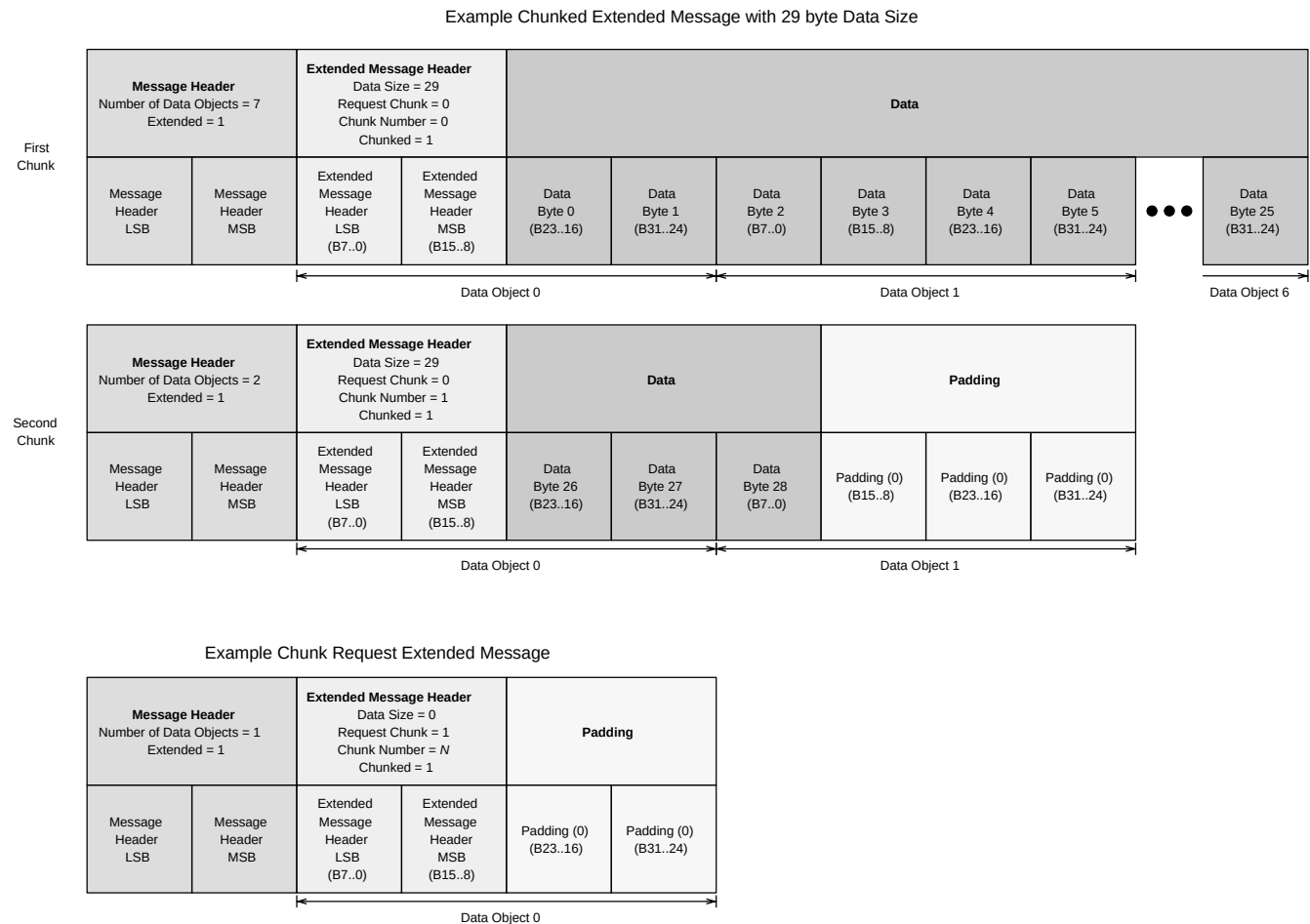
Table 6.1. Revision Interoperability during an Explicit Contract

Port 1 Revision	Cable Plug Revision	Port 2 Revision	Port to Port Operating Revision	Port to Cable Plug Operating Revision
2	2	2	2	2
2	2	3	2	2
2	3	2	2	2
2	3	3	2	2
3	2	2	2	2
3	2	3	3	2
3	3	2	2	2
3	3	3	3	3

6.1.4. Chunking

With USBPD Revision 2, the longest PD Message that can be transferred is 7 data objects, totaling 28 bytes. The [Extended Message](#) Type was introduced in USBPD Revision 3.0 V1.0, and may transfer up to 260 bytes per Message. Since Extended Messages may be longer than [Data Messages](#), Chunking allows USBPD devices with older PHY hardware to send and receive [Extended Messages](#). Chunking splits an Extended Message into "chunks" that can be sent and received similarly to [Data Messages](#). Each Chunk is no longer than [MaxExtendedMsgChunkLen](#) bytes long. See [Section 6.5.1.1](#).

Figure 6.1. Construction of Chunked Extended and Chunk Request Messages



6.1.5. Vendor and Product IDs

Vendor IDs (VID) and Product IDs (PID) are 16-bit values used by USBPD to identify an individual Device. Vendor IDs are assigned by USB-IF and Product IDs are assigned by the Vendor.

Multiple Messages contain VID/PID fields that may either describe the Device Port itself, or a Battery Attached to the Device. USBPD Messages describing the VID/PID associated with the Device Port **Shall** all respond with the same values. Messages that describe VID/PID include:

- [Discover Identity Message](#) Response.
- [Source_Capabilities_Extended Message](#).
- [Sink_Capabilities_Extended Message](#).
- [Manufacturer_Info Message](#).

For USB Devices or Hubs which support USB Communications, the USB Vendor ID field **Shall** be identical to the Vendor ID field defined in the product's USB Device Descriptor (see [USB2] and [USB3]).

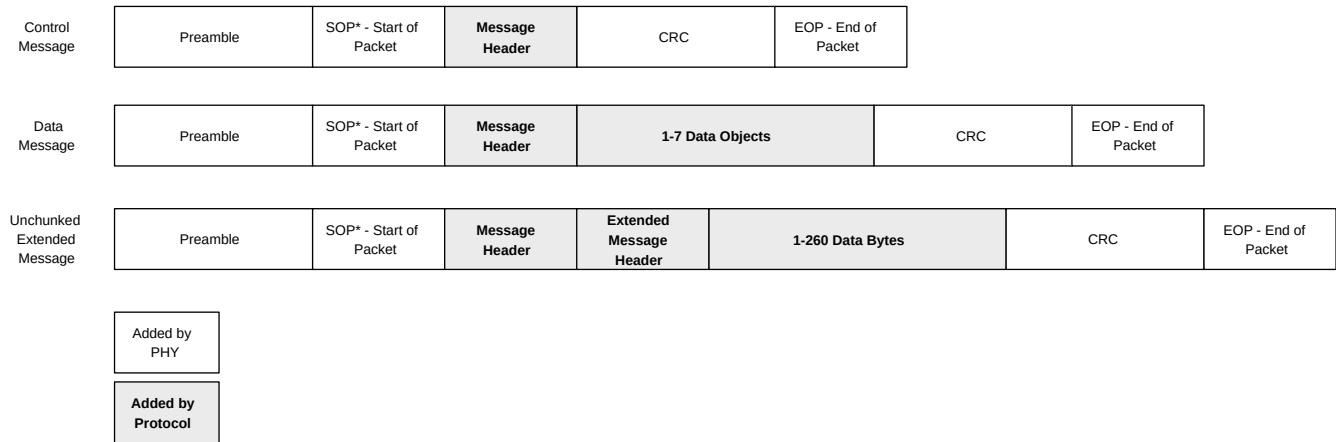
If the vendor does not have a VID, VID fields in Messages **Shall** be set to FFFFh. If the VID field is FFFFh, a PID field in the same Message **Shall** be set to 0000h.

[Vendor Defined Messages](#) contain an SVID field that is similar to VID. However, the SVID field is used to describe the contents of the [Vendor Defined Message](#) and may not match the VID of the Device Port that is transmitting the Message.

6.2. Message Construction

All Messages **Shall** begin with a Message Header and **May** be followed by a variable length (including zero) data portion. The following figure illustrates the three different messages types as they exist inside a PD Packet.

Figure 6.2. Format of PD Packets for Different Message Types



The 16-bit Message Header, 16-bit [Extended Message](#) Header, and all 32-bit Data Objects are transmitted least-significant byte first.

6.2.1. Message Header

Each PD Message starts with a 16-bit Message Header as defined in [Table 6.2](#).

Table 6.2. Message Header

Bit(s)	Field Name	Applicable SOP	Description
15	Extended	All	0 - Indicates a Control or Data Message 1 - Indicates an Extended Message .
14...12	Number of Data Objects	All	See Section 6.2.2 .
11...9	MessageID	All	Value generated by a rolling counter maintained by the originator of the Message.
8	Port Power Role	SOP	Present Power Role of the Port that is sending this Message: 0 - Sink 1 - Source Shall Not be verified by the receiver and therefore Shall Not lead to Soft Reset , Hard Reset or Error Recovery if it is incorrect. At Attachment or Hard Reset , this field is set to 0 for the Port presenting Rd and set to 1 for the Port presenting Rp.
	Cable Plug	SOP/ SOP"	Indicates Message sender type: 0 - Port (DFP/UFP) 1 - Cable Plug or VPD
7...6	Specification Revision	All	00b - Revision 1.0 (Deprecated , receiver Shall interpret as Revision 2.0). 01b - Revision 2.0 10b - Revision 3.x 11b - Reserved .
5	Port Data Role	SOP	Present Data Role of the Port that is sending this Message: 0 - UFP 1 - DFP At Attachment or Hard Reset , this field is set to 0 for the Port presenting Rd and set to 1 for the Port presenting Rp. If a USB Type-C Port receives a Message with the Port Data Role field set to the same Data Role as its current Data Role, except for the GoodCRC Message , USB Type-C Error Recovery actions as defined in [USB-C] Shall be performed.
	Reserved	SOP/ SOP"	Reserved , receiver ignores this field.
4...0	Message Type	All	Indicates type of Message being sent. To determine the exact Message type: <ul style="list-style-type: none"> • If Extended is set, refer to Section 6.5. • Otherwise, if Number of Data Objects is greater than 0, refer to Table 6.5. • Otherwise, refer to Table 6.4.

6.2.2. Number of Data Objects

This field specifies the number of 32-bit data objects in the Message.

Table 6.3. Number of Data Objects

Message Header: Extended bit ¹	Extended Message Header: Chunked bit ²	Number of Data Objects
0	X	For Control Messages: Shall be 0 For Data Messages: The number of 32-bit Data Objects
1	0	Reserved , receiver Shall ignore
1	1	Number of 32-bit data objects in the Message, created by padding the number of bytes sent in this Message to the 4-byte boundary. This includes the 16-bit Extended Message Header. $[(\text{Number of Bytes in this Message, including 16-bit } \text{Extended Message Header}) + 3] \div 4.$ See Figure 6.1 .
1. See Section 6.2.1 2. See Section 6.5.1		

6.3. Control Message

A Control Message has the Extended and Number of Data Objects fields in the Message header set to zero. The Control Message consists only of a Message Header and a CRC.

The following table shows the Control Message Types:

Table 6.4. Control Message Types

Value	Message Type
00000b	Reserved , receiver Shall respond with Not_Supported Message .
00001b	GoodCRC
00010b	GotoMin (Deprecated)
00011b	Accept
00100b	Reject
00101b	Ping (Deprecated)
00110b	PS_RDY
00111b	Get_Source_Cap
01000b	Get_Sink_Cap
01001b	DR_Swap
01010b	PR_Swap
01011b	VCONN Swap
01100b	Wait
01101b	Soft_Reset
01110b	Data_Reset
01111b	Data_Reset_Complete
10000b	Not_Supported
10001b	Get_Source_Cap_Extended
10010b	Get_Status
10011b	FR_Swap

Value	Message Type
10100b	Get_PPS_Status
10101b	Get_Country_Codes
10110b	Get_Sink_Cap_Extended
10111b	Get_Source_Info
11000b	Get_Revision
11001b..11111b	Reserved , receiver Shall respond with Not_Supported

6.3.1. GoodCRC Message

The [GoodCRC Message](#) acknowledges that the previous Message was correctly received. The [GoodCRC](#) sender **Shall** set the MessageID value to that of the received Message, so the sender can determine that the correct Message is being acknowledged.

Usage: [Section 7.6.1](#)

6.3.2. GotoMin Message (Deprecated)

The GotoMin Message has been **Deprecated**. This Message Type is no longer **Valid** and **Shall** be responded to by a [Not_Supported Message](#).

6.3.3. Accept Message

The [Accept Message](#) is used as a positive acknowledgment by a receiver

This Message is used in the following sequences:

- Source Capabilities
- Request
- EPR Source Capabilities
- EPR Request
- [Soft Reset](#)
- [Data Reset](#)
- [Power Role Swap](#)
- [Fast Role Swap](#)
- [Data Role Swap](#)
- [VCONN Swap](#)
- [Enter USB](#)

Usage: [Section 7.4](#)

6.3.4. Reject Message

The [Reject Message](#) is used as a negative acknowledgment by a receiver

This Message is used in the following sequences:

- Source Capabilities

- Request
- EPR Source Capabilities
- EPR Request
- [Power Role Swap](#)
- [Data Role Swap](#)
- [VCONN Swap](#)
- [Enter USB](#)
- [EPR Get Source Capabilities](#)
- [EPR Get Sink Capabilities](#)

Note: The [Reject Message](#) is not a **Valid** response when a Message is not supported. In this case the [Not_Supported Message](#) is returned.

Usage: [Section 7.4](#)

6.3.5. Ping Message (Deprecated)

The [Ping Message](#) has been **Deprecated**. This Message Type is no longer **Valid**.

A Port that receives a [Ping \(Deprecated\)](#) Message **May** respond with a [Not_Supported Message](#) or **Ignore** the [Ping \(Deprecated\)](#) Message. A Cable Plug that receives a [Ping \(Deprecated\)](#) Message **Shall Ignore** the [Ping \(Deprecated\)](#) Message.

6.3.6. PS_RDY Message

The [PS_RDY](#) (Power Supply Ready) Message indicates the Source's power supply has reached the desired operating condition.

This Message is used in the following sequences:

- Source Capabilities
- Request
- EPR Source Capabilities
- EPR Request
- [Data Reset](#)
- [Power Role Swap](#)
- [Fast Role Swap](#)
- [VCONN Swap](#)

Usage: See relevant AMS section

6.3.7. Get_Source_Cap Message

The [Get_Source_Cap](#) (Get Source Capabilities) Message is used by a Port to Request the Source Capabilities of its Port Partner.

This Message is used in the following sequences:

- Get Source Capabilities

Usage: [Section 7.17](#)

6.3.8. Get_Sink_Cap Message

The [Get_Sink_Cap](#) (Get Sink Capabilities) Message is used by a Port to Request the Sink Capabilities of its Port Partner.

This Message is used in the following sequences:

- Get Sink Capabilities

Usage: [Section 7.18](#)

6.3.9. DR_Swap Message

The [DR_Swap](#) ([Data Role Swap](#)) Message is used to exchange DFP and UFP operations between Dual Role Data (DRD) Port Partners. The Source of VBUS and VCONN Source **shall** remain unchanged as well as the Rp/Rd resistors on the CC wire during the Data Role Swap process.

This Message is used in the following sequences:

- [Data Role Swap](#)

Usage: [Section 7.12](#)

6.3.10. PR_Swap Message

The [PR_Swap](#) ([Power Role Swap](#)) Message is used to Request an exchange of Power Roles.

The DFP (Host), UFP (Device) Data Roles, and VCONN Source **shall** remain unchanged by the [Power Role Swap](#) process. During the [Power Role Swap](#) process, the Initial Sink does not disconnect even though VBUS drops below [vSafe5V](#).

This Message is used in the following sequences:

- [Power Role Swap](#)

Usage: [Section 7.10](#)

6.3.11. VCONN_Swap Message

The [VCONN_Swap Message](#) is used to Request an exchange of VCONN Source ownership.

This Message is used in the following sequences:

- [VCONN Swap](#)

Usage: [Section 7.13](#)

6.3.12. Wait Message

The [Wait Message](#) is a response used to Request the Port Partner to wait and (optionally) retry its Message later.

This Message is used in the following sequences:

- Source Capabilities
- Request
- EPR Source Capabilities
- EPR Request
- [Power Role Swap](#)
- [Data Role Swap](#)
- [VCONN Swap](#)
- [Enter USB](#)

Usage: [Section 7.4](#)

6.3.13. Soft_Reset Message

The [Soft_Reset Message](#) is used by a Port to recover from Protocol Layer errors; putting the Message counters and Policy Engine into a known State.

This Message is used in the following sequences:

- [Soft_Reset](#)

Usage: [Section 7.7](#)

6.3.14. Data_Reset Message

The [Data_Reset Message](#) is used by a Port to reset the USB data connection and exit all Alternate Modes with its Port Partner while preserving the power on VBUS. USB4 Mode capable ports **Shall** support the [Data_Reset Message](#) and other ports **May** support the [Data_Reset Message](#).

This Message is used in the following sequences:

- [Data_Reset](#)

Usage: [Section 7.8](#)

6.3.15. Data_Reset_Complete Message

The [Data_Reset_Complete Message](#) is used to indicate the completion of the Data Reset process.

This Message is used in the following sequences:

- [Data_Reset](#)

Usage: [Section 7.8](#)

6.3.16. Not_Supported Message

The [Not_Supported Message](#) **Shall** be sent by a Port or Cable Plug in response to any Message it does not support or is unable to interpret. Returning a [Not_Supported](#) Message is assumed in this specification and has not been called out explicitly except in [Section 7.5](#) which defines cases where the [Not_Supported Message](#) is returned.

This Message is used in the following sequences:

- [Unsupported Sequence](#)

6.3.17. Get_Source_Cap_Extended Message

The [Get_Source_Cap_Extended Message](#) is sent by a Port to Request additional information about a Port's Source Capabilities.

This Message is used in the following sequences:

- [Get Source Capabilities Extended](#)

Usage: [Section 7.19.1](#)

6.3.18. Get_Status Message

The [Get_Status Message](#) is sent by a Port using SOP to Request the Port Partner's present status.

The [Get_Status Message](#) **May** also be sent to an Active Cable to get its present status using SOP'/SOP". Passive cables are allowed to respond to a [Get_Status](#) message with a [Status message](#) as well.

This Message is used in the following sequences:

- [Get Status](#)
- [Alert](#)

Usage: [Section 7.15](#)

6.3.19. FR_Swap Message

The [FR_Swap Message](#) is sent by the New Source as a part of the [Fast Role Swap](#) sequence to re-synchronize the State machines with the Port Partner. See [Chapter 10](#) for complete details.

This Message is used in the following sequences:

- [Fast Role Swap](#)

Usage: [Section 7.11](#)

6.3.20. Get_PPS_Status

The [Get_PPS_Status Message](#) is used by a Sink Port to Request additional information about a Source's PPS Status.

This Message is used in the following sequences:

- [Get PPS Status](#)

Usage: [Section 7.16](#)

6.3.21. Get_Country_Codes

The [Get_Country_Codes Message](#) is used by a Port to Request the alpha-2 country codes its Port Partner supports as defined in [ISO 3166].

This Message is used in the following sequences:

- Get [Country Codes](#)

Usage: [Section 7.23.1](#)

6.3.22. Get_Sink_Cap_Extended Message

The [Get_Sink_Cap_Extended Message](#) is used by a Port to Request additional information about a its Port Partner's Sink Capabilities.

This Message is used in the following sequences:

- Get Sink Capabilities Extended

Usage: [Section 7.19.3](#)

6.3.23. Get_Source_Info Message

The [Get_Source_Info Message](#) is used by a Port to Request the Port type, maximum Capabilities, and present Capabilities of its Port Partner when operating as a Source.

This Message is used in the following sequences:

- Get [Source Information](#)

Usage: [Section 7.26](#)

6.3.24. Get_Revision Message

The [Get_Revision Message](#) is sent by a Port to Request the Revision and Version of the Power Delivery Specification the Port Partner (if SOP is used) or Cable Plug (if SOP'/SOP" is used) supports.

This Message is used in the following sequences:

- Get Revision

Usage: [Section 7.25](#)

6.4. Data Message

A [Data Message](#) Shall consist of a Message Header , followed by one to seven Data Objects.

There are many types of Data Objects used to compose [Data Messages](#). Some examples are:

- Power Data Object (PDO) used to expose a Source Port's power Capabilities or a Sink's power requirements.
- Request Data Object (RDO) used by a Sink Port to Negotiate an Explicit Contract.
- Vendor Data Object (VDO) used to convey vendor specific information.
- BIST Data Object (BDO) used for PHY Layer compliance testing.
- [Battery Status](#) Data Object (BSDO) used to convey Battery status information.

- [Alert](#) Data Object (ADO) used to indicate events occurring on the Source or Sink.

Table 6.5. Data Message Types

Value	Message Type
00000b	Reserved , receiver Shall respond with Not_Supported Message
00001b	Source_Capabilities
00010b	Request
00011b	BIST
00100b	Sink_Capabilities
00101b	Battery_Status
00110b	Alert
00111b	Get_Country_Info
01000b	Enter_USB
01001b	EPR_Request
01010b	EPR_Mode
01011b	Source_Info
01100b	Revision
01101b ...01110b	Reserved , receiver Shall respond with Not_Supported Message .
01111b	Vendor_Defined
10000b...11111b	Reserved , receiver Shall respond with Not_Supported Message

6.4.1. Capabilities Messages

6.4.1.1. Source_Capabilities Message

The [Source_Capabilities Message](#) is used by a Port to describe its power Capabilities when acting as a Source in SPR Mode. This Message **Shall** contain 1 to 7 Power Data Object (PDOs), ordered as follows:

1. The [vSafe5V](#) Fixed Supply PDO **Shall** always be the first PDO.
2. The remaining Fixed Supply PDOs, if present, **Shall** be sent in voltage order; lowest to highest.
3. The Battery Supply PDOs, if present, **Shall** be sent in Minimum voltage order; lowest to highest.
4. The Variable Supply (non-Battery) PDOs, if present, **Shall** be sent in Minimum voltage order; lowest to highest.
5. The SPR AVS APDO, if present, **Shall** be sent.
6. The Programmable Power Supply APDOs, if present, **Shall** be sent in Maximum voltage order, lowest to highest.

A Source **Shall Not** offer multiple PDOs of the same type and the same voltage, but **Shall** instead offer one PDO with the highest available current or power for that voltage.

This Message is used in the following sequences:

- Source Capabilities
- [EPR Mode](#) Exit
- Get Source Capabilities

Usage: [Section 7.9.1](#)

6.4.1.2. Sink_Capabilities Message

The [Sink_Capabilities Message](#) is used by a Port to describe its power and voltage requirements when acting as a Sink in SPR Mode. Construction of this Message **Shall** follow the same rules as [Source_Capabilities Message](#).

A Sink **Shall Not** respond with multiple PDOs of the same type and the same voltage, but **Shall** instead respond with one PDO with the highest required current or power for that voltage.

This Message is used in the following sequences:

- Get Sink Capabilities

6.4.1.3. Power Data Objects

A Power Data Object (PDO) is formatted differently depending on the capability being described. PDOs are used to describe power Capabilities in the following messages:

- [Source_Capabilities](#)
- [Sink_Capabilities](#)
- [EPR_Request](#)
- [EPR_Source_Capabilities](#)
- [EPR_Sink_Capabilities](#)

There are three types of Power Data Objects. They contain additional information beyond that encoded in the Message Header to identify each of the three types of Power Data Objects:

- Fixed Supply is used to expose well-regulated fixed voltage power supplies.
- Variable Supply is used to expose very poorly regulated power supplies.
- Battery Supply is used to expose batteries that can be directly Connected to VBUS.

Additionally, there are three types of Augmented Power Data Objects:

- SPR PPS is used to expose a power supply whose output voltage can be programmatically adjusted over the Advertised voltage range and limited by the Source to a programmable Current Limit.
- SPR AVS and EPR AVS are used to expose a power supply whose output voltage can be adjusted over the Advertised voltage range but otherwise is equivalent to a Fixed Supply (AVS does not support a programmable Current Limit).

6.4.1.3.1. PDO Format

Table 6.6. PDO Format

Bit(s)	Field Name	Static	Description
31...30	PDO Type	No	00b - Fixed Voltage 01b - Battery 10b - Variable (non-Battery) 11b - Augmented PDO (APDO)
29...0	PDO Specific	No	This field is described by the PDOs in the following sections.

6.4.1.3.2. Augmented PDO Format

Table 6.7. Augmented PDO Format

Bit(s)	Field Name	Static	Description
31...30	PDO Type	Yes	11b - Augmented PDO (APDO) Other values are used to describe other PDOs.
29...28	APDO Type	No	00b - PPS 01b - EPR AVS 10b - SPR AVS 11b - Invalid , receiver Should consider this Power Data Object as Invalid .
27...0	APDO Data	No	Specific Power Capabilities are described by the APDOs in the following sections.

6.4.1.3.3. Fixed 5V Source PDO Format

Table 6.8. Fixed 5V Source PDO Format

Bit(s)	Field Name	Static	Description
31...30	PDO Type	Yes	00b - Fixed Voltage Other values are used to describe other PDOs.
29	Dual-Role Power	Yes	0b - Port cannot change Power Role via the PR_Swap_Message process 1b - Port may change Power Role via the PR_Swap_Message process This bit Shall match the Dual-Role Power bit set in "5V Fixed Voltage PDO - Sink"
28	USB Suspend Supported	Yes	0b - Sink Shall Not apply the [USB2], [USB3] or [USB4] rules for suspend and May continue to draw the Negotiated power. 1b - Sink Shall follow the [USB2], [USB3] or [USB4] rules for suspend and resume.
27	Unconstrained Power	No	0b - The Source is limiting its available output power based on its own internal power usage. 1b - An external Source of power is available that is sufficient to adequately power the Source system while charging external devices, or when the Source's primary function is to charge external devices
26	USB Communications Capable	Yes	0b - Port is not capable of communication over the USB data lines 1b - Port is capable of communication over the USB data lines
25	Dual-Role Data	Yes	0b - Port cannot change Data Role via the DR_Swap_Message process 1b - Port may change Data Role via the DR_Swap_Message process For DRP capable devices, this bit Shall match the Dual-Role Data bit set in "5V Fixed Voltage PDO - Sink"
24	Unchunked Extended Messages Supported	Yes	0b - Port does not support Unchunked Extended Message 1b - Port supports both Chunked and Unchunked Extended Message
23	EPR Capable	Yes	0b - Source will only provide power Capabilities in the Standard Power Range. 1b - Source is capable of providing power Capabilities in the Extended Power Range. The Source may enter EPR_Mode when requested by an EPR Capable Sink.

Bit(s)	Field Name	Static	Description
22	Reserved	Yes	Reserved , receiver Shall ignore this field.
21...20	Peak Current	No	See Section 6.4.1.3.13 .
19...10	Voltage	Yes	Voltage, in 50mV units. 100 - Required voltage value (5V). All other values are Invalid .
9...0	Maximum Current	No	Maximum amount of current supported at this voltage, in 10mA units. 0..500 - Allowed current values (0 - 5A). All other values are Invalid .

6.4.1.3.4. Fixed 5V Sink PDO

Table 6.9. Fixed 5V Sink PDO Format

Bit(s)	Field Name	Static	Description
31...30	PDO Type	Yes	00b - Fixed Voltage Other values are used to describe other PDOs.
29	Dual-Role Power	Yes	0b - Port cannot change Power Role via the PR_ Swap process. 1b - Port may change Power Role via the PR_ Swap process. This bit Shall match the Dual-Role Power bit set in "5V Fixed Voltage PDO - Source".
28	Higher Capability	Yes	0b - Sink can operate with full functionality at vSafe5V . 1b - Sink requires higher than vSafe5V to operate with full functionality.
27	Unconstrained Power	No	0b - The Sink may require power from VBUS on this Port. 1b - The Device's power requirements could be fulfilled by another Source of power (e.g. a different USBC Port, barrel jack, etc).
26	USB Communications Capable	Yes	0b - Port is not capable of communication over the USB data lines. 1b - Port is capable of communication over the USB data lines.
25	Dual-Role Data	Yes	0b - Port cannot change Data Role via the DR_Swap Message process. 1b - Port may change Data Role via the DR_Swap Message process. For DRP capable devices, this bit Shall match the Dual-Role Data bit set in "5V Fixed Voltage PDO - Source".
24...23	Fast Role Swap required USB Type-C Current	No	00b - Fast Role Swap not supported (default) 01b - Default USB Port 10b - 1.5A@5V 11b - 3.0A@5V
22...20	Reserved	Yes	Reserved , receiver Shall ignore this field.
19...10	Voltage	Yes	Voltage, in 50mV units. 100 - Required voltage value (5V). Fixed PDOs with Voltages other than 5V are defined in Table 6.10 .
9...0	Maximum Current	No	Maximum amount of current consumed at this voltage, in 10mA units. 0..500 - Allowed current values (0 - 5A). All other values are Invalid .

6.4.1.3.5. Fixed PDO (>5V)

Table 6.10. Fixed PDO (>5V)

Bit(s)	Field	Static	Description
31...30	PDO Type	Yes	00b - Fixed Voltage Other values are used to describe other PDOs.
29...22	Device Flags	No	Reserved , receiver Shall ignore this field. This field is populated for the 5V Fixed PDO in position 1 of the Capabilities Message. See Table 6.8 and Table 6.9 for details.
21...20	Peak Current	No	When transmitted by Source: See Section 6.4.1.3.13 . When transmitted by Sink: Reserved , receiver Shall ignore this field.
19...10	Voltage	No	Voltage, in 50mV units. See Section 3.2.4 and Section 3.2.5 for allowed values.
9...0	Maximum/Operational Current	No	Maximum amount of current supported (for Source) or consumed (for Sink) at this voltage, in 10mA units. 0..500 - Allowed current values (0 - 5A). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .

6.4.1.3.6. Battery PDO

Table 6.11. Battery PDO

Bit(s)	Field Name	Static	Description
31...30	PDO Type	Yes	01b - Battery Other values are used to describe other PDOs.
29...20	Maximum Voltage	No	Maximum voltage supported, in 50mV units. See Section 3.2.5.1 for allowed values.
19...10	Minimum Voltage	No	Minimum voltage supported, in 50mV units. See Section 3.2.5.1 for allowed values.
9...0	Maximum Power	No	Maximum amount of power supported (for Source) or consumed (for Sink), in 250mW units. 0..400 - Allowed power values (0 - 100W). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid . Note: This field describes Power, not Current.

6.4.1.3.7. Variable PDO

Table 6.12. Variable PDO

Bit(s)	Field Name	Static	Description
31...30	PDO Type	Yes	10b - Variable (non-Battery) Other values are used to describe other PDOs.
29...20	Maximum Voltage	No	Maximum voltage supported, in 50mV units. See Section 3.2.5.1 for allowed values.
19...10	Minimum Voltage	No	Minimum voltage supported, in 50mV units. For a Source, this field Shall Not be less than 80% of the Maximum Voltage field value. See Section 3.2.5.1 for allowed values.
9...0	Maximum Current	No	Maximum amount of current supported (for Source) or consumed (for Sink) in this voltage range, in 10mA units. 0..500 - Allowed current values (0 - 5A). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid . For a Sink, this field May describe either the maximum current the Sink will ever require or the current that would be sufficient to operate the Sink in one of its modes of operations.

6.4.1.3.8. SPR Programmable Power Supply Source APDO

Table 6.13. SPR Programmable Power Supply Source APDO

Bit(s)	Field	Static	Description
31...30	PDO Type	Yes	11b - Augmented Power Data Object (APDO) Other values are used to describe other PDOs.
29...28	APDO Type	Yes	00b - PPS Other values are used to describe other APDOs.
27	PPS Power Limited	No	When set to 1, the Source May supply power that exceeds the Source's rated PDP, within its Optional operating area.
26...25	Reserved	Yes	Reserved , receiver Shall ignore this field.
24...17	Maximum Voltage	No	Maximum voltage supported, in 100mV increments 110 - Allowed voltage value (11V). 160 - Allowed voltage value (16V). 210 - Allowed voltage value (21V). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid . See Table 4.2 .
16	Reserved	Yes	Reserved , receiver Shall ignore this field.
15...8	Minimum Voltage	No	Minimum voltage supported, in 100mV increments. 33 - Deprecated voltage value (3.3V). 50 - Required voltage value (5V). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .
7	Reserved	No	Reserved , receiver Shall ignore this field.
6...0	Maximum Current	No	Maximum amount of current supported in this voltage range, in 50mA units. 0..100 - Allowed current values (0 - 5A). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .

6.4.1.3.9. SPR Programmable Power Supply Sink APDO

Table 6.14. SPR Programmable Power Supply Sink APDO

Bit(s)	Field	Static	Description
31...30	PDO Type	Yes	11b - Augmented Power Data Object (APDO) Other values are used to describe other PDOs.
29...28	APDO Type	Yes	00b - PPS Other values are used to describe other APDOs.
27...25	Reserved	Yes	Reserved , receiver Shall ignore this field.
24...17	Maximum Voltage	No	Maximum voltage supported, in 100mV increments. 50..210 - Allowed voltage values (5 - 21V). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .
16	Reserved	Yes	Reserved , receiver Shall ignore this field.
15...8	Minimum Voltage	No	Minimum voltage supported, in 100mV increments. 33 - Deprecated voltage value (3.3V). 50 - Required voltage value (5V). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .
7	Reserved	Yes	Reserved , receiver Shall ignore this field.
6...0	Maximum Current	No	Maximum amount of current supported in this voltage range, in 50mA units. 0..100 - Allowed current values (0 - 5A). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .

6.4.1.3.10. SPR Adjustable Voltage Supply APDO

Table 6.15. SPR Adjustable Voltage Supply APDO

Bit(s)	Field	Static	Description
31...30	PDO Type	Yes	11b - Augmented Power Data Object (APDO) Other values are used to describe other PDOs.
29...28	APDO Type	Yes	10b - SPR AVS Other values are used to describe other APDOs.
27...26	Peak Current	No	When transmitted by Source: See Section 6.4.1.3.13 . When transmitted by Sink: Reserved , receiver Shall ignore this field.
25...20	Reserved	Yes	Reserved , receiver Shall ignore this field.
19...10	Maximum Current 15V	No	Maximum current supported when operating from 9-15V, in 10mA increments. 0..500 - Allowed current values (0 - 5A). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid . When transmitted by Source, Shall be equal to the "Maximum Current" field in the 15V Fixed PDO.
9...0	Maximum Current 20V	No	Maximum current supported when operating from 15 - 20V, in 10mA increments. 0..500 - Allowed current values (0 - 5A). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid . Shall be 0 if the maximum voltage supported is 15V. When transmitted by Source, Shall be equal to the "Maximum Current" field in the 20V Fixed PDO.

6.4.1.3.11. EPR Adjustable Voltage Supply Source APDO

Table 6.16. EPR Adjustable Voltage Supply Source APDO

Bit(s)	Field	Static	Description
31...30	PDO Type	Yes	11b - Augmented Power Data Object (APDO) Other values are used to describe other PDOs.
29...28	APDO Type	Yes	01b - EPR AVS Other values are used to describe other APDOs.
27...26	Peak Current	No	See Section 6.4.1.3.13 .
25...17	Maximum Voltage	No	Maximum voltage supported, in 100mV increments. 280 - Allowed voltage values (28V). 360 - Allowed voltage values (36V). 480 - Allowed voltage values (48V). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid . See Table 4.4 .
16	Reserved	Yes	Reserved , receiver Shall ignore this field.
15...8	Minimum Voltage	No	Minimum voltage supported, in 100mV increments. 150 - Allowed voltage value (15V). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .
7...0	PDP	No	Port Present PDP, in 1W increments. 0..240 - Allowed power values (0 - 240W). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid . See Section 3.2.5.3 and Figure 3.3 for more information regarding how PDP in the AVS APDO relates to maximum available current.

6.4.1.3.12. EPR Adjustable Voltage Supply Sink APDO

Table 6.17. EPR Adjustable Voltage Supply Sink APDO

Bit(s)	Field	Static	Description
31...30	PDO Type	Yes	11b - Augmented Power Data Object (APDO) Other values are used to describe other PDOs.
29...28	APDO Type	Yes	01b - EPR AVS Other values are used to describe other APDOs.
27...26	Reserved	Yes	Reserved , receiver Shall ignore this field.
25...17	Maximum Voltage	No	Maximum voltage supported, in 100mV increments. 150..480 - Allowed voltage values (15 - 48V). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .
16	Reserved	Yes	Reserved , receiver Shall ignore this field.
15...8	Minimum Voltage	No	Minimum voltage supported, in 100mV increments. 150..480 - Allowed voltage values (15 - 48V). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid .
7...0	Maximum Power	No	Maximum power consumed, in 1W increments. 0..240 - Allowed power values (0 - 240W). All other values are Invalid ; receiver Should consider this Power Data Object as Invalid . Shall be \leq EPR Sink Maximum PDP field value in Table 6.61

6.4.1.3.13. PDO Peak Current

The USB Power Delivery Fixed Supply is only required to deliver the amount of current requested in the Operating Current field (IoC) of an RDO. In some usages however, for example computer systems, where there are short bursts of activity, it might be desirable to overload the Source for short periods. For example, when a computer system tries to maintain average power consumption, the higher the peak current, the longer the low current (see [Section 4.2.8](#)) period needed to maintain such average power. The Peak Current field allows a Source to Advertise this additional capability. This capability is intended for direct Port to Port connections only and **Shall Not** be offered to downstream Sinks via a Hub.

Every Fixed Supply PDO **Shall** contain a Peak Current field. Supplies that want to offer a set of overload Capabilities **Shall** Advertise this through the Peak Current field in the corresponding Fixed Supply PDO (see [Table 6.18](#)). Supplies that do not support an overload capability **Shall** set these bits to 00b in the corresponding Fixed Supply PDO.

Supplies that support an extended overload capability specified in the PeakCurrent1...3 fields of the [Source_Capabilities_Extended Message](#) (see [Section 6.5.2](#)) **Shall** also set these bits to 00b. Sinks wishing to utilize these [Extended Capabilities](#) **Shall** first send the [Get_Source_Cap_Extended Message](#) to determine what Capabilities, if any are supported by the Source.

Table 6.18. PDO Peak Current

PDO Peak Current Field	Description
00b	Peak current equals IoC (default) or look at the Source Capabilities Extended Message (send Get Source Cap Extended Message).
01b	Overload Capabilities: <ol style="list-style-type: none"> 1. Peak current equals 150% IoC for 1ms @ 5% duty cycle (low current equals 97% IoC for 19ms). 2. Peak current equals 125% IoC for 2ms @ 10% duty cycle (low current equals 97% IoC for 18ms). 3. Peak current equals 110% IoC for 10ms @ 50% duty cycle (low current equals 90% IoC for 10ms).
10b	Overload Capabilities: <ol style="list-style-type: none"> 1. Peak current equals 200% IoC for 1ms @ 5% duty cycle (low current equals 95% IoC for 19ms). 2. Peak current equals 150% IoC for 2ms @ 10% duty cycle (low current equals 94% IoC for 18ms). 3. Peak current equals 125% IoC for 10ms @ 50% duty cycle (low current equals 75% IoC for 10ms).
11b	Overload Capabilities: <ol style="list-style-type: none"> 1. Peak current equals 200% IoC for 1ms @ 5% duty cycle (low current equals 95% IoC for 19ms). 2. Peak current equals 175% IoC for 2ms @ 10% duty cycle (low current equals 92% IoC for 18ms). 3. Peak current equals 150% IoC for 10ms @ 50% duty cycle (low current equals 50% IoC for 10ms).

6.4.2. Request Message

A [Request Message](#) is used by the Sink to Request a change in VBUS from the Source. A [Request Message](#) **Shall** contain a single Request Data Object (RDO) that **Shall** identify the Power Data Object being requested.

The Request uses a different format depending on the kind of power that is being requested.

- The Fixed Supply Power Data Object and Variable Supply Power Data Object share a common format shown in [Table 6.19](#).
- The Battery Supply Power Data Object uses the format shown in [Table 6.20](#).
- The PPS Request Data Object's format is shown in [Table 6.21](#).
- The AVS Request Data Object's format is shown in [Table 6.22](#).

This Message is used in the following sequences:

- Source Capabilities
- Request
- Get Source Capabilities

Usage: [Section 7.9.2](#)

6.4.2.1. Sink Request Data Objects

Sink Request Data Objects (RDOs) are used to describe a Sink's power Request to the Source in the following messages:

- Request
- [EPR_Request](#)

6.4.2.1.1. Fixed and Variable RDO

Table 6.19. Fixed and Variable RDO

Bit(s)	Field Name	Static	Description
31...28	Object Position	No	See Section 6.4.2.1.5 .
27	Giveback	No	Deprecated , receiver Shall ignore this field.
26	Capability Mismatch	No	See Section 6.4.2.1.6 .
25	USB Communications Cable	Yes	See Section 6.4.2.1.7 .
24	No USB Suspend	Yes	See Section 6.4.2.1.8 .
23	Unchunked Extended Messages Supported	Yes	See Section 6.4.2.1.9 .
22	EPR Capable	No	See Section 6.4.2.1.10 .
21...20	Reserved	Yes	Reserved , receiver Shall ignore this field.
19...10	Operating Current	No	Operating current, in 10mA units Indicates the highest current the Sink will draw during the Explicit Contract. A new Request / EPR_Request Message , with an updated Operating Current value, Shall be issued whenever the Sink's power needs change. This value Shall Not exceed the value in the Maximum Current field of the Source_Capabilities Message . For EPR AVS, the Operating Current field Shall Not exceed the $PDP \div \text{Output voltage}$ rounded down to the nearest 50 mA.
9...0	Maximum Operating Current	No	Deprecated , transmitter Shall set this field equal to "Operating Current", receiver should ignore this field.

6.4.2.1.2. Battery RDO

Table 6.20. Battery RDO

Bit(s)	Field Name	Static	Description
31...28	Object Position	No	See Section 6.4.2.1.5 .
27	Giveback	No	Deprecated , receiver Shall ignore this field.
26	Capability Mismatch	No	See Section 6.4.2.1.6 .
25	USB Communications Cable	Yes	See Section 6.4.2.1.7 .
24	No USB Suspend	Yes	See Section 6.4.2.1.8 .
23	Unchunked Extended Messages Supported	Yes	See Section 6.4.2.1.9 .
22	EPR Capable	No	See Section 6.4.2.1.10 .
21...20	Reserved	Yes	Reserved , receiver Shall ignore this field.
B19...10	Operating Power	No	Operating Power, in 250mW units Indicates the highest power the Sink will draw throughout the Explicit Contract.
B9...0	Maximum Operating Power	No	Deprecated , transmitter Shall set this field equal to "Operating Current", receiver should ignore this field.

6.4.2.1.3. Programmable Power Supply RDO

Table 6.21. Programmable Power Supply RDO

Bit(s)	Field Name	Static	Description
31...28	Object Position	No	See Section 6.4.2.1.5 .
27	Giveback	No	Deprecated , receiver Shall ignore this field.
26	Capability Mismatch	No	See Section 6.4.2.1.6 .
25	USB Communications Cable	Yes	See Section 6.4.2.1.7 .
24	No USB Suspend	Yes	See Section 6.4.2.1.8 .
23	Unchunked Extended Messages Supported	Yes	See Section 6.4.2.1.9 .
22	EPR Capable	No	See Section 6.4.2.1.10 .
21	Reserved	Yes	Reserved , receiver Shall ignore this field.
20...9	Output Voltage	No	Requested output voltage at the Source's connector, in 20mV units. Voltage Shall be greater than or equal to the Minimum. Voltage field and less than or equal to the Maximum. Voltage field in the selected APDO.
8...7	Reserved	Yes	Reserved , receiver Shall ignore this field.
6...0	Operating Current	No	Operating current, in 50mA units. Indicates the highest current the Sink will draw during the Explicit Contract. A new Request / EPR_Request Message , with an updated Operating Current value, Shall be issued whenever the Sink's power needs change. This value Shall Not exceed the value in the Maximum Current field of the Source_Capabilities Message .

6.4.2.1.4. Adjustable Voltage Supply RDO

Table 6.22. Adjustable Voltage Supply RDO

Bit(s)	Field Name	Static	Description
31...28	Object Position	No	See Section 6.4.2.1.5 .
27	Giveback	No	Deprecated , receiver Shall ignore this field.
26	Capability Mismatch	No	See Section 6.4.2.1.6 .
25	USB Communications Cable	Yes	See Section 6.4.2.1.7 .
24	No USB Suspend	Yes	See Section 6.4.2.1.8 .
23	Unchunked Extended Messages Supported	Yes	See Section 6.4.2.1.9 .
22	EPR Capable	No	See Section 6.4.2.1.10 .
21	Reserved	Yes	Reserved , receiver Shall ignore this field.
20...9	Output Voltage	No	Requested output voltage at the Source's connector, in 25mV units. Bits 10..9 Shall be set to 00b, making the effective voltage step size 100mV. Voltage Shall be greater than or equal to the Minimum Voltage field and less than or equal to the Maximum Voltage field in the selected APDO.
8...7	Reserved	Yes	Reserved , receiver Shall ignore this field.

Bit(s)	Field Name	Static	Description
6...0	Operating Current	No	<p>Operating current, in 50mA units. Indicates the highest current the Sink will draw during the Explicit Contract.</p> <p>A new Request / EPR_Request Message, with an updated Operating Current value, Shall be issued whenever the Sink's power needs change.</p> <p>This value is limited by the contents of the data object within the received Source_Capabilities Message referenced by the Object Position field. For an SPR AVS, this value Shall Not exceed the value in the Maximum Current field of the Source_Capabilities Message. For EPR AVS, the Operating Current field Shall Not exceed the (PDP field from the Source_Capabilities Message) ÷ (Output voltage from this RDO) rounded down to the nearest 50 mA.</p>

6.4.2.1.5. Object Position

The value in the Object Position field **Shall** indicate the corresponding PDO location in the [Source_Capabilities Message](#) when responding with a [Request Message](#) or the [EPR_Source_Capabilities Message](#) when responding with an [EPR_Request Message](#).

The Object Position field values 0001b...0111b **Shall** only be used to refer to SPR PDOs and APDOs. SPR PDOs and APDOs **May** be requested by either a Request or an [EPR_Request Message](#). Object positions 1000b...1011b **Shall** only be used to refer to EPR PDOs and APDOs. EPR PDOs and APDOs **Shall** only be requested by an [EPR_Request Message](#). If the Object Position field in a [Request Message](#) contains a value greater than 0111b, the Source **Shall** send [Hard Reset](#) Signaling.

6.4.2.1.6. Capability Mismatch

A Capability Mismatch occurs when the Source cannot satisfy the Sink's power or voltage requirements based on the Source Capabilities it has offered. In this case the Sink **Shall** make a **Valid** Request from the offered Source Capabilities and **Shall** set the Capability Mismatch bit. When a Capabilities Mismatch condition does not exist, the Sink **Shall Not** set the Capability Mismatch bit.

6.4.2.1.7. USB Communications Capable

When the USB Communications Capable flag is set to 0b, the Sink does not have USB data lines or is otherwise incapable of communicating using either [USB2], [USB3] or [USB4] protocols.

When USB Communications Capable flag is set to 1b, the Sink has USB data lines and is capable of communicating using either [USB2], [USB3] or [USB4] protocols.

The Source uses this bit to determine operation in certain cases such as USB suspend. If the USB Communications Capable flag has been set to 0b by a Sink, then the Source needs to be aware that USB Suspend rules cannot be observed by the Sink.

6.4.2.1.8. No USB Suspend

When the No USB Suspend flag is set to 0b, the Sink **May** reduce its power consumption during USB Suspend.

When set to 1b, it indicates to the Source that this Device is requesting to continue its Explicit Contract during USB Suspend. Sinks setting this flag typically have functionality that can use power for purposes other than USB Communication, e.g., for charging a Battery.

The Source uses this flag to evaluate whether it **Should** re-issue the [Source_Capabilities Message](#) with the USB Suspend Supported flag cleared.

6.4.2.1.9. Unchunked Extended Messages Supported

When the Unchunked [Extended Messages](#) Supported bit set to 0b, Port does not support Unchunked [Extended Messages](#).

When the Unchunked [Extended Messages](#) Supported bit set to 1b, Port supports both Chunked and Unchunked [Extended Messages](#).

6.4.2.1.10. EPR Capable

When the EPR Capable bit is set to 0b, Sink will only consume power Capabilities in the Standard Power Range. Sink **Shall** not Request [EPR Mode](#) entry with this bit cleared.

When the EPR Capable bit is set to 1b, Sink is capable of consuming power Capabilities in the Extended Power Range. The Sink may Request to enter EPR Mode.

6.4.3. BIST Message

The [BIST Message](#) is sent to Request the Port to enter a PHY Layer test Mode (see [Section 5.4](#)) that performs one of the following functions:

- Enters a Continuous BIST Mode to send a continuous stream of test data to the Tester.
- Enters and leave a Shared Capacity Port test Mode.

Table 6.23. BIST Modes

Bit(s)	Field Name	Static	Description
B31...28	BIST Test Mode	No	0101b - BIST Carrier Mode 1000b - BIST Test Data 1001b - BIST Shared Test Mode Entry 1010b - BIST Shared Test Mode Exit All other values are Invalid ; receiver Shall ignore this Message.
B27...0	Reserved	Yes	Reserved , receiver Shall ignore this field.

6.4.3.1. BIST Test Data Mode

Upon receipt of a [BIST Message](#), with a BIST Test Data BIST Data Object, the UUT **Shall** return a [GoodCRC Message](#) and **Shall** enter BIST Test Data Mode in which it sends no further Messages except for [GoodCRC Messages](#) in response to received Messages. The test **Shall** be ended by sending [Hard Reset](#) Signaling to reset the UUT.

6.4.3.2. BIST Shared Capacity Test Mode

A Shared Capacity Port of Ports share a common power Source that is not capable of simultaneously powering all the ports to their full Source Capabilities (see [USB-C]). The BIST Shared Capacity Test Mode **Shall** only be implemented by ports in a Shared Capacity Port.

The UUT Shared Capacity Port of Ports **Shall** contain one or more Ports, designated as Master Ports, that recognize both the BIST Shared Test Mode Entry BIST Data Object and the BIST Shared Test Mode Exit BIST Data Object.

6.4.3.3. BIST Shared Test Mode Entry

When any master Port in a Shared Capacity Port receives a [BIST Message](#) with a BIST Shared Test Mode Entry BIST Data Object, while in the PE_SRC_Ready State, the UUT **Shall** enter a compliance test Mode where the

maximum Source Capabilities are always offered on every Port, regardless of the availability of shared power (i.e., all shared power management is disabled).

Ports in the Shared Capacity Port that are not Master Ports **Shall Not** enter compliance Mode on receiving the BIST Shared Test Mode Entry BIST Data Object.

Upon receipt of a [BIST Message](#), with a BIST Shared Test Mode Entry BIST Data Object, the UUT **Shall** return a [GoodCRC Message](#) and **Shall** enter the BIST Shared Capacity Test Mode.

On entering this Mode, the UUT **Shall** send a New [Source_Capabilities Message](#) from each Port in the Shared Capacity Port within [tBISTSharedTestMode](#) . The Tester will not exceed the shared capacity during this Mode.

6.4.3.4. BIST Shared Test Mode Exit

Upon receipt of a [BIST Message](#), with a BIST Shared Test Mode Exit BIST Data Object, the UUT **Shall** return a [GoodCRC Message](#) and **Shall** exit the BIST Shared Capacity Test Mode. If any other Message, aside from a [BIST Message](#), with a BIST Shared Test Mode Exit BIST Data Object, is received while in BIST Shared Capacity Test Mode , this **Shall Not** cause the UUT to exit the BIST Shared Capacity Test Mode.

On exiting the Mode, the UUT **May** send a New [Source_Capabilities Message](#) to each Port in the Shared Capacity Port or the UUT **May** perform ErrorRecovery on each Port.

Ports in the Shared Capacity Port that are not Master Ports **Shall Not** exit compliance Mode on receiving the BIST Shared Test Mode Entry BIST Data Object.

Ports in the Shared Capacity Port that are not Master Ports **Should Not** exit compliance Mode on receiving the BIST Shared Test Mode Exit BIST Data Object.

- The UUT **Shall** exit BIST Shared Capacity Test Mode when It is powered off.
- The UUT **Shall** remain in BIST Shared Capacity Test Mode for any PD event (except when a BIST Shared Test Mode Exit BIST Data Object, is received); specifically the UUT **Shall** remain in BIST Shared Capacity Test Mode when any of the following PD events occurs:
 - [Hard Reset](#)
 - [Cable Reset](#)
 - [Soft Reset](#)
 - [Data Role Swap](#)
 - [Power Role Swap](#)
 - [Fast Role Swap](#)
 - [VCONN Swap](#).
- The UUT **May** leave BIST Shared Capacity Test Mode if the Tester makes a [Request](#) that exceeds the Capabilities of the UUT.

6.4.4. Battery_Status Message

The [Battery_Status](#) is used to convey current information about a Battery Attached to the system.

This Message is used in the following sequences:

- Get [Battery Status](#)

Usage: [Section 7.21](#)

Table 6.24. Battery Status Data Object (BSDO)

Bit(s)	Field Name	Static	Description
31...16	Battery Present Capacity	No	0000h-FFFEh - Battery's State of Charge (SoC) in 0.1 WH increments FFFFh - Battery's SoC is unknown.
15...12	Reserved	Yes	Reserved , receiver Shall ignore this field.
11...10	Battery Charging Status	No	When Battery Present is 1b : 00b - Battery is Charging. 01b - Battery is Discharging. 10b - Battery is Idle. 11b - Invalid , receiver Shall ignore this field. When Battery Present is 0b : Reserved , receiver Shall ignore this field.
9	Battery Present	No	0b - Battery is not present. 1b - Battery is present.
8	Invalid Battery Reference	No	0b - Get_Battery_Status_Message contained a valid Battery reference or slot. 1b - Get_Battery_Status_Message contained a Battery reference or slot that does not exist.
7...0	Reserved	Yes	Reserved , receiver Shall ignore this field.

6.4.5. Alert Message

The [Alert Message](#) is provided to allow Port Partners to inform each other when there is a status change event.

This Message is used in the following sequences:

- [Alert](#)

Usage: [Section 7.14](#)

Table 6.25. Alert Data Object (ADO)

Bit(s)	Field Name	Static	Description
31	Extended Alert Event	No	When set to 1b, see "Extended Alert Event Type" for more information.
30	OVP Event	No	Shall be set to 1b when the Sink detects its output voltage exceeds its limits triggering its protection circuitry. May be set to 1b when the Source detects its output voltage exceeds its limits triggering its protection circuitry.

Bit(s)	Field Name	Static	Description
29	Source Input Change Event	No	<p>Shall be set to 1b when the Source/Sink's input changes.</p> <p>Examples include:</p> <ul style="list-style-type: none"> The AC input is removed, and the Source/Sink continues to be powered from one or more of its batteries. The AC input returns and the Source/Sink transitions from Battery to AC operation. The Source/Sink changes operation from one (or more) Battery to another (or more) Battery.
28	Operating Condition Change	No	<p>Shall be set to 1b when a Source or Sink detects its Operating Condition enters or exits either the 'warning' or 'over temperature' temperature states.</p> <p>Shall be set to 1b when the Source operating in the Programmable Power Supply Mode detects it has changed its operating condition between Constant Voltage (CV) and Current Limit (CL).</p>
27	OTP Event	No	Shall be set to 1b when a Source or Sink shuts down due to over-temperature triggering its protection circuitry.
26	OCP Event	No	<p>Shall be set to 1b when a Source detects its output current exceeds its limits triggering its protection circuitry.</p> <p>This bit is Reserved for a Sink.</p>
25	Battery Status Change Event	No	<p>Shall be set to 1b when any Battery's power State changes between charging, discharging, neither.</p> <p>For Hot Swappable Batteries, it Shall also be set to 1b when a Battery is Attached or Detached.</p>
24	Reserved	Yes	Reserved , receiver Shall ignore if this field is set.
23...20	Fixed Batteries	No	<p>When Battery Status Change Event is 1b:</p> <p>xxx1b - Event corresponds with Battery 0.</p> <p>xx1xb - Event corresponds with Battery 1.</p> <p>x1xxb - Event corresponds with Battery 2.</p> <p>1xxxb - Event corresponds with Battery 3.</p> <p>When Battery Status Change Event is 0b:</p> <p>Reserved, receiver Shall ignore this field.</p>
19...16	Hot Swappable Batteries	No	<p>When Battery Status Change Event is 1b:</p> <p>xxx1b - Event corresponds with Battery 4.</p> <p>xx1xb - Event corresponds with Battery 5.</p> <p>x1xxb - Event corresponds with Battery 6.</p> <p>1xxxb - Event corresponds with Battery 7.</p> <p>When Battery Status Change Event is 0b:</p> <p>Reserved, receiver Shall ignore this field.</p>
15...4	Reserved	Yes	Reserved , receiver Shall ignore this field.

Bit(s)	Field Name	Static	Description
3...0	Extended Alert Event Type	No	<p>When the Extended Alert Event is 1b:</p> <p>1 - Power State change (DFP only).</p> <p>2 - Power button press (UFP only).</p> <p>3 - Power button release (UFP only).</p> <p>4 - Controller initiated wake e.g., Wake on LAN (UFP only).</p> <p>5 - Source is about to reduce Source Capabilities (Source Only)</p> <p>All other values Reserved, receiver Shall ignore this value.</p> <p>When the Extended Alert Event is 0b:</p> <p>Reserved, receiver Shall ignore this field.</p>

6.4.5.1. Power State Change Extended Event

The Power State change event value **May** be set when the DFP transitions into a new power State. The new power State **Shall** be communicated via the Power State change byte in the [Status Message](#). This Message **Should** be sent by the Host in response to any system power State change.

6.4.5.2. Power Button Press Extended Event

The Power button press event value **May** be set when the power button on the UFP is pressed. The press and release events are separated into two different events so that devices that respond differently to a long button press will see a long button press. On the Host-side, the power button press event typically initiates the same behavior as a power button press of the Host's power button.

6.4.5.3. Power Button Release Extended Event

If a Power button press event was sent, then the Power button release event value **Shall** be sent by the UFP following the Power button press event. If a physical power button press initiated the Power button press event, then the Power button release event **Should** be sent when the physical button is released.

6.4.5.4. Controller Initiated Wake Extended Event

The Controller initiated wake is used to communicate a wake event from the UFP to the DFP such as Wake on LAN from a NIC or another controller. This event doesn't need the press/release form of the Power button press, because it only needs to communicate the presence of the event, and not the timing.

6.4.5.5. Source Reducing Capabilities Extended Event

Every Port in the Source role **Should** send an [Alert Message](#) within tReducePowerAlert before transmitting any new, reduced Source Capabilities. Upon sending this [Alert Message](#), the Status and [Source Info Message](#) fields **Shall** also be updated to reflect the Source's new status. If tReducePowerAlert can't be met, the [Alert Message](#) **Should** be sent as soon as possible before the power reduction. This warning allows the Sink to query for more information or otherwise prepare before receiving the reduced [Source Capabilities Message](#).

6.4.6. Get_Country_Info Message

The [Get_Country_Info Message](#) **Shall** be sent by a Port to get country specific information from its Port Partner.

This Message is used in the following sequences:

- Get Country Info

Usage: [Section 7.24](#)

Table 6.26. Country Code Data Object (CCDO)

Bit(s)	Field Name	Static	Description
31...24	First Character	No	First character of the Alpha-2 Country Code defined by [ISO 3166].
23...16	Second Character	No	Second character of the Alpha-2 Country Code defined by [ISO 3166].
15...0	Reserved	Yes	Reserved , receiver Shall ignore this field.

As an example, if the Request is for China information, then the Country Code Data Object (CCDO) would be CCDO [31:0] = 434E0000h for "CN" country code.

6.4.7. Enter_USB Message

The [Enter_USB Message](#) **Shall** be sent by the DFP to its UFP Port Partner and to the Cable Plug(s) of an Active Cable, when in an Explicit Contract, to enter a specified USB Mode of operation.

This Message is used in the following sequences:

- [Enter_USB](#)

Usage: [Section 7.29](#)

Table 6.27. Enter_USB Data Object (EUDO)

Bit(s)	Field Name	Static	Description
31	Reserved	Yes	Reserved , receiver Shall ignore this field.
30...28	USB Mode	No	Used by the DFP to direct the USB Mode the Port Partner is to enter. 000b - USB2.0 001b - USB3.2 010b - USB4 All other values are Invalid ; receiver Shall treat as 010b (USB4). Entry into USB3.2 and USB4 include entry into USB2.0.
27	Reserved	Yes	Reserved , receiver Shall ignore this field.
26	[USB4] DRD ¹	No	Shall be set to 1b when the Host DFP is capable of operating as a [USB4] Device. A [USB4] Host DFP that sets the USB4 DRD field Shall also be capable of operating as a [USB2] Device.
25	USB3 DRD ¹	No	Shall be set to 1b when the Host DFP is capable of operating as a [USB3] Device. A [USB3] Host DFP that sets the USB3DRD field Shall also be capable of operating as a [USB2] Device.
24	Reserved	Yes	Reserved , receiver Shall ignore this field.

Bit(s)	Field Name	Static	Description
23...21	Cable Speed ¹	No	Used to indicate the cable's maximum speed. The value is read from the Cable Plug and interpreted by the DFP. 000b - [USB2]only, no SuperSpeed support. 001b - [USB3] Gen1. 010b - [USB3] Gen2 and [USB4] Gen2. 011b - [USB4] Gen3. 100b - [USB4] Gen4. All other values are Invalid ; receiver Shall treat as 100b (USB4 Gen 4).
20...19	Cable Type ¹	No	The value is read from the Cable Plug and interpreted by the DFP: 00b - Passive. 01b - Active Re-timer. 10b - Active Re-driver. 11b - Optically Isolated.
18...17	Cable Current ¹	No	Used to indicate the cable's current carrying capability: 00b - VBUS is not supported. 01b - Invalid , receiver Shall treat as 00b (VBUS is not supported). 10b - 3A 11b - 5A
16	PCIe Support ¹	No	0b - PCIe tunneling not supported. 1b - [USB4] PCIe tunneling supported by the Host.
15	DP Support ¹	No	0b - DP tunneling not supported 1b - [USB4] DP tunneling supported by the Host.
14	TBT Support ¹	No	0b - [TBT3] Thunderbolt is not supported 1b - [TBT3] Thunderbolt is supported by the Host's USB4 Connection Manager.
13	Host Present ¹		0b - No Host is present 1b - A Host is present at the top of the USB tree. When this bit is set to 1b, PCIe Support, DP Support and TBT Support represent the Host's Capabilities that Shall be propagated down the Hub tree.
12...0	Reserved	Yes	Reserved , receiver Shall ignore this field.
1. Field Shall be ignored when received by a Cable Plug			

Cable Speed, Cable Type, and Cable Current **Shall** be determined by the DFP as defined by [USB-C] in the USB4 Discovery and Entry Section.

6.4.8. EPR_Request Message

The [EPR_Request Message](#) **Shall** return a Request Data Object (RDO) that **Shall** identify the Power Data Object followed by a copy of the Power Data Object being requested.

This Message is used in the following sequences:

- EPR Source Capabilities

- EPR Request
- EPR Get Source Capabilities

Usage: [Section 7.30.3](#)

Figure 6.3. Figure: EPR_Request Message



Note: This differs from the [Request Message](#), which only contains a single RDO.

Note: The requested Power Data Object (PDO) **May** be an EPR PDO or APDO or an SPR PDO or APDO.

6.4.9. EPR_Mode Message

The [EPR_Mode Message](#) is used to enter, acknowledge, and exit the [EPR Mode](#).

This Message is used in the following sequences:

- [EPR Mode](#) Enter
- [EPR Mode](#) Exit

Usage: [Section 7.30](#)

Table 6.28. EPR Mode Data Object (EPRMDO)

Bit(s)	Field Name	Static	Description
31...24	Action	No	01h - Enter EPR Mode (sent by Sink only). 02h - Enter Acknowledged (sent by Source only). 03h - Enter Succeeded (sent by Source only). 04h - Enter Failed (sent by Source only). 05h - Exit EPR Mode (sent by Source/Sink). All other values are Invalid ; receiver Should ignore this Message.
23...16	Data	No	If Action is 01h (Enter EPR Mode): 00h-FFh - EPR Sink Operational PDP (this Shall match the "EPR Sink Operational PDP" field in Sink_Capabilities_Extended Message). If Action is 02h (Enter Acknowledged): Reserved , receiver Shall ignore this field. If Action is 03h (Enter Succeeded): Reserved , receiver Shall ignore this field. If Action is 04h (Enter Failed): 00h - Unknown cause of failure. 01h - Cable not EPR Capable. 02h - Source failed to become VCONN Source. 03h - EPR Capable bit not set in RDO. 04h - Source unable to enter EPR Mode . The Sink May retry entering EPR Mode after receiving this Enter Failed response. 05h - EPR Capable bit not set in PDO. All other values are Invalid ; receiver Should ignore this field. If Action is 05h (Exit EPR Mode): Reserved , receiver Shall ignore this field.
15...0	Reserved	Yes	Reserved , receiver Shall ignore this field.

6.4.10. Source_Info Message

Used to convey the overall Capabilities of a Port to act as a Source Bit(s) Field Name **Static** Description. The [Source_Info Message](#) contains two [Source Information](#) Data Objects (SIDO1 and SIDO2).

This Message is used in the following sequences:

- Get [Source Information](#)

Usage: [Section 7.26](#)

Table 6.29. Source_Info Data Object 1 (SIDO1)

Bit(s)	Field Name	Static	Description
31	Port Type	Yes	Indicates whether the amount of power the Port can provide is fixed or can change dynamically. 0b - Managed Capability Port. 1b - Guaranteed Capability Port. See Section 3.2 for description of Managed vs Guaranteed Capability.
30...24	Reserved	Yes	Reserved , receiver Shall ignore this field.
23...16	Port Maximum PDP	Yes	Maximum power the Port will provide in 1W steps.
15...8	Port Present PDP	No	Power the Port is presently capable of supplying in 1W steps, including limitations due to Cable Capabilities or abnormal operating conditions (e.g., elevated temperature, low input voltage, etc.).
7...0	Port Reported PDP	No	Power the Port is offering, in 1W steps, equal to the power offered in the Source_Capabilities Message or EPR_Source_Capabilities Message . Note: Computed as the integer part of, the largest of the products of the voltage times current of the Fixed Supply PDOs returned in the Source_Capabilities Message or EPR_Source_Capabilities Messages . May be Static for Guaranteed Capability Ports.

Table 6.30. Source_Info Data Object 2 (SIDO2)

Bit(s)	Field Name	Static	Description
31	Port Type	Yes	Indicates whether the amount of power the Port can provide is fixed or can change dynamically. 0b - Managed Capability Port. 1b - Guaranteed Capability Port. See Section 3.2 for description of Managed vs Guaranteed Capability.
30	DPS Port	Yes	Indicates whether the Source will behave as a Dynamic Power Source. 0b - Non DPS Port 1b - DPS Port (Port Type Shall be set to 0b in both SIDO1 and SIDO2)
29...18	Reserved	Yes	Reserved , receiver Shall ignore this field.
17...9	Port Maximum PDP	Yes	Maximum power the Port will provide in 0.5W steps.
8...0	Port Guaranteed PDP	Yes	Minimum power the Port is guaranteed to always be able to provide in 0.5W steps.

6.4.11. Revision Message

This Message is used to identify the highest PD Specification Revision at which the Port is capable of operating.

This Message is used in the following sequences:

- Get Revision

Usage: [Section 7.25](#)

Table 6.31. Revision Message Data Object (RMDO)

Bit(s)	Field Name	Static?	Description
31...28	Revision Major	Yes	Major Value of PD Revision.
27...24	Revision Minor	Yes	Minor Value of PD Revision.
23...20	Version Major	Yes	Major Value of PD Version.
19...16	Version Minor	Yes	Minor Value of PD Version.
15...0	Reserved	Yes	Reserved , receiver Shall ignore this field.

As an example, Revision 3.2, Version 1.2 would be represented as: 0x32120000.

6.4.12. Vendor Defined Message

The Vendor_Defined Message (VDM) is provided to allow vendors to exchange information outside of that defined by this specification.

A Vendor_Defined Message **Shall** consist of at least one Vendor Data Object (VDO), the VDM Header, and **May** contain up to a maximum of six additional VDOs.

To ensure vendor uniqueness of Vendor_Defined Messages, all Vendor_Defined Messages **Shall** contain a **Valid** USB Standard or Vendor ID (SVID) allocated by USB-IF in the VDM Header.

Two types of Vendor_Defined Messages are defined: Structured VDMs and Unstructured VDMs. A [Structured VDM](#) defines an extensible structure designed to support Modal Operation. An Unstructured VDM does not define any structure and Messages **May** be created in any manner that the vendor chooses.

Vendor_Defined Messages **Shall Not** be used for direct power Negotiation. They **May** however be used to alter Local Policy, affecting what is offered or consumed via the normal PD Messages. The Message format **Shall** be as shown in [Figure 6.4](#).

Figure 6.4. Figure: Vendor Defined Message

The VDM Header **Shall** be the first 4-byte object in a [Vendor Defined Message](#). The VDM Header provides Command space to allow vendors to customize Messages for their own purposes. Additionally, vendors **May** make use of the Commands in a Structured VDM.

The fields in the VDM Header for an Unstructured VDM, when the VDM Type Bit is set to zero, **Shall** be as defined in [Table 6.32](#) . The fields in the VDM Header for a [Structured VDM](#), when the VDM Type Bit is set to one **Shall** be as defined in [Table 6.33](#) .

This Message is used in the following sequences:

- [Discover Identity](#) (SVDM)
- [Discover SVIDs](#) (SVDM)
- [Discover Modes](#) (SVDM)
- [Enter Mode](#) (SVDM)

- [Exit Mode](#) (SVDM)
- [Structured VDM](#) (Other)

6.4.12.1. Unstructured VDM

The Unstructured VDM does not define the contents of bits 14...0 in the VDM Header. Their definition and use are the sole responsibility of the vendor indicated by the VID. The Port Partners and Cable Plugs **Shall** exit any states entered using an Unstructured VDM when a [Hard Reset](#) appears on PD.

Usage: [Section 8.4](#)

Table 6.32. Unstructured VDM Header

Bit(s)	Parameter	Static	Description
31...16	Vendor ID (VID)	Yes	Unique 16-bit unsigned integer. Assigned by the USB-IF to the Vendor. The Vendor ID (VID) field Shall contain the 16-bit Vendor ID value assigned to the vendor by the USB-IF (VID). No other value Shall be present in this field.
15	VDM Type	Yes	0 = Unstructured VDM The VDM Type field Shall be set to zero indicating that this is an Unstructured VDM.
14...0	Available for Vendor Use	No	Content of this field is defined by the vendor.

6.4.12.2. Structured VDM

Setting the VDM Type field to 1 ([Structured VDM](#)) defines the use of bits B14...0 in the [Structured VDM](#) Header. The fields in the [Structured VDM](#) Header are defined in [Table 6.33](#) .

Usage: [Section 8.5](#)

Table 6.33. Structured VDM Header

Bit(s)	Field Name	Static	Description
31...16	Standard or Vendor ID (SVID)	Yes	16-bit USB Standard ID value (SID) or the 16-bit assigned to the vendor by the USB-IF (VID). No other value Shall be present in this field. SVID values referenced by this specification include 0xFF00 - PD SID allocated to this specification by USB-IF 0xFF01 - DPTC SID allocated to [DPTC] by USB-IF.
15	VDM Type	Yes	1 = Structured VDM .
14...13	Structured VDM Version (Major)	Yes	Version Number (Major) of the Structured VDM (not this specification). This field indicates the expected content for this VDO. 00b - Version 1.0 (Used by USB PD implementations conforming to USB PD Revision 3.0, Version 1.0 - Deprecated for new designs). 01b - Version 2.x (Used by USB PD R3.x implementations starting with USB PD Revision 3.0, Version 1.0a). 10b...11b - Invalid , receiver uses 01b (Version 2.x)

Bit(s)	Field Name	Static	Description
12...11	Structured VDM Version (Minor)	Yes	<p>Version Number (Minor) of the Structured VDM (not this specification). This field indicates the expected content for this VDO.</p> <p>When Command = 0...15:</p> <p>00b - Version 2.0 (Used for ports implemented prior to USB PD Revision 3.1, Version 1.6).</p> <p>01b - Version 2.1 (Used for ports implemented starting with USB PD Revision 3.1, Version 1.6)</p> <p>10b...11b - Invalid, Shall Not be used</p> <p>When Command = 16...31 :</p> <p>00b...11b - Defined by the SVID.</p>
10...8	Object Position	No	<p>Shall be used by the Enter Mode, Exit Mode and Attention commands to indicate which VDO (e.g., Alternate Mode) the Command refers to.</p> <p>When Command = 0...3, 7...15:</p> <p>000b - Shall be used.</p> <p>001b...111b - Invalid; receiver Should ignore this field.</p> <p>When Command = 4...6:</p> <p>000b = Invalid, Shall Not be used.</p> <p>001b...110b = Index into the list of VDOs to identify the desired Alternate Mode VDO. Values greater than the number of VDOs supplied by Discover Modes ACK are Invalid.</p> <p>111b = Exit all Active Modes (equivalent of a power on reset). Shall only be used with the Exit Mode Command.</p> <p>When Command = 16...31:</p> <p>000b - 111b - defined by the SVID</p> <p>See Table 6.33 .</p>
7...6	Command Type	No	<p>Shall be used to indicate the type of Command Request/response being sent.</p> <p>00b - REQ (Request from Initiator Port)</p> <p>01b - ACK (Acknowledge Response from Responder Port)</p> <p>10b - NAK (Negative Acknowledge Response from Responder Port)</p> <p>11b - BUSY (Busy Response from Responder Port)</p> <p>See Table 6.33 .</p>
5	Reserved	Yes	Reserved , receiver Shall ignore this field.

Bit(s)	Field Name	Static	Description
4...0	Command	No	<p>VDM Command being sent.</p> <p>0 - Invalid; receiver Shall respond with NAK.</p> <p>1 - Discover Identity</p> <p>2 - Discover SVIDs</p> <p>3 - Discover Modes</p> <p>4 - Enter Mode</p> <p>5 - Exit Mode</p> <p>6 - Attention</p> <p>7...15 - Invalid; receiver Shall respond with NAK.</p> <p>16...31 - SVID Specific Commands</p> <p>See Table 6.32 .</p>

6.4.12.2.1. Structured VDM Version

The [Structured VDM](#) Version (Major)/[Structured VDM](#) Version (Minor) fields indicate the level of functionality supported in the [Structured VDM](#) part of the specification. This is not the same Version as the Version of this specification. The [Structured VDM](#) Version (Major) **Shall** be set to 01b to indicate Version 2.x with the [Structured VDM](#) Version (Minor) field set as appropriate based on whether the Port is implemented to USB PD Revision 3.1, Version 1.6 (or newer) or a prior Version.

6.4.12.2.2. Object Position

The Object Position field **Shall** be used by the [Enter Mode](#) and [Exit Mode](#) Commands. The [Discover Modes](#) Command returns a list of zero to six VDOs, each of which describes an Alternate Mode. The value in Object Position field is an index into that list that indicates which VDO (e.g., Alternate Mode) in the list the [Enter Mode](#) and [Exit Mode](#) Command refers to. The Object Position **Shall** start with one for the first Alternate Mode in the list.

This field **Shall** be set to zero in the Request or Response (REQ, ACK, NAK or BUSY) when not required by the specification of the individual Command.

6.4.12.2.3. Command Type

This Command Type field **Shall** be used to indicate the type of Command Request/response being sent.

An Initiator **Shall** set the Command Type field to REQ to indicate that this is a Command Request from an Initiator. If Structured VDMs are supported, then the responses are as follows:

- "Responder ACK" is the normal return and **Shall** be sent to indicate that the Command Request was received and handled normally.
- "Responder NAK" **Shall** be returned when the Command Request meets one of the following criteria:
 - Has an **Invalid** parameter (e.g., **Invalid** SVID or Alternate Mode).
 - Cannot be acted upon because the configuration is not correct (e.g., an Alternate Mode which has a dependency on another Alternate Mode or a Request to exit an Alternate Mode which is not an Active Mode).
 - Is an Unrecognized Message.

- The handling of "Responder NAK" is left up to the Initiator.
- A "Responder BUSY" **Shall** be sent in the response to a VDM when the Responder is unable to immediately respond to the Command Request, but the Command Request **May** be retried.

6.4.12.2.4. Command

The Command field contains the value for the VDM Command being sent. The Commands explicitly listed in the Command field are used to identify devices and manage their operational Modes. There is a further range of Command values left for the vendor to use to manage additional extensions.

6.4.12.3. Discover Identity

The [Discover Identity](#) Command is provided to enable an Initiator to identify its Port Partner and for an Initiator (VCONN Source) to identify the Responder (Cable Plug or VPD). The [Discover Identity](#) Command is also used to determine whether a Cable Plug or VPD is PD-Capable by looking for a [GoodCRC Message](#) Response.

If the product is a DRD, the ID Header VDO **Shall** declare both a Product Type (UFP) and a Product Type (DFP). The [Discover Identity](#) ACK **Shall** contain Product Type VDOs for both UFP and DFP beginning with the UFP VDO, followed by a 32-bit Pad Object (defined as all 0s), followed by the DFP VDO as shown in [Figure 6.5](#).

Figure 6.5. Discover Identity Response Format

Product Type: UFP	Message Header Message Type = Vendor_Defined Number of Data Objects = 5	VDM Header Command = Discover Identity Command Type = ACK VDM Type = Structured SVID = PD	ID Header VDO	Cert Stat VDO	Product VDO	UFP VDO		
Product Type: DFP	Message Header Message Type = Vendor_Defined Number of Data Objects = 5	VDM Header Command = Discover Identity Command Type = ACK VDM Type = Structured SVID = PD	ID Header VDO	Cert Stat VDO	Product VDO	DFP VDO		
Product Type: DRD	Message Header Message Type = Vendor_Defined Number of Data Objects = 7	VDM Header Command = Discover Identity Command Type = ACK VDM Type = Structured SVID = PD	ID Header VDO	Cert Stat VDO	Product VDO	UFP VDO	Padding (0)	DFP VDO
Product Type: Cable/VPD	Message Header Message Type = Vendor_Defined Number of Data Objects = 5	VDM Header Command = Discover Identity Command Type = ACK VDM Type = Structured SVID = PD	ID Header VDO	Cert Stat VDO	Product VDO	Passive Cable/ ActiveCable/ VPD VDO		

Usage: [Section 8.6.1](#)

6.4.12.3.1. ID Header VDO

The ID Header VDO contains information corresponding to the Power Delivery Product.

Table 6.34. ID Header VDO

Bit(s)	Field Name	Static	Description
31	Communications Capable as USB Host	Yes	0b - Product is not capable of enumerating USB Devices. 1b - Product is capable of enumerating USB Devices.
30	Communications Capable as USB Device	Yes	0b - Product is not capable of being enumerated as a USB Device. 1b - Product is capable of being enumerated as a USB Device.

Bit(s)	Field Name	Static	Description
29...27	Product Type (UFP/Cable Plug/VPD)	Yes	SOP (UFP): 000b - Not a UFP 001b - PDUSB Hub 010b - PDUSB Peripheral 011b - PSD 100b - Invalid ; receiver Should ignore this field 101b - Deprecated , Alternate Mode Adapter (AMA) 110b...111b - Invalid ; receiver Should ignore this field SOP' (Cable Plug/VPD): 000b - Not a Cable Plug/VPD 001b...010b - Invalid ; receiver Should ignore this field 011b - Passive Cable 100b - Active Cable 101b - Invalid ; receiver Should ignore this field 110b - VCONN Powered USB Device (VPD) 111b - Invalid, Shall Not be use
26	Modal Operation Supported	Yes	Shall indicate whether or not the Product (either a Cable Plug or a Device that can operate in the UFP role) is capable of supporting Modes. A product that supports Modal Operation Shall respond to the Discover SVIDs Command with a list of SVIDs for all of the Modes it is capable of supporting whether or not those Modes can currently be entered. 0b - Product is not capable of supporting Modal Operation. 1b - Product is capable of supporting Modal Operation. Note: The Modal Operation Supported bit does not describe a DFP's Alternate Mode Controller functionality.
25...23	Product Type (DFP)	Yes	SOP (DFP): 000b - Not a DFP 001b - PDUSB Hub 010b - PDUSB Host 011b - Power Brick 100b - Deprecated , Alternate Mode Controller (AMC). 101b...111b - Invalid, Shall Not be used. SOP': Reserved, Shall Not be used. See Section 6.4.12.3.1 .
22...21	Connector Type	Yes	00b - Deprecated , Unknown connector type 01b - Invalid, Shall Not be used. 10b - USB Type-C Receptacle 11b - USB Type-C Plug

Bit(s)	Field Name	Static	Description
20...16	Reserved	Yes	Reserved , receiver Shall ignore this field.
15...0	USB Vendor ID	Yes	Vendor ID assigned to the product vendor by USB-IF. For USB Devices or Hubs which support USB Communications the USB Vendor ID field Shall be identical to the Vendor ID field defined in the product's USB Device Descriptor (see [USB2] and [USB3]). See Section 6.1.5 .

6.4.12.3.2. Product Type (UFP/Cable Plug/VPD)

6.4.12.3.2.1. SOP (UFP)

The SOP Product Type (UFP) field indicates the type of Product when in UFP Data Role, whether a VDO will be returned and if so the type of VDO to be returned. The Product Type indicated in the SOP Product Type (UFP) field **Shall** be the closest categorization of the main functionality of the Product in UFP Data Role or "Undefined" when there is no suitable category for the product. For DRD Products this field **Shall** always indicate the Product Type when in UFP role regardless of the present Data Role. [Table 6.35](#) defines the Product Type VDOs which **Shall** be returned.

Table 6.35. Product Types (UFP)

Field Name	Product Type	Description	VDO
Product Type (UFP/Cable Plug/VPD)	Not a UFP	Shall be used when this is not a UFP.	None
Product Type (UFP/Cable Plug/VPD)	PDUSB Hub	Shall be used when the Product is a PDUSB Hub.	UFP VDO
Product Type (UFP/Cable Plug/VPD)	PDUSB Peripheral	Shall be used when the Product is a PDUSB Device other than a PDUSB Hub.	UFP VDO
Product Type (UFP/Cable Plug/VPD)	PSD	Shall be used with the Product is a PSD (e.g., power bank).	None

6.4.12.3.2.2. SOP' (Cable Plug/VPD)

The SOP' Product Type (Cable Plug/VPD) field indicates the type of Product when the Product is a Cable Plug or VPD, whether a VDO will be returned and if so the type of VDO to be returned. [Table 6.36](#) defines the Product Type VDOs which **Shall** be returned.

Table 6.36. Product Types (Cable Plug/VPD)

Field Name	Product Type	Description	VDO
Product Type (UFP/Cable Plug/VPD)	Not a Cable Plug/VPD	Shall be used where no other Product Type value is appropriate.	None
Product Type (UFP/Cable Plug/VPD)	Active Cable	Shall be used when the Product is a cable that incorporates signal conditioning circuits.	Active Cable VDO
Product Type (UFP/Cable Plug/VPD)	Passive Cable	Shall be used when the Product is a cable that does not incorporate signal conditioning circuits.	Passive Cable VDO
Product Type (UFP/Cable Plug/VPD)	VCONN Powered USB Device	Shall be used when the Product is a PDUSB VCONN Powered USB Device.	VPD VDO

6.4.12.3.3. Product Type (DFP)

6.4.12.3.3.1. SOP (DFP)

The SOP - Product Type (DFP) field indicates the type of Product when in DFP Data Role, whether a VDO will be returned and if so the type of VDO to be returned. The Product Type indicated in the SOP - Product Type (DFP) field **Shall** be the closest categorization of the main functionality of the Product in DFP Data Role or "Undefined" when there is no suitable category for the product. For DRD Products this field **Shall** always indicate the Product Type when in DFP role regardless of the present Data Role. [Table 6.37](#) defines the Product Type VDOs which **Shall** be returned.

In SOP' Communication (Cable Plugs and VPDs) this bit field is **Reserved** and **Shall** be set to zero.

Where there is no Product Type VDO defined for a specific Product Type, no VDOs **Shall** be sent as part of the [Discover Identity](#) Command response.

Table 6.37. Product Type (DFP)

Field Name	Product Type	Description	VDO
Product Type (DFP)	Not a DFP	Shall be used where no other Product Type value is appropriate.	None
Product Type (DFP)	PDUSB Hub	Shall be used when the Product is a PDUSB Hub.	DFP VDO
Product Type (DFP)	PDUSB Host	Shall be used when the Product is a PDUSB Host or a PDUSB Host that supports one or more Alternate Modes as and AMC.	DFP VDO
Product Type (DFP)	Power Brick	Shall be used when the Product is a Power Brick.	DFP VDO

6.4.12.3.4. Cert Stat VDO

The Cert Stat VDO **Shall** contain the XID assigned by USB-IF to the product before certification in binary format. If the vendor does not have an XID, then it **Shall** return zero in this field.

Table 6.38. Cert Stat VDO

Bit(s)	Field Name	Static	Description
31...0	XID	Yes	Value provided by the USB-IF and assigned to the product by the vendor.

6.4.12.3.5. Product VDO

The Product VDO contains identity information relating to the product.

Table 6.39. Product VDO

Bit(s)	Field Name	Static	Description
31...16	USB Product ID	Yes	Product ID assigned to the Port by the Vendor. See Section 6.1.5
15...0	bcdDevice	Yes	Device Version assigned to the Port by the Vendor

Manufacturers **Should** set the USB Product ID field to a unique value identifying the product and **Should** set the bcdDevice field to a Version number relevant to the release Version of the product.

6.4.12.3.6. UFP VDO

The UFP VDO defined in this section **Shall** be returned by Ports capable of operating as a UFP including traditional USB peripherals, USB Hub's upstream Port and DRD capable Host Ports. The UFP VDO defined in this section **Shall** be sent when the Product Type (UFP) field in the ID Header VDO is given as a PDUSB Peripheral or PDUSB Hub.

A [USB4] UFP **Shall** support the [Structured VDM Discover Identity](#) Command.

Table 6.40. UFP VDO

Bit(s)	Field Name	Static	Description
31...29	UFP VDO Version	Yes	Version Number of the VDO (not this specification Version). This field indicates the expected content for the UFP VDO. 000b - Invalid, Shall Not be used. 001b - Deprecated , Version 1.1 010b - Deprecated , Version 1.2 011b - Version 1.3 100b...111b - Invalid, Shall Not be used.
28	Reserved	Yes	Reserved , receiver Shall ignore this field.
27	USB4 Device Capability	Yes	UFP's Capability when operating as either a PDUSB Device or PDUSB Hub. 0b - Device is not USB4 capable. 1b - Device is USB4 capable.
26	USB 3.2 Device Capability	Yes	UFP's Capability when operating as either a PDUSB Device or PDUSB Hub. 0b - Device is not USB 3.2 capable 1b - Device is USB 3.2 capable
25..24	USB 2.0 Device Capability	Yes	UFP's Capability when operating as either a PDUSB Device or PDUSB Hub. 00b - Device is not USB 2.0 capable. 01b - Device is USB 2.0 capable of USB 2.0 as a billboard Device only. 10b - Device is capable of USB 2.0. 11b - Invalid , receiver Shall assume 00b.
23...22	Connector Type (Legacy)	Yes	Deprecated. Shall be set to 00b, and Shall be Ignored by the receiver. The Connector Type (Legacy) field was previously used for the UFP VDO's Connector Type. The receiver can find this information in the Connector Type field in the ID Header VDO.
21...11	Reserved	Yes	Reserved , receiver Shall ignore this field.

Bit(s)	Field Name	Static	Description
10...8	VCONN Power	Yes	<p>When the VCONN Required field is set to "Yes" the VCONN Power Field indicates the VCONN power needed by the AMA for full functionality:</p> <p>000b - 1W</p> <p>001b - 1.5W</p> <p>010b - 2W</p> <p>011b - 3W</p> <p>100b - 4W</p> <p>101b - 5W</p> <p>110b - 6W</p> <p>111b - Invalid, Shall Not be used.</p> <p>When the VCONN Required field is set to "No" the VCONN Power field is Reserved and Shall be set to zero.</p>
7	VCONN Required	Yes	<p>Indicates whether the AMA requires VCONN in order to Function.</p> <p>0b - No</p> <p>1b - Yes</p> <p>When the Alternate Mode fields indicate no modes are supported, the VCONN Required field is Reserved and Shall be set to zero.</p>
6	VBUS Required	Yes	<p>Indicates whether the AMA requires VBUS in order to function.</p> <p>0b - Yes</p> <p>1b - No</p> <p>When the Alternate Mode fields indicate no modes are supported, the VBUS Required field is Reserved and Shall be set to zero.</p>
5	No Signal Reconfig. Alternate Mode Support	Yes	<p>Indicates support for Alternate Modes that do not reconfigure the signals on the [USB-C] connector.</p> <p>0b - No</p> <p>1b - Yes</p>
4	Non-[TBT3] Signal Reconfig. Alternate Mode Support	Yes	<p>Indicates support for Alternate Modes that reconfigure the signals on the [USB-C] connector - except for [TBT3]</p> <p>0b - No</p> <p>1b - Yes</p>
3	[TBT3] Alternate Mode Support	Yes	<p>Indicates support for [TBT3] Alternate Mode</p> <p>0b - No</p> <p>1b - Yes</p>

Bit(s)	Field Name	Static	Description
2...0	USB Highest Speed	Yes	<p>Port's highest speed capability.</p> <p>000b - [USB2] only, no SuperSpeed support</p> <p>001b - [USB3] Gen1</p> <p>010b - [USB3]/[USB4] Gen2</p> <p>011b - [USB4] Gen3</p> <p>100b - [USB4] Gen4</p> <p>101b...111b - Invalid, Shall Not be used.</p> <p>Note: The DFP Shall consider all values, including Invalid, indicated in this field that are higher than the highest value that the DFP recognizes as being Valid and functionally compatible with the highest speed that the DFP supports.</p>

6.4.12.3.7. DFP VDO

The DFP VDO **Shall** be returned by Ports capable of operating as a DFP; including those implemented by Hosts, Hubs and Power Bricks. The DFP VDO **Shall** be returned when the Product Type (DFP) field in the ID Header VDO is given as Power Brick, PDUSB Host or PDUSB Hub.

Table 6.41. DFP VDO

Bit(s)	Field Name	Static	Description
31...29	DFP VDO Version	Yes	Version Number of the VDO (not this specification Version). This field indicates the expected content for the DFP VDO. 000b - Invalid, Shall Not be used. 001b - Deprecated , Version 1.1 010b - Version 1.2 011b...111b - Invalid, Shall Not be used.
28...27	Reserved	Yes	Reserved , receiver Shall ignore this field.
26	USB 4 Host Capability	Yes	DFP's Capability when operating as a PDUSB Hub. 0b - Host is not USB 4 capable 1b - Host is USB 4 capable
25	USB 3.2 Host Capability	Yes	DFP's Capability when operating as a PDUSB Host. 0b - Host is not USB 3.2 capable. 1b - Host is USB 3.2 capable. Power Bricks and PDUSB Hubs Shall set to zero.
24	USB 2.0 Host Capability	Yes	DFP's Capability when operating as a PDUSB Host. 0b - Device is not USB 2.0 capable. 1b - Device is USB 2.0 capable. Power Bricks and PDUSB Hubs Shall set to zero.
23...22	Connector Type (Legacy)	Yes	Deprecated. Shall be set to 00b, and Shall be Ignored by the receiver. The Connector Type (Legacy) field was previously used for the UFP VDO's Connector Type. The receiver can find this information in the Connector Type field in the ID Header VDO (Section 6.4.12.3.1).
21...5	Reserved	Yes	Reserved , receiver Shall ignore this field.
4...0	Port Number	Yes	Unique number that unambiguously identifies each [USB-C] DFP, including DRPs, on the Device. This number is independent of the USB Port number.

6.4.12.3.8. Passive Cable VDO

The Passive Cable VDO defined in this section **Shall** be sent when the Product Type is given as Passive Cable.

A Passive Cable has a USB Plug on each end at least one of which is a Cable Plug supporting SOP' Communication. A Passive Cable **Shall Not** incorporate data bus signal conditioning circuits and hence has no concept of Super Speed Directionality. A Passive Cable **Shall** include a VBUS wire and **Shall** only respond to SOP' Communication. Passive Cables **Shall** support the [Structured VDM Discover Identity](#) Command and **Shall** return the Passive Cable VDO in a [Discover Identity](#) Command ACK.

Table 6.42. Passive Cable VDO

Bit(s)	Field Name	Static	Description
31...28	Hardware Version	Yes	0000b...1111b - HW Version assigned by the VID owner.
27...24	Firmware Version	Yes	0000b...1111b - FW Version assigned by the VID owner.

Bit(s)	Field Name	Static	Description
23...21	VDO Version	Yes	Version Number of the VDO (not this specification Version). This field indicates the expected content for this VDO. 000b - Version 1.0 001b...111b - Invalid, Shall Not be used.
20	Reserved	Yes	Reserved , receiver Shall ignore this field.
19...18	USB Type-C plug to USB Type-C/Captive (Passive Cable)	Yes	Indicates whether the opposite end from the USB Type-C plug is another USB Type-C plug (i.e., a detachable Standard USB Type-C Cable Assembly) or is a Captive Cable Assembly. 00b - Deprecated , USB Type-A 01b - Deprecated , USB Type-B 10b - USB Type-C 11b - Captive
17	EPR Capable (Passive Cable)	Yes	Indicates when the cable is specifically designed for safe operation when carrying up to 48 volts at 5 amps. 0b - Cable is not EPR Capable 1b - Cable is EPR Capable
16...13	Cable Latency (Passive Cable)	Yes	Signal latency through the cable which can be used as an approximation for its length. 0000b - Invalid, Shall Not be used 0001b - <10ns (~1m) 0010b - 10ns to 20ns (~2m) 0011b - 20ns to 30ns (~3m) 0100b - 30ns to 40ns (~4m) 0101b - 40ns to 50ns (~5m) 0110b - 50ns to 60ns (~6m) 0111b - 60ns to 70ns (~7m) 1000b - > 70ns (>~7m) 1001b...1111b - Invalid, Shall Not be used
12...11	Cable Termination Type (Passive Cable)	Yes	Indicates whether the Passive Cable needs VCONN only initially in order to support the Discover Identity Command, after which it can be removed, or the Passive Cable needs VCONN to be continuously applied in order to power some feature of the Cable Plug. 00b - VCONN not required. Cable Plugs that only support Discover Identity Commands Shall set these bits to 00b. 01b - VCONN required 10b...11b - Invalid, Shall Not be used.

Bit(s)	Field Name	Static	Description
10...9	Maximum VBUS Voltage (Passive Cable)	Yes	<p>Maximum voltage that Shall be Negotiated using a Fixed Supply over the cable as part of an Explicit Contract where the maximum voltage that Shall be applied to the cable is $v_{SrcNew} \max + v_{SrcValid} \max$.</p> <p>For example, when the Maximum VBUS Voltage (Passive Cable) field is 20V, a Fixed Supply of 20V can be Negotiated as part of an Explicit Contract where the absolute maximum voltage that can be applied to the cable is 21.55V. Similarly, when the Maximum VBUS Voltage (Passive Cable) field is 50V, a Fixed Supply of 48V can be Negotiated as part of an Explicit Contract where the absolute maximum voltage that can be applied to the cable is 50.9V.</p> <p>EPR Sinks with a captive cable Shall report 50V.</p> <p>00b - 20V</p> <p>01b..10b - Deprecated, receiver Shall assume 00b (20V).</p> <p>11b - 50V</p>
8...7	Reserved	Yes	Reserved , receiver Shall ignore this field.
6...5	VBUS Current Handling Capability (Passive Cable)	Yes	<p>Indicates whether the cable is capable of carrying 3A or 5A.</p> <p>00b - Invalid; receiver Shall assume 01b (3A).</p> <p>01b - 3A</p> <p>10b - 5A</p> <p>11b - Invalid; receiver Shall assume 01b (3A).</p>
4...3	Reserved	Yes	Reserved , receiver Shall ignore this field.
2...0	USB Highest Speed (Passive Cable)	Yes	<p>Highest data rate the cable supports.</p> <p>000b - [USB2] only, no SuperSpeed support.</p> <p>001b - [USB3] Gen1</p> <p>010b - [USB3]/[USB4] Gen2</p> <p>011b - [USB4] Gen3</p> <p>100b - [USB4] Gen4</p> <p>101b...111b - Invalid, Shall Not be used.</p> <p>Note: The DFP Shall consider all values, including Invalid, indicated in this field that are higher than the highest value that the DFP recognizes as being Valid and functionally compatible with the highest speed that the DFP supports.</p>

6.4.12.3.9. Active Cable VDOs

An Active Cable has a USB Plug on each end at least one of which is a Cable Plug supporting SOP' Communication. An Active Cable **Shall** incorporate data bus signal conditioning circuits and **May** have a concept of Super Speed Directionality on its Super Speed wires. An Active Cable **May** include a VBUS wire.

An Active Cable:

- **Shall** respond to SOP' Communication.
- **May** respond to SOP" Communication.
- **Shall** support the [Structured VDM Discover Identity](#) Command.
- In the [Discover Identity](#) Command ACK:
 - **Shall** set the Product Type in the ID Header VDO to Active Cable.

- **Shall** return the Active Cable VDOs.

Table 6.43. Active Cable VDO1

Bit(s)	Field Name	Static	Description
31...28	Hardware Version	Yes	0000b...1111b - HW Version assigned by the VID owner.
27...24	Firmware Version	Yes	0000b...1111b - FW Version assigned by the VID owner.
23...21	VDO Version	Yes	Version Number of the VDO (not this specification Version). This field indicates the expected content for this VDO. 000b - Deprecated , Version 1.0 001b - Invalid, Shall Not be used. 010b - Deprecated , Version 1.2 011b - Version 1.3 100b...111b - Invalid, Shall Not be used.
20	Reserved	Yes	Reserved , receiver Shall ignore this field.
19...18	USB Type-C plug to USB Type-C/Captive (Active Cable)	Yes	Indicates whether the opposite end from the USB Type-C plug is another USB Type-C plug (i.e., a detachable Standard USB Type-C Cable Assembly) or is a Captive Cable Assembly. 00b - Deprecated , USB Type-A 01b - Deprecated , USB Type-B 10b - USB Type-C 11b - Captive
17	EPR Capable (Active Cable)	Yes	Indicates when the cable is specifically designed for safe operation when carrying up to 48 volts at 5 amps. 0b - Cable is not EPR Capable 1b - Cable is EPR Capable
16...13	Cable Latency	Yes	Signal latency through the cable which can be used as an approximation for its length. 0000b - Invalid, Shall not be used. 0001b - <10ns (~1m) 0010b - 10ns to 20ns (~2m) 0011b - 20ns to 30ns (~3m) 0100b - 30ns to 40ns (~4m) 0101b - 40ns to 50ns (~5m) 0110b - 50ns to 60ns (~6m) 0111b - 60ns to 70ns (~7m) 1000b - 1000ns (~100m) 1001b - 2000ns (~200m) 1010b - 3000ns (~300m) 1011b1111b - Invalid, Shall Not be used. Note: Includes latency of electronics in Active Cable.

Bit(s)	Field Name	Static	Description
12...11	Cable Termination Type (Active Cable)	Yes	<p>Shall contain a value corresponding to whether the Active Cable has one or two Cable Plugs requiring power from VCONN.</p> <p>00b...01b - Invalid, Shall Not be used.</p> <p>10b - One end Active, one end passive, VCONN required.</p> <p>11b - Both ends Active, VCONN required.</p>
10...9	Maximum VBUS Voltage (Active Cable)	Yes	<p>Maximum voltage that Shall be Negotiated as part of an Explicit Contract where the maximum voltage that Shall be applied to the cable is vSrcNew max + vSrcValid max. When this field is set to 20V, the cable will safely carry a Programmable Power Supply APDO of 20V where the absolute maximum voltage that can be applied to the cable is 21.55V. Similarly, when the Maximum VBUS Voltage (Active Cable) field is 50V, a Fixed Supply of 48V can be Negotiated as part of an Explicit Contract where the absolute maximum voltage that can be applied to the cable is 50.9V. EPR Sinks with a captive cable Shall report 50V.</p> <p>00b - 20V</p> <p>01b..10b - Deprecated, receiver Shall assume 00b (20V).</p> <p>11b - 50V</p>
8	SBU Supported	Yes	<p>Indicates whether the cable supports the SBUs in the cable.</p> <p>0b - SBU connections supported.</p> <p>1b - SBU connections are not supported.</p>
7	SBU Type	Yes	<p>Indicates whether the SBUs are passive or active (e.g., digital). When SBU Supported = 1 (SBU connections are not supported) this bit Shall be Ignored.</p> <p>When SBU Supported = 0 (SBU connections are supported):</p> <p>0b - SBU is passive</p> <p>1b - SBU is active</p>
6...5	VBUS Current Handling Capability (Active Cable)	Yes	<p>Indicates whether the cable is capable of carrying 3A or 5A. Shall be ignored when VBUS Through Cable = 0</p> <p>00b - Invalid, Shall Not be used</p> <p>01b - 3A</p> <p>10b - 5A</p> <p>11b - Invalid, Shall Not be used.</p>
4	VBUS Through Cable	Yes	<p>Indicates whether the cable contains an end-to-end VBUS wire.</p> <p>0b - No</p> <p>1b - Yes</p>
3	SOP'' Controller Present	Yes	<p>Indicates whether one of the Cable Plugs is capable of SOP'' Communication in addition to the Required SOP' Communication.</p> <p>0b - No SOP'' controller present</p> <p>1b - SOP'' controller present</p>

Bit(s)	Field Name	Static	Description
2...0	USB Highest Speed (Active Cable)	Yes	<p>Highest data rate the cable supports.</p> <p>000b - [USB2] only, no SuperSpeed support</p> <p>001b - [USB3] Gen1</p> <p>010b - [USB3]/[USB4] Gen2</p> <p>011b - [USB4] Gen3</p> <p>100b - [USB4] Gen4</p> <p>101b...111b - Invalid, and Shall Not be used</p> <p>Note: The DFP Shall consider all values, including Invalid, indicated in this field that are higher than the highest value that the DFP recognizes as being Valid and functionally compatible with the highest speed that the DFP supports.</p>

Table 6.44. Active Cable VDO2

Bit(s)	Field Name	Static	Description
31...24	Maximum Operating Temperature	Yes	Maximum allowable operating temperature inside the plug in °C. It may reflect the plug's skin temperature
23...16	Shutdown Temperature	Yes	Temperature inside the plug, in °C, at which the plug will shut down its active Signaling components. When this temperature is reached, it will be reported in the Active Cable Status Message through the Thermal Shutdown bit.
15	Reserved	Yes	Reserved , receiver Shall ignore this field.
14...12	U3/CLd Power	Yes	<p>Power the cable consumes while in [USB3] U3 or [USB4] CLd.</p> <p>000b - >10mW</p> <p>001b - 5-10mW</p> <p>010b - 1-5mW</p> <p>011b - 0.5-1mW</p> <p>100b - 0.2-0.5mW</p> <p>101b - 50-200μW</p> <p>110b - <50μW</p> <p>111b - Invalid; receiver Should assume 000b (>10mW)</p>
11	U3 to U0 transition Mode	Yes	<p>Indicates which U3 to U0 Mode the cable supports. This does not include the power in U3S if supported.</p> <p>0b - U3 to U0 direct</p> <p>1b - U3 to U0 through U3S</p>
10	Physical Connection	Yes	<p>Cable's construction, whether the connection between the active elements is copper or optical.</p> <p>0b - Copper</p> <p>1b - Optical</p>
9	Active Element	Yes	<p>Cable's active element, whether the active element is a re-timer or a re-driver.</p> <p>0b - Active Re-driver</p> <p>1b - Active Re-timer</p>

Bit(s)	Field Name	Static	Description
8	USB4 Supported	Yes	Indicates whether or not the cable supports [USB4] operation. 0b - [USB4] supported 1b - [USB4] not supported.
7...6	USB 2.0 Hub Hops Consumed	Yes	Number of USB 2.0 'Hub hops' that are lost due to the transmission time of the cable. Shall be set to zero if USB 2.0 not supported.
5	USB 2.0 Supported	Yes	Indicates whether or not the cable supports [USB2] Signaling. 0b - [USB2] supported 1b - [USB2] not supported.
4	USB 3.2 Supported	Yes	Indicates whether or not the cable supports [USB3] SuperSpeed Signaling. 0b - [USB3] SuperSpeed supported. 1b - [USB3] SuperSpeed not supported.
3	USB Lanes Supported	Yes	Indicates whether the cable supports one or two lanes of [USB3] SuperSpeed Signaling. 0b - One lane 1b - Two lanes
2	Optically Isolated Active Cable	Yes	Indicates whether this cable is an optically isolated Active Cable or not (as defined in [USB-C]). Optically Isolated Active Cables Shall have a re-timer or linear re-driver (LRD) as the active element and do not support [USB2] or carry VBUS. 0b - No 1b - Yes
1	[USB4] Asymmetric Mode Supported	Yes	Active Cable supports asymmetric Mode as defined in [USB4] and [USB-C]. 0b - No 1b - Yes
0	USB Gen	Yes	Signaling Gen the cable supports. Gen 1 Shall only be used by [USB3] cables as indicated by the USB 3.2 Supported field. Gen 2 or higher May be used by either [USB3] or [USB4] cables as indicated by their respective supported fields. When Gen 2 or higher is indicated the USB Highest Speed (Active Cable) field in VDO1 Shall indicate the actual Gen supported. 0b - Gen 1 1b - Gen 2 or higher Note: See VDO1 USB Highest Speed for details of Gen supported.

6.4.12.3.10. Alternate Mode Adapter

The Alternate Mode Adapter (AMA) VDO has been **Deprecated**. PDUSB Devices which support one or more Alternate Modes **Shall** set an appropriate Product Type (UFP), and **Shall** set the Modal Operation Supported bit to '1'.

6.4.12.3.11. VCONN Powered USB Device (VDO)

The VCONN Powered USB Device (VPD) VDO defined in this section **Shall** be sent when the Product Type is given as VCONN Powered USB Device.

Table 6.45. VCONN Powered USB Device VDO

Bit(s)	Field Name	Static	Description
31...28	HW Version	Yes	0000b...1111b - HW Version assigned by the VID owner.
27...24	Firmware Version	Yes	0000b...1111b - FW Version assigned by the VID owner.
23...21	VDO Version	Yes	Version Number of the VDO (not this specification Version). This field indicates the expected content for this VDO. 000b - Version 1.0 001b...111b - Invalid, Shall Not be used
20...17	Reserved	Yes	Reserved , receiver Shall ignore this field.
16...15	Maximum VBUS Voltage	Yes	Maximum voltage that a Sink Shall Negotiate through the VPD Charge Through Port as part of an Explicit Contract. 00b - 20V 01b..11b - Deprecated , receiver Shall assume 00b (20V).
14	Charge Through Current Support	Yes	Supported charge through current. When Charge Through Support = 0b, this is Reserved and Shall be set to 0. When Charge Through Support = 1b: 0b - 3A capable. 1b - 5A capable
13	Reserved	Yes	Reserved , receiver Shall ignore this field.
12...7	VBUS Impedance	Yes	Impedance the VPD adds in series between the Source and the Sink. The Sink Shall take this value into account when requesting current so as to not to exceed the VBUS IR Drop limit of 0.5V between the Source and itself. If the Sink can tolerate a larger IR Drop on VBUS it May do so. When Charge Through Support = 0b, this is Reserved and Shall not be used. When Charge Through Support = 1b: VBUS impedance through the VPD in 2 mΩ increments. Values less than 10 mΩ are Reserved and Shall Not be used.
6...1	Ground Impedance	Yes	Impedance the VPD adds in series between the Source and the Sink. The Sink Shall take this value into account when requesting current so as to not to exceed the Ground IR Drop limit of 0.25V between the Source and itself. When Charge Through Support = 0b, this is Reserved and Shall not be used. When Charge Through Current Support = 1b: Ground impedance through the VPD in 1 mΩ increments. Values less than 10 mΩ are Reserved and Shall Not be used.
0	Charge Through Support	Yes	Indicates whether the VPD supports Charge Through. 0b - the VPD does not support Charge Through. 1b - the VPD supports Charge Through.

6.4.12.4. Discover SVIDs

The [Discover SVIDs](#) Command is used by an Initiator to determine the SVIDs for which a Responder has Modes. The [Discover SVIDs](#) Command is used in conjunction with the [Discover Modes](#) Command in the Discovery Process to determine which Modes a Device supports. The list of SVIDs is always terminated with one or two 0x0000 SVIDs.

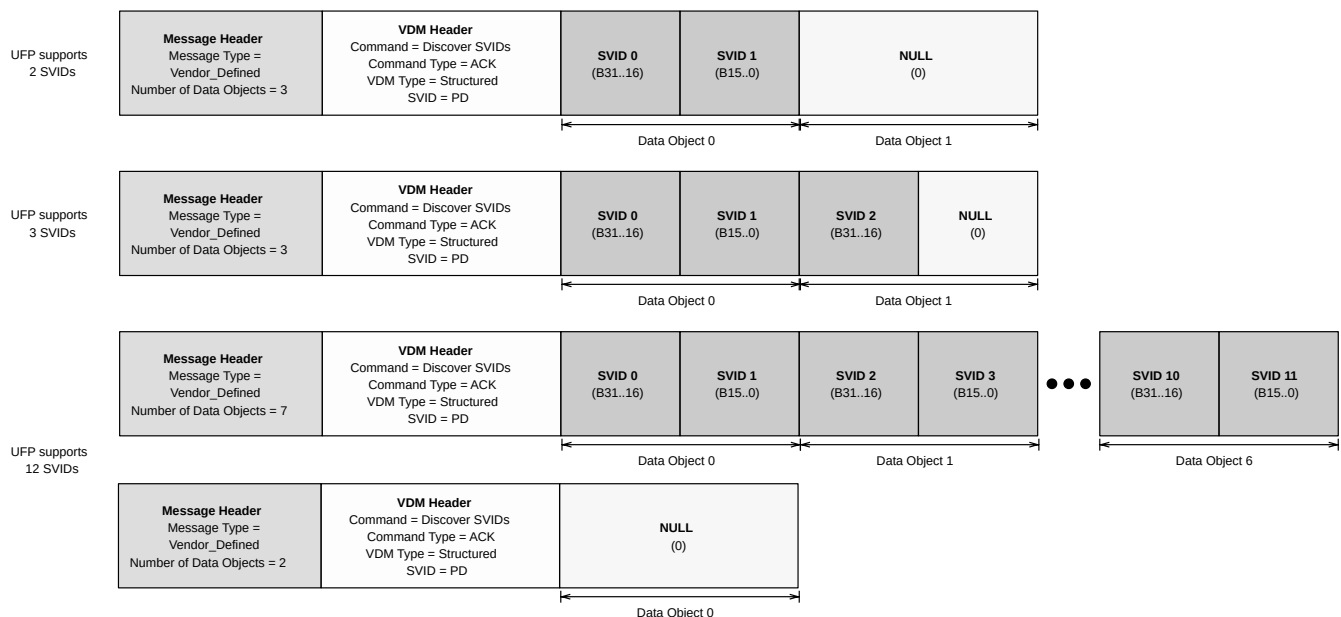
The SVID in the [Discover SVIDs](#) Command **Shall** be set to the PD SID (see [Table 6.33](#)) by both the Initiator and the Responder for this Command.

The Number of Data Objects field in the Message Header in the [Discover SVIDs](#) Command Request **Shall** be set to 1 since the [Discover SVIDs](#) Command Request **Shall Not** contain any VDOs.

The [Discover SVIDs](#) Command ACK sent back by the Responder **Shall** contain one or more SVIDs. The SVIDs are returned 2 per VDO (see [Table 6.46](#) "). If there are an odd number of supported SVIDs, the [Discover SVIDs](#) Command is returned ending with a SVID value of 0x0000 in the last part of the last VDO. If there are an even number of supported SVIDs, the [Discover SVIDs](#) Command is returned ending with an additional VDO containing two SVIDs with values of 0x0000. A Responder **Shall** only return SVIDs for which a [Discover Modes](#) Command Request for that SVID will return at least one Alternate Mode.

The Number of Data Objects field in the Message Header in the [Discover SVIDs](#) Command NAK and BUSY responses **Shall** be set to 1 since they **Shall Not** contain any VDOs.

Figure 6.6. Discover SVIDs Response Format



Usage: [Section 8.6.2](#)

6.4.12.4.1. Responder VDO

A Device that supports SVIDs **Shall** return them in the [Discover SVIDs](#) Command ACK. The SVIDs are returned 2 per VDO (see [Table 6.46](#) "). If there are an odd number of supported SVIDs, the [Discover SVIDs](#) Command is returned ending with a SVID value of 0x0000 in the last part of the last VDO.

Table 6.46. Discover SVIDs Responder VDO

Bit(s)	Field	Description
B31...16	SVID n	16-bit unsigned integer, assigned by the USB-IF or 0x0000 if this is the last VDO and the Responder supports an even number of SVIDs.
B15...0	SVID n+1	16-bit unsigned integer, assigned by the USB-IF or 0x0000 if this is the last VDO and the Responder supports an odd or even number of SVIDs.

6.4.12.5. Discover Mode

The [Discover Modes](#) Command is used by an Initiator to determine the Modes a Responder supports for a given SVID.

The SVID in the [Discover Modes](#) Command **Shall** be set to the SVID for which Modes are being requested by both the Initiator and the Responder for this Command.

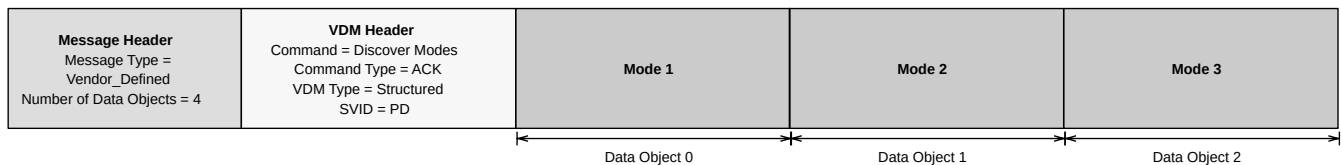
The Number of Data Objects field in the Message Header in the [Discover Modes](#) Command Request **Shall** be set to 1 since the [Discover Modes](#) Command Request **Shall Not** contain any VDOs.

The [Discover Modes](#) CommandACK sent back by the Responder **Shall** contain one or more Modes. The [Discover Modes](#) Command ACK **Shall** contain a Message Header with the Number of Data Objects field set to a value of 2 to 7 (the actual value is the number of Alternate Mode objects plus one).

The Number of Data Objects field in the Message Header in the [Discover Modes](#) Command NAK and BUSY responses **Shall** be set to 1 since they **Shall Not** contain any VDOs.

[Figure 6.7](#) shows an example of a [Discover Modes](#) Command response from a Responder which supports three Modes for a given SVID.

Figure 6.7. Example Discover Modes response for a given SVID with 3 Modes



Usage: [Section 8.6.3](#)

6.4.12.6. Enter Mode

The [Enter Mode](#) Command is used by an Initiator (DFP) to Command a Responder (UFP or Cable Plug) to enter a specified Alternate Mode of operation.

The value in the Object Position field in the VDM Header **Shall** indicate to which Alternate Mode in the [Discover Modes](#) Command the VDO refers. The value 1 always indicates the first Alternate Mode as it is the first object following the VDM Header. The value 2 refers to the next Alternate Mode and so forth.

The Number of Data Objects field in the Message Header in the Command Request **Shall** be set to either 1 or 2 since the [Enter Mode](#) Command Request **Shall Not** contain more than 1 VDO. When a VDO is included in an [Enter Mode](#) Command Request the contents of the 32-bit VDO is defined by the Alternate Mode.

The Number of Data Objects field in the Command response **Shall** be set to 1 since an [Enter Mode](#) Command response (ACK, NAK) **Shall Not** contain any VDOs.

Usage: [Section 8.6.4](#)

6.4.12.7. Exit Mode

The [Exit Mode](#) Command is used by an Initiator (DFP) to Command a Responder (UFP or Cable Plug) to exit its Active Mode and return to normal USB operation. Only the DFP **Shall** initiate the [Exit Mode](#) Process.

The value in the Object Position field **Shall** indicate to which Alternate Mode in the [Discover Modes](#) Command the VDO refers and **Shall** have been used previously in an [Enter Mode](#) Command Request for an Active Mode. The value 1 always indicates the first Alternate Mode as it is the first object following the VDM Header. The value 2

refers to the next Alternate Mode and so forth. A value of 111b in the Object Position field **Shall** indicate that all Active Modes **Shall** be exited.

The Number of Data Objects field in both the Command Request and Command response (ACK, NAK) **Shall** be set to 1 since an [Exit Mode](#) Command **Shall Not** contain any VDOs.

Usage: [Section 8.6.5](#)

6.4.12.8. Attention

The [Attention](#) Command **May** be used by the Initiator to notify the Responder that it requires service.

The Number of Data Objects field in the Message Header **Shall** be set to 1 or 2 since the [Attention](#) Command **Shall Not** contain more than 1 VDO. When a VDO is included in an [Attention](#) Command the contents of the 32-bit VDO is defined by the Alternate Mode.

The value in the Object Position field **Shall** indicate to which Alternate Mode in the [Discover Modes](#) Command the VDO refers (see [Figure 6.7](#)) and **Shall** have been used previously in an [Enter Mode](#) Command Request for an Active Mode. The value 1 always indicates the first Alternate Mode as it is the first object following the VDM Header. The value 2 refers to the next Alternate Mode and so forth. A value of 000b or 111b in the Object Position field **Shall Not** be used by the [Attention](#) Command.

Usage: [Section 8.6.6](#)

6.5. Extended Message

An [Extended Message](#) **Shall** contain a Message Header, an [Extended Message](#) Header (indicated by the Extended field in the Message Header being set), and be followed by zero or more data bytes. Additional bytes that might be added to existing Messages in future Revision of this specification **Shall** be **Ignored**.

Table 6.47. Extended Message Types

Value	Message Type
00000b	Reserved , receiver Shall respond with Not_Supported Message
00001b	Source_Capabilities_Extended
00010b	Status Message
00011b	Get_Battery_Cap
00100b	Get_Battery_Status
00101b	Battery_Capabilities
00110b	Get_Manufacturer_Info
00111b	Manufacturer_Info
01000b	Security_Request
01001b	Security_Response
01010b	Firmware_Update_Request
01011b	Firmware_Update_Response
01100b	PPS_Status
01101b	Country_Info
01110b	Country_Codes
01111b	Sink_Capabilities_Extended
10000b	Extended_Control
10001b	EPR_Source_Capabilities
10010b	EPR_Sink_Capabilities

Value	Message Type
10011b..11101b	Reserved , receiver Shall respond with Not_Supported_Message .
11110b	Vendor_Defined_Extended
11111b	Reserved , receiver Shall respond with Not_Supported_Message .

6.5.1. Extended Message Header

An [Extended Messages](#) (indicated by the Extended field being set in the Message Header) **Shall** contain an [Extended Message](#) Header following the Message Header as shown in [Figure 6.2](#) and defined in [Table 6.48](#).

Table 6.48. Extended Message Header Fields

Bit(s)	Field Name	Applicable SOP	Description
15	Chunked	All	See Section 6.5.1.1 .
14...11	Chunk Number	All	<p>When Chunked is 0b:</p> <p>Shall be 0</p> <p>When Chunked is 1b and Request Chunk is 1b:</p> <p>0..9 - Chunk Number being requested.</p> <p>10..15 - Invalid, receiver Shall ignore this Message.</p> <p>When Chunked is 1b and Request Chunk is 0b:</p> <p>0..9 - Number of Chunk being transmitted.</p> <p>10..15 - Invalid, receiver Shall ignore this Message.</p>
10	Request Chunk	All	<p>When Chunked is 0b: Shall be 0</p> <p>When Chunked is 1b:</p> <p>0 - Message contains a data Chunk</p> <p>1 - Message is requesting a data Chunk</p> <p>Except for Chunk 0, a requested Chunk of a Data Block Shall only be returned as a Chunk response to a corresponding Request for that Chunk. Both the Chunk Request and the Chunk response Shall contain the same value in the Message Type field.</p>
9	Reserved	All	Reserved , receiver Shall ignore this field.
8...0	Data Size	All	<p>Indicates total number of bytes of data in the Data Block being returned. The total number of data bytes in the Message Shall Not exceed MaxExtendedMsgLen .</p> <p>If less than MaxExtendedMsgLegacyLen , and the Chunked bit is set, then the Packet Payload Shall be padded to the next 4-byte Data Object boundary with 00h.</p> <p>If greater than the expected size for a given Extended Message Type but less than or equal to MaxExtendedMsgLen , then the expected fields in the Message Shall be processed appropriately, and the additional fields Shall be Ignored.</p> <p>If Request Chunk is 1, Shall be 0.</p>

6.5.1.1. Chunked

Determination to use Unchunked [Extended Messages](#) is accomplished by evaluating the [Source_Capabilities_Message](#) and [Request Messages](#) as part of the initial power Negotiation after an Attach or [Hard Reset](#).

Table 6.49. Chunked

Source Capabilities 5V PDO: Unchunked Extended Messages Supported	Request RDO: Unchunked Extended Messages Supported	Extended Message Format
0	0	Chunked Only
0	1	Chunked Only
1	0	Chunked Only
1	1	Unchunked Only

When either Port Partner only supports Chunked [Extended Messages](#):

- The Chunked bit in every [Extended Message](#) **Shall** be set to one.
- Every [Extended Message](#) of Data Size > [MaxExtendedMsgLegacyLen](#) **Shall** be transmitted between the Port Partners in Chunks
- Each Chunk in the series, except for the last Chunk, **Shall** contain [MaxExtendedMsgChunkLen](#) bytes. The last Chunk in the series **Shall** contain the remainder of the Data Block and so could be less than [MaxExtendedMsgChunkLen](#) bytes and **Shall** be padded to the next 4-byte Data Object boundary.
- The Number of Data Objects in the Message Header **Shall** indicate the number of Data Objects in the Message padded to the 4-byte boundary including the [Extended Message](#) Header as part of the first Data Object.

When [Extended Message](#) Format is Unchunked:

- The Chunked bit in every [Extended Message](#) **Shall** be set to 0b.
- Every [Extended Message](#) **Shall** be transmitted between the Port Partners Unchunked.
- The Number of Data Objects in the Message Header is **Reserved**.

The condition listed above **Shall** apply until the Port Pair is Detached or there is a [Hard Reset](#).

A Port that supports [Extended Messages](#), (i.e., all Sink devices since Sink_cap_ext is mandatory), **Shall** support Chunking. When sending Extended Messages to the Cable Plug the VCONN Source **Shall** only send Chunked Extended Messages. Cable Plugs **Shall** always send [Extended Messages](#) of Data Size > [MaxExtendedMsgLegacyLen](#) Chunked and **Shall** set the Chunked bit in every [Extended Message](#) to one.

6.5.2. Source_Capabilities_Extended Message

The [Source_Capabilities_Extended Message](#) enables a Source or a DRP to inform the Sink about its Capabilities as a Source.

This Message is used in the following sequences:

- Get Source Capabilities Extended

Usage: [Section 7.19.2](#)

Table 6.50. Source Capabilities Extended Data Block (SCEDB)

Byte(s)	Field Name	Static	Description
1...0	VID	Yes	Vendor ID (See Section 6.1.5).
3...2	PID	Yes	Product ID (See Section 6.1.5).
7...4	XID	Yes	Value provided by the USB-IF and assigned to the product by the vendor. Shall match XID field in the CertStat VDO.
8	FW Version	Yes	Firmware Version number assigned by the vendor.
9	HW Version	Yes	Hardware Version number assigned by the vendor.

Byte(s)	Field Name	Static	Description
10	Voltage Regulation	Yes	Bits 1..0: 00b - 150mA/μs Load Step (default) 01b - 500mA/μs Load Step 10b..11b - Invalid , receiver Shall assume default Bit 2: 0b - 25% IoC (default) 1b - 90% IoC Bits 7..3: Reserved , receiver Shall ignore these bits.
11	Holdup Time	Yes	0 - feature not supported 1..255 - Output will stay with regulated limits for this number of milliseconds after removal of the AC from the input. Note: A value of at least 3ms Should be used (see Section 4.1.7.3).
12	Compliance	Yes	Bits 2..0: xx1b - Indicates LPS compliance x1xb - Indicates PS1 compliance 1xxb - Indicates PS2 compliance Bits 7..3: Reserved , receiver Shall ignore these bits
13	Touch Current	Yes	Bits 2..0: xx1b - Indicates low touch current External Power Supply (EPS) x1xb - Indicates ground pin supported 1xxb - Indicates ground pin intended for protective earth Bits 7..3: Reserved , receiver Shall ignore these bits.
15...14	Peak Current 1	Yes	Bits 4..0: Percent Overload 0..25 - Percent overload in 10% increments 26..31 - Invalid , receiver Shall treat as 25 Bits 10..5: Overload Period 0..63 - Period in 20ms increments Bits 14..11: Duty Cycle 0..31 - Duty cycle in 5% increments Bit 15: VBUS Droop 0b - VBUS is provided in the range of vSrcNew when overload conditions occur 1b - VBUS may droop an additional 5% during overload conditions
17...16	Peak Current 2	Yes	See Peak Current 1
19...18	Peak Current 3	Yes	See Peak Current 1

Byte(s)	Field Name	Static	Description
20	Touch Temp	Yes	<p>Indicates the IEC standard used to determine the surface temperature of the Source's enclosure. Safety limits for the Source's touch temperature are set in applicable product safety standards (e.g., [IEC 60950-1] or [IEC 62368-1]). The Source May report when its touch temperature performance conforms to the TS1 or TS2 limits described in [IEC 62368-1].</p> <p>0 - [IEC 60950-1] (default)</p> <p>1 - [IEC 62368-1] TS1</p> <p>2 - [IEC 62368-1] TS2</p> <p>All other values Invalid, receiver Shall assume default.</p>
21	Source Inputs	No	<p>Bit 0:</p> <p>0b - No External Supply present</p> <p>1b - External Supply present</p> <p>Bit 1:</p> <p>0b - External Supply is constrained.</p> <p>1b - External Supply is unconstrained</p> <p>Bit 1 Shall be Reserved, receiver Shall ignore this bit if Bit 0 is set to 0b.</p> <p>Bit 2:</p> <p>0b - No internal Battery present</p> <p>1b - Internal Battery present</p> <p>Bits 7..3:</p> <p>Reserved, receiver Shall ignore these bits.</p>
22	Number of Batteries/Battery Slots	Yes	<p>Bits 3..0:</p> <p>0..4 - Count of Fixed Batteries</p> <p>5..15 - Invalid, receiver Shall assume 0 (no fixed batteries).</p> <p>Bits 7..4:</p> <p>0..4 - Count of Hot Swappable Battery Slots</p> <p>5..15 - Invalid, receiver Shall assume 0 (no hot swappable batteries).</p> <p>This Field Shall match that reported in the Battery Info field of the Sink Capabilities Extended Message. The number assigned to a given Battery Slot Shall Not change between Attach and Detach.</p>
23	SPR Source PDP Rating	Yes	<p>0..100 - Integer portion of the PDP Rating of the Port operating in SPR Mode, in watts</p> <p>101..255 - Invalid, receiver Shall ignore this field.</p>
24	EPR Source PDP Rating	Yes	<p>0..240 - Integer portion of the PDP Rating of the Port operating in EPR Mode, in watts</p> <p>241..255 - Invalid, receiver Shall ignore this field.</p> <p>An SPR Source Shall set this field to 0.</p>

6.5.2.1. Touch Current Field

The Touch Current field reports whether the Source meets certain leakage current levels and if it has a ground pin.

A Source **Shall** set the Touch Current bit (bit 0) when their leakage current is less than 65µA rms when Source's maximum capability is less than or equal to 30W, or when their leakage current is less than 100 µA rms when its power capability is between 30W and 100W. The total combined leakage current **Shall** be measured in accordance with [IEC 60950-1] when tested at 250V AC rms at 50 Hz.

6.5.2.2. Peak Current Fields

The Peak Current1/Peak Current2/Peak Current3 fields **Shall** contain the combinations of Peak Current that the Source supports (see [Section 4.1.6](#)).

Peak Current provides a means for Source report its ability to provide current in excess of the Negotiated amount for short periods. The Peak Current descriptor defines up to three combinations of % overload, duration and duty cycle defined as Peak Current1, Peak Current2 and Peak Current3 that the Source supports. A Source **May** offer no Peak Current capability. A Source **Shall** populate unused Peak Current bit fields with zero.

The Duty Cycle Bit fields **Should** be 5%, 10% and 50% for PeakCurrent1, PeakCurrent2, and PeakCurrent3, respectively.

The Source **Should** provide VBUS in the range of [vSrcNew](#) when overload conditions occur and set the VBUS Droop bit to zero.

6.5.3. Status Message

When sent to SOP the [Status Message](#) returns the status of the Port's Port Partner.

All fields contained in this Message are informational.

This Message is used in the following sequences:

- [Get Status](#)

Usage: [Section 7.15](#)

Table 6.51. SOP Status Data Block (SDB)

Byte(s)	Field Name	Static	Description
0	Internal Temp	No	Source or Sink's internal temperature: 0 - Feature not supported 1 - Temperature is less than 2°C 2..255 - Temperature in °C

Byte(s)	Field Name	Static	Description
1	Present Input	No	<p>Indicates which supplies are presently powering the Source or Sink.</p> <p>Bit 0:</p> <p>Reserved, receiver Shall ignore this bit</p> <p>Bits 2..1:</p> <p>00b - Internally Powered</p> <p>01b - Indicates DC External Power Source is present</p> <p>10b - Invalid, receiver Shall ignore these bits</p> <p>11b - Indicates AC External Power Source is present</p> <p>Bit 3:</p> <p>1b - Internal Power provided by Battery</p> <p>Bit 4:</p> <p>1b - Internal Power provided by a non-Battery power Source</p> <p>Bits 7..5:</p> <p>Reserved, receiver Shall ignore these bits</p>
2	Present Battery Input	No	<p>When Present Input Bit 3 is 1b:</p> <p>Indicates which Battery or Batteries are providing power.</p> <p>1xxxxxxb - HotSwappable Battery 3 is providing power</p> <p>x1xxxxxb - HotSwappable Battery 2 is providing power</p> <p>xx1xxxxb - HotSwappable Battery 1 is providing power</p> <p>xxx1xxxxb - HotSwappable Battery 0 is providing power</p> <p>xxxx1xxb - Fixed Battery 3 is providing power</p> <p>xxxxx1xb - Fixed Battery 2 is providing power</p> <p>xxxxxx1b - Fixed Battery 1 is providing power</p> <p>xxxxxxx1b - Fixed Battery 0 is providing power</p> <p>When Present Input Bit 3 is 0b:</p> <p>Reserved, receiver Shall ignore this field.</p>

Byte(s)	Field Name	Static	Description
3	Event Flag	No	<p>Bit 0:</p> <p>Reserved, receiver Shall ignore this bit</p> <p>Bit 1:</p> <p>1b - Overcurrent Event (OCP)</p> <p>Shall be cleared after sending Status Message</p> <p>Bit 2:</p> <p>1b - Overtemperature Event (OTP) Shall be cleared after sending Status Message</p> <p>When set to 1b, Bits 2..1 in Temperature Status Shall be set to 11b.</p> <p>Bit 3:</p> <p>1b - Overvoltage Event (OVP)</p> <p>Shall be cleared after sending Status Message</p> <p>Bit 4: (PPS Mode only)</p> <p>0b - Constant Voltage (CV) Mode</p> <p>1b - Current Limit (CL) Mode</p> <p>Shall be ignored when not operating in PPS Mode.</p> <p>Bits 7..5:</p> <p>Reserved, receiver Shall ignore these bits.</p>
4	Temperature Status	No	<p>Bit 0:</p> <p>Reserved, receiver Shall ignore this bit</p> <p>Bit 2..1:</p> <p>00b - Not Supported</p> <p>01b - Normal</p> <p>10b - Warning Note: DPS Source that is sending an Alert Message to reduce power due to Temperature Shall set this field to Warning.</p> <p>11b - Over-temperature. When set, Event Flags Bit 2 Shall also be set</p> <p>Bits 7..3:</p> <p>Reserved, receiver Shall ignore these bits.</p>

Byte(s)	Field Name	Static	Description
5	Power Status	No	<p>Bit 0:</p> <p>Reserved, receiver Shall ignore this bit</p> <p>Bit 1:</p> <p>1b - Source power limited due to cable supported current</p> <p>Bit 2:</p> <p>1b - Source power limited due to insufficient power available while sourcing other ports</p> <p>Bit 3:</p> <p>1b - Source power limited due to insufficient external power</p> <p>Bit 4:</p> <p>1b - Source power limited due to Event Flags in place (Event Flags must also be set)</p> <p>Bit 5:</p> <p>1b - Source power limited due to temperature. Note: A DPS Source that is sending an Alert Message to reduce power due to Temperature Shall set this bit.</p> <p>Bits 7..6:</p> <p>Reserved, receiver Shall ignore these bits</p> <p>Sinks Shall set this field to 0</p>

Byte(s)	Field Name	Static	Description
6	Power State Change	No	<p>Bits 2..0:</p> <p>Indicates a power State change to one of the specified power states. Any Device that supports the ACPI standard system power states Shall use the ACPI states. For devices that do not support the ACPI power states, the following mapping Should be used:</p> <p>High power (on) State -> S0</p> <p>Sleep State -> S3</p> <p>Low power (off) State -> S5 or G3</p> <p>0 - Status Not Supported</p> <p>1 - S0</p> <p>2 - Modern Standby</p> <p>3 - S3</p> <p>4 - S4</p> <p>5 - S5 (Off with Battery, wake events supported)</p> <p>6 - G3 (Off with no Battery, wake events not supported)</p> <p>7 - Invalid, receiver Shall assume 0 (Not Supported)</p> <p>Bits 5..3:</p> <p>Defines the Host's desired indicator for the specified power State. This indicator allows several possibilities for predefined behaviors that the Host can specify to indicate its system power State to the user via the downstream Device. The New Power State indicator is a "best effort" indicator. If the Device cannot provide the requested indicator, then it provides the best indicator that it can. If a Breathing indicator cannot be provided, then a Blinking indicator Should be provided. If a Blinking indicator cannot be provided, then a constant on indicator Should be provided.</p> <p>0 - Off LED</p> <p>1 - On LED</p> <p>2 - Blinking LED</p> <p>3 - Breathing LED</p> <p>4..7 - Invalid, receiver Shall ignore these bits</p>

When sent to SOP' or SOP" the [Status Message](#) returns the status of the Cable's corresponding Cable Plugs.

Table 6.52. SOP'/SOP'' Status Message Byte Definitions

Byte(s)	Field Name	Static	Description
0	Internal Temp	No	Cable Plugs's internal temperature 0 - Feature not supported 1 - Temperature is less than 2°C 2..255 - Temperature in °C
1	Flags	No	Indicates which supplies are presently powering the Source or Sink. Bit 0: 1b - Indicated Thermal Shutdown Shall also be set when the plug's internal temperature exceeds the Internal Maximum Temperature reported in the Active Cable VDO. Once this bit has been set, it Shall remain set and the plug Shall remain in Thermal Shutdown until there is a Hard Reset or the Active Cable's power is removed. The Thermal Shutdown flag Shall Not be cleared by a Cable Reset . Passive Cable Plugs Shall Not indicate Thermal Shutdown. Bits 7..1: Reserved , receiver Shall ignore these bits.

6.5.4. Get_Battery_Cap Message

The [Get_Battery_Cap Message](#) is used to Request the capability of a Battery present in its Port Partner.

This Message is used in the following sequences:

- Get [Battery Capabilities](#)

Usage: [Section 7.20](#)

Table 6.53. Get Battery Cap Data Block (GBCDB)

Byte(s)	Field Name	Static	Description
0	Battery Cap Ref	No	Number of the Battery indexed from 0. Valid indices are determined from the "Number of Batteries/Battery Slots" field in Section 6.5.2 or the "Battery Info" field in Section 6.5.16 0..3 - Fixed Battery 0-3 4..7 - Hot Swappable Battery 0-3 8..255 - Invalid , receiver Shall send Battery_Capabilities Message with Battery Type field bit 0 set to 1b.

6.5.5. Get_Battery_Status Message

The [Get_Battery_Status](#) (Get [Battery Status](#)) Message is used to Request the status of a Battery present in its Port Partner.

This Message is used in the following sequences:

- Get [Battery Status](#)

Usage: [Section 7.21](#)

Table 6.54. Get Battery Status Data Block (GBSDB)

Byte(s)	Field Name	Static	Description
0	Battery Status Ref	No	<p>Number of the Battery indexed from 0. Valid indices are determined from the "Number of Batteries/Battery Slots" field in Section 6.5.2 or the "Battery Info" field in Section 6.5.16</p> <p>0..3 - Fixed Battery.</p> <p>4..7 - Hot Swappable Battery.</p> <p>8..255 - Invalid, receiver Shall send Battery_Status Message with In-valid Battery Reference field set to 1b.</p>

6.5.6. Battery_Capabilities Message

The [Battery_Capabilities Message](#) is sent in response to a [Get_Battery_Cap Message](#). The [Battery_Capabilities Message](#) contains one Battery Capability Data Block (BCDB) for one of the Batteries it supports as reported by Number of Batteries/Battery Slots field in the [Source_Capabilities_Extended Message](#). The returned BCDB **Shall** correspond to the Battery requested in the Battery Cap Ref field contained in the [Get_Battery_Cap Message](#).

This Message is used in the following sequences:

- Get [Battery Capabilities](#)

Usage: [Section 7.20](#)

Table 6.55. Battery Capability Data Block (BCDB)

Byte(s)	Field Name	Static	Description
1...0	VID	No	Vendor ID associated with the Battery manufacturer (see Section 6.1.5)
3...2	PID	No	Product ID (See Section 6.1.5)
5...4	Battery Design Capacity	No	<p>0000h - Battery not present</p> <p>0001h..FFFEh - Battery's Design capacity in 0.1 WH</p> <p>FFFFh - Design capacity is unknown.</p>
7...6	Battery Last Full Charge Capacity	No	<p>0000h - Battery not present</p> <p>0001h..FFFEh - Battery's last full charge capacity in 0.1 WH</p> <p>FFFFh - Last full charge capacity is unknown.</p>
8	Battery Type	No	<p>Bit 0:</p> <p>1b - Invalid Battery Reference.</p> <p>Bits 7..1:</p> <p>Reserved, receiver Shall ignore these bits.</p>

6.5.7. Get_Manufacturer_Info Message

The [Get_Manufacturer_Info Message](#) is sent by a Port to Request manufacturer- specific information relating to its Port Partner, Cable Plug or of a Battery behind a Port.

This Message is used in the following sequences:

- Get Manufacturer Info

Usage: [Section 7.22](#)

Table 6.56. Get_Manufacturer_Info Message

Byte(s)	Field Name	Static?	Description
0	Manufacturer Info Target	No	0 - Port/Cable Plug 1 - Battery 2..255 - Invalid , receiver Shall send Manufacturer_Info Message with "Not Supported" in the Manufacturer String field.
1	Manufacturer Info Ref	No	When Manufacturer Info Target is set to 0: Reserved , receiver Shall ignore this field. When Manufacturer Info Target is set to 1: Number of the Battery indexed from 0. 0..3 - Fixed Battery 0-3 4..7 - Hot Swappable Battery 0-3 8..255 - Invalid , receiver Shall send Manufacturer_Info Message with "Not Supported" in the Manufacturer String field.

6.5.8. Manufacturer_Info Message

The [Manufacturer_Info Message](#) **Shall** be sent in response to a [Get_Manufacturer_Info Message](#).

This Message is used in the following sequences:

- Get Manufacturer Info

Usage: [Section 7.22](#)

Table 6.57. Manufacturer_Info Message

Byte(s)	Field Name	Static	Description
1...0	VID	No	Vendor ID associated with the Battery manufacturer (see Section 6.1.5)
3...2	PID	No	Product ID (See Section 6.1.5)
N...4	Manufacturer String	No	Vendor defined null terminated string of 0..21 characters. If the Manufacturer Info Target field or Manufacturer Info Ref field in the Get_Manufacturer_Info Message is unrecognized the field Shall return a null terminated ASCII text string "Not Supported".

6.5.9. Security_Request Message

The [Security_Request Message](#) is used by a Port to pass a security data structure to its Port Partner or a Cable Plug.

The authentication process between Port Partner or Port and Cable Plug is described in [USBC Auth].

This Message is used in the following sequences:

- [Security](#) Request

Usage: [Section 7.27](#)

6.5.10. Security_Response Message

The [Security_Response Message](#) is used by a Port or Cable Plug to pass a security data structure to the Port that sent the [Security_Request Message](#).

The authentication process between Port Partner or Port and Cable Plug is described in [USBC Auth].

This Message is used in the following sequences:

- [Security](#) Request

Usage: [Section 7.27](#)

6.5.11. Firmware_Update_Request Message

The firmware update process between Port Partners or a Port and Cable Plug to pass a [PDFU] data structure to the Port that sent the [Firmware_Update_Request Message](#).

The Firmware_Update process between Port Partner or Port and Cable Plug is described in [USB DFU].

This Message is used in the following sequences:

- [Firmware Update](#) Request

Usage: [Section 7.28](#)

6.5.12. Firmware_Update_Response Message

The [Firmware_Update_Response Message](#) is used by a Port or Cable Plug to pass a firmware update data structure to the Port that sent the [Firmware_Update_Request Message](#).

The Firmware_Update process between Port Partner or Port and Cable Plug is described in [USB DFU].

This Message is used in the following sequences:

- [Firmware Update](#) Request

Usage: [Section 7.28](#)

6.5.13. PPS_Status Message

The [PPS_Status Message](#) enables a Sink to query the Source to get additional information about its operational status.

This Message is used in the following sequences:

- [Get PPS Status](#)

Usage: [Section 7.16](#)

Table 6.58. PPS Status Data Block (PPSSDB)

Byte(s)	Field Name	Static	Description
1...0	Output Voltage	No	<p>The Output Voltage field Shall return the Source's output voltage at the time of the Request. The output voltage is measured either at the Source's receptacle or, if the Source has a captive cable, where the voltage is applied to the cable.</p> <p>The measurement accuracy Shall be +/-3% rounded to the nearest 20mV in PPS Mode.</p> <p>0000h..FFFEh - Source's output voltage in 20mV units</p> <p>FFFFh - Field not supported</p>
2	Output Current	No	<p>The Output Current field Shall return the Source's output current at the time of the Request measured at the Source's receptacle.</p> <p>The measurement accuracy Shall be +/-150mA.</p> <p>00h..FEh - Source's output current in 50mA units.</p> <p>FFh - Field not supported.</p>
3	Real Time Flags	No	<p>Bit 0:</p> <p>Reserved, receiver Shall ignore this bit</p> <p>Bits 2..1:</p> <p>Present Temperature Flag (PTF)</p> <p>00b - Not Supported</p> <p>01b - Normal</p> <p>10b - Warning</p> <p>11b - Over-temperature</p> <p>Bit 3:</p> <p>Operating Mode Flag (OMF)</p> <p>0b - Constant Voltage (CV) Mode</p> <p>1b - Current Limit (CL) Mode</p> <p>Shall be set to 0 when not operating in PPS Mode.</p> <p>Shall match the Status Message field Event Flags - Bit 4.</p> <p>Bits 7..4:</p> <p>Reserved, receiver Shall ignore these bits</p>

6.5.14. Country_Info Message

The [Country_Info Message](#) enables a Port to get additional country specific information from its Port Partner.

This Message is used in the following sequences:

- Get Country Info

Usage: [Section 7.24](#)

Table 6.59. Country Info Data Block (CIDB)

Byte(s)	Field Name	Static	Description
1...0	Country Code	No	Bits 7..0: First character of the Alpha-2 Country Code received in the corresponding Get_Country_Info Message Bits 15..8: Second character of the Alpha-2 Country Code received in the corresponding Get_Country_Info Message .
3...2	Reserved	Yes	Reserved , receiver Shall ignore this field.
N...4	Country Specific Data	No	1..22 bytes of content defined by and formatted in a manner determined by an official agency of the country indicated in the Country Code field. If the Country Code field in the Get_Country_Info Message is unrecognized then Country Specific Data field Shall return the null terminated ASCII text string "Unsupported Code".

6.5.15. Country_Codes Message

The [Country_Codes Message](#) enables a Port to query its Port Partner to get a list of alpha-2 country codes as defined in [ISO 3166] for which the Port Partner has country specific information.

This Message is used in the following sequences:

- Get [Country Codes](#)

Usage: [Section 7.23](#)

Table 6.60. Country Codes Data Block (CCDB)

Byte(s)	Field Name	Static	Description
0	Length	Yes	1..12 - Number of country codes in the Message.
1	Reserved	Yes	Reserved , receiver Shall ignore this field.
3...2	Country Code 1	Yes	Bits 7..0: First character of the Alpha-2 Country Code defined by [ISO 3166] Bits 15..8: Second character of the Alpha-2 Country Code defined by [ISO 3166]
5...4	Country Code 2	Yes	If Length is greater than 1: Bits 7..0: First character of the Alpha-2 Country Code defined by [ISO 3166] Bits 15..8: Second character of the Alpha-2 Country Code defined by [ISO 3166]

6.5.16. Sink_Capabilities_Extended Message

The [Sink_Capabilities_Extended Message](#) enables a Sink or a DRP to inform the Source about its Capabilities as a Sink.

This Message is used in the following sequences:

- Get Sink Capabilities Extended

Usage: [Section 7.19.4](#)**Table 6.61. Sink Capabilities Extended Data Block (SKEDB)**

Byte(s)	Field Name	Static	Description
1..0	VID	Yes	Vendor ID (see Section 6.1.5)
3..2	PID	Yes	Product ID (see Section 6.1.5)
7..4	XID	Yes	Value provided by the USB-IF and assigned to the product by the vendor. Shall match XID field in the CertStat VDO.
8	FW Version	Yes	Firmware Version number assigned by the vendor.
9	HW Version	Yes	Hardware Version number assigned by the vendor.
10	SKEDB Version	Yes	Version of this Message format (not specification Version) 0 - Invalid , receiver Shall ignore Message. 1 - Version 1.0 2..255 - Invalid , receiver Shall ignore Message.
11	Load Step	Yes	Bits 1..0: 00b - 150mA/μs Load Step (default) 01b - 500mA/μs Load Step 10b..11b - Invalid , receiver Shall assume default. Bits 7..2: Reserved , receiver Shall ignore these bits.
13...12	Sink Load Characteristics	Yes	Preferred load characteristics of the Sink. Operation Shall Not exceed Capabilities reported in the Source_Capabilities_Extended Message . Bits 4..0: 0..25 - Percent overload in 10% increments 26..31 - Invalid , receiver Shall treat as 25 Bits 10..5: 0..63 - Overload period in 20ms increments. Receiver Should Ignore if Bits 4..0 are 00000b. Bits 14..11: 0..31 - Duty cycle in 5% increments. Receiver Should Ignore if Bits 4..0 are 00000b. Bit 15: 0b - VBUS droop is not tolerated. 1b - Sink can tolerate VBUS droop of an additional 5% during overload conditions.
14	Compliance	Yes	Bits 2..0: xx1b - Indicates LPS compliance x1xb - Indicates PS1 compliance 1xxb - Indicates PS2 compliance Bits 7..3: Reserved , receiver Shall ignore these bits.

Byte(s)	Field Name	Static	Description
15	Touch Temp	Yes	<p>Reports the IEC standard used to determine the surface temperature of the Sink's enclosure. Safety limits for the Sink's touch temperature are set in applicable product safety standards (e.g., [IEC 60950-1] or [IEC 62368-1]).</p> <p>The Sink May report when its touch temperature performance conforms to the TS1 or TS2 limits described in [IEC 62368-1].</p> <p>0 - No applicable standard</p> <p>1 - [IEC 60950-1] (default)</p> <p>2 - [IEC 62368-1] TS1</p> <p>3 - [IEC 62368-1] TS2</p> <p>4..255 - Invalid, receiver Shall assume default.</p>
16	Battery Info	Yes	<p>Bits 3..0:</p> <p>0..4 - Count of Fixed Batteries</p> <p>5..15 - Invalid, receiver Shall assume 0 (no fixed batteries).</p> <p>Bits 7..4:</p> <p>0..4 - Count of Hot Swappable Battery Slots</p> <p>5..15 - Invalid, receiver Shall assume 0 (no hot swappable batteries).</p> <p>This Field Shall match that reported in the Number of Batteries/Battery Slots field of the Source_Capabilities_Extended Message.</p>
17	Sink Modes	Yes	<p>xx1xxxxb - AVS supported</p> <p>xxx1xxxxb - Battery essentially unlimited</p> <p>xxxx1xxx - Battery powered</p> <p>xxxxx1xxb - AC Supply powered</p> <p>xxxxxx1xb - VBUS powered</p> <p>xxxxxxx1b - PPS charging supported.</p>
18	SPR Sink Minimum PDP	Yes	Integer portion of the Minimum PDP of the Port operating in SPR Mode, in watts. See Minimum PDP for more details.
19	SPR Sink Operational PDP	Yes	<p>0..100 - Integer portion of the Operational PDP of the Port operating in SPR Mode, in watts</p> <p>101..255 - Invalid, receiver Shall ignore this field.</p> <p>See Operational PDP for more details.</p>
20	SPR Sink Maximum PDP	Yes	<p>0..100 - Integer portion of the Maximum PDP of the Port operating in SPR Mode, in watts</p> <p>101..255 - Invalid, receiver Shall ignore this field.</p> <p>See Maximum PDP for more details.</p>
21	EPR Sink Minimum PDP	Yes	<p>0..240 - Integer portion of the Minimum PDP of the Port operating in EPR Mode, in watts</p> <p>241..255 - Invalid, receiver Shall ignore this field.</p> <p>See Minimum PDP for more details.</p>

Byte(s)	Field Name	Static	Description
22	EPR Sink Operational PDP	Yes	0..240 - Integer portion of the Operational PDP of the Port operating in EPR Mode , in watts 241..255 - Invalid , receiver Shall ignore this field. See Operational PDP for more details.
23	EPR Sink Maximum PDP	Yes	0..240 - Integer portion of the Maximum PDP of the Port operating in EPR Mode , in watts 241..255 - Invalid , receiver Shall ignore this field. See Maximum PDP for more details.

6.5.16.1. Minimum PDP

The SPR Sink Minimum PDP field **Shall** contain the minimum power Source PDP needed by the Sink, rounded up to the next Watt, to operate at its lowest level of functionality without drawing power from, or charging any present Battery The Minimum PDP field **Shall** be less than or equal to the SPR Sink Operational PDP.

If the Sink is self-powered, such that it doesn't need power from a Source, then it **Shall** set this field to 0.

If the Sink is not EPR Capable, or if the Sink is self-powered, such that it doesn't need power from a Source, this field **Shall** be set to 0.

If the Sink is EPR Capable and is unable to operate at PDPs less than 100W, it **Shall** set the SPR Sink Minimum PDP field to the minimum power to sustain PD communication.

Possible examples of SPR Sink Minimum PDP could be:

- The power required to have basic functionality by a batteryless Sink,
- On a Device with a Battery, it can power the minimum functionality of the Device.

Possible examples of EPR Sink Minimum PDP could be:

- The power required to have basic functionality by a batteryless Sink,
- On a Device with a Battery, it can power the minimum functionality of the Device.

Note: EPR Sink Minimum PDP can be the same as its SPR Sink Minimum PDP.

6.5.16.2. Operational PDP

The Operational PDP field **Shall** contain the Source PDP that the manufacturer recommends for the normal functionality of the Sink, rounded up to the next integerWatt. This corresponds to the PDP Rating of Sources that the Sink is designed to operate with (See [Section 3.4.2](#)). The Operational PDP field **Shall** be sufficient to operate all the Sink's functional modes normally AND charge the Sink's Battery if present. For Sinks with a Battery(s), the SPR Sink Operational PDP field **Shall** correspond to the PDP Rating of the Charger shipped with the Sink or the recommended Charger's PDP Rating.

If the Sink is self-powered, such that it doesn't need power from a Source, then it **Shall** set this field to 0.

If the Sink is not EPR Capable, or if the Sink is self-powered, such that it doesn't need power from a Source, this field **Shall** be set to 0.

If the Sink is EPR Capable and is unable to operate at PDPs less than 100W, it **Shall** set the SPR Sink Minimum PDP field to the minimum power to sustain PD communication.

6.5.16.3. Maximum PDP

The SPR Sink Maximum PDP field **Shall** contain the highest PDP the Sink will ever Request under any operating condition, rounded up to the next integer, including charging its Battery if present. The SPR Sink Maximum PDP field **Shall Not** be less than the SPR Sink Operational PDP field, but **May** be the same. The value is used by the Source to determine the maximum amount of power it has to budget for the Attached Sink.

If the Sink is self-powered, such that it doesn't need power from a Source, then it **Shall** set this field to 0.

If the Sink is not EPR Capable, it **Shall** set the EPR Maximum PDP field to 0.

If the Sink is EPR Capable and is unable to operate at PDPs less than 100W, it **Shall** set SPR Maximum PDP field to the minimum power to sustain PD communication.

6.5.17. Extended_Control Message

The [Extended_Control Message](#) extends the Control Message space.

Table 6.62. Extended Control Data Block (ECDB)

Byte(s)	Field Name	Static	Description
0	Type	No	Extended Control Message Type (see Table 6.63 for details).
1	Data	No	Additional Extended Control Message Data (see Table 6.63 for details).

Table 6.63. Extended Control Message Types

Type	Data	Message Type
0	X	Invalid , receiver Shall respond with Not_Supported .
1	0	EPR_Get_Source_Cap
2	0	EPR_Get_Sink_Cap
3	0	EPR_Keep_Alive
4	0	EPR_Keep_Alive_Ack
5..255	X	Invalid , receiver Shall respond with Not_Supported .

6.5.17.1. EPR_Get_Source_Cap Message

The [EPR_Get_Source_Cap](#) (EPR Get Source Capabilities) Message **May** only be sent by a Port capable of operating as a Sink and that supports [EPR Mode](#) to Request the Source Capabilities and Dual-Role Power capability of its Port Partner.

An EPR Capable Sink Port that is operating in SPR Mode **Shall** treat the [EPR_Source_Capabilities Message](#) response as informational only and **Shall Not** respond with an [EPR_Request Message](#). The [EPR_Get_Source_Cap Message](#) **Shall Not** be sent by a Port that does not support [EPR Mode](#).

This Message is used in the following sequences:

- EPR Get Source Capabilities

Usage: [Section 7.30.8](#)

6.5.17.2. EPR_Get_Sink_Cap Message

The [EPR_Get_Sink_Cap](#) (EPR Get Sink Capabilities) Message **May** only be sent by a Port capable of operating as a Source and that supports [EPR Mode](#) to Request the Sink Capabilities and Dual-Role Power capability of its Port Partner. The [EPR_Get_Sink_Cap Message](#) **Shall Not** be sent by a Port that does not support EPR Mode

This Message is used in the following sequences:

- EPR Get Sink Capabilities

Usage: [Section 7.30.9](#)

6.5.17.3. EPR_Keep_Alive Message

The [EPR_Keep_Alive Message](#) **May** be sent by a Sink operating in [EPR Mode](#) to meet the requirement for periodic traffic.

This Message is used in the following sequences:

- EPR Keep Alive

Usage: [Section 7.30.6](#)

6.5.17.4. EPR_Keep_Alive_Ack Message

The [EPR_Keep_Alive_Ack Message](#) **Shall** be sent by a Source operating in [EPR Mode](#) in response to an [EPR_Keep_Alive Message](#).

This Message is used in the following sequences:

- EPR Keep Alive

Usage: [Section 7.30.6](#)

6.5.18. EPR_Source_Capabilities Message

The [EPR_Source_Capabilities Message](#) is used by a Port to describe its power Capabilities when acting as a Source in [EPR Mode](#). This Message contains 1 to 11 Power Data Object(PDOs), ordered as follows:

1. The SPR PDOs and APDOs as reported in the SPR Capabilities Message.
2. If the SPR Capabilities Message contains fewer than 7 PDOs, the unused Data Objects **Shall** be zero filled.
3. The EPR PDOs and APDOs as defined in [Section 6.4.1.3](#) **Shall** start at Data Object position 8 and **Shall** be sent in the following order:

Fixed Supply PDOs that offer 28V, 36V or 48V, if present, **Shall** be sent in voltage order; lowest to highest.

One EPR AVS APDO **Shall** be sent.

Figure 6.8. Source Capabilities Message Format

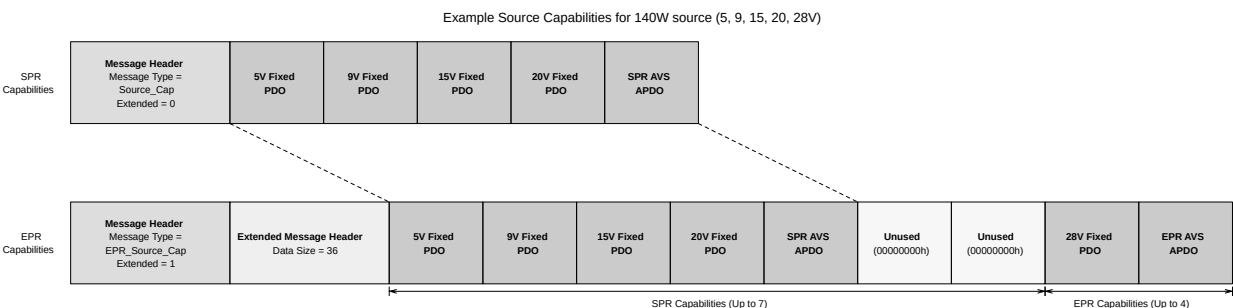
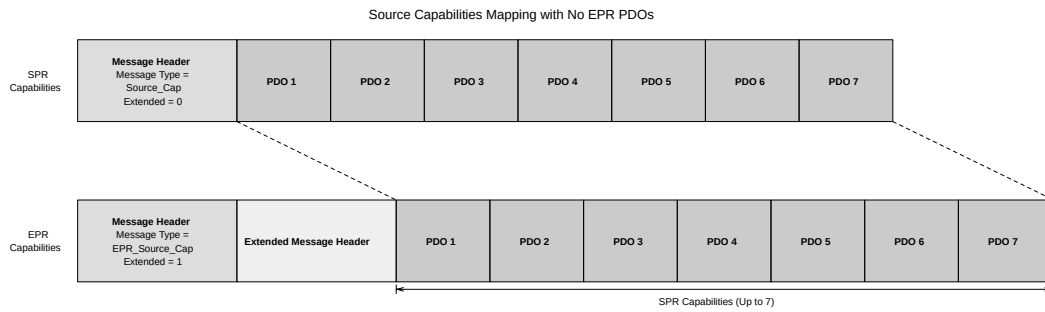


Figure 6.9. Source Capabilities Message Format (with no EPR PDOs)

This Message is used in the following sequences:

- EPR Source Capabilities
- EPR Get Source Capabilities

Usage: [Section 7.30.2](#)

6.5.19. EPR_Sink_Capabilities Message

The [EPR_Sink_Capabilities](#) is an EPR Capabilities Message that contains a list of Power Data Objects that the EPR Sink requires to operate. It is sent by an EPR Sink in order to convey its power requirements to an EPR Source. The EPR Sink **shall** only send the [EPR_Sink_Capabilities Message](#) in response to an [EPR_Get_Sink_Cap Message](#).

Construction of this Message **shall** follow the same rules as [EPR_Source_Capabilities Message](#) (see [Section 6.5.18](#)).

This Message is used in the following sequences:

- EPR Sink Capabilities
- EPR Get Sink Capabilities

Usage: [Section 7.30.4](#)

6.5.20. Vendor_Defined_Extended Message

The [Vendor_Defined_Extended Message](#) (VDEM) allows vendors to exchange information outside of the parameters defined by this specification using the [Extended Message](#) format.

To ensure vendor uniqueness, all [Vendor_Defined_Extended Messages](#) **shall** contain a **Valid** USB Standard or Vendor ID (SVID) that is allocated by USB-IF in the VDM Header.

Table 6.64. Vendor_Defined_Extended Message+

Byte(s)	Field Name	Static	Description
1...0	SVID	No	Standard/Vendor ID (assigned by the USB-IF).
3...2	Command Space	No	Bit 15: Reserved , shall be 0b Bits 14..0: Vendor Command Data.
N...4	Vendor Defined Data	No	Vendor defined field of 0..256 bytes.

6.6. Value Parameters

Table 6.65. Value Parameters

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
MaxExtendedMsgLen		260		Byte	Maximum length of an Extended Message as expressed in the Data Size field.
MaxExtendedMsgChunkLen		26		Byte	Maximum length of an Extended Message Chunk.
MaxExtendedMsgLegacyLen		26		Byte	Maximum length of an Extended Message that can be sent without Chunking.

Chapter 7. PD Communications Protocol Message Usage

7.1. Reset

Resets are a necessary response to protocol or other error conditions. USB Power Delivery defines four different types of reset:

- [Soft Reset](#), which resets protocol.
- [Data Reset](#), which resets the USB communications.
- [Hard Reset](#), which resets both the power supplies and protocol.
- [Cable Reset](#) which resets the cable.

7.1.1. Soft Reset and Protocol Error

A [Soft Reset Message](#) is used to cause a [Soft Reset](#) of the communication protocol when it has broken down in some way. The breakdown in communication or the Protocol Error may take several forms:

- An Unexpected Message is received during an AMS.
- An expected Message fails to arrive.
- A timeout occurs while waiting for a Message or event.

The [Soft Reset](#) **Shall Not** have any impact on power supply operation but is used to correct a Protocol Error occurring during an Atomic Message Sequence (AMS). After a [Soft Reset](#) has completed, an Explicit Contract Negotiation occurs, in order to re-establish PD Communication and to bring State operation for both Port Partners back to either the PE_SNK_Ready or PE_SRC_Ready states as appropriate (see [Section 9.2.5](#)). The [Soft Reset](#) **May** be triggered by either Port Partner in response to the Protocol Error. An [Accept Message](#) is sent by the recipient of a [Soft Reset](#) Message to indicate that it has completed the [Soft Reset](#).

The Protocol Error may result in:

- a [Hard Reset](#).
- a [Soft Reset](#).
- or no response.

An Unrecognized Message or Unsupported Message received in a Protocol Engine ready State causes a [Not_Supported Message](#) to be generated.

If the error is not corrected by the [Soft Reset](#), [Hard Reset](#) Signaling **Shall** be issued (see [Section 7.1.3](#)).

A [Soft Reset](#) may be sent by either the Source or Sink in response to a Protocol Error irrespective of the value of Rp, either [SinkTxOK](#) or [SinkTxNG](#). If the [Soft_Reset Message](#) fails a [Hard Reset](#) **Shall** be initiated within [tHardReset](#) of the last [CRCReceiveTimer](#) expiring after [nRetryCount](#) retries have been completed.

A [Soft_Reset Message](#) **Shall** be targeted at a specific entity depending on the type of SOP* Packet used. [Soft_Reset Messages](#) sent using SOP Packets **Shall** [Soft_Reset](#) the Port Partner only. [Soft_Reset Messages](#) sent using SOP' Packet/ SOP'' Packets **Shall** [Soft_Reset](#) the corresponding Cable Plug only.

After a [VCONN Swap](#) the VCONN Source needs to reset the Cable Plug's Protocol Layer to ensure MessageID synchronization. If after a [VCONN Swap](#) the VCONN Source wants to communicate with a Cable Plug using SOP'

Packets, it **Shall** issue a [Soft_Reset Message](#) using a SOP' Packet in order to reset the Cable Plug's Protocol Layer. If the VCONN Source wants to communicate with a Cable Plug using SOP" Packets, it **Shall** issue a [Soft_Reset Message](#) using a SOP" Packet in order to reset the Cable Plug's Protocol Layer.

See [Section 9.2, "Policy Engine Layer State Diagrams"](#) for more details.

[Table 7.1](#) and [Table 7.2](#) summarize the responses that **Shall** be made to an incoming Message including VDMs.

Table 7.1. Response to an incoming Message (except VDM)

Recipient's Power Role	Recipient's State	Incoming Message			
		Recognized			Unrecognized
		Supported		Unsupported	
		Expected	Unexpected		
Source	PE_SRC_Ready	Process Message	Soft_Reset Message¹	Not_Support-ed Message²	Not_Supported Mes-sage² (except for VDM) See Section 8.4 for UVDM. See Section 8.5 for SVDM
	During AMS (power not transitioning ³)	Process Message	Soft_Reset Message¹		
	During AMS (power transitioning ³)	Process Message	Hard_Reset Signaling		
Sink	PE_SNK_Ready	Process Message	Soft_Reset Message¹	Not_Support-ed Message²	Not_Supported Mes-sage² (except for VDM) See Section 8.4 for UVDM. See Section 8.5 for SVDM
	During AMS (not power transitioned)	Process Message	Soft_Reset Message¹		
	During AMS (power transitioned)	Process Message	Hard_Reset Signaling		
<div>1. The Soft_Reset Message Shall be sent using the SOP* of the incoming Message.</div> <div>2. The Not_Supported Message Shall be sent using the SOP* of the incoming Message.</div> <div>3. “Power transitioning” means the Policy Engine is in PE_SRC_Transition_Supply State or PE_SNK_Transition_Sink State or PE_FRS_SNK_SRC_Start_AMS State.</div>					

Table 7.2. Response to an incoming VDM

Recipient's Role	Unstructured VDM			Structured VDM		
	Supported	Unsupported	Unrecognized	Supported	Unsupported	Unrecognized
DFP or UFP	Defined by vendor	Not_Support-ed Message	Not_Support-ed Message	See: Table 7.4	Not_Support-ed Message	NAK Command
Cable Plug	Defined by vendor	Message Ignored	Message Ignored	See: Table 7.5	Message Ignored	NAK Command

A failure to see a [GoodCRC Message](#) in response to any Message within [tReceive](#) (after [nRetryCount](#) retries), when a Port Pair is Connected, is indicative of a communications failure resulting in a [Soft Reset](#) (see [Section 7.1.1](#)).

A [Soft Reset](#) **Shall** impact the USB Power Delivery layers in the following ways:

- PHY Layer: Reset not required since the PHY Layer resets on each Packet transmission/reception.
- Protocol Layer: Reset MessageIDCounter, RetryCounter and State machines.
- Policy Engine: Reset State dependent behavior by performing an Explicit Contract Negotiation.

- Power supply: Does not change.

Note: When in SPR Mode the Source sends a [Source_Capabilities Message](#) and when in [EPR Mode](#) the Source sends an [EPR_Source_Capabilities Message](#).

A [Soft Reset](#) is performed using an AMS (see [Section 7.7](#)).

7.1.2. Data Reset

The [Data_Reset Message](#) **May** be sent by either the DFP or UFP and **Shall** reset the USB data connection and exit all Alternate Modes with its Port Partner while preserving the power on VBUS. USB4 Mode capable ports **Shall** support the [Data_Reset Message](#) and other ports **May** support the [Data_Reset Message](#).

The [Data_Reset Message](#) **Shall Not** change the existing:

- Power Contract
- Data Roles (i.e., which Port is the DFP or UFP)

The receiver of the [Data_Reset Message](#) **Shall** respond by sending an [Accept Message](#) and then follow the process outlined in the following steps. Neither the sender nor receiver **Shall** initiate a [VCONN Swap](#) until the [Data_Reset](#) process is complete, and the [Data_Reset_Complete Message](#) has been sent. Following receipt of the [Accept Message](#), or [GoodCRC Message](#) following the [Accept Message](#), depending which Port sends the [Data_Reset Message](#):

1. The DFP **Shall**:
 - Disconnect the Port's [USB2] D+/D- signals.
 - If operating in [USB3] remove the Port's Rx Terminations (see [USB3]).
 - If operating in [USB4] drive the Port's SBTX to a logic low (see [USB4]).
2. Both the DFP and UFP **Shall** exit all Alternate Modes if any.
3. Reset the cable:
 - If the VCONN Source Port is also the UFP, then it **Shall** run the UFP VCONN Power Cycle process described in [Section 4.6](#).
 - If the VCONN Source Port is also the DFP, then it **Shall** run the DFP VCONN Power Cycle process described in [Section 4.6](#).
 - The DFP **Shall** exit the VCONN Power Cycle process as the VCONN Source and be sourcing VCONN.
4. After tDataReset the DFP **Shall**:
 - Reconnect the [USB2] D+/D- signals.
 - If the Port was operating in [USB3] or [USB4] reapply the Port's Rx Terminations (see [USB3]).
5. The [Data_Reset](#) process is complete; the DFP **Shall** send a [Data_Reset_Complete Message](#) and enter the USB4 Discovery and Entry Flow (See [USB-C]).

If the Initiator of the [Data_Reset Message](#) does not receive a **Valid** response within tSenderResponse it **Shall** enter the ErrorRecovery State.

7.1.3. Hard Reset

Hard Resets are signaled by an ordered set as defined in [Table 5.4](#). Both the sender and recipient **Shall** cause their power supplies to return to their default states (see [Figure 4.14](#) for details of voltage transitions). In addition, their respective Protocol Layers **Shall** be reset as for the [Soft Reset](#). This allows the Attached devices to be in a State

where they can re-establish USB PD communication. Hard Reset is retried up to [nHardResetCount](#) times (see also [Section 7.31.5.4](#) and [Section 7.32.5](#)).

Note: Even though VBUS drops to [vSafe0V](#) during a [Hard Reset](#) a Sink will not see this as a disconnect since this is expected behavior.

A [Hard Reset](#) **Shall Not** cause any change to either the Rp/Rd resistor being asserted.

If there has been a [Data Role Swap](#) the [Hard Reset](#) **Shall** cause the Port Data Role to be changed back to DFP for a Port with the Rp resistor asserted and UFP for a Port with the Rd resistor asserted.

When VCONN is supported (see [USB-C]) the [Hard Reset](#) **Shall** cause the Port with the Rp resistor asserted to supply VCONN and the Port with the Rd resistor asserted to turn off VCONN.

If the [Hard Reset](#) is insufficient to clear the error condition, then the Port **Shall** use USB Type-C ErrorRecovery as defined in [USB-C].

A Sink **Shall** be able to send [Hard Reset](#) Signaling regardless of the value of Rp (see [Section 5.2.2](#)").

7.1.3.1. Cable Plugs and Hard Reset

Cable Plugs **Shall Not** generate [Hard Reset](#) Signaling but **Shall** monitor for Hard Reset Signaling between the Port Partners and **Shall** reset when this is detected (see [Section 9.2.26.2.2](#)). The Cable Plugs **Shall** perform the equivalent of a power cycle returning to their initial power up State. This allows the Port Partners to be in a State where they can re-establish USB PD communication.

7.1.3.2. Modal Operation and Hard Reset

A [Hard Reset](#) **Shall** cause [EPR Mode](#) and all Active Modes to be exited by both Port Partners and any Cable Plugs (see [Section 8.6.4](#)").

7.1.4. Cable Reset

Cable Resets are signaled by an ordered set as defined in [Table 5.4](#). Both the sender and recipient of [Cable Reset](#) Signaling **Shall** reset their respective Protocol Layers. The Cable Plugs **Shall** perform the equivalent of a power cycle returning to their initial power up State. This allows the Port Partners to be in a State where they can re-establish USB PD communication.

Only a DFP **Shall** generate [Cable Reset](#) Signaling. The DFP must be supplying VCONN prior to a [Cable Reset](#). If VCONN has been turned off the DFP **Shall** turn on VCONN prior to generating [Cable Reset](#) Signaling. If there has been a [VCONN Swap](#) and the UFP is currently supplying VCONN, the DFP **Shall** perform a [VCONN Swap](#) such that it is supplying VCONN prior to generating [Cable Reset](#) Signaling. A DFP **Shall** only generate [Cable Reset](#) Signaling within an Explicit Contract.

A [Cable Reset](#) **Shall** cause all Active Modes in the Cable Plugs to be exited (see [Section 8.6.4](#)").

7.2. Collision Avoidance

To avoid Message collisions due to asynchronous Messaging sent from the Sink, the Source sets Rp to SinkTxOK to indicate to the Sink that it is OK to initiate an AMS. When the Source wishes to initiate an AMS, it sets Rp to SinkTxNG. When the Sink detects that Rp is set to SinkTxOK it **May** initiate an AMS. When the Sink detects that Rp is set to SinkTxNG it **Shall Not** initiate an AMS once [tSinkDelay](#) has elapsed after SinkTxNG is asserted, and **Shall** only send Messages that are part of a Source-initiated AMS.

Note: This restriction applies to SOP* AMSs i.e., for both Port to Port and Port to Cable Plug communications.

Note: A Sink can still send [Hard Reset](#) Signaling at any time.

7.3. Message Discarding

On receiving a received Message on SOP, the Protocol Layer **Shall Discard** any pending SOP* Messages. A received Message on SOP'/SOP'' **Shall Not** cause any pending SOP* Messages to be **Discarded**.

It is assumed that Messages using SOP'/SOP'' constitute a simple Request/response AMS, with the Cable Plug providing the response so there is no reason for a pending SOP* Message to be **Discarded**. There can only be one AMS between the Port Partners, and these also take priority over Cable Plug communications so a Message received on SOP will always cause a Message pending on SOP* to be **Discarded**.

[Table 7.3](#) for details of the Messages that **Shall/ Shall Not** be **Discarded**.

Table 7.3. Message Discarding

Message pending transmission	Message received	Message to be Discarded
SOP	SOP	Outgoing Message
SOP	SOP'/SOP''	Incoming Message
SOP'	SOP	Outgoing Message
SOP'	SOP'	Incoming Message
SOP'	SOP''	Incoming Message
SOP''	SOP	Outgoing Message
SOP''	SOP'	Incoming Message
SOP''	SOP''	Incoming Message

7.4. Common Messages

7.4.1. Accept

Usage details for the [Accept Message](#) are given in each of the different AMS sequences where it is used.

7.4.2. Reject

Most usage details for the [Reject Message](#) are given in each of the different AMS sequences where it is used.

The sender of a [Request](#), [EPR_Request Message](#), [PR_Swap Message](#), [DR_Swap Message](#), [VCONN_Swap](#), or [Enter_USB Message](#), on receiving a [Reject Message](#) response, **Shall Not** send this same Message to the recipient until one of the following has occurred:

- A New Explicit Contract Negotiation as a result of the Source sending a [Source_Capabilities Message](#) or [EPR_Source_Capabilities Message](#). This can be triggered by:
 - The Source's Device Policy Manager.
 - A [Get_Source_Cap Message](#) sent from the Sink to the Source in SPR Mode.
 - An [EPR_Get_Source_Cap Message](#) sent from the Sink to the Source in EPR Mode.
 - A [Power Role Swap](#).
 - A [Soft Reset](#).
 - A [Hard Reset](#).
 - A Disconnect/Re-Connect.
- A [Data Role Swap](#).

- A [Data Reset](#).

7.4.3. Wait

Usage details for the [Wait Message](#) are given in each of the different AMS sequences where it is used.

7.5. Applicability Tables

[Table 7.4](#) and [Table 7.5](#) lists the AMS and Message requirements for a Port/Cable Plug.

The "Initiator Applicability" column refers to a Port's requirement to be able to start an AMS, while the "Responder Applicability" column refers to a Port's requirement to respond to an AMS. The "Transmitter" column specifies the transmitter of that specific Message in the sequence. The Initiator of an AMS is the transmitter of the first Message in that sequence. Ports **May** initiate or respond to any AMS not explicitly forbidden.

When multiple messages are allowed in an AMS (e.g. [Accept/Reject/Wait](#)), transmitters **May** choose to not implement all of the options. Receivers **Shall** be able to understand any of the Message options.

See [Chapter 9](#) for detailed State diagrams explaining the AMS and messaging options.

Table 7.4. Port Sequence Applicability

AMS	Initiator Applicability	Responder Applicability	General Message Sequence	Transmitter
Source Capabilities	Required for ports that can operate as Source	Required	Source_Capabilities	Source
			Request	Sink
			Accept/Reject/Wait	Source
			PS_RDY	Source
Request	Required for ports that can operate as Sink	Required	Request	Sink
			Accept/Reject/Wait	Source
			PS_RDY	Source
EPR Source Capabilities	Required for ports that can operate as EPR Source	Required for EPR Capable ports	EPR_Source_Capabilities	Source
			EPR_Request	Sink
			Accept/Reject/Wait	Source
			PS_RDY	Source
EPR Request	Required for ports that can operate as EPR Sink	Required for EPR Capable ports	EPR_Request	Sink
			Accept/Reject/Wait	Source
			PS_RDY	Source
EPR Mode Enter	Required for ports that can operate as EPR Sink	Required for EPR Capable ports	EPR_Mode (Enter)	Sink
			EPR_Mode (Enter Acknowledge)	Source
			VCONN Swap Sequence	-
			Cable Plug Discover Identity (SVDM) Sequence	-
			EPR_Mode (Enter Succeeded/Enter Failed)	Source
			EPR_Source_Capabilities Sequence	-
EPR Keep Alive	Required for ports that can operate as EPR Sink	Required for EPR Capable ports	EPR_Keep_Alive	Sink
			EPR_Keep_Alive_Ack	Source
EPR Mode Exit	Recommended for EPR Capable ports	Required for EPR Capable ports	EPR_Mode (Exit)	Any
			Source Capabilities Sequence	-
Unsupported Sequence		Required	Initiating Message	Any
			Not_Supported	Any

AMS	Initiator Applicability	Responder Applicability	General Message Sequence	Transmitter
Soft Reset	Required	Required	Soft_Reset	Any
			Accept	Responder
			Source Capabilities/EPR Source Capabilities Sequence	-
Data Reset	Required for USB4 capable ports	Required for USB4 capable ports	Data_Reset	Any
			Accept	Responder
			PS_RDY	UFP VCONN Source
			Data_Reset_Complete	DFP
Power Role Swap	Required for DRP	Required for DRP Unsupported by Source only/Sink only	PR_Swap	Any
			Accept/Reject/Wait	Responder
			PS_RDY	Initial Source
			PS_RDY	New Source
Fast Role Swap	Recommended for DRP	Optional for DRP Unsupported by Source only/Sink only	FR_Swap	Sink
			Accept	Source
			PS_RDY	Initial Source
			PS_RDY	New Source
Data Role Swap	Required for DRD	Required for DRD	DR_Swap	Any
			Accept/Reject/Wait	Responder
VCONN Swap	Recommended	Required for ports that can supply VCONN	VCONN_Swap	Any
			Accept/Reject/Wait	Responder
			PS_RDY	New VCONN Source
Alert	Required for ports that support SPR PPS Mode	Recommended	Alert	Any
			Optional Get Status Sequence	-
			Optional Get Battery Status Sequence	-
Get Status	Recommended	Required for Source ports that support Alert	Get_Status	Any
			Status	Responder
Get PPS Status	Required for ports that support SPR PPS Mode	Required for ports that support SPR PPS Mode	Get_PPS_Status	PPS Sink
			PPS_Status	PPS Source
Get Source Capabilities	Recommended for ports that can operate as Sink	Required for ports that can operate as Source	Get_Source_Cap	Any
			Source_Capabilities	Responder
			If in SPR Mode and AMS was initiated by the Sink: Request Sequence	-
Get Sink Capabilities	Recommended for ports that can operate as Source	Required for ports that can operate as Sink	Get_Sink_Cap	Any
			Sink_Capabilities	Responder
EPR Get Source Capabilities	Recommended for ports that can operate as EPR Sink	Required for ports that can operate as EPR Source	EPR_Get_Source_Cap	Any
			EPR_Source_Capabilities/Reject	Responder
			If in EPR Mode and AMS was initiated by the Sink: EPR Request Sequence	-
EPR Get Sink Capabilities	Recommended for ports that can operate as EPR Source	Required for ports that can operate as EPR Sink	EPR_Get_Sink_Cap	Any
			EPR_Sink_Capabilities/Reject	Responder

AMS	Initiator Applicability	Responder Applicability	General Message Sequence	Transmitter
Get Source Capabilities Extended	Recommended for ports that can operate as Sink	Recommended for ports that can operate as Source	Get_Source_Cap_Extended	Any
			Source_Capabilities_Extended	Responder
Get Sink Capabilities Extended	Recommended for ports that can operate as Source	Required for ports that can operate as Sink	Get_Sink_Cap_Extended	Any
			Sink_Capabilities_Extended	Responder
Get Battery Capabilities	Recommended	Required if product contains batteries	Get_Battery_Cap	Any
			Battery_Capabilities	Responder
Get Battery Status	Recommended	Required if product contains batteries	Get_Battery_Status	Any
			Battery_Status	Responder
Get Manufacturer Info	Recommended	Recommended	Get_Manufacturer_Info	Any
			Manufacturer_Info	Responder
Get Country Codes	Required if mandated by a country authority	Required if mandated by a country authority	Get_Country_Codes	Any
			Country_Codes	Responder
Get Country Info	Required if mandated by a country authority	Required if mandated by a country authority	Get_Country_Info	Any
			Country_Info	Responder
Get Revision	Recommended	Required	Get_Revision	Any
			Revision	Responder
Get Source Information	Recommended for ports that can operate as Sink	Required for ports that can operate as Source (except single Port SPR Chargers with Invariant PDOs)	Get_Source_Info	Any
			Source_Info	Responder
Security Request	Required by ports that support USB security communication as defined in [USBC Auth].	Required by ports that support USB security communication as defined in [USBC Auth].	Security_Request	Any
			Security_Response	Responder
Firmware Update Request	Required by ports that support USB firmware update communication as defined in [USB DFU].	Required by ports that support USB firmware update communication as defined in [USB DFU].	Firmware_Update_Request	Any
			Firmware_Update_Response	Responder
Discover Identity (SVDM)	Recommended for ports that can operate as Source	Required for USB4 capable ports, products with more than one DFP, or ports that support Modal Operation	Discover_Identity Vendor_Defined INIT	Any
			Discover_Identity Vendor_Defined ACK/NAK/BUSY	Responder
Discover SVIDs (SVDM)	Required for ports that support Modal Operation	Required for ports that support Modal Operation	Discover_SVIDs Vendor_Defined INIT	Any
			Discover_SVIDs Vendor_Defined ACK/NAK/BUSY	Responder
Discover Modes (SVDM)	Required for ports that support Modal Operation	Required for ports that support Modal Operation	Discover_Modes Vendor_Defined INIT	Any
			Discover_Modes Vendor_Defined ACK/NAK/BUSY	Responder
Enter Mode (SVDM)	Required for ports that support Modal Operation	Required for ports that support Modal Operation	Enter_Mode Vendor_Defined INIT	DFP
			Enter_Mode Vendor_Defined ACK/NAK/BUSY	UFP
Exit Mode (SVDM)	Required for ports that support Modal Operation	Required for ports that support Modal Operation	Exit_Mode Vendor_Defined INIT	DFP
			Exit_Mode Vendor_Defined ACK/NAK/BUSY	UFP
Structured VDM (Other)	Optional	Optional	Vendor_Defined INIT	Any
			Vendor_Defined ACK/NAK/BUSY	Responder

AMS	Initiator Applicability	Responder Applicability	General Message Sequence	Transmitter
Enter_USB	Required for USB4 capable ports	Required for USB4 capable ports	Enter_USB	DFP
			Accept/Reject/Wait	UFP

[Table 7.5](#) applies to all Cable Plugs and to any Ports that are otherwise required to communicate with the Cable Plug. Only Ports acting as a VCONN Source are allowed to initiate AMSs listed in this table. The Port is always the Initiator of any AMS to the Cable Plug. For a port that is not currently the VCONN Source to communicate with the Cable Plug, initiate the [VCONN Swap](#) AMS to become the VCONN Source.

Cable Plugs **May** respond to any AMS not explicitly forbidden.

Table 7.5. Cable Plug Sequence Applicability

AMS	Port Initiator Applicability	SOP' Responder Applicability	SOP'' Responder Applicability	Message Sequence	Transmitter
Unsupported Sequence		Required	Required	Initiating Message	Port (VCONN Source)
				Not_Supported	Cable
Soft_Reset	Required	Required	Required	Soft_Reset	Port (VCONN Source)
				Accept	Cable
Get_Status	Optional	Required for Active Cables	Required for Active Cables	Get_Status	Port (VCONN Source)
				Status	Cable
Get_Manufacturer_Info	Optional	Recommended	Not Allowed	Get_Manufacturer_Info	Port (VCONN Source)
				Manufacturer_Info	Cable
Get_Revision	Optional	Required for Active Cables	Not Allowed	Get_Revision	Port (VCONN Source)
				Revision	Cable
Security_Request	Required by ports that support USB security communication as defined in [USBC Auth].	Required by cables that support USB security communication as defined in [USBC Auth].	Not Allowed	Security_Request	Port (VCONN Source)
				Security_Response	Cable
Firmware_Update_Request	Required by ports that support USB firmware update communication as defined in [USB DFU].	Required by cables that support USB firmware update communication as defined in [USB DFU].	Optional	Firmware_Update_Request	Port (VCONN Source)
				Firmware_Update_Response	Cable
Discover_Identity (SVDM)	Required for ports that can operate as Source and offer >3A. Required for ports that support Modal Operation. Required for USB4 capable ports. Recommended for all ports.	Required	Not Allowed	Discover_Identity Vendor_Defined INIT	Port (VCONN Source)
				Discover_Identity Vendor_Defined ACK/NAK/BUSY	Cable

AMS	Port Initiator Applicability	SOP' Responder Applicability	SOP'' Responder Applicability	Message Sequence	Transmitter
Discover SVIDs (SVDM)	Required for ports that support Modal Operation.	Required , responds NAK if Modal Operation is not supported	Not Allowed	Discover SVIDs Vendor_Defined INIT	Port (VCONN Source)
				Discover SVIDs Vendor_Defined ACK/NAK/BUSY	Cable
Discover Modes (SVDM)	Required for ports that support Modal Operation.	Required , responds NAK if Modal Operation is not supported	Not Allowed	Discover Modes Vendor_Defined INIT	Port (VCONN Source)
				Discover Modes Vendor_Defined ACK/NAK/BUSY	Cable
Enter Mode (SVDM)	Required for ports that support Modal Operation.	Required , responds NAK if Modal Operation is not supported	Optional	Enter Mode Vendor_Defined INIT	Port (VCONN Source)
				Enter Mode Vendor_Defined ACK/NAK/BUSY	Cable
Exit Mode (SVDM)	Required for ports that support Modal Operation.	Required , responds NAK if Modal Operation is not supported	Optional	Exit Mode Vendor_Defined INIT	Port (VCONN Source)
				Exit Mode Vendor_Defined ACK/NAK/BUSY	Cable
Structured VDM (Other)	Optional	Optional	Optional	Vendor_Defined INIT	Port (VCONN Source)
				Vendor_Defined ACK/NAK/BUSY	Cable
Enter USB	Required for USB4 capable ports.	Required for Active Cables that support USB4	Required for Active Cables that support USB4	Enter_USB	Port (VCONN Source)
				Accept/Reject/Wait	Cable

[Table 7.6](#) and [Table 7.7](#) cover messages that may not be a part of an AMS, but still have applicability for given ports/cable plugs.

Table 7.6. Port Message Applicability

Message	Transmit Applicability	Receive Applicability
GoodCRC	Required	Required
Attention (SVDM)	Optional	Optional
Vendor_Defined (Unstructured)	Optional	Optional
Vendor_Defined_Extended	Optional	Optional
BIST	Required	Required
Hard_Reset	Required	Required

Table 7.7. Cable Plug Message Applicability

Message	Transmit Applicability	Receive Applicability
GoodCRC	Required	Required
Attention (SVDM)	Not Allowed	Ignore
Vendor_Defined (Unstructured)	Optional	Optional
Vendor_Defined_Extended	Optional	Optional
BIST	Not Allowed	Required
Hard_Reset	Not Allowed	Required
Cable_Reset	Not Allowed	Required

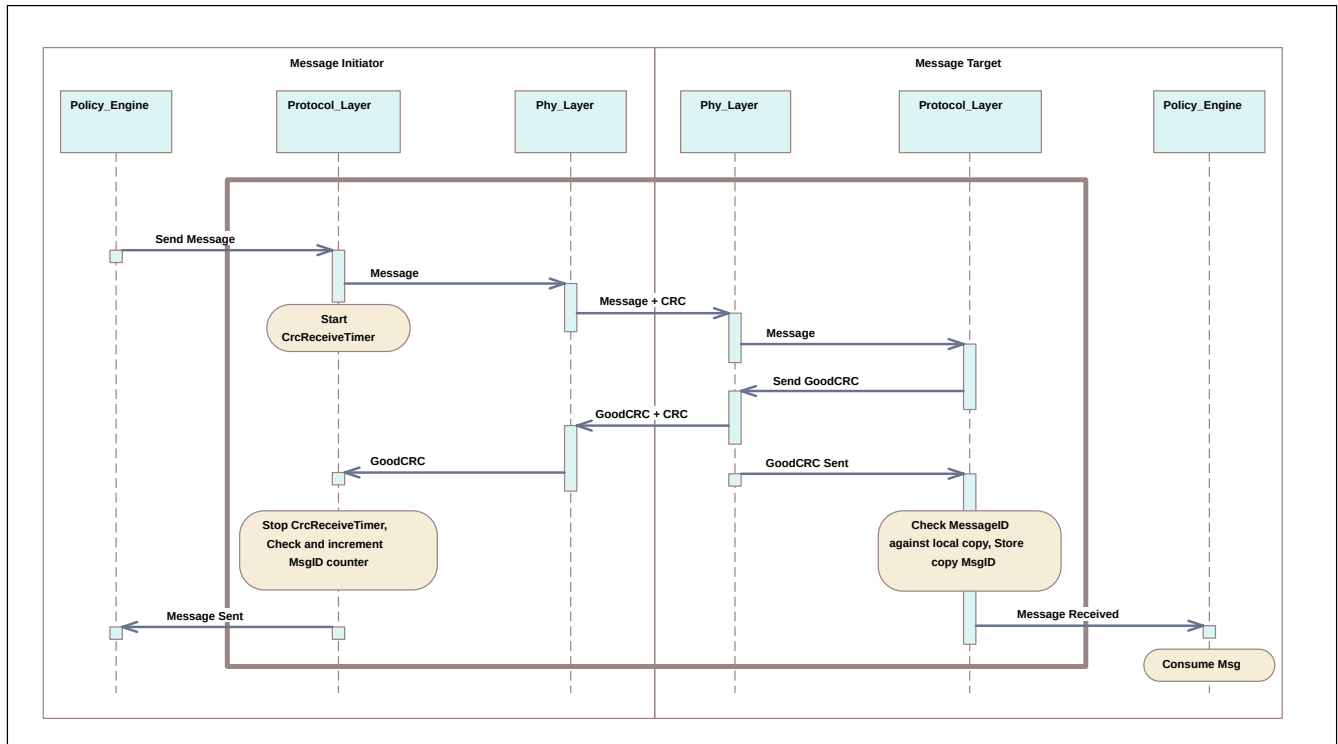
7.6. AMS Usages

7.6.1. Basic Message Exchange

[Figure 7.1](#) illustrates how a Message is sent. Note: The sender might be either a Source or Sink while the receiver might be either a Sink or Source. The basic Message sequence is the same. It starts when the Message Initiator's Policy Engine forms a Message that it passes to the Protocol Layer. The "Protocol to protocol core" sequence fragment defines where:

- the Message Request arrives at the Initiator Protocol Layer on the left side,
- the Request is delivered to the target Policy Engine on the right side, and
- the Message-sent event is delivered to the Initiator Protocol Layer on the left side.

This fragment is used in subsequent diagrams when the Message flow through the protocol and physical layers are identical.

Figure 7.1. Sequence diagram of messaging core between protocol layers

7.6.2. Possible Points of Failure for Message Delivery

There are various points during the Message flow where failures in communication or other issues can occur. [Figure 7.2](#) below is an annotated Version of [Figure 7.1](#) indicating at which points issues can occur.

Figure 7.2. Basic Message flow indicating possible errors

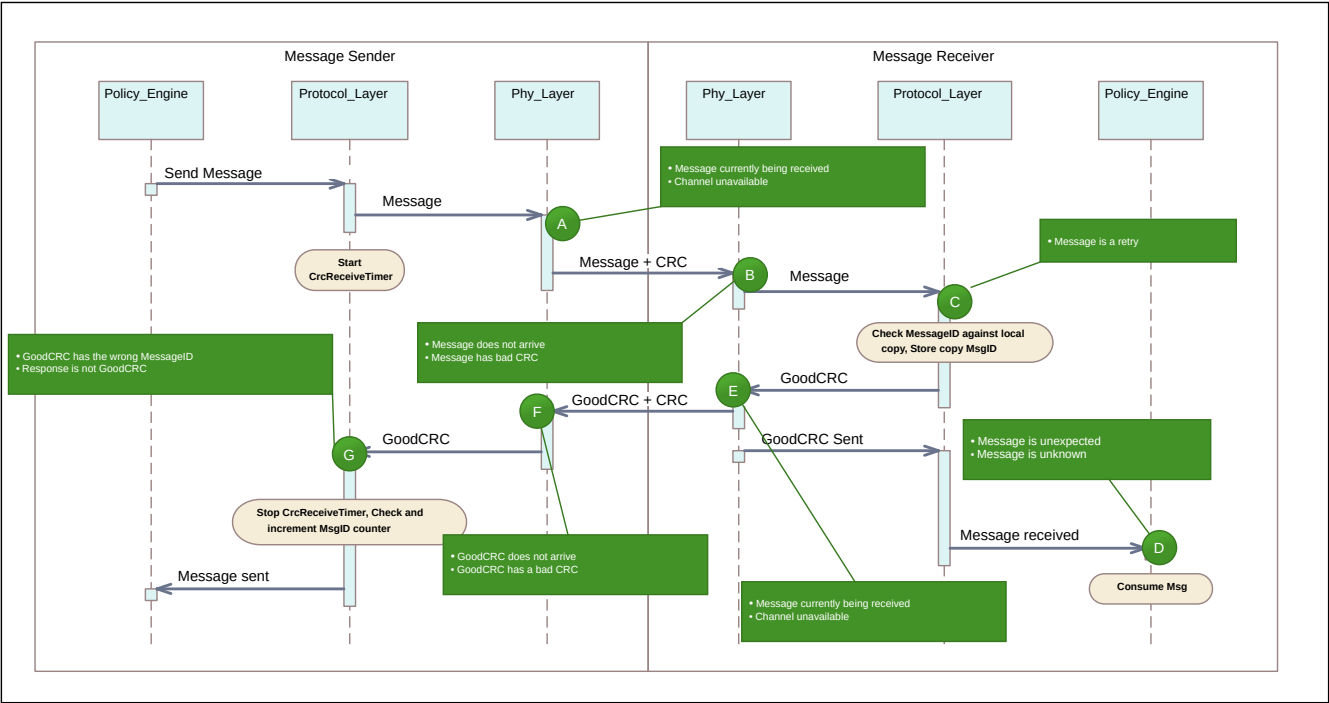


Table 7.8. Potential Issues in the Basic Message Flow

Point	Possible Issues
A	<ol style="list-style-type: none"> 1. There is an incoming Message on the channel, meaning that the PHY Layer is unable to send. In this case the outgoing Message is removed from the queue and the incoming Message processed. 2. Due to some sort of noise on the line it is not possible to transmit. In this case the outgoing Message is Discarded by the PHY Layer. Retransmission is via the Protocol Layer's normal mechanism.
B	<ol style="list-style-type: none"> 1. Message does not arrive at the PHY Layer due to noise on the channel. 2. Message arrives but has been corrupted and has a bad CRC. <p>There is no Message to pass up to the Protocol Layer on the receiver which means a GoodCRC Message is not sent. This leads to a CRCReceiveTimer timeout in the Message Sender.</p>
C	Same as Point A, but at the Message Receiver side.
D	<ol style="list-style-type: none"> 1. GoodCRC Message response does not arrive at the Message Sender side due to the noise on the channel. 2. GoodCRC Message response arrives but has a bad CRC. <p>A GoodCRC Message is not received by the Message Sender's Protocol Layer. This leads to a CRCReceiveTimer timeout in the Message Sender.</p>
E	MessageID of received Message matches stored MessageID so this is a retry. Message is not passed up to the Policy Engine.
F	<ol style="list-style-type: none"> 1. GoodCRC Message is received but does contain the same MessageID as the transmitted Message. 2. A Message is received but it is not a GoodCRC Message (similar case to that of an unexpected or unknown Message but this time detected in the Protocol Layer). <p>Both of these issues indicate errors in receiving an expected GoodCRC Message which will lead to a CRCReceiveTimer timeout in the Protocol Layer and a subsequent retry (except for communications with Cable Plugs).</p>
G	<ol style="list-style-type: none"> 1. Policy Engine receives a known Message that it was not expecting. 2. Policy Engine receives an Unrecognized Message. <p>These cases are errors in the protocol which could lead to the generation of a Soft_Reset Message.</p>

7.7. Soft Reset AMS

Message numbers **Shall** be set to zero prior to sending the [Soft_Reset/Accept Message](#) since the issue might be with the counters. The sender of a [Soft_Reset Message](#) **Shall** reset its MessageIDCounter and RetryCounter to 0. The receiver of the Message **Shall** reset its MessageIDCounter and RetryCounter to 0 before sending the [Accept Message](#) response. A Port Partner observing a Protocol Error during an SOP **Soft Reset** **Shall** initiate a [Hard Reset](#); a DFP observing a Protocol Error during an SOP/SOP" **Soft Reset** **Shall** initiate a [Cable Reset](#) (see [Section 7.1.3](#) and [Section 7.1.4](#)).

7.8. Data Reset AMS

In a [Data Reset AMS](#), the Initiator sends a [Data_Reset Message](#). The Responder will respond with an [Accept Message](#). Both Port partners will then follow the process given in [Section 7.1.2, "Data Reset"](#). When the [Data Reset](#) process is complete, the DFP will complete the AMS by sending a [Data_Reset_Complete Message](#) to the UFP regardless of the role of the Initiator.

7.9. Power Negotiation (SPR) AMS

7.9.1. Source_Capabilities Message

7.9.1.1. Power Data Objects

[Section 6.4.1.1](#) and [Section 6.4.1.3](#) describe the Power Data Objects (PDOs) used in the construction of a Capabilities Message for both SPR Mode and [EPR Mode](#).

Power Data Objects are also used to expose additional Capabilities that **May** be utilized, such as in the case of a [Power Role Swap](#).

A list of one or more Power Data Objects **Shall** be sent by the Source to convey its Capabilities. The Sink **May** then [Request](#) one of these Capabilities by returning a Request Data Object that contains an index to a Power Data Object, to Negotiate a mutually agreeable Explicit Contract.

Where Maximum and Minimum voltage and current values are given in PDOs these **Shall** be taken to be absolute values.

The Source and Sink **Shall Not** Negotiate a power level that would allow the current to exceed the maximum current supported by their receptacles or the Attached plug (see [USB-C]). The Source **Shall** limit its offered Capabilities to the maximum current supported by its receptacle and Attached plug. A Sink **Shall** only make a Request from any of the Capabilities offered by the Source. For further details see [Section 8.6.1.2.2](#).

Sources expose their power Capabilities by sending a [Source_Capabilities Message](#). Sinks expose their power requirements by sending a [Sink_Capabilities Message](#). Both are composed of several 32-bit Power Data Objects (see [Section 7.9.1.1](#)).

The Augmented Power Data Object (APDO) is defined to allow support for more than the four PDO types by extending the Power Data Object field from 2 to 4 bits when the B31...B30 are 11b. The generic APDO structure is shown in [Table 6.7](#)"

7.9.1.2. USB Suspend Supported

Prior to an Explicit Contract or when the USB Communications Capable bit is set to zero, the USB Suspend Supported flag is undefined and Sinks **Shall** follow the rules for suspend as defined in [USB2], [USB 3.2], [USB4], [USB-C] or [USB BC]. After an Explicit Contract has been Negotiated:

- A PDUSB Peripheral **May** draw up to [pSnkSusp](#) during suspend; a PDUSB Hub **May** draw up to [pHubSusp](#) during suspend (see [Section 4.2.3](#)").

Note: When USB is suspended, the USB Device State is also suspended.

Sinks **May** indicate to the Source that they would prefer to have the USB Suspend Supported flag cleared by setting the No USB Suspend flag in a [Request Message](#) (see [Section 6.4.2.1.8](#)).

7.9.1.3. Unconstrained Power

To set the Unconstrained Power bit because of an external Source, the external Source of power **Should** be either:

- An AC Supply, e.g., a Charger, directly Connected to the Sink.
- Or, in the case of a PDUSB Hub:
 - A PD Source with its Unconstrained Power bit set.

- Multiple PD Sources all with their Unconstrained Power bits set.

7.9.1.4. EPR Mode Capable

When this bit is set, an EPR Source:

- Operating in SPR Mode **Shall** only send an [EPR_Source_Capabilities_Message](#) in response to an [EPR_Get_Source_Cap_Message](#)

7.9.1.5. Source Offering No Capabilities

A Source offering no Capabilities (0mA or 0W) **Shall** offer a single Fixed Supply PDO with Voltage set to 5V and Maximum Current set to 0mA. Examples of products that might do this:

- A Dual-Role Power product that offers no Capabilities as a Source or in the absence of external power.
- A Source that wants the Sink to draw no more than pSnkSusp.
- A Sink with Accessory Support that does not support VBUS but is sourcing VCONN to an Accessory (see [USB-C]).

7.9.2. Request Message

A [Request Message](#) **Shall** be sent by a Sink to Request power during the Request phase of an SPR power Negotiation. The Request Data Object **Shall** be returned by the Sink making a Request for power. It **Shall** be sent in response to the most recent [Source_Capabilities_Message](#) when in SPR Mode.

The [Request Message](#) includes the requested power level. For example, if the [Source_Capabilities_Message](#) includes a Fixed Supply PDO that offers 9V @ 1.5A and if the Sink only wants 9V @ 0.5A, it will set the Operating Current field to 50 (i.e., 10mA * 50 = 0.5A).

Note: A Source in AVS Mode, unlike the SPR Source in PPS Mode, does not support Current Limit; the Sink is responsible not to take more current than it requested.

The [Accept Message](#) **Shall** be sent by the Source, in SPR Mode, to signal the Sink that the Source is willing to meet the [Request Message](#).

The [Reject Message](#) **Shall** be sent to signal the Sink, in SPR Mode, that the Source is unable to meet the [Request Message](#). This **May** be due an **Invalid** Request or because the Source can no longer provide what it previously Advertised.

A Source operating in [EPR Mode](#) that receives a [Request Message](#) **Shall** initiate a [Hard Reset](#).

The Sink **May** send a different [Request Message](#) to the one which was rejected but **Shall Not** repeat the same [Request Message](#), using the same RDO, unless there has been a New Explicit Contract Negotiation, [Data Role Swap](#), or [Data Reset](#).

7.9.2.1. Capability Mismatch

Capabilities Mismatch occurs when the Source cannot satisfy the Sink's power requirements based on the offered Source Capabilities. In this case the Sink **Shall** make a **Valid** Request from the offered Source Capabilities and **Shall** set the Capability Mismatch bit. When a Capabilities Mismatch condition does not exist, the Sink **Shall Not** set the Capability Mismatch bit.

When a Sink returns a Request Data Object with the Capability Mismatch bit set in response to a Source Capabilities Message, it indicates that it wants more power than the Source is currently offering. This can be due to a lack of a specific offered voltage or insufficient current for the offered voltages.

Sources whose Port Reported PDP is less than their Port Present PDP (see [Section 6.4.10](#)) **Shall** respond to Requests with the Capability Mismatch bit set by issuing a New Source Capabilities Message within [tCapabilitiesMismatchResponse](#) of the [PS_RDY Message](#) that offers either:

1. The set of Source Capabilities to minimally satisfy the Sink's requirements based on what it actually requires for full operation by evaluating the:
 - i. [Sink_Capabilities_Extended Message](#) (for Sinks > PD2) and/or
 - ii. [Sink_Capabilities Message](#) or [EPR_Sink_Capabilities Message](#).
2. The set of Source Capabilities the Source can supply at this time based on the Port Present PDP.

To prevent looping, Sources **Should Not** send a New Source Capabilities Message in response to subsequent [Request Message](#) with the Capability Mismatch flag set until its Port Present PDP changes.

Once a Guaranteed Capability Source that has responded to a Capability Mismatch, it **Shall Not** subsequently send out another Source Capabilities Message at a lower PDP unless the power required by the Sink (as indicated in its Sink Capabilities Message) has also been reduced. Sources wishing to manage their power **May** periodically check the Sink Capabilities Message to determine whether this has changed.

Note: A Source Capabilities Message refers to a [Source_Capabilities Message](#) or an [EPR_Source_Capabilities Message](#), and a Sink Capabilities Message refers to a [Sink_Capabilities Message](#) or [EPR_Sink_Capabilities Message](#), Request refers to a [Request Message](#) or [EPR_Request Message](#) depending on operating Mode.

7.9.2.2. Wait in Response to Request Message

The [Wait Message](#) allows the Source time to recover the power it requires to meet the Request, e.g., through Re-Negotiation with other Sinks or an upstream Source. A Source **Should** only send a [Wait Message](#) in response to a [Request Message](#) when an Explicit Contract exists between the Port Partners. The Sink is allowed to repeat the [Request Message](#) using the [SinkRequestTimer](#) and **Shall** ensure that at least [tSinkRequest](#) has elapsed after receiving the [Wait Message](#) before sending another [Request Message](#).

7.9.3. PS_Rdy

The [PS_RDY Message](#) **Shall** be sent by the Source (or by both the New Sink and New Source during the [Power Role Swap AMS](#) or [Fast Role Swap AMS](#)) to indicate its power supply has reached the desired operating condition.

7.10. Power Role Swap AMS

The [PR_Swap Message](#) **May** be sent by either Port Partner to Request an exchange of Power Roles. The recipient of the Message **Shall** respond by sending an [Accept Message](#), a [Wait Message](#) or a [Reject Message](#).

- The [Accept Message](#) **Shall** be sent by the recipient of the [PR_Swap Message](#) to signal that it is willing to do a [Power Role Swap](#) and has begun the [Power Role Swap AMS](#). If an [Accept Message](#) is sent, the Source and Sink **Shall** do a [Power Role Swap](#).
- The [Reject Message](#) **Shall** be sent by the recipient of the [PR_Swap Message](#) to signal that it is unable or unwilling to do a [Power Role Swap](#). If a [Reject Message](#) is sent, no action **Shall** be taken.
- For a [Wait Message](#), see section, [Section 7.10.1](#)

The [PR_Swap Message](#) **Shall Not** be sent while in EPR Mode. While in [EPR Mode](#) if a [Power Role Swap](#) is required, an [EPR Mode Exit](#) **Shall** be done first. After a successful [Power Role Swap](#) the Port Partners **Shall** reset their respective Protocol Layers (equivalent to a [Soft Reset](#)): resetting their MessageIDCounter, RetryCounter and Protocol Layer State machines before attempting to establish the First Explicit Contract. At this point the New Source **Shall** also reset its [CapsCounter](#).

The New Source **Shall** have Rp asserted on the CC wire and the New Sink **Shall** have Rd asserted on the CC wire as defined in [USB-C]. When performing a Power Role Swap from Source to Sink, the Port **Shall** change its CC wire resistor from Rp to Rd. When performing a [Power Role Swap](#) from Sink to Source, the Port **Shall** change its CC wire resistor from Rd to Rp. The DFP (Host), UFP (Device) Data Roles and VCONN Source **Shall** remain unchanged by the [Power Role Swap](#) process.

During the [Power Role Swap AMS](#), for the Initial Source Port, the Port Power Role field **Shall** be set to Sink in the [PS_RDY Message](#) indicating that the Initial Source's power supply is turned off.

During the [Power Role Swap AMS](#), for the Initial Sink, the Port Power Role field **Shall** be set to Source for Messages initiated by the Policy Engine after receiving the [PS_RDY Message](#) from the Initial Source.

Note: During the [Power Role Swap](#) process the Initial Sink does not disconnect even though VBUS drops below [vSafe5V](#).

For more information regarding the [Power Role Swap](#), refer to:

[Section 4.3.3, "Power Role Swap Sequence"](#)

[Section 7.1.2, "Data Reset"](#).

[Section 9.2.20.3, "Policy Engine in Source to Sink Power Role Swap State Diagram"](#)

[Section 9.2.20.4, "Policy Engine in Sink to Source Power Role Swap State Diagram"](#)

7.10.1. Wait in response to a PR_Swap Message

The [Wait Message](#) is used when responding to a [PR_Swap Message](#) to indicate that a [Power Role Swap](#) might be possible in the future. This can occur in any case where the Device receiving the [PR_Swap Message](#) needs to evaluate the Request further e.g., by requesting Sink Capabilities from the originator of the [PR_Swap Message](#). Once it has completed this evaluation one of the Port Partners **Should** initiate the [Power Role Swap](#) process again by sending a [PR_Swap Message](#). Once it has completed this evaluation one of the Port Partners **Should** initiate the [Power Role Swap](#) process again by sending a [PR_Swap Message](#).

The [Wait Message](#) is also used where a Hub is operating in hybrid Mode when a Request cannot be satisfied (see [UCSI]). A Port that receives a [Wait Message](#) in response to a [PR_Swap Message](#) **Shall** wait [tPRSwapWait](#) after receiving the [Wait Message](#) before sending another [PR_Swap Message](#).

7.11. Fast Role Swap AMS

See [Chapter 10](#) for details regarding [Fast Role Swap](#).

7.12. Data Role Swap AMS

7.12.1. DR_Swap

The [Data Role Swap](#) process can be used by Port Partners whether or not they support USB Communications capability. A DFP that supports USB Communication capability starts as the USB Host on Attachment. A UFP that supports USB Communication capability starts as the USB Device on Attachment.

[USB-C] Dual-Role Data (DRD) Ports **Shall** have the capability to perform a [Data Role Swap](#) from the PE_SRC_Ready or PE_SNK_Ready states. DFPs and UFPs **May** have the capability to perform a [Data Role Swap](#) from the PE_SRC_Ready or PE_SNK_Ready states. A [Data Role Swap](#) **Shall** be regarded in the same way as a cable Detach/Re-Attach in relation to any USB Communication which is ongoing between the Port Partners. If there are any Active Modes between the Port Partners when a [DR_Swap Message](#) is received, then a [Hard Reset](#) **Shall** be

performed (see [Section 8.6.4](#)"). If the Cable Plug has any Active Modes then the DFP **Shall Not** issue a [DR_Swap Message](#) and **Shall** cause all Active Modes in the Cable Plug to be exited before accepting a [Data Role Swap Request](#). The Source of VBUS and VCONN Source **Shall** remain unchanged as well as the Rp/Rd resistors on the CC wire during the [Data Role Swap](#) process.

The [DR_Swap Message](#) **May** be sent by either Port Partner. The recipient of the [DR_Swap Message](#) **Shall** respond by sending an [Accept Message](#), a [Wait Message](#) or a [Reject Message](#).

- The [Accept Message](#) **Shall** be sent by the recipient of the [DR_Swap Message](#) to signal that it is willing to do a [Data Role Swap](#) and has begun the [Data Role Swap AMS](#). If an [Accept Message](#) is sent, the Source and Sink **Shall** exchange Data Roles.
- The [Reject Message](#) **Shall** be sent by the recipient of the [DR_Swap Message](#) to signal that it is unable or unwilling to do a [Data Role Swap](#). If a [Reject Message](#) is sent, no action **Shall** be taken.
- For a [Wait Message](#), see [Section 7.12.2](#).

Before a [Data Role Swap](#), the initial DFP **Shall** have its Port Data Role bit set to DFP, and the initial UFP **Shall** have its Port Data Role bit set to UFP.

7.12.2. Wait in response to a DR_Swap Message

The [Wait Message](#) is used when responding to a [DR_Swap Message](#) to indicate that a [Data Role Swap](#) might be possible in the future. This can occur in any case where the Device receiving the [DR_Swap Message](#) needs to evaluate the Request further. Once it has completed this evaluation one of the Port Partners **Should** initiate the Data Role Swap process again by sending a [DR_Swap Message](#).

A Port that receives a [Wait Message](#) in response to a [DR_Swap Message](#), **Shall** wait [tDRSwapWait](#) after receiving the [Wait Message](#) before sending another [DR_Swap Message](#).

7.13. VCONN Swap AMS

The [VCONN_Swap Message](#) **Shall** be supported by any Port that can operate as a VCONN Source.

The [VCONN_Swap Message](#) **May** be sent by either Port Partner to Request an exchange of VCONN Source. The recipient of the Message **Shall** respond by sending an [Accept Message](#), [Reject Message](#) (see [Section 7.4](#)), [Wait Message](#) (see [Section 7.13.1](#)), or [Not_Supported Message](#).

- The [Accept Message](#) **Shall** be sent by the recipient of the [VCONN_Swap Message](#) to signal that it is willing to do a [VCONN Swap](#) and has begun the [VCONN Swap AMS](#). The new VCONN Source **Shall** send a [PS_RDY Message](#) within [tVconnSourceOn](#) to indicate that it is now sourcing VCONN. The initial VCONN Source **Shall** cease sourcing VCONN within [tVconnSourceOff](#) of receipt of the last bit of the EOP of the [PS_RDY Message](#).
- The [Reject Message](#) **Shall** be sent by the recipient of a [VCONN_Swap Message](#), that is not presently the VCONN Source, to indicate that it is unable to do a [VCONN Swap](#), and no action **Shall** be taken. The Port that is presently the VCONN Source **Shall Not** send a [Reject Message](#) in response to a [VCONN_Swap Message](#).
- For a [Wait Message](#), see [Section 7.13.1](#).
- If a [Not_Supported Message](#) is sent, the requester is informed that [VCONN Swap](#) is not supported. The Port that is not presently the VCONN Source **May** turn on VCONN when a [Not_Supported Message](#) is received in response to a [VCONN_Swap Message](#).

The DFP (Host), UFP (Device) Data Roles and Source of VBUS **Shall** remain unchanged as well as the Rp/Rd resistors on the CC wire during the [VCONN Swap](#) process.

VCONN **Shall** be continually sourced during the [VCONN Swap](#) process to maintain power to the Cable Plug(s) i.e., make before break.

Before communicating with a Cable Plug a Port **Shall** ensure that it is the VCONN Source and that the Cable Plugs are powered, by performing a [VCONN Swap](#) if necessary. Since it cannot be guaranteed that the present VCONN Source is supplying VCONN, the only means to ensure that the Cable Plugs are powered is for a Port wishing to communicate with a Cable Plug to become the VCONN Source. If a [Not_Supported_Message](#) is returned in response to the [VCONN_Swap_Message](#), then the Port is allowed to become the VCONN Source until a [Hard_Reset](#) or Detach.

A VCONN Source that is also a Source can attempt to send a [Discover_Identity](#) Command using SOP' to a Cable Plug prior to the establishment of the First Explicit Contract.

Note: Even when it is currently the VCONN Source, the Sink is not permitted to initiate an AMS with a Cable Plug unless Rp is set to [SinkTxOK](#).

7.13.1. Wait in response to a VCONN_Swap Message

The [Wait_Message](#) is used when responding to a [VCONN_Swap_Message](#) to indicate that a [VCONN_Swap](#) might be possible in the future. This can occur in any case where the Device receiving the [VCONN_Swap_Message](#) needs to evaluate the Request further. Once it has completed this evaluation, one of the Port Partners **Should** initiate the VCONN Swap process again by sending a [VCONN_Swap_Message](#).

A Port that receives a [Wait_Message](#) in response to a [VCONN_Swap_Message](#) **Shall** wait [tVCONNSwapWait](#) after receiving the [Wait_Message](#) before sending another [VCONN_Swap](#) Message.

A Port that is currently the VCONN Source **Shall** respond with an [Accept_Message](#) (rather than a [Wait_Message](#)) if the Port Partner's Revision and Version, as reported in the [Revision_Message](#), is earlier than R3.2 V1.1. A Port Partner supporting an earlier Revision and Version will not expect a [Wait_Message](#) and will generate a Soft Reset in response.

7.14. Alert AMS Message

The [Alert_Message](#) is provided to allow Port Partners to inform each other when there is a status change event. Some of the events are critical such as OCP, OVP and OTP, while others are informational such as a change in a Battery's status from charging to either charging or discharging.

The [Alert_Message](#) **Shall** only be sent when the Source or Sink detects a status change.

7.14.1. Alert Types

Multiple event bits **May** be set in one [Alert_Message](#). Once the [Alert_Message](#) has been sent the event bits **Shall** be cleared.

A [Get_Status_Message](#) **Should** be sent in response to a non-Battery status change in an [Alert_Message](#) to get the details of the change. A [Get_Battery_Status_Message](#) **Should** be sent in response to a Battery status change in an [Alert_Message](#) to get the details of the change.

7.15. Get Status AMS

An [Alert_Message](#) indicates that the Sender's status has changed, and the Receiver **Should** re-read it using a [Get_Status_Message](#).

The Active Cable **Shall** respond to the [Get_Status_Message](#) by returning an SOP'/SOP" [Status_Message](#) (see [Table 6.52](#)).

To make sure that the internal temperature value in the [Status Message](#) is up to date, the Cable Plug **Shall** update its internal temperature at least every [tACTempUpdate](#).

7.16. Get PPS Status AMS

The Sink sends the [Get_PPS_Status Message](#) to Request information about the Source's PPS status. The Source **Shall** respond with a [PPS_Status Message](#). A Source that supports this AMS **Shall** support sending the [Alert Message](#). A Sink that supports this AMS **Shall** support receiving the [Alert Message](#).

7.17. Get Source Capabilities (SPR) AMS

The Port **Shall** respond by returning a [Source_Capabilities Message](#).

Sinks with the [Dual-Role Power](#) bit set, **Shall** respond to a [Get_Source_Cap Message](#) by declaring their Source Capabilities, without limiting them based on the cable's Capabilities.

7.18. Get Sink Capabilities (SPR) AMS

The Port **Shall** respond by returning a [Sink_Capabilities Message](#).

7.19. Extended Capabilities AMS

7.19.1. Get_Source_Cap_Extended

The Port **Shall** respond by returning a [Source_Capabilities_Extended Message](#).

7.19.2. Source_Capabilities_Extended

The [Source_Capabilities_Extended Message](#) **Should** be sent in response to a [Get_Source_Cap_Extended Message](#).

7.19.2.1. Touch Current Field

The Touch Current field reports whether the Source meets certain leakage current levels and if it has a ground pin.

A Source **Shall** set the Touch Current bit (bit 0) when their leakage current is less than 65µA rms when Source's maximum capability is less than or equal to 30W, or when their leakage current is less than 100 µA rms when its power capability is between 30W and 100W. The total combined leakage current **Shall** be measured in accordance with [IEC 60950-1] when tested at 250VAC rms at 50 Hz.

7.19.2.2. Peak Current Field

The Peak Current1/Peak Current2/Peak Current3 fields **Shall** contain the combinations of Peak Current that the Source supports (see [Section 4.1.6](#)).

Peak Current provides a means for Source report its ability to provide current in excess of the Negotiated amount for short periods. The Peak Current descriptor defines up to three combinations of percent overload, duration and duty cycle defined as Peak Current1, Peak Current2, and Peak Current3 that the Source supports. A Source **May** offer no Peak Current capability. A Source **Shall** populate unused Peak Current bit fields with zero.

The Bit Fields within Peak Current1, Peak Current2 and Peak Current3 contain the following subfields:

- Duty Cycle

- Duty Cycle **Shall** be the maximum percentage of overload period reported in 5% increments. The values **Should** be 5%, 10% and 50% for PeakCurrent1, PeakCurrent2, and PeakCurrent3, respectively.
- VBUS Droop
 - **Shall** be set to '1' to indicate there is an additional 5% voltage droop on VBUS when the overload conditions occur as defined by [vSrcPeak](#). However, it is recommended that the Source **Should** provide VBUS in the range of [vSrcNew](#) when overload conditions occur and set this bit to '0'.

7.19.2.3. Number of Batteries/Battery Slots Field

The number assigned to a given Battery Slot **Shall Not** change between Attach and Detach.

7.19.3. Get_Sink_Cap_Extended

The Port **Shall** respond by returning a [Sink_Capabilities_Extended Message](#).

7.19.4. Sink_Capabilities_Extended

The [Sink_Capabilities_Extended Message](#) **Shall** be sent in response to a [Get_Sink_Cap_Extended Message](#).

7.19.4.1. XID Field

If the vendor does not have an XID, then it **Shall** return zero in this field (see [USB2] and [USB3]).

7.19.4.2. Battery Info

Fixed Batteries **Shall** be numbered consecutively from 0 to 3. The number assigned to a given Fixed Battery **Shall Not** change between Attach and Detach. Battery Slots **Shall** be numbered consecutively from 4 to 7. The number assigned to a given Battery Slot **Shall Not** change between Attach and Detach.

7.19.4.3. Sink Modes

Bits 1-4 **May** be set independently of one another. The combination indicates what sources of power the Sink can utilize. For example, some Sinks are only powered by a Battery (e.g., an automobile Battery) rather than the more common AC Supply and some Sinks are only powered from VBUS or VCONN.

7.20. Battery Capabilities AMS

7.20.1. Get_Battery_Cap Message

The Port **Shall** respond by returning a [Battery_Capabilities Message](#) containing a [Battery Capabilities](#) Data Block (BCDB) for the targeted Battery.

7.21. Battery Status AMS

7.21.1. Get_Battery_Status Message

The Port **Shall** respond by returning a [Battery_Status Message](#) containing a [Battery Status](#) Data Object (BSDO) for the targeted Battery.

7.21.2. Battery Status Message

The [Battery_Status Message](#) **Shall** be sent in response to a [Get_Battery_Status Message](#).

7.22. Manufacturer Information AMS

7.22.1. Get_Manufacturer_Info Message

The Port **Shall** respond by returning a [Manufacturer_Info Message](#) containing a Manufacturer Info Data Block (MIDB). Support for this feature by the Cable Plug is **Optional**.

7.22.2. Manufacturer_Info Message

The [Manufacturer_Info Message](#) **Shall** be sent in response to a [Get_Manufacturer_Info Message](#).

7.23. Country Codes AMS

7.23.1. Get_Country_Codes Message

The Port Partner **Shall** respond by returning a [Country_Codes Message](#).

7.23.2. Country_Codes Message

The [Country_Codes Message](#) **Shall** be sent in response to a [Get_Country_Codes Message](#).

7.24. Country Information AMS

7.24.1. Country_Info Message

The [Country_Info Message](#) **Shall** be sent in response to a [Get_Country_Info Message](#).

7.25. Revision Information AMS

7.25.1. Get_Revision Message

The Port Partner **Shall** respond by returning a [Revision Message](#).

The Active Cable **Shall** respond by returning a [Revision Message](#).

7.25.2. Revision Message

The [Revision Message](#) **Shall** be sent in response to the [Get_Revision Message](#) sent by the Port Partner.

7.26. Source Information AMS

7.26.1. Get_Source_Info Message

The Port **Shall** respond by returning the [Source_Info Message](#).

7.26.2. Source_Info Message

The [Source_Info Message](#) **Shall** be sent in response to a [Get_Source_Info Message](#).

7.26.2.1. Port Maximum PDP Field

A Guaranteed Capability Port (as indicated by the Port Type field being set to '1') **Shall** always be capable of supplying this amount of power. A Managed Capability Port (as indicated by the Port Type field being set to '0') **Shall** be able to offer this amount of power at some time.

The Port Maximum PDP **Shall** be the same as the larger of the SPR Source PDP Rating and the EPR Source PDP Rating in the [Source_Capabilities_Extended Message](#).

7.26.2.2. Port Present PDP Field

A Guaranteed Capability Port **Shall** always set its Port Present PDP to be the same as its Port Maximum PDP or the highest possible value when limited.

A Managed Capability Port that is a Shared Capacity Group **Shall** set its Port Present PDP to Shared Port Power Available as defined in [USB-C] or to a lower value when limited.

A Managed Capability Port, that is an Assured Capacity Port, **Shall** set its Port Present PDP to the Port Maximum PDP or the highest value possible when limited.

7.26.2.3. Port Guaranteed PDP Field

Managed Capability Ports may offer less power than the value defined in the Port Guaranteed PDP but **Shall** be able to provide at least this power level once the Sink initiates the process of requesting more power by setting the mismatch bit, and following the process indicated in [Section 7.9.2.1](#). While there is an Explicit Contract for Port Guaranteed PDP, the Source **Shall Not** reduce the advertised power.

The only scenario where the Source **May** not be able to reach Port Guaranteed PDP is if the Cable Capabilities limit the current.

7.27. Security AMS

The authentication process between Port Partners or a Port and Cable Plug is fully described in [USBC Auth]. This specification describes two Extended Messages used by the authentication process when applied to PD. In the authentication process described in [USBC Auth] there are three basic exchanges that serve to:

- Get the Port or Cable Plug's certificates.
- Get the Port or Cable Plug's digest.
- Challenge the Port Partner or Cable Plug.

Certificates are used to convey information, attested to by a signer, which attests to the Port Partner's or Cable Plug's authenticity. The Port's or Cable Plug's certificates are needed when a Port encounters a Port Partner or Cable Plug it has not been Attached to before. To minimize calculations after the initial Attachment, a Port can also use a digest consisting of hashes of the certificates rather than the certificates themselves. Once the Port has the certificates and has calculated the hashes, it stores the hashes and uses the digest in future exchanges. After the Port gets the certificates or digest, it challenges its Port Partner or the Cable Plug to detect replay attacks. For further details refer to [USBC Auth].

7.28. Firmware Update AMS

The firmware update process between Port Partners or a Port and Cable Plug is fully described in [USB DFU]. This specification describes two Extended Messages used by the firmware update process when applied to PD.

7.29. Enter USB AMS

The recipient of the Message **Shall** respond by sending an [Accept Message](#), a [Wait Message](#) or a [Reject Message](#).

- The [Accept Message](#) **Shall** be sent by the recipient of the [Enter_USB Message](#) to indicate that it has begun the [Enter USB AMS](#).
- The [Reject Message](#) **Shall** be sent by UFP on receiving an [Enter_USB Message](#) to indicate it is unable to enter the requested USB Mode.
- For a [Wait Message](#), see [Section 7.29.1](#).

When entering [USB4] operation, the [Enter_USB Message](#) **Shall** be sent by a [USB4] PDUSB Hub's DFP(s) or [USB4] PDUSB Host's DFP(s) within [tEnterUSB](#):

- following a PD Connection.
- after a [Data Reset](#) to enter [USB4] operation is completed.
- after a [Data Role Swap](#) is completed.

The [Enter_USB Message](#) **May** be sent by a PDUSB Hub's DFP(s) or PDUSB Host's DFP(s) within [tEnterUSB](#) following a PD Connection or after a [Data Reset](#) to enter [USB3] or [USB2] operation. The [Enter_USB Message](#) **Shall** be used by a PDUSB Hub's DFP(s) to speculatively train the USB links or enter [DPTC] or [TBT3] Alternate Modes prior to the presence of a Host. In this case, the Host Present bit **Shall** be cleared. When the Host is Connected the [Enter_USB Message](#) **Shall** be resent with the Host Present bit set. The [Enter_USB Message](#)'s Enter USB Data Object (EUDO), received from the Root Hub when the USB Host is Connected, **Shall** be propagated down through the Hub tree. See [USB-C] [USB4] Hub Connection Requirements.

7.29.1. Wait in Response to an Enter_USB Message

The [Wait Message](#) is used, by the UFP, when responding to an [Enter_USB Message](#) to indicate that entering the requested USB Mode might be possible in the future. This can occur, for example, in any case where the UFP needs to Negotiate more power to enter the Mode. Once the UFP has completed this the DFP **Should** initiate the [Enter_USB](#) process again by sending an [Enter_USB Message](#). A DFP that receives a [Wait Message](#) in response to an [Enter_USB Message](#) **Shall** wait [tEnterUSBWait](#) after receiving the [Wait](#) Message before sending another [Enter_USB Message](#).

7.30. EPR Mode AMS

7.30.1. Process to Enter EPR Mode AMS

An EPR Source **Shall** enter [EPR Mode](#) upon Request by an EPR Sink Connected with an EPR Cable when able to offer the Source Capabilities as defined in the Power Rules ([Table 3.5](#)" and [Table 3.6](#)).

For Port Partners to successfully enter [EPR Mode](#), the following conditions must be met:

- The Sink **Shall** Request entry into the [EPR Mode](#).
- The Source **Shall** verify the cable is EPR Capable.
- A Sink **Shall Not** be Connected to the Source through a Charge Through VPD (CT-VPD).
- The Source and Sink **Shall** already be in an SPR Explicit Contract.
- The EPR Capable bit **Shall** be set in the Fixed Supply 5V PDO.

- The EPR Capable bit **Shall** have been set in the RDO in the last [Request Message](#) received by the Source.

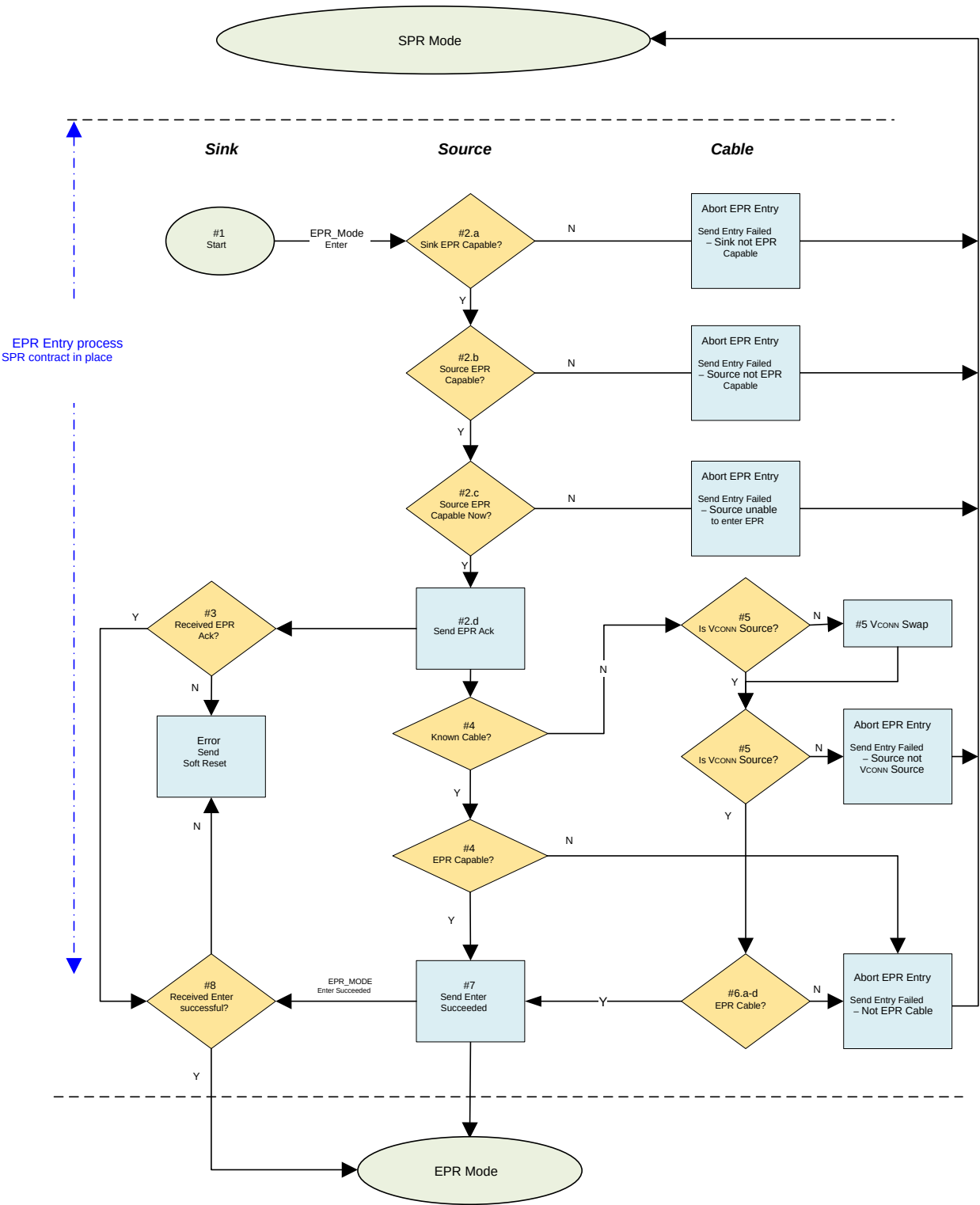
To verify the cable is EPR Capable, the EPR Source **Shall** have already done the following (see [Section 7.31.18.3](#)):

- Discover the cable prior to entering its First Explicit Contract
- Alternatively, within [tEPRSourceCableDiscovery](#) of entry into the First Explicit Contract
 - If it is the VCONN Source, discover the cable.
 - If not the VCONN Source, do a [VCONN Swap](#) then discover the cable and verify the cable is EPR Capable by completing steps 5 and 6 in the entry process in [Figure 7.3](#).

The [EPR Mode](#) Entry process is a multi-Message AMS. An illustration of this AMS is shown in [Figure 7.3](#).

Figure 7.3. Illustration of process to enter EPR Mode.

The [EPR Mode](#) Entry process is illustrated for **Informative** purposes.



The entry process **Shall** follow these steps in order:

1. The Sink **Shall** send the [EPR_Mode Message](#) with the Action field set to 1 (Enter) and the Data field set to its Operational PDP. If the EPR Source receives an [EPR_Mode Message](#) with the Action field not set to Enter it **Shall** initiate a Soft Reset.
2. The Source **Shall** do the following:
 - a. Verify the EPR Capable bit was set in the most recent RDO. If not set, the Source **Shall** do the following:
 - i. Send an [EPR_Mode Message](#) with the Action field set to 4 (Enter Failed) and the Data field set to 3 ("EPR Mode Capable bit not set in the RDO").
 - ii. Abort the [EPR_Mode](#) Entry process and remain in the existing SPR Explicit Contract.
 - b. Verify the EPR Capable bit was set in the most recent 5V Fixed Supply PDO. If not set, the Source **Shall** do the following:
 - i. Send an [EPR_Mode Message](#) with the Action field set to 4 (Enter Failed) and the Data field set to 5 ("EPR Mode Capable bit not set in the Fixed Supply 5V PDO").
 - ii. Abort the [EPR_Mode](#) Entry process and remain in the existing SPR Explicit Contract.
 - c. Verify the Source is still able to support [EPR_Mode](#). If not, the Source **Shall** do the following:
 - i. Send an [EPR_Mode Message](#) with the Action field set to 4 (Enter Failed) and Data field set to 4 ("Unable at this time").
 - ii. Abort the [EPR_Mode](#) Entry process and remain in the existing SPR Explicit Contract. The Sink **May** at some time in the future send another Request to enter [EPR_Mode](#).
 - d. Send an [EPR_Mode Message](#) with the Action field set to 2 (Enter Acknowledged).
3. If the Sink receives any Message, other than an [EPR_Mode Message](#) with the Action Field set to 2, the Sink **Shall** initiate a [Soft Reset](#).
4. When the EPR Source has used the [Discover Identity](#) Command to determine and remembers the Cable Capabilities or the EPR Source is Connected with a captive cable:
 - a. If the cable is EPR Capable it **Should** go directly to Step 7, but **May** continue to Step 5.
 - b. If the cable is not EPR Capable it **Shall** do the following:
 - c. Send an [EPR_Mode Message](#) with the Action field set to 4 (Enter Failed) and the Data field set to 1 ("Cable not EPR Capable").
 - d. Abort the [EPR_Mode](#) Entry process and remain in the existing SPR Explicit Contract.
5. If the Source is not the VCONN Source, it **Shall** send a [VCONN_Swap](#) Message.
 - a. If the Source fails to become the VCONN Source, it **Shall**:
 - i. Send an [EPR_Mode Message](#) with the Action field set to 4 (Enter Failed) and the Data field set to 2 (not VCONN Source).
 - ii. Abort the [EPR_Mode](#) Entry process and remain in the existing SPR Explicit Contract.
6. The Source **Shall** use the [Discover Identity](#) Command to read the cable's e-Marker and verify the following:
 - a. Cable VDO - Maximum VBUS Voltage (Passive Cable)/Maximum VBUS Voltage (Active Cable) field is 11b (50V)
 - b. Cable VDO - VBUS Current Handling Capability (Passive Cable)/VBUS Current Handling Capability (Active Cable) field is 10b (5A)

- c. Cable VDO - EPR Capable (Passive Cable)/EPR Capable (Active Cable) field is 1b (EPR Capable)
- d. If the cable fails to respond to the [Discover Identity](#) Command or is not EPR Capable, the Source **Shall** do the following:
 - i. Send an [EPR_Mode Message](#) with the Action field set to 4 (Enter Failed) and the Data field to ("Cable not EPR Capable").
 - ii. Abort the [EPR_Mode](#) Entry process and remain in the existing SPR Explicit Contract.
- 7. The Source **Shall** send the [EPR_Mode Message](#) with the Action field set to 3 (Enter Succeeded) and **Shall** enter EPR Mode.
- 8. If the Sink receives an [EPR_Mode Message](#) with the Action field set to 3 (Enter Succeeded) it **Shall** enter [EPR_Mode](#), otherwise it **Shall** initiate a [Soft Reset](#).

If the [EPR_Mode](#) Entry process successfully completes within [tEnterEPR](#) of the last bit of the [GoodCRC Message](#) sent in response to the [EPR_Mode Message](#) with the Action field set to 1 (Enter), the Source **Shall** send an [EPR_Source_Capabilities Message](#) within [tFirstSourceCap](#).

If the [EPR_Mode](#) Entry process has not been aborted or does not complete within [tEnterEPR](#) of the last bit of the [GoodCRC Message](#) sent in response to the [EPR_Mode Message](#) with the Action field set to 1 (Enter), the Sink **Shall** initiate a Soft Reset.

7.30.2. EPR_Source_Capabilities Message

An EPR Source **Shall** send the [EPR_Source_Capabilities Message](#):

- When entering [EPR_Mode](#)
- While in EPR Modes when its Capabilities change
- In response to an [EPR_Get_Source_Cap Message](#)
- After a [Soft Reset](#) while in [EPR_Mode](#)

An EPR Sink operating in [EPR_Mode](#) **Shall** evaluate every [EPR_Source_Capabilities Message](#) it receives and **Shall** respond with a [EPR_Request Message](#). If its power consumption exceeds the Source Capabilities, it **Shall** Re-Negotiate so as not to exceed the Source's most recently Advertised Source Capabilities.

While operating in SPR Mode, an EPR Sink receiving an [EPR_Source_Capabilities Message](#) in response to an [EPR_Get_Source_Cap Messages](#) **Shall Not** respond with an [EPR_Request Message](#).

The PDOs or APDOs in an [EPR_Source_Capabilities Message](#) **Shall** only be requested using the [EPR_Request Message](#) and only when in [EPR_Mode](#).

When Source wants to exit [EPR_Mode](#), if not already in power Contract with an SPR PDO or APDO, it **Shall** send an [EPR_Source_Capabilities Message](#) with no EPR PDOs or APDOs (i.e. seven SPR PDOs or APDOs including any zero padded ones). See [Section 6.5.18](#).

7.30.3. EPR_Request Message

An [EPR_Request Message](#) **Shall** be sent by a Sink, operating in [EPR_Mode](#), to Request power, typically during the Request phase of a power Negotiation. The [EPR_Request Message](#) **Shall** be sent in response to the most recent [EPR_Source_Capabilities Message](#).

The Source **Shall** verify the PDO in the [EPR_Request Message](#) exactly matches the PDO in the latest [EPR_Source_Capabilities Message](#) pointed to by the Object Position field in the RDO.

The Source **Shall** respond to an [EPR_Request Message](#) in the same manner as it responds to a [Request Message](#) with an [Accept Message](#), a [Reject Message](#), or a [Wait Message](#). The Explicit Contract Negotiation process for EPR is the same as the process for SPR Mode except that the [Source_Capabilities Message](#) is replaced by the [EPR_Source_Capabilities Message](#) and the [Request Message](#) is replaced by the [EPR_Request Message](#).

- The [Accept Message](#) **Shall** be sent by the Source, in [EPR Mode](#), to signal the Sink that the Source is willing to meet the [EPR_Request Message](#).
- The [Reject Message](#) **Shall** be sent to signal the Sink, in [EPR Mode](#), that the Source is unable to meet the [EPR_Request Message](#). This **May** be due an **Invalid** Request or because the Source can no longer provide what it previously Advertised. A mismatch between the requested and advertised PDOs is an example of an **Invalid** Request.
- The [Wait Message](#) **May** be sent by the Source, in [EPR Mode](#), to signal the Sink that the Source is currently unable to meet the [EPR_Request Message](#).

The [Wait Message](#) **Shall** be sent to signal the Sink, in response to a [EPR_Request Message](#) in [EPR Mode](#) during Negotiation, to indicate that the Source is currently unable to meet the Request.

7.30.4. EPR_Sink_Capabilities Message

The EPR Sink **Shall** only send the [EPR_Sink_Capabilities Message](#) in response to an [EPR_Get_Sink_Cap Message](#).

7.30.5. Operation in EPR Mode

While operating in [EPR Mode](#), the Source **Shall** only send [EPR_Source_Capabilities Messages](#) to Advertise its power Capabilities and the Sink **Shall** only respond with [EPR_Request Message](#) Messages to Negotiate Explicit Contracts. The [EPR_Request Message](#) **May** be for either an SPR or EPR PDO or APDO.

If the Source sends a [Source_Capabilities Message](#), that is not in response to a Sink [Get_Source_Cap Message](#), the Sink **Shall** initiate a [Hard Reset](#). If the Sink sends a [Request Message](#), the Source **Shall** initiate a [Hard Reset](#).

The Source **Shall** monitor the CC communications path to ensure that there is periodic traffic. The Sink **Shall** send an [EPR_Keep_Alive Message](#) when it has not sent any Messages for more than [tSinkEPRKeepAlive](#) to ensure there is timely periodic traffic. If there is no traffic for more than [tSourceEPRKeepAlive](#), the Source **Shall** initiate a [Hard Reset](#).

A [PR_Swap](#) is not allowed while in EPR Mode. The recipient of a [PR_Swap Message](#) **Shall** respond with a [Reject Message](#).

7.30.6. EPR_Keep_Alive Message

The Source operating on [EPR Mode](#) responds by returning an [EPR_Keep_Alive_Ack Message](#) to the Sink.

See [Section 7.31.18.1](#) and [Section 7.31.18.2](#) for more information on the [EPR_Keep_Alive Message](#).

7.30.7. Exiting EPR Mode

7.30.7.1. Commanded Exit

While in [EPR Mode](#), either the Source or Sink **May** exit [EPR Mode](#) by sending an [EPR_Mode Message](#) with the Action field set to 5 (Exit).

The ports **Shall** be in an Explicit Contract with an SPR PDO or APDO prior to the EPR Mode Exit process by either:

- The Source sending an [EPR_Source_Capabilities Message](#) with no EPR PDO or APDO s (e.g., only SPR PDOs or APDOs) or

- The Sink negotiating a new Explicit Contract with bit 31 in the RDO set to zero (e.g., only SPR PDOs or APDOs)).

The process to exit [EPR Mode](#) is a multi-Message AMS and **Shall** follow these steps in order:

1. The Port Partners **Shall** be in an Explicit Contract with an SPR PDO or APDO.
2. Either the Source or Sink **Shall** send an [EPR_Mode Message](#) with the Action field set to 5 (Exit) to exit the [EPR Mode](#).
3. The Source **Shall** send a [Source_Capabilities Message](#) within [tFirstSourceCap](#) of the [GoodCRC Message](#) in response to the [EPR_Mode Message](#) with the Action field set to 5 (Exit).
4. If the Sink does not receive a [Source_Capabilities Message](#) within [tTypeCSinkWaitCap](#) of the last bit of the [GoodCRC Message](#) in response to the [EPR_Mode Message](#) with the Action field set to 5 (Exit), Sink **Shall** initiate a [Hard Reset](#).

7.30.7.2. Implicit Exit

[EPR Mode](#) **Shall** be exited as the side-effect of the [Fast Role Swap](#) process. This is because at the end of this process VBUS will be at [vSafe5V](#) and the Ports will be in an Implicit Contract. The New Source will then send a [Source_Capabilities Message](#) (not an [EPR_Source_Capabilities Message](#)) to begin the process of negotiating an SPR Explicit Contract. Once an SPR Explicit Contract is entered, the Source and Sink can then enter [EPR Mode](#) if needed.

7.30.7.3. Exits due to errors

Other critical errors can occur while in [EPR Mode](#); these errors **Shall** result in [Hard Reset](#) being initiated by the Port that detects the error. Some of these errors include:

- An [EPR_Mode Message](#) with the Action field set to 5 (Exit) to exit [EPR Mode](#) is received by a Port in an Explicit Contract with an EPR PDO or APDO.
- The Sink receives an [EPR_Source_Capabilities Message](#) with an EPR PDO or APDO in any of the first seven object positions.
- The PDO or APDO in the [EPR_Request Message](#) does not match the PDO or APDO in the latest [EPR_Source_Capabilities Message](#) pointed to by the Object Position field in the RDO.
- The Source receives a [Request Message](#).
- The Sink receives a [Source_Capabilities Message](#) not in response to a [Get_Source_Cap Message](#).

7.30.8. EPR_Get_Source_Cap Message

A Port that can operate as an EPR Source **Shall** respond by returning an [EPR_Source_Capabilities Message](#).

A Port that does not support [EPR Mode](#) as a Source **Shall** have the following behavior:

- If the Port supports [EPR Mode](#) as a Sink and is a DRP Port, it **Shall** return a [Reject Message](#) or [Not_Supported Message](#).
- If the Port does not support [EPR Mode](#) as a Sink or is not a DRP Port, it **Shall** return a [Not_Supported Message](#).

An EPR Capable Sink Port that is operating in SPR Mode **Shall** treat the [EPR_Source_Capabilities Message](#) as informational only and **Shall Not** respond with an [EPR_Request Message](#).

7.30.9. EPR_Get_Sink_Cap Message

A Port that is EPR Capable operating as a Sink **Shall** respond by returning an [EPR_Sink_Capabilities Message](#).

A Port that does not support [EPR Mode](#) as a Sink **Shall** behave as follows:

- If the Port supports [EPR Mode](#) as a Source and is a DRP Port, it **Shall** return a [Reject Message](#) or [Not_Supported Message](#).
- If the Port does not support [EPR Mode](#) as a Source or is not a DRP Port, it **Shall** return a [Not_Supported Message](#).

7.31. Timers

All the following timers are defined in terms of bits on the bus regardless of where they are implemented in terms of the logical architecture. This is to ensure a fixed reference for the starting and stopping of timers. It is left to the implementer to ensure that this timing is observed in a real system.

7.31.1. CRCReceiveTimer

The [CRCReceiveTimer](#) **Shall** be used by the sender's Protocol Layer to ensure that a Message has not been lost. Failure to receive an acknowledgment of a Message (a [GoodCRC Message](#)) whether caused by a bad [GoodCRC Message](#) on the receiving end or by a garbled Message within [tReceive](#) is detected when the [CRCReceiveTimer](#) expires. The sender's Protocol Layer response when a [CRCReceiveTimer](#) expires **Shall** be to retry [nRetryCount](#) times.

Note: Cable Plugs do not retry Messages and large [Extended Messages](#) that are not Chunked are not retried (see [Section 7.32.4](#)). Sending of the Preamble corresponding to the retried Message **Shall** start within [tRetry](#) of the [CRCReceiveTimer](#) expiring.

The [CRCReceiveTimer](#) **Shall** be started when the last bit of the Message EOP has been transmitted by the PHY Layer. The [CRCReceiveTimer](#) **Shall** be stopped when the last bit of the EOP corresponding to the [GoodCRC Message](#) has been received by the PHY Layer. The Protocol Layer receiving a Message **Shall** respond with a [GoodCRC Message](#) within [tTransmit](#) in order to ensure that the sender's [CRCReceiveTimer](#) does not expire. The [tTransmit](#) time **Shall** be measured from when the last bit of the Message EOP has been received by the PHY Layer until the first bit of the Preamble of the [GoodCRC Message](#) has been transmitted by the PHY Layer.

7.31.2. SenderResponseTimer

The [SenderResponseTimer](#) **Shall** be used by the sender's Policy Engine to ensure that a Message requesting a response (e.g., [Get_Source_Cap Message](#)) is responded to within a bounded time of [tSenderResponse](#). Failure to receive the expected response is detected when the [SenderResponseTimer](#) expires. For [Extended Messages](#) received as Chunks, the [SenderResponseTimer](#) will also be started and stopped by the Chunked Rx State Machine. See [Section 9.2.2](#) for more details of the [SenderResponseTimer](#) operation. The Policy Engine's response when the [SenderResponseTimer](#) expires **Shall** be dependent on the Message sent. The [SenderResponseTimer](#) **Shall** be started from the time the last bit of the [GoodCRC Message](#) EOP, corresponding to the Message requesting a response, has been received by the PHY Layer. The [SenderResponseTimer](#) **Shall** be stopped when the last bit of the EOP of the [GoodCRC Message](#), corresponding to the expected response Message, has been transmitted by the PHY Layer. The receiver of a Message requiring a response **Shall** respond within [tReceiverResponse](#) in order to ensure that the sender's [SenderResponseTimer](#) does not expire. The [tReceiverResponse](#) time **Shall** be measured from the time the last bit of the [GoodCRC Message](#) EOP, corresponding to the expected [Request Message](#), has been transmitted by the PHY Layer until the first bit of the response Message Preamble has been transmitted by the PHY Layer.

7.31.3. Capability Timers

Sources and Sinks use Capability Timers to determine Attachment of a PD Capable Device. By periodically sending or requesting Capabilities, it is possible to determine PD Device Attachment when a response is received.

7.31.3.1. SourceCapabilityTimer

Prior to the First Explicit Contract a Source **Shall** use the [SourceCapabilityTimer](#) to periodically send out a [Source_Capabilities Message](#) every [tTypeCSendSourceCap](#) while:

- The Port is Attached.
- The Source is not in an active connection with a PD Sink Port.

Whenever there is a [SourceCapabilityTimer](#) timeout the Source **Shall** send a [Source_Capabilities Message](#). It **Shall** then re-initialize and restart the [SourceCapabilityTimer](#). The [SourceCapabilityTimer](#) **Shall** be stopped when the last bit of the EOP corresponding to the [GoodCRC Message](#) has been received by the PHY Layer since a PD connection has been established. At this point, the Source waits for a [Request Message](#) or a response timeout.

Note: The Source can also stop sending [Source_Capabilities Message](#) after [nCapsCount](#) Messages have been sent without a [GoodCRC Message](#) response (see [Section 7.32.6](#)). See [Section 9.2.3](#) for more details of when [Source_Capabilities Messages](#) are transmitted.

7.31.3.2. SinkWaitCap Timer

While in a Default Contract or an Implicit Contract when a Sink observes an absence of [Source_Capabilities Messages](#), after VBUS is present, for a duration of [tTypeCSinkWaitCap](#) the Sink **May** issue [Hard Reset](#) Signaling in order to restart the sending of [Source_Capabilities Messages](#) by the Source (see [Section 7.32.6](#)) or continue to operate at USB Type-C current. When a Sink, entering [EPR Mode](#), observes an absence of [EPR_Source_Capabilities Messages](#), after the [GoodCRC Message](#) acknowledging the [EPR_Mode Message](#) with the Action field set to 3 (Enter Succeeded), for a duration of [tTypeCSinkWaitCap](#) the Sink **Shall** issue [Hard Reset](#) Signaling in order to exit [EPR Mode](#) (see [Section 6.4.9](#)).

When a Sink, exiting [EPR Mode](#), observes an absence of [Source_Capabilities Messages](#), after the [GoodCRC Message](#) acknowledging the [EPR_Mode Message](#) with the Action field set to 5 (Exit), for a duration of [tTypeCSinkWaitCap](#) the Sink **Shall** issue [Hard Reset](#) Signaling in order to restart the sending of [Source_Capabilities Messages](#) by the Source (see [Section 7.32.6](#)). See [Section 9.2.4](#) for more details of when the SinkWaitCapTimer is run.

7.31.3.3. tFirstSourceCap

After Port Partners are Attached or after a [Hard Reset](#) or after a [Power Role Swap](#) or after a [Fast Role Swap](#) a Source **Shall** send its first [Source_Capabilities Message](#) within [tFirstSourceCap](#) of VBUS reaching [vSafe5V](#). After [Soft Reset](#), a Source **Shall** send its first Source Capabilities Message within [tFirstSourceCap](#) after the last bit of the [GoodCRC Message](#) EOP corresponding to the [Accept Message](#). This ensures that the Sink receives a Source Capabilities Message before the Sink's [SinkWaitCapTimer](#) expires.

A Source entering [EPR Mode](#) **Shall** send its first [EPR_Source_Capabilities Message](#) within [tFirstSourceCap](#) of the [GoodCRC Message](#) acknowledging the [EPR_Mode Message](#) with the Action field set to 3 (Enter Succeeded). A Source exiting [EPR Mode](#) **Shall** send its first [Source_Capabilities Message](#) within [tFirstSourceCap](#) of the [GoodCRC Message](#) acknowledging the [EPR_Mode Message](#) with the Action field set to 5 (Exit).

7.31.4. Wait Timers and Times

7.31.4.1. SinkRequestTimer

The [SinkRequestTimer](#) is used to ensure that the time before the next Sink [Request Message](#), after a [Wait Message](#) has been received from the Source in response to a Sink [Request Message](#) (see [Section 7.9.2.2](#)), is a minimum of [tSinkRequest](#). The [SinkRequestTimer](#) **Shall** be started when the EOP of a [Wait Message](#) has been received and **Shall** be stopped if any other Message is received or during a [Hard Reset](#). The Sink **Shall** wait at least [tSinkRequest](#), after receiving the EOP of a [Wait Message](#) sent in response to a Sink [Request Message](#), before sending a new [Request Message](#). Whenever there is a [SinkRequestTimer](#) timeout, the Sink **May** send a [Request Message](#). It **Shall** then re-initialize and restart the [SinkRequestTimer](#).

7.31.4.2. tPRSwapWait

The time before the next [PR_Swap Message](#), after a [Wait Message](#) has been received in response to a [PR_Swap Message](#) (see [Section 7.10.1](#)) is a minimum of [tPRSwapWait](#) min. The Port **Shall** wait at least [tPRSwapWait](#) after receiving the EOP of a [Wait Message](#) sent in response to a [PR_Swap Message](#), before sending a new [PR_Swap Message](#).

7.31.4.3. tDRSwapWait

The time before the next [DR_Swap Message](#), after a [Wait Message](#) has been received in response to a [DR_Swap Message](#) (see [Section 7.12.2](#)) is a minimum of [tDRSwapWait](#) min. The Port **Shall** wait at least [tDRSwapWait](#) after receiving the EOP of a [Wait Message](#) sent in response to a [DR_Swap Message](#), before sending a new [DR_Swap Message](#).

7.31.4.4. tVconnSwapWait

The time before the next [VCONN_Swap Message](#), after a [Wait Message](#) has been received in response to a [VCONN_Swap Message](#) (see [Section 7.13.1](#)) is a minimum of [tVconnSwapWait](#) min. The Port **Shall** wait at least [tVconnSwapWait](#) after receiving the EOP of a [Wait Message](#) sent in response to a [VCONN_Swap Message](#), before sending a new [VCONN_Swap Message](#).

7.31.4.5. tVconnSwapDelayDFP

The time delay for DFP after losing VCONN Source role due to an incoming [VCONN_Swap](#) Request from UFP and before sending the next [VCONN_Swap Message](#). The DFP **Shall** wait at least [tVconnSwapDelayDFP](#) after sending the EOP of the [GoodCRC Message](#) in response to [PS_RDY Message](#) received at the end of the previous [VCONN_Swap](#) AMS.

7.31.4.6. tVconnSwapDelayUFP

The time delay for UFP after losing VCONN Source role due to an incoming [VCONN_Swap](#) Request from DFP and before sending the next [VCONN_Swap Message](#). The UFP **Shall** wait at least [tVconnSwapDelayUFP](#) after sending the EOP of the [GoodCRC Message](#) in response to [PS_RDY Message](#) received at the end of the previous [VCONN_Swap](#) AMS.

7.31.4.7. tEnterUSBWait

The time before the next [Enter_USB Message](#), after a [Wait Message](#) has been received in response to a [Enter_USB Message](#) (see [Section 7.29.1](#)) is a minimum of [tEnterUSBWait](#) min. The DFP **Shall** wait at least [tEnterUSBWait](#) after receiving the EOP of a [Wait Message](#) sent in response to an [Enter_USB Message](#), before sending a new [Enter_USB Message](#).

7.31.5. Power Supply Timers

See [Section 4.5](#) for diagrams showing the usage of the timers in this section.

7.31.5.1. PSTransition Timer

The [PSTransitionTimer](#) is used by the Policy Engine to timeout on a [PS_RDY Message](#). It is started when a Request for New Source Capabilities has been accepted and will timeout after [tPSTransition](#) if a [PS_RDY Message](#) has not been received. This condition leads to a [Hard Reset](#) and a return to USB Default Operation. The [PSTransitionTimer](#) relates to the time taken for the Source to transition from one voltage, or current level, to another (see [Section 4.1](#)).

The [PSTransitionTimer](#) **Shall** be started when the last bit of the [GoodCRC Message](#) EOP, corresponding to an [Accept Message](#), has been transmitted by the PHY Layer. The [PSTransitionTimer](#) **Shall** be stopped when the last bit of the [GoodCRC Message](#) EOP, corresponding to the [PS_RDY Message](#), has been transmitted by the PHY Layer.

7.31.5.2. PSSourceOffTimer

Use during [Power Role Swap](#)

The [PSSourceOffTimer](#) is used by the Policy Engine in Dual-Role Power Device that is currently acting as a Sink to timeout on a [PS_RDY Message](#) during a [Power Role Swap AMS](#). This condition leads to USB Type-C Error Recovery. If a [PR_Swap Message](#) Request has been sent by the Dual-Role Power Device currently acting as a Source the Sink can respond with an [Accept Message](#). When the last bit of the [GoodCRC Message](#) EOP, corresponding to this transmitted [Accept Message](#), is received by the Sink's PHY Layer, then the [PSSourceOffTimer](#) **Shall** be started.

If a [PR_Swap Message](#) Request has been sent by the Dual-Role Power Device currently acting as a Sink the Source can respond with an [Accept Message](#). When the last bit of the [GoodCRC Message](#) EOP, corresponding to this received [Accept Message](#), is transmitted by the Sink's PHY Layer, then the [PSSourceOffTimer](#) **Shall** be started. The [PSSourceOffTimer](#) **Shall** be stopped when the last bit of the [GoodCRC Message](#) EOP, corresponding to the received [PS_RDY Message](#), is transmitted by the PHY Layer. The [PSSourceOffTimer](#) relates to the time taken for the remote Dual-Role Power Device to stop supplying power (see [Section 4.3.3](#)). The timer **Shall** time out if a [PS_RDY Message](#) has not been received from the remote Dual-Role Power Device within [tPSSourceOff](#) indicating this has occurred.

Use during [Fast Role Swap](#)

The [PSSourceOffTimer](#) is used by the Policy Engine in Dual-Role Power Device that is the Initial Sink (currently providing [vSafe5V](#)) to timeout on a [PS_RDY Message](#) during a [Fast Role Swap AMS](#). This condition leads to USB Type-C Error Recovery. When the [FR_Swap Message](#) Request has been sent by the Initial Sink, the Initial Source **Shall** respond with an [Accept Message](#). When the last bit of the [GoodCRC Message](#) EOP, corresponding to this [Accept Message](#) is received by the Initial Sink's PHY Layer, then the [PSSourceOffTimer](#) **Shall** be started. The [PSSourceOffTimer](#) **Shall** be stopped when the last bit of the [GoodCRC Message](#) EOP, corresponding to the received [PS_RDY Message](#), is transmitted by the PHY Layer. The [PSSourceOffTimer](#) relates to the time taken for the Initial Source to stop supplying power and for VBUS to revert to [vSafe5V](#) (see [Figure 10.3](#)). The timer **Shall** time out if a [PS_RDY Message](#) has not been received from the Initial Source within [tPSSourceOff](#) indicating this has occurred.

7.31.5.3. PSSourceOnTimer

Use during [Power Role Swap](#)

The [PSSourceOnTimer](#) is used by the Policy Engine in Dual-Role Power Device that has just stopped sourcing power and is waiting to start sinking power to timeout on a [PS_RDY Message](#) during a [Power Role Swap](#). This condition leads to USB Type-C Error Recovery.

The [PSSourceOnTimer](#) **Shall** be started when:

- The last bit of the [GoodCRC Message](#) EOP, corresponding to the transmitted [PS_RDY Message](#), is received by the PHY Layer.
- The [PSSourceOnTimer](#) **Shall** be stopped when:
- The last bit of the [GoodCRC Message](#) EOP, corresponding to the received [PS_RDY Message](#), is transmitted by the PHY Layer.

The [PSSourceOnTimer](#) relates to the time taken for the remote Dual-Role Power Device to start sourcing power (see [Section 4.3.3](#)) and will time out if a [PS_RDY Message](#) indicating this has not been received within [tPSSourceOn](#).

Use during [Fast Role Swap](#)

The [PSSourceOnTimer](#) is used by the Policy Engine in Dual-Role Power Device that has just stopped sourcing power and is waiting to start sinking power to timeout on a [PS_RDY Message](#) during a [Fast Role Swap](#). This condition leads to USB Type-C Error Recovery.

The [PSSourceOnTimer](#) **Shall** be started when:

- The last bit of the [GoodCRC Message](#) EOP, corresponding to the transmitted [PS_RDY Message](#), is received by the PHY Layer.
- The [PSSourceOnTimer](#) **Shall** be stopped when:
- The last bit of the [GoodCRC Message](#) EOP, corresponding to the received [PS_RDY Message](#), is transmitted by the PHY Layer.

The [PSSourceOnTimer](#) relates to the time taken for the remote Dual-Role Power Device to start sourcing power (see [Section 10.1](#)) and will time out if a [PS_RDY Message](#) indicating this has not been received within [tPSSourceOn](#).

7.31.5.4. NoResponseTimer

The [NoResponseTimer](#) is used by the Policy Engine in a Source to determine that its Port Partner is not responding after a Hard Reset. When the [NoResponseTimer](#) times out, the Policy Engine **Shall** issue up to [nHardResetCount](#) additional Hard Resets before determining that the Port Partner is non-responsive to USB Power Delivery messaging. If the Source fails to receive a [GoodCRC Message](#) in response to a [Source_Capabilities Message](#) within [tNoResponse](#) of:

- The last bit of a [Hard Reset](#) Signaling being sent by the PHY Layer if the Hard Reset Signaling was initiated by the Sink.
- The last bit of a [Hard Reset](#) Signaling being received by the PHY Layer if the [Hard Reset](#) Signaling was initiated by the Source.
- Then the Source **Shall** issue additional Hard Resets up to [nHardResetCount](#) times (see [Section 7.1.3](#)).

For a non-responsive Device, the Policy Engine in a Source **May** either decide to continue sending [Source_Capabilities Messages](#) or to go to non-USB Power Delivery operation and cease sending [Source_Capabilities Messages](#).

7.31.6. BIST Timers

7.31.6.1. tBISTCarrierMode

[tBISTCarrierMode](#) is used to define the maximum time that a UUT has to enter BIST Carrier Mode when requested by a Tester. A UUT **Shall** enter BIST Carrier Mode within [tBISTCarrierMode](#) of the last bit of the [GoodCRC Message](#) EOP, corresponding to the received the [BIST Message](#) used to initiate the test, being transmitted by the PHY Layer. In BIST Carrier Mode when transmitting a continuous carrier signal transmission **Shall** start as soon as the UUT enters BIST Mode.

7.31.6.2. BISTContModeTimer

The [BISTContModeTimer](#) is used by a UUT to ensure that a Continuous BIST Mode (i.e., BIST Carrier Mode) is exited in a timely fashion. A UUT that has been put into a Continuous BIST Mode **Shall** return to normal operation (either PE_SRC_Transition_to_default, PE_SNK_Transition_to_default, or PE_CBL_Ready) within [tBISTContMode](#) of starting to transmit a continuous carrier signal.

7.31.6.3. tBISTSharedTestMode

[tBISTSharedTestMode](#) is used to define the maximum time that a UUT has to enter BIST Shared Capacity Test Mode when requested by a Tester. A UUT **Shall** enter BIST Shared Capacity Test Mode and send a new [Source_Capabilities Message](#) from all Ports within the Shared Capacity Port within [tBISTSharedTestMode](#) of the last bit of the [GoodCRC Message](#). Message EOP, corresponding to the received the [BIST Message](#) used to initiate the test, being transmitted by the PHY Layer.

7.31.7. Power Role Swap Timers

7.31.7.1. SwapSourceStartTimer

The [SwapSourceStartTimer](#) **Shall** be used by the New Source, after a [Power Role Swap](#) or [Fast Role Swap](#), to ensure that it does not send [Source_Capabilities Message](#) before the New Sink is ready to receive the [Source_Capabilities Message](#). The New Source **Shall Not** send the [Source_Capabilities Message](#) earlier than [tSwapSourceStart](#) after the last bit of the EOP of [GoodCRC Message](#) sent in response to the [PS_RDY Message](#) sent by the New Source indicating that its power supply is ready. The Sink **Shall** be ready to receive a [Source_Capabilities Message](#) [tSwapSinkReady](#) after having sent the last bit of the EOP of [GoodCRC Message](#) sent in response to the [PS_RDY Message](#) sent by the New Source indicating that its power supply is ready.

7.31.8. Soft Reset Timers

7.31.8.1. tSoftReset

A failure to see a [GoodCRC Message](#) in response to any Message within [tReceive](#) (after [nRetryCount](#) retries), when a Port Pair is Connected, is indicative of a communications failure. This **Shall** cause the Source or Sink to send a [Soft_Reset Message](#), transmission of which **Shall** be completed within [tSoftReset](#) of the [CRCReceiveTimer](#) expiring.

7.31.8.2. tProtErrSoftReset

If the Protocol Error occurs that causes the Source or Sink to send a [Soft_Reset Message](#), the transmission of the [Soft_Reset Message](#) **Shall** be completed within [tProtErrSoftReset](#) of the EOP of the [GoodCRC Message](#) sent in response to the Message that caused the Protocol Error.

7.31.9. Data Reset Timers

7.31.9.1. VCONNDischargeTimer

The [VCONNDischargeTimer](#) is used by the Policy Engine in the DFP to ensure the UFP actively discharges VCONN in a timely manner to ensure the cable will restore Ra. Once the UFP has discharged VCONN below vRaReconnect (see [USB-C]) it sends a [PS_RDY Message](#) (see also [Section 4.6](#)).

If the DFP does not receive a [PS_RDY Message](#) from the UFP within [tVCONNSourceDischarge](#) of the last bit of the [GoodCRC Message](#) acknowledging the [Accept Message](#) in response to the [Data_Reset Message](#), the [VCONNDischargeTimer](#) will time out and the Policy Engine **Shall** enter the ErrorRecovery State.

7.31.9.2. tDataReset

The DFP **Shall** complete the [Data_Reset](#) process (see [Section 7.1.2](#)) within [tDataReset](#) of the last bit of the [GoodCRC Message](#) EOP, corresponding to the [Accept Message](#), being transmitted by the PHY Layer.

7.31.9.3. DataResetFailTimer

The [DataResetFailUFPTimer](#) **Shall** be used by the DFP's Policy Engine to ensure the [Data_Reset](#) process completes within [tDataResetFailUFP](#) of the last bit of the [GoodCRC Message](#) acknowledging the [Accept Message](#) in response to the [Data_Reset Message](#). If the DFP's [DataResetFailUFPTimer](#) expires, the DFP **Shall** enter the ErrorRecovery State.

7.31.9.4. DataResetFailUFPTimer

The [DataResetFailUFPTimer](#) **Shall** be used by the UFP's Policy Engine to ensure the [Data_Reset](#) process completes within [tDataResetFailUFP](#) of the last bit of the [GoodCRC Message](#) acknowledging the [Accept Message](#) in response to the [Data_Reset Message](#). If the UFP's [DataResetFailUFPTimer](#) expires, the UFP **Shall** enter the ErrorRecovery State.

7.31.10. Hard Reset Timers

7.31.10.1. HardResetCompleteTimer

The [HardResetCompleteTimer](#) is used by the Protocol Layer in the case where it has asked the PHY Layer to send [Hard Reset](#) Signaling and the PHY Layer is unable to send the Signaling within a reasonable time due to a non-Idle channel. If the PHY Layer does not indicate that the [Hard Reset](#) Signaling has been sent within [tHardResetComplete](#) of the Protocol Layer requesting transmission, then the Protocol Layer **Shall** inform the Policy Engine that the [Hard Reset](#) Signaling has been sent in order to ensure the power supply is reset in a timely fashion.

7.31.10.2. PSHardResetTimer

The [PSHardResetTimer](#) is used by the Policy Engine in a Source to ensure that the Sink has had sufficient time to process Hard Reset Signaling before turning off its power supply to VBUS. When a [Hard Reset](#) occurs the Source, stops driving VCONN, removes Rp from the CC pin and starts to transition the VBUS voltage to [vSafe0V](#) either:

- [tPSHardReset](#) after the last bit of the [Hard Reset](#) Signaling has been received from the Sink or
- [tPSHardReset](#) after the last bit of the [Hard Reset](#) Signaling has been sent by the Source. See [Section 4.4](#).

7.31.10.3. tDRSwapHardReset

If a [DR_Swap Message](#) is received during Modal Operation then a [Hard Reset](#) **Shall** be initiated by the recipient of the unexpected [DR_Swap Message](#); [Hard Reset](#) Signaling **Shall** be generated within [tDRSwapHardReset](#) of the EOP of the [GoodCRC Message](#) sent in response to the [DR_Swap Message](#).

7.31.10.4. tProtErrHardReset

If a Protocol Error occurs that directly leads to a [Hard Reset](#), the transmission of the [Hard Reset](#) Signaling **Shall** be completed within [tProtErrHardReset](#) of the EOP of the [GoodCRC Message](#) sent in response to the Message that caused the Protocol Error.

7.31.10.5. tSendHardReset

When a timer expiration leads to sending a [Hard Reset](#), the transmission of the last bit of the [Hard Reset](#) Message **Shall** be completed within [tSendHardReset](#). [tSendHardReset](#) applies to the expiration of:

- [SenderResponseTimer](#),
- [SourcePPSCommTimer](#) or
- [SourceEPRKeepAliveTimer](#).

7.31.10.6. tInitiateErrorRecovery

When an event or timer expiration leads to ErrorRecovery, the ErrorRecovery State **Shall** be entered within [tInitiateErrorRecovery](#).

7.31.11. Structured VDM Timers

7.31.11.1. VDMResponseTimer

The [VDMResponseTimer](#) **Shall** be used by the Initiator's Policy Engine to ensure that a Structured VD Command Request needing a response (e.g. [Discover Identity](#) Command Request) is responded to with a bounded time of [tVDMSenderResponse](#). The [VDMResponseTimer](#) **Shall** be applied to all Structure VDM Commands except the [Enter Mode](#) and [Exit Mode](#) Commands which have their own timer ([VDMModeEntryTimer](#) and [VDMModeExitTimer](#) respectively). Failure to receive the expected response is detected when the [VDMResponseTimer](#) expires.

The Policy Engine's response when the [VDMResponseTimer](#) expires **Shall** depend on the Message sent (see [Section 7.5](#) and [Section 9.2](#)). The [VDMResponseTimer](#) **Shall** be started from the time the last bit of the [GoodCRC Message](#) EOP corresponding to the VDM Command requesting a response, has been received by the PHY Layer. The [VDMResponseTimer](#) **Shall** be stopped when the last bit of the EOP of the [GoodCRC Message](#), corresponding to the expected VDM Command response, has been transmitted by the PHY Layer. The receiver of a Message requiring a response **Shall** respond within [tVDMReceiverResponse](#) in order to ensure that the sender's [VDMResponseTimer](#) does not expire. The [tVDMReceiverResponse](#) time **Shall** be measured from the time the last bit of the Message EOP has been transmitted by the PHY Layer until the first bit of the response Message Preamble has been transmitted by the PHY Layer.

7.31.11.2. VDMModeEntryTimer

The [VDMModeEntryTimer](#) **Shall** be used by the Initiator's Policy Engine to ensure that the response to a [Structured VDM](#) Enter Mode Command Request (ACK or NAK with ACK indicating that the requested Alternate Mode has been entered) arrives within a bounded time of [tVDMWaitModeExit](#). Failure to receive the expected response is detected when the [VDMModeEntryTimer](#) expires. The Policy Engine's response when the [VDMModeEntryTimer](#) expires is to inform the Device Policy Manager (see [Section 9.2.24.1.1](#)).

The [VDMModeEntryTimer](#) **Shall** be started from the time the last bit of the EOP of the [GoodCRC Message](#), corresponding to the VDM Command Request, has been received by the PHY Layer. The [VDMModeEntryTimer](#) **Shall** be stopped when the last bit of the EOP of the [GoodCRC Message](#), corresponding to the expected [Structured VDM](#) Command response (ACK, NAK or BUSY), has been transmitted by the PHY Layer.

The receiver of a Message requiring a response **Shall** respond within [tVDMEnterMode](#) in order to ensure that the sender's [VDMModeEntryTimer](#) does not expire. The [tVDMEnterMode](#) time **Shall** be measured from the time the last bit of the EOP of the [GoodCRC Message](#), corresponding to VDM Command Request, has been transmitted by the PHY Layer until the first bit of the response Message Preamble has been transmitted by the PHY Layer.

7.31.11.3. VDMModeExitTimer

The [VDMModeExitTimer](#) **Shall** be used by the Initiator's Policy Engine to ensure that the ACK response to a [Structured VDM](#) Exit Mode Command, indicating that the requested Alternate Mode has been exited, arrives within a bounded time of [tVDMWaitModeExit](#). Failure to receive the expected response is detected when the [VDMModeExitTimer](#) expires. The Policy Engine's response when the [VDMModeExitTimer](#) expires is to inform the Device Policy Manager (see [Section 9.2.24.2.1](#)). The [VDMModeExitTimer](#) **Shall** be started from the time the last bit of the [GoodCRC Message](#) EOP, corresponding to the VDM Command requesting a response, has been received by the PHY Layer. The [VDMModeExitTimer](#) **Shall** be stopped when the last bit of the [GoodCRC Message](#) EOP, corresponding to the expected [Structured VDM](#) Command response ACK, has been transmitted by the PHY Layer. The receiver of a Message requiring a response **Shall** respond within [tVDMExitMode](#) in order to ensure that the sender's [VDMModeExitTimer](#) does not expire. The [tVDMExitMode](#) time **Shall** be measured from the time the last bit of the Message EOP has been received by the PHY Layer until the first bit of the response Message Preamble has been transmitted by the PHY Layer.

7.31.11.4. tVDMBusy

The Initiator **Shall** wait at least [tVDMBusy](#), after receiving a BUSY Command response, before repeating the [Structured VDM](#) Request again.

7.31.12. VCONN Timers

7.31.12.1. VconnOnTimer

The [VconnOnTimer](#) is used during a [VCONN Swap](#).

The [VconnOnTimer](#) **Shall** be started when:

- The last bit of [GoodCRC Message](#) EOP, corresponding to the [Accept Message](#), is transmitted or received by the PHY Layer.

The [VconnOnTimer](#) **Shall** be stopped when:

- The last bit of the [GoodCRC Message](#) EOP, corresponding to the [PS_RDY Message](#), is transmitted by the PHY Layer.

Prior to sending the [PS_RDY Message](#), the Port **Shall** have turned VCONN On.

7.31.12.2. tVconnSourceOff

The [tVconnSourceOff](#) time applies during a [VCONN Swap](#). The initial VCONN Source **Shall** cease sourcing VCONN within [tVconnSourceOff](#) of the last bit of the [GoodCRC Message](#) EOP, corresponding to the [PS_RDY Message](#), being transmitted by the PHY Layer.

7.31.12.3. tCableMessage

When Ports on both ends are compliant with Revision 3.x of the specification, the ports **Shall Not** wait [tCableMessage](#) before sending an SOP' Packet or SOP" Packet even when communicating using [PD2] with a Cable Plug. This specification defines Collision Avoidance mechanisms that obviate the need for this time. Cable Plugs **Shall** only wait [tCableMessage](#) before sending an SOP' Packet or SOP" Packet when operating at [PD2]. When operating at Revisions higher than [PD2] Cable Plugs **Shall Not** wait [tCableMessage](#) before sending an SOP' Packet or SOP" Packet.

7.31.12.4. DiscoverIdentityTimer

The [DiscoverIdentityTimer](#) is used prior to or during an Explicit Contract when discovering whether a Cable Plug is PD Capable using SOP'. When performing Cable Discovery during an Explicit Contract the [Discover Identity](#) Command Request **Shall** be sent every [tDiscoverIdentity](#). No more than [nDiscoverIdentityCount](#) [Discover Identity](#) Messages without a [GoodCRC Message](#) response **Shall** be sent. If no [GoodCRC Message](#) response is received after [nDiscoverIdentityCount](#) [Discover Identity](#) Command requests have been sent by a Port, the Port **Shall Not** send any further SOP'/SOP" Messages.

7.31.13. Collision Avoidance Timers

7.31.13.1. SinkTxTimer

The [SinkTxTimer](#) is used by the Protocol Layer in a Source to allow the Sink to complete its transmission before initiating an AMS. The Source **Shall** wait a minimum of [tSinkTx](#) after changing Rp from SinkTxOK to SinkTxNG before initiating an AMS by sending a Message. A Sink **Shall** only initiate an AMS when it has determined that Rp is set to SinkTxOK.

7.31.13.2. tSrcHoldsBus

If a transition into the PE_SRC_Ready State will result in an immediate transition out of the PE_SRC_Ready State within [tSrcHoldsBus](#) e.g. it is due to a Protocol Error that has not resulted in a [Soft Reset](#), then the notifications of the end of AMS and first Message in an AMS **May Not** be sent to avoid changing the Rp value unnecessarily.

7.31.14. Fast Role Swap Timers

See [Section 10.1](#) for more information on the use of [Fast Role Swap](#) timers.

7.31.14.1. tFRSwap5V

The [tFRSwap5V](#) time **Shall** be measured from:

The later of:

- The last bit of the [GoodCRC Message](#) EOP, corresponding to the [Accept Message](#) or
- VBUS being within [vSafe5V](#).
- Until the first bit of the response [PS_RDY Message](#) Preamble has been transmitted by the PHY Layer.

During a [Fast Role Swap](#), the Initial Source **Shall** start the [PS_RDY Message](#) within [tFRSwap5V](#) after both:

- The Initial Source has sent the [Accept Message](#), and
- VBUS is at or below [vSafe5V](#).

7.31.14.2. tFRSwapComplete

During a [Fast Role Swap](#), the Initial Sink **Shall** respond with a the [PS_RDY Message](#) within [tFRSwapComplete](#) after it has received the [PS_RDY Message](#) from the Initial Source. The [tFRSwapComplete](#) time **Shall** be measured from the time the last bit of the [GoodCRC Message](#) EOP, corresponding to the [PS_RDY Message](#), has been transmitted by the PHY Layer until the first bit of the response [PS_RDY Message](#) Preamble has been transmitted by the PHY Layer.

7.31.14.3. tFRSwapInit

That last bit of the EOP of the [FR_Swap Message](#) **Shall** be transmitted by the New Source no later than [tFRSwapInit](#) after the Fast Role Swap Request has been detected (see [Section 10.3](#))

7.31.15. Chunking Timers

7.31.15.1. ChunkingNotSupportedTimer

The [ChunkingNotSupportedTimer](#) is used by a Source or Sink which does not support multi-Chunk Chunking but has received a Message Chunk.

The [ChunkingNotSupportedTimer](#) **Shall** be started when:

The last bit of the [GoodCRC Message](#) EOP, corresponding to a Message Chunk of a multi-Chunk Message, is transmitted by the PHY Layer. The Policy Engine **Shall Not** send its [Not_Supported Message](#) before the [ChunkingNotSupportedTimer](#) expires.

7.31.15.2. ChunkSenderRequestTimer

The [ChunkSenderResponseTimer](#) is used during a Chunked Message transmission. The [ChunkSenderResponseTimer](#) **Shall** be used by the sender's Chunking State machine to ensure that a Chunk Response is responded to within a bounded time of [tChunkSenderResponse](#). Failure to receive the expected response is detected when the [ChunkSenderResponseTimer](#) expires.

The [ChunkSenderResponseTimer](#) **Shall** be started when:

The last bit of the [GoodCRC Message](#) EOP, corresponding to the Chunk Response Message, is received by the PHY Layer.

The [ChunkSenderResponseTimer](#) **Shall** be stopped when:

- The last bit of the EOP of the [GoodCRC Message](#), corresponding to the Chunk [Request Message](#), is transmitted by the PHY Layer.

- A Message other than a Chunk Request is received from the Protocol Layer Rx.

The receiver of a Chunk Response requiring a Chunk Request **Shall** respond with a Chunk Request within [tChunkReceiverRequest](#) in order to ensure that the sender's [ChunkSenderRequestTimer](#) does not expire.

The [tChunkReceiverRequest](#) time **Shall** be measured from the time the last bit of the EOP of the [GoodCRC Message](#), corresponding to the Chunk Response Message, has been transmitted by the PHY Layer until the first bit of the response Message Preamble has been transmitted by the PHY Layer.

7.31.15.3. ChunkSenderResponseTimer

The [ChunkSenderResponseTimer](#) is used during a Chunked Message transmission. The [ChunkSenderResponseTimer](#) **Shall** be used by the sender's Chunking State machine to ensure that a Chunk Request is responded to within a bounded time of [tChunkSenderResponse](#). Failure to receive the expected response is detected when the [ChunkSenderResponseTimer](#) expires.

The [ChunkSenderResponseTimer](#) **Shall** be started when:

The last bit of the [GoodCRC Message](#) EOP, corresponding to the Chunk [Request Message](#), is received by the PHY Layer.

The [ChunkSenderResponseTimer](#) **Shall** be stopped when:

The last bit of the [GoodCRC Message](#) EOP, corresponding to the Chunk Response Message, is transmitted by the PHY Layer.

A Message other than a Chunk is received from the Protocol Layer. The receiver of a Chunk Request requiring a Chunk Response **Shall** respond with a Chunk Response within [tChunkReceiverResponse](#) in order to ensure that the sender's [ChunkSenderResponseTimer](#) does not expire. The [tChunkReceiverResponse](#) time **Shall** be measured from the time the last bit of the EOP of the [GoodCRC Message](#), corresponding to the Chunk [Request Message](#), has been transmitted by the PHY Layer until the first bit of the response Message Preamble has been transmitted by the PHY Layer.

7.31.16. Programmable Power Supply Timers

7.31.16.1. SinkPPSPeriodicTimer

The [SinkPPSPeriodicTimer](#) **Shall** be used by the Sink's Policy Engine to ensure that communication between the Sink and Source occurs within a bounded time of [tPPSRequest](#) when in PPS Mode. In the absence of any other traffic, a [Request Message](#) requesting a PPS APDO is sent periodically as a keep alive mechanism.

[SinkPPSPeriodicTimer](#) **Shall** be re-initialized and restarted on transmission, by the PHY Layer, of the last bit of the [GoodCRC Message](#) EOP, corresponding to any received Message, that causes the Sink to enter the PE_SNK_Ready State. The Sink **Shall** stop the [SinkPPSPeriodicTimer](#) on transmission, by the PHY Layer, of the last bit of the [GoodCRC Message](#) EOP, corresponding to any Message, or the last bit of any Signaling is received, by the PHY Layer, from the Source and by the Sink that causes the Sink to leave the PE_SNK_Ready State.

7.31.16.2. SourcePPSCommTimer

The [SourcePPSCommTimer](#) **Shall** be used by the Source's Policy Engine to ensure that communication between the Sink and Source occurs within a bounded time of [tPPSTimeout](#) when in PPS Mode. In the absence of any other traffic, a [Request Message](#) requesting a PPS APDO is received periodically as a keep alive mechanism.

[SourcePPSCommTimer](#) **Shall** be re-initialized and restarted when, after receiving any Message that causes the Source to enter the PE_SRC_Ready State, the last bit of the corresponding [GoodCRC Message](#) EOP is transmitted by the PHY Layer.

The Source **Shall** stop the [SourcePPSCCommTimer](#) when:

- After receiving any Message that causes the Source to leave the PE_SRC_Ready State, the last bit of the corresponding [GoodCRC Message](#) EOP is sent by the PHY Layer, or
- The last bit of any Signaling is received by the PHY Layer from the Sink by the Source that causes the Source to leave the PE_SRC_Ready State. When the [SourcePPSCCommTimer](#) times out the Source **Shall** issue [Hard Reset](#) Signaling.

7.31.17. tEnterUSB

The DFP **Shall** send the [Enter_USB Message](#) within [tEnterUSB](#) of either:

- The last bit of the [GoodCRC Message](#) acknowledging the [Data_Reset_Complete Message](#) in response to the [Data_Reset Message](#) or
- A PD Connection, specifically the last bit of the [GoodCRC Message](#) acknowledging the [Source_Capabilities Message](#) after the initial entry into the PE_SRC_Send_Capabilities State or

The last bit of the [GoodCRC Message](#) acknowledging the [Accept Message](#) in response to the [DR_Swap Message](#) Failure by the DFP to meet this timeout parameter can result in the ports not transitioning into [USB4] operation. Any AMS initiated by the UFP prior to receiving the [Enter_USB Message](#) will delay reception of the [Enter_USB Message](#) and [USB4] operation, therefore a USB4 -capable UFP **Should Not** initiate any AMS until the DFP has been given time to send the [Enter_USB Message](#).

7.31.18. EPR Timers

7.31.18.1. SinkEPREnterTimer

The [SinkEPREnterTimer](#) is used to ensure the [EPR Mode](#) Entry process completes within [tEnterEPR](#). The Sink **Shall** start the timer when it sees the last bit of the [GoodCRC Message](#) in response to the [EPR_Mode Message](#) with the Action field set to 1 (Enter). The Sink **Shall** stop the timer when the last bit of the corresponding [GoodCRC Message](#) EOP, corresponding to the received [EPR_Mode Message](#) with the Action field set to 3 (Enter Succeeded), has been transmitted by the PHY Layer. If the timer expires the Sink **Shall** send a [Soft_Reset Message](#).

7.31.18.2. SourceEPRKeepAlive Timer

The [SinkEPRKeepAliveTimer](#) **Shall** be used by the Sink's Policy Engine to ensure that communication between the Sink and Source occurs within a bounded time of [tSinkEPRKeepAlive](#). The Sink **Shall** initialize and run this timer upon entry into the PE_SNK_Ready State when in [EPR Mode](#) and **Shall** stop it upon exit from the PE_SNK_Ready when in [EPR Mode](#).

While operating in [EPR Mode](#), the Sink **Shall** stop the [SinkEPRKeepAliveTimer](#) timer whenever:

- The last bit of the [GoodCRC Message](#) EOP, in response any Message from the Source, is transmitted by the PHY Layer.
- The PHY Layer receives the last bit of the [GoodCRC Message](#) EOP in response to any Message sent to the Source.

If the timer expires the Sink **Shall** send an [EPR_Keep_Alive Message](#).

7.31.18.3. tEPRSourceCableDiscovery

After Port Partners are Attached or after a [Hard Reset](#) or after a [Power Role Swap](#) or after a [Fast Role Swap](#) an EPR Source **Shall** discover the Cable Plug within [tEPRSourceCableDiscovery](#) of entering the First Explicit Contract.

The EPR Source **Shall** send the [Discover Identity](#) REQ Command, to the Cable Plug, within [tEPRSourceCableDiscovery](#) of receiving the [GoodCRC Message](#) acknowledging the [PS_RDY Message](#) as part of the Explicit Contract Negotiation.

Note: If the EPR Source is not the VCONN Source, [tEPRSourceCableDiscovery](#), will also include the time needed for the [VCONN Swap](#).

7.31.19. Time Values and Timers

Table 7.9. Timer Values

Parameter Name		Min Value	Nom Value	Max Value	Unit
tACTempUpdate				500	ms
tBISTContMode		30	45	60	ms
tBISTCarrierMode				300	ms
tBISTSharedTestMode				1	s
tCableMessage		750			µs
tCapabilitiesMismatchResponse				2	s
tChunkingNotSupported		40	45	50	ms
tChunkReceiverRequest				15	ms
tChunkReceiverResponse				15	ms
tChunkSenderRequest		24	27	30	ms
tChunkSenderResponse		24	27	30	ms
tDataReset		200	225	250	ms
tDataResetFail		300		400	ms
tDataResetFailUFP		450		550	ms
tDiscoverIdentity		40		50	ms
tDRSwapHardReset				15	ms
tDRSwapWait		100			ms
tEnterUSB				500	ms
tEnterUSBWait		100			ms
tEnterEPR		450	500	550	ms
tEPRSourceCableDiscovery				2	ms
tFirstSourceCap				250	ms
tHardReset				5	ms
tHardResetComplete		4000	4500	5000	µs
tSourceEPRKeepAlive		750	875	1000	ms
tSinkEPRKeepAlive		250	375	500	ms
tNoResponse		4.5	5.0	5.5	s
tPPSRequest					s
tPPSTimeout		12.0	13.5	5.5	s
tProtErrHardReset				15	ms
tSendHardReset				15	ms
tProtErrSoftReset				15	ms
tPRSwapWait		100			ms
tPSHardReset		25	30	35	ms
tInitiateErrorRecovery				15	ms
tPSSourceOff	SPR Mode	750	835	920	ms
	EPR Mode	1120	1260	1400	ms

Parameter Name		Min Value	Nom Value	Max Value	Unit
tPSSourceOn	SPR Mode	390	435	480	ms
tPSTransition	SPR Mode	450	500	550	ms
	EPR Mode	830	925	1020	ms
tReceive		900	1000	1100	μs
tReceiverResponse				15	ms
tRetry				195	μs
tSenderResponse		27 ¹		50 ¹	ms
tSinkRequest		100			ms
tSinkTx		16	18	20	ms
tSoftReset				15	ms
tSinkDelay				5	ms
tSrcHoldsBus				50	ms
tSwapSinkReady				15	ms
tSwapSourceStart		20			ms
tTransmit				195	μs
tTypeCSendSourceCap		100	150	200	ms
tTypeCSinkWaitCap		310	465	620	ms
tVconnSourceDischarge		160	200	240	ms
tVconnSourceOff				25	ms
tVconnSourceOn				50	ms
tVconnSourceTimeout		100	150	200	ms
tVconnSwapWait		100			ms
tVconnSwapDelayDFP		100			ms
tVconnSwapDelayUFP		500			ms
tVDMBusy		50			ms
tVDMEnterMode				25	ms
tVDMEExitMode				25	ms
tVDMReceiverResponse				15	ms
tVDMSenderResponse		24		50 ¹	ms
tVDMWaitModeEntry		40	45	50	ms
tVDMWaitModeExit		40	45	50	ms

1. This timing can also be used when USBPD Device is in a PD2 explicit Contract

Table 7.10. Timer Specifications

Timer Name	Parameter	Used By
BISTContModeTimer	tBISTContMode	Policy Engine
ChunkingNotSupportedTimer	tChunkingNotSupported	Policy Engine
ChunkSenderRequestTimer	tChunkSenderRequest	Protocol Layer
ChunkSenderResponseTimer	tChunkSenderResponse	Protocol Layer
CRCReceiveTimer	tReceive	Protocol Layer
DataResetFailTimer	tDataResetFail	Policy Engine
DataResetFailUFPTimer	tDataResetFailUFP	Policy Engine
DiscoverIdentityTimer	tDiscoverIdentity	Policy Engine
HardResetCompleteTimer	tHardResetComplete	Protocol Layer
NoResponseTimer	tNoResponse	Policy Engine

Timer Name	Parameter	Used By
PSHardResetTimer	tPSHardReset	Policy Engine
PSSourceOffTimer	tPSSourceOff	Policy Engine
PSSourceOnTimer	tPSSourceOn	Policy Engine
PSTransitionTimer	tPSTransition	Policy Engine
SenderResponseTimer	tSenderResponse	Policy Engine
SinkEPREnterTimer	tEnterEPR	Policy Engine
SinkEPRKeepAliveTimer	tSinkEPRKeepAlive	Policy Engine
SinkPPSPeriodicTimer	tPPSRequest	Policy Engine
SinkRequestTimer	tSinkRequest	Policy Engine
SinkWaitCapTimer	tTypeCSinkWaitCap	Policy Engine
SourceCapabilityTimer	tTypeCSendSourceCap	Policy Engine
SourceEPRKeepAliveTimer	tSourceEPRKeepAlive	Policy Engine
SourcePPSCmnTimer	tPPSTimeout	Policy Engine
SinkTxTimer	tSinkTx	Protocol Layer
SwapSourceStartTimer	tSwapSourceStart	Policy Engine
VconnDischargeTimer	tVconnSourceDischarge	Policy Engine
VconnOnTimer	tVconnSourceTimeout	Policy Engine
VDMModeEntryTimer	tVDMWaitModeEntry	Policy Engine
VDMModeExitTimer	tVDMWaitModeExit	Policy Engine
VDMResponseTimer	tVDMSenderResponse	Policy Engine

7.32. Counters

7.32.1. MessageID Counter

The MessageIDCounter is a rolling counter, ranging from 0 to [nMessageIDCount](#), used to detect duplicate Messages. This value is used for the MessageID field in the Message Header of each transmitted Message. Each Port **Shall** maintain a copy of the last MessageID value received from its Port Partner. Devices that support multiple ports, such as Hubs, **Shall** maintain copies of the last MessageID on a per Port basis. A Port which communicates using SOP* Packets **Shall** maintain copies of the last MessageID for each type of SOP* it uses. The transmitter **Shall** use the MessageID in a [GoodCRC Message](#) to verify that a particular Message was received correctly. The receiver **Shall** use the MessageID to detect duplicate Messages.

7.32.2. Transmitter Usage

The Transmitter **Shall** use the MessageID as follows:

- Upon receiving either [Hard Reset](#) Signaling, or a [Soft_Reset Message](#), the transmitter **Shall** set its MessageIDCounter to zero and re-initialize its retry mechanism.
- If a [GoodCRC Message](#) with a MessageID matching the MessageIDCounter is not received before the [CRCReceiveTimer](#) expires, it **Shall** retry the same Packet up to [nRetryCount](#) times using the same MessageID.
- If a [GoodCRC Message](#) is received with a MessageID matching the current MessageIDCounter before the [CRCReceiveTimer](#) expires, the transmitter **Shall** re-initialize its retry mechanism and increment its MessageIDCounter.
- If the Message is aborted by the Policy Engine, the transmitter **Shall** delete the Message from its transmit buffer, re-initialize its retry mechanism and increment its MessageIDCounter.

7.32.3. Receiver Usage

The Receiver **Shall** use the MessageID as follows:

- When the first good Packet is received after a reset, the receiver **Shall** store a copy of the received MessageID value.
- For subsequent Messages, if MessageID value in a received Message is the same as the stored value, the receiver **Shall** return a [GoodCRC Message](#) with that MessageID value and drop the Message (this is a retry of an already received Message).

Note: This **Shall Not** apply to the [Soft_Reset Message](#) which always has a MessageID value of zero.

- If MessageID value in the received Message is different than the stored value, the receiver **Shall** return a [GoodCRC Message](#) with the new MessageID value, store a copy of the new MessageID value and process the Message.

7.32.4. Retry Counter

The RetryCounter is used by a Port whenever there is a Message transmission failure (timeout of [CRCReceive_Timer](#)). If the [nRetryCount](#) retry fails, then the link **Shall** be reset using the [Soft_Reset](#) mechanism.

The following rules apply to retries when there is a Message transmission failure (see [Section 9.1.2.2](#)):

- Cable Plugs **Shall Not** retry Messages.
- [Extended Messages](#) of Data Size > [MaxExtendedMsgLegacyLen](#) that are not Chunked (Chunked flag set to zero) **Shall Not** be retried.
- [Extended Messages](#) of Data Size ≤ [MaxExtendedMsgLegacyLen](#) (Chunked flag set to zero or one) **Shall** be retried.
- [Extended Messages](#) of Data Size > [MaxExtendedMsgLegacyLen](#) that are Chunked (Chunked flag set to one) individual Chunks **Shall** be retried.

When Messages are not retried, then the RetryCounter is not used. Higher layer protocols are expected to accommodate Message delivery failure or failure to receive a [GoodCRC Message](#).

7.32.5. Hard Reset Counter

The HardResetCounter is used to retry the [Hard_Reset](#) whenever there is no response from the remote Device (see [Section 7.31.5.4](#)). Once the Hard_Reset has been retried [nHardResetCount](#) times then it **Shall** be assumed that the remote Device is non-responsive.

7.32.6. Capabilities Counter

The [CapsCounter](#) is used to count the number of [Source_Capabilities Messages](#) which have been sent by a Source at power up or after a [Hard_Reset](#). Implementation of the [CapsCounter](#) is **Optional** but **May** be used by any Source which wishes to preserve power by not sending [Source_Capabilities Messages](#) after a period of time. When the [CapsCounter](#) is implemented and the Source detects that a Sink is Attached then after [nCapsCount](#) [Source_Capabilities Messages](#) have been sent the Source **Shall** decide that the Sink is non-responsive, stop sending [Source_Capabilities Messages](#) and disable PD.

A Sink **Shall** use the [SinkWaitCapTimer](#) to trigger the resending of [Source_Capabilities Messages](#) by a USB Power Delivery capable Source which has previously stopped sending [Source_Capabilities Messages](#). Any Sink which is Attached and does not detect a [Source_Capabilities Message](#), **Shall** issue [Hard_Reset](#) Signaling when the [SinkWait-](#)

[CapTimer](#) times out in order to reset the Source. Resetting the Source **Shall** also reset the [CapsCounter](#) and restart the sending of [Source_Capabilities Messages](#).

7.32.7. Discover Identity Counter

When sending [Discover Identity](#) Messages to a Cable Plug a Port **Shall** maintain a count of Messages sent ([DiscoverIdentityTimer](#)). No more than [nDiscoverIdentityCount](#) [Discover Identity](#) Messages **Shall** be sent by the Port without receiving a [GoodCRC Message](#) Message response. A [VCONN Swap](#) **Shall** reset the [DiscoverIdentityTimer](#).

7.32.8. VDMBusyCounter

When sending Responder BUSY responses to a Structured [Vendor_Defined Message](#) a UFP or Cable Plug **Shall** maintain a count of Messages sent ([VDMBusyCounter](#)). No more than [nBusyCount](#) Responder BUSY responses **Shall** be sent. The [VDMBusyCounter](#) **Shall** be reset on sending a non-BUSY response. Products wishing to meet [USB-C] requirements for Alternate Mode Entry **Should** use an [nBusyCount](#) of 1.

7.32.9. Counter Values and Counters

[Table 7.12](#) lists the counters used in this section and [Table 7.11](#) shows the corresponding parameters.

Table 7.11. Counter Parameters

Parameter	Value
nBusyCount	5
nCapsCount	50
nDiscoverIdentityCount	20
nHardResetCount	2
nMessageIDCount	7
nRetryCount	2

Table 7.12. Counters

Counter	Max
CapsCounter	nCapsCount
DiscoverIdentityCounter	nDiscoverIdentityCount
HardResetCounter	nHardResetCount
MessageIDCounter	nMessageIDCount
RetryCounter	nRetryCount
VDMBusyCounter	nBusyCount

Chapter 8. Vendor Defined Message Usage and Alternate Modes

8.1. Overview

This chapter describes Vendor-Defined Message (VDM) usage and Alternate Modes. VDMs allow devices to discover information from Connected Ports and Cable Plugs, and they enable Modal Operation. Additionally, VDMs can support proprietary, vendor-defined functions. Vendor-Defined Messages include [Data Messages](#) with Message Type set to [Vendor_Defined](#) and [Extended Messages](#) with Message Type set to [Vendor_Defined_Extended](#).

8.2. Alternate Modes

A Device **May** support multiple Alternate Modes with one or more active at any point in time. Any interactions between them are the responsibility of the Standard or Vendor. Where there are multiple Active Modes at the same time Modal Operation **Shall** start on entry to the first Alternate Mode.

A [DR_Swap Message](#) **Shall Not** be sent during Modal Operation between the Port Partners (see [Section 6.3.9](#)).

8.3. General VDM Rules

VDMs **Shall Not** be used for direct power Negotiation. They **May** however, be used to alter Local Policy, affecting what is offered or consumed via the normal PD messages. VDMs **Should Not** be used where the PD Specification provides equivalent functionality, e.g., authentication or firmware update.

The following sections use the terms Initiator and Responder to identify messaging roles that participants take on relative to each other for the duration of a VDM AMS. These roles are independent of Power Capabilities, Power Role, or Data Role. The Initiator is the Port sending the initial Command Request and the Responder is the Port replying with the Command response. See: [Section 8.7](#).

During Default Contract or Implicit Contract, Ports **Shall Not** Initiate VDMs, and Ports and Cables **Shall Ignore** any VDMs received, with the exception of [Discover Identity](#) on SOP*. See [Section 8.6.1.2](#) for rules related to SOP* [Discover Identity](#).

8.4. Unstructured VDM Rules

The following rules apply to the use of Unstructured VDMs:

- Unstructured VDMs **May** be used with SOP* Packets.
- Prior to establishing an Explicit Contract, Unstructured VDMs **Shall Not** be sent and **Shall** be Ignored if received.
- Only the DFP **Shall** be an Initiator of Unstructured VDMs.
- Only the UFP or a Cable Plug **Shall** be a Responder to an Unstructured VDM.
- A Port **Shall Not** initiate an Unstructured VDM AMS except during Modal Operation, where the SVID of the Unstructured VDMs is one corresponding to an Active Mode.
- When a DFP or UFP does not support Unstructured VDMs or does not recognize the VID, it **Shall** Respond with a [Not_Supported Message](#).

8.5. Structured VDM Rules

The following rules apply to the use of [Structured VDM](#) messages:

- Either Port **May** Initiate a [Structured VDM AMS](#), except that the UFP **Shall Not** Initiate [Enter Mode](#) or [Exit Mode](#) commands
- A Cable Plug **Shall Not** Initiate a [Structured VDM AMS](#).
- When a Port does not support Structured VDMs, it **Shall** Respond to any received Structured VDMs with [Not_Supported Message](#).
- When a Cable Plug does not support Structured VDMs, any Structured VDMs received **Shall** be **Ignored**.
- When using any of the SVID-Specific Commands in the [Structured VDM](#) Header (Command field, values 16 - 31), the Responder **Shall** NAK VDMs where the Command is not recognized for the SVID.
- A Responder that supports Structured VDMs **Shall** NAK a [Structured VDM](#) with an SVID that it does not recognize.

8.5.1. Structured VDM Version

To ensure interoperability with existing USB PD products, USB PD products **Shall** support every [Structured VDM](#) Version number starting from Version 1.0.

On receipt of a VDM Header with a higher Version number than it supports, a Responder **Shall** respond using the highest Version number that the Responder supports. On receipt of a VDM Header with a lower Version number than it supports, a Responder **Shall** respond using the same Version number it received.

The [Structured VDM](#) Version (Major)/[Structured VDM](#) Version (Minor) fields of the [Discover Identity](#) Commands sent and received during the [Section 8.7.1](#) **Shall** be used to determine the highest common [Structured VDM](#) Versions supported by the Port Partners or by the Port Partner and the Cable Plugs, respectively. After discovering the common [Structured VDM](#) Version for the Port Partner or Cable Plugs, each Port Partner **Shall** continue to use this [Structured VDM](#) Version when Initiating Structured VDMs until Detach, [Hard Reset](#), or Error Recovery occurs.

8.5.2. Object Position

The Object Position field **Shall** be used by the [Enter Mode](#) and [Exit Mode](#) Commands. The [Discover Modes](#) Command returns a list of 0-6 VDOs, each of which describe an Alternate Mode. The value in Object Position field is an index into that list that indicates which VDO (e.g., Alternate Mode) in the list the Enter Mode and [Exit Mode](#) Command refers to. The Object Position **Shall** start with one for the first Alternate Mode entries in the list.

8.5.3. SVID

The content of the Mode VDOs for an Alternate Mode **Shall** be defined by the standard or vendor associated with the SVID. The Mode VDOs' content **May** be as simple as a numeric value or as complex as a bit-mapped description of Capabilities of the Alternate Mode. In all cases, the [Discover Modes](#) Responder is responsible for interpreting the VDOs to know whether or not it supports the Alternate Mode at the Object Position.

8.6. Command Usage

Ports use sequences of [Structured VDM](#) commands to discover, manage, and raise events associated with Alternate Modes.

In general, a [Structured VDM AMS](#) consists of a Command Request and a Command response (ACK, NAK or BUSY). A [Structured VDM AMS](#) is deemed to be completed (and if applicable, the transition to the requested functionality is made) when the Responder's Command response has been successfully transmitted.

The [Attention](#) AMS is an exception to this Request/response flow. See [Section 8.6.6](#).

[Table 8.1](#) details the responses a Responder **May** issue to each Command Request. Responses not listed for a given Command **Shall Not** be sent by a Responder. An Initiator that receives a NAK response **Should** take that response as an indication not to retry that particular Command REQ.

Table 8.1. Commands and Responses

Command	Allowed Response	Reference
Discover Identity	ACK, NAK, BUSY	Section 8.6.1
Discover SVIDs	ACK, NAK, BUSY	Section 8.6.2
Discover Modes	ACK, NAK, BUSY	Section 8.6.3
Enter Mode	ACK, NAK	Section 8.6.4
Exit Mode	ACK, NAK	Section 8.6.5
Attention	None	Section 8.6.6

All Ports that support Modal Operation **Shall** support the [Discover Identity](#), [Discover SVIDs](#), the [Discover Modes](#), the [Enter Mode](#) and [Exit Mode](#) Commands.

The Responder **Shall** respond with:

- ACK if it recognizes the SVID and is able to process it at the expected time.
- NAK if any of the following are true:
 - It does not recognize the SVID.
 - It does not support the Command with respect to the SVID.
 - It recognizes the SVID but cannot process the Command Request.
 - A VDO in the Command Request contains a field which is **Invalid**.
 - Structured VDMs are supported, but the [Structured VDM](#) Command Request is an Unrecognized Message.
- BUSY if it recognizes the SVID and the Command but cannot process the Command Request at the present time.

The Initiator **Shall** wait [tVDMBusy](#) after a "Responder BUSY" response is received before retrying the Command Request.

The ACK, NAK or BUSY response **Shall** contain the same SVID as the Command Request.

The Responder **Shall** respond to:

- [Enter Mode](#) requests within [tVDMEnterMode](#) .
- [Exit Mode](#) requests within [tVDMExitMode](#) .
- Other requests within [tVDMReceiverResponse](#) .

An Initiator not receiving a response within the expected allotted time **Shall** generate a timeout and return to either the PE_SRC_Ready or PE_SNK_Ready State (as appropriate):

- [Enter Mode](#) requests within [tVDMWaitModeEntry](#) .
- [Exit Mode](#) requests within [tVDMWaitModeExit](#) .

- Other requests within [tVDMSenderResponse](#) .

8.6.1. Discover Identity

8.6.1.1. SOP Use

The [Discover Identity](#) Command, sent to the Port Partner, enables a Port to identify its Port Partner and determine its suitability for [USB4] or Alternate Mode entry, among other Capabilities.

The following products **Shall** respond with a [Discover Identity](#) Command ACK in response to a [Discover Identity](#) Command Request sent to SOP:

- A PD-Capable UFP that supports Modal Operation.
- A PD-Capable product that has multiple DFPs.
- A PD-Capable [USB4] product.

8.6.1.2. SOP' Usage

The [Discover Identity](#) Command that is sent to SOP', enables a Port to identify a cable and determine its suitability for [USB4] or Alternate Mode entry, similarly to SOP use with respect to the Port Partner.

Additionally, the [Discover Identity](#) Command sent to SOP', enables a Port to, among other things:

- Determine whether a cable or VPD is PD-Capable.
- Establish a Specification Revision level with the cable.
- Determine the cable's maximum supported VBUS voltage and current.
- Determine whether the cable supports EPR.

A PD-Capable Cable Plug or VPD **Shall** support the [Discover Identity](#) Command.

8.6.1.2.1. PD Capability

A Port that communicates with Cable Plug(s) **Shall** use the following process to determine whether a given Cable Plug or VPD is PD Capable before communicating with the Cable Plug(s)

- The Initial Source/VCONN Source **Shall** initiate a Discover Identity REQ Message to SOP' during Default Contract. The Cable Plug and Sink/UFP will power up independently after the Port sources VBUS and VCONN, and either target might be ready for PD Communication first. The Source/VCONN Source **May** initiate additional SOP' [Discover Identity](#) REQ messages until it receives a valid [GoodCRC Message](#), subject to the limits described below.

Note: A Cable Plug or VPD will not be ready for PD Communication until tVCONNStable after VCONN has been applied (see [USB-C]).

- Prior to the initial power Contract, the initial VCONN Source **May** continue to send SOP' [Discover Identity](#) REQ until the Initial Source receives a [GoodCRC](#), up to a maximum of [nDiscoverIdentityCount](#) times. While doing this, the Port must continue to send Source Capabilities at the times required by [SourceCapabilityTimer](#) .
- If a Port that communicates with the Cable Plug(s) enters an Explicit Contract before receiving a [GoodCRC Message](#) for SOP' [Discover Identity](#), it **Shall** continue to send SOP' Discover Identity REQ up to a maximum

of [nDiscoverIdentityCount](#) times at a rate defined by the [DiscoverIdentityTimer](#), while the Port is VCONN Source.

- If the Cable Plug or VPD does not respond with a [GoodCRC Message](#) before [nDiscoverIdentityCount](#) expires, the Port **Shall** consider the Cable Plug or VPD not to be PD Capable, and the Port **Shall Not** send any further messages to SOP or SOP' until a Detach, [Hard Reset](#), or Error Recovery happens.
- If a Cable Plug does not respond to a Revision 3.x Discover Identity REQ with a [Discover Identity](#) ACK or BUSY, the VCONN Source **May** repeat the [Discover Identity](#) REQ using PD Revision 2.0 before establishing that the Cable Plug is not PD Capable, until [nDiscoverIdentityCount](#) expires.
- If a Port does not receive a [GoodCRC Message](#) for an SOP' Discover Identity REQ, it **Shall** not send an SOP' [Soft Reset](#), because the Cable or VPD **May Not** be PD Capable.
- If the Cable Plug or VPD responds to a [Discover Identity](#) REQ with a [GoodCRC Message](#) and a [Discover Identity](#) ACK or BUSY, the initiating Port **Shall** consider the Responder to be PD Capable.

See [Figure 9.76](#) and [Figure 9.93](#).

Note: If a Port becomes VCONN Source via a [VCONN Swap](#), it **Shall** issue an SOP' and/or SOP'' [Soft Reset Message](#) before otherwise communicating with the Cable Plug(s). If a Cable Plug responds to this [Soft Reset](#) with a [GoodCRC Message](#), the Port **May** consider the Responder to be PD Capable. See [Section 7.1.1](#).

During Implicit Contract, the VCONN Source **May** Initiate SOP' [Discover Identity](#), subject to the rules in this section.

This process is closely linked to the process for establishing a common Specification Revision with a Cable Plug. The Port **Shall** reset its PD Capability determination whenever it resets its Specification Revision determination. (See [Section 6.1.3](#).)

8.6.1.2.2. Maximum Voltage and Current Capability

Standard USB Type-C cable assemblies are rated for PD voltages higher than [vSafe5V](#) and current levels of at least 3A (See [USB-C]). The Source **Shall** limit maximum Capabilities it offers so as not to exceed the Capabilities of the type of cabling detected, using the process described here.

The Cable VDO, returned as part of the [Discover Identity](#) ACK, specifies the maximum current and voltage values that the Responder supports. A Source capable of offering more than 3A **Shall** discover the Attached cable via the SOP' [Discover Identity](#) Command and limit its Source Capabilities based on the maximum current supported by the cable. ([Section 8.6.1](#)).

8.6.1.2.3. Cable EPR Capability

The Cable VDO specifies whether or not the cable is EPR-capable. An EPR Source **Shall** discover the Attached cable via the SOP' [Discover Identity](#) Command and limit its Source Capabilities based on the cable's EPR support. (See [Section 7.30.1](#).)

Any additional VDOs received by the Initiator **Shall** be Ignored.

8.6.2. Discover SVIDs

A Responder that does not support any SVIDs **Shall** return a NAK.

If the Responder supports 12 or more SVIDs, then the Initiator **Shall** repeat the [Discover SVIDs](#) REQ until the Responder responds with a [Discover SVIDs](#) ACK ending either with a SVID value of 0x0000 in the last part of the last VDO or with a VDO containing two SVIDs with values of 0x0000. Each Discover SVID ACK Message, other than the one containing the terminating 0x0000 SVID, **Shall** convey 12 SVIDs. The Responder **Shall** restart the list of SVIDs each time a Discover Identity Command Request is received from the Initiator.

8.6.2.1. SOP' Usage

Note: Since a Cable Plug does not retry Messages if the [GoodCRC Message](#) from the Initiator becomes corrupted the Cable Plug will consider the [Discover SVIDs](#) Command ACK unsent and will send the same list of SVIDs again.

8.6.3. Discover Modes

A Responder that does not support any Modes **Shall** return a NAK.

8.6.4. Enter Mode Command

A Device **May** support multiple Modes with one or more active at any point in time. Any interactions between them are the responsibility of the relevant Standards or Vendors. Where there are multiple Active Modes at the same time Modal Operation **Shall** start on entry to the first Alternate Mode.

On receiving an [Enter Mode](#) Command Request, the Responder **Shall** respond with either an ACK or a NAK response. The Responder is not allowed to return a BUSY response. The value in the Object Position field of the [Enter Mode](#) Command response **Shall** contain the same value as the received [Enter Mode](#) Command Request.

Before entering an Alternate Mode, by sending the [Enter Mode](#) Command Request that requires the reconfiguring of any pins on entry to that Alternate Mode, the Initiator **Shall** ensure that those pins being reconfigured are placed into the USB Safe State. Before entering an Alternate Mode that requires the reconfiguring of any pins, the Responder **Shall** ensure that those pins being reconfigured are placed into either USB operation or the USB Safe State.

If the Responder responds to the [Enter Mode](#) Command Request with an ACK, the Responder **Shall** enter the Alternate Mode before sending the ACK. The Initiator **Shall** enter the Alternate Mode on reception of the ACK. Successful transmission of the Message confirms to the Responder that the Initiator will enter an Active Mode.

If the Responder responds to the [Enter Mode](#) Command Request with a NAK, the Alternate Mode is not entered. If not presently in Modal Operation, the Initiator **Shall** return to USB operation. If not presently in Modal Operation, the Responder **Shall** remain in either USB operation or the USB Safe State.

If the Initiator fails to receive a response within [tVDMWaitModeEntry](#) it **Shall Not** enter the Alternate Mode but return to USB operation.

[Figure 8.1](#) shows the sequence of events during the transition between USB operation and entering an Alternate Mode. It illustrates when the Responder's Alternate Mode changes and when the Initiator's Alternate Mode changes. [Figure 8.2](#) illustrates that when the Responder returns a NAK the transition to an Alternate Mode does not take place and the Responder and Initiator remain in their default USB roles.

Figure 8.1. Successful Enter Mode sequence

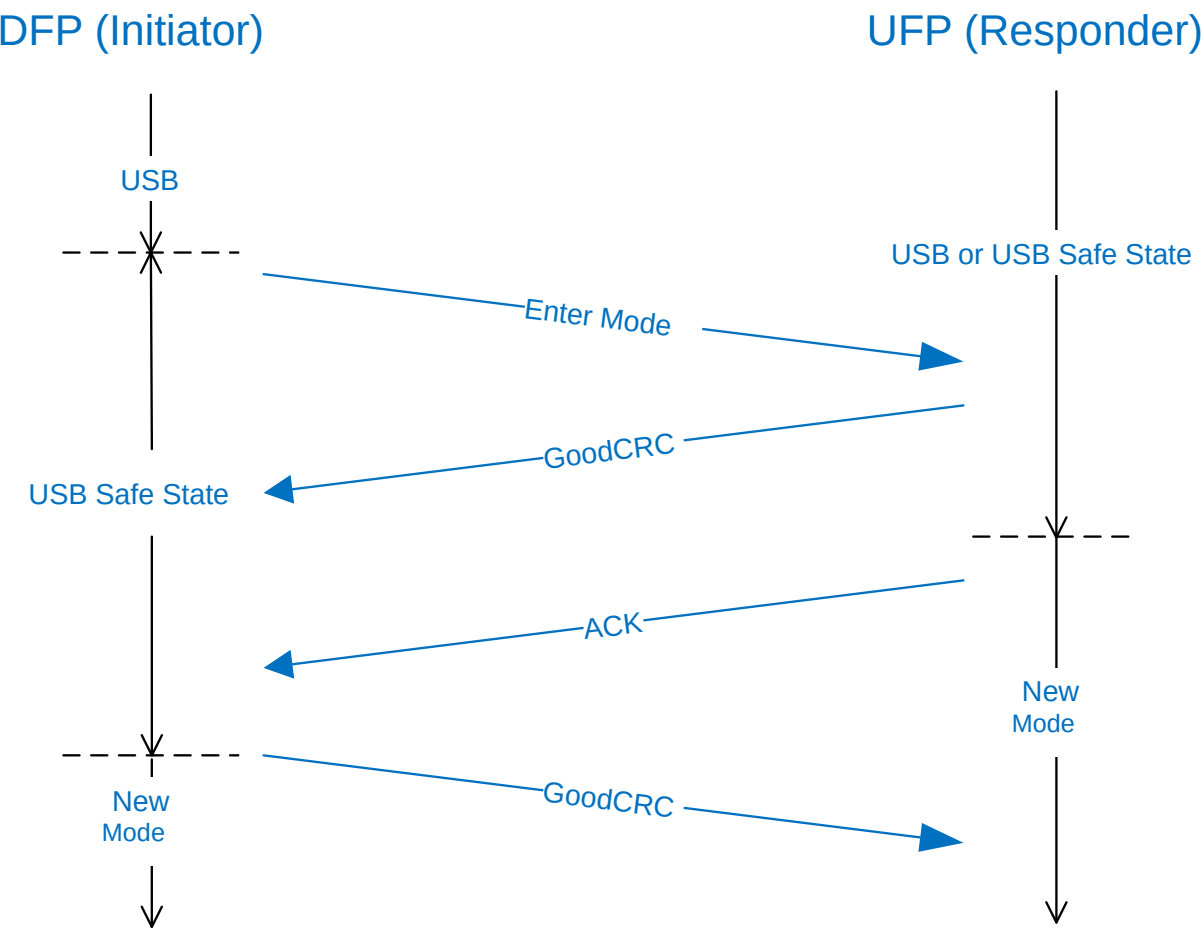
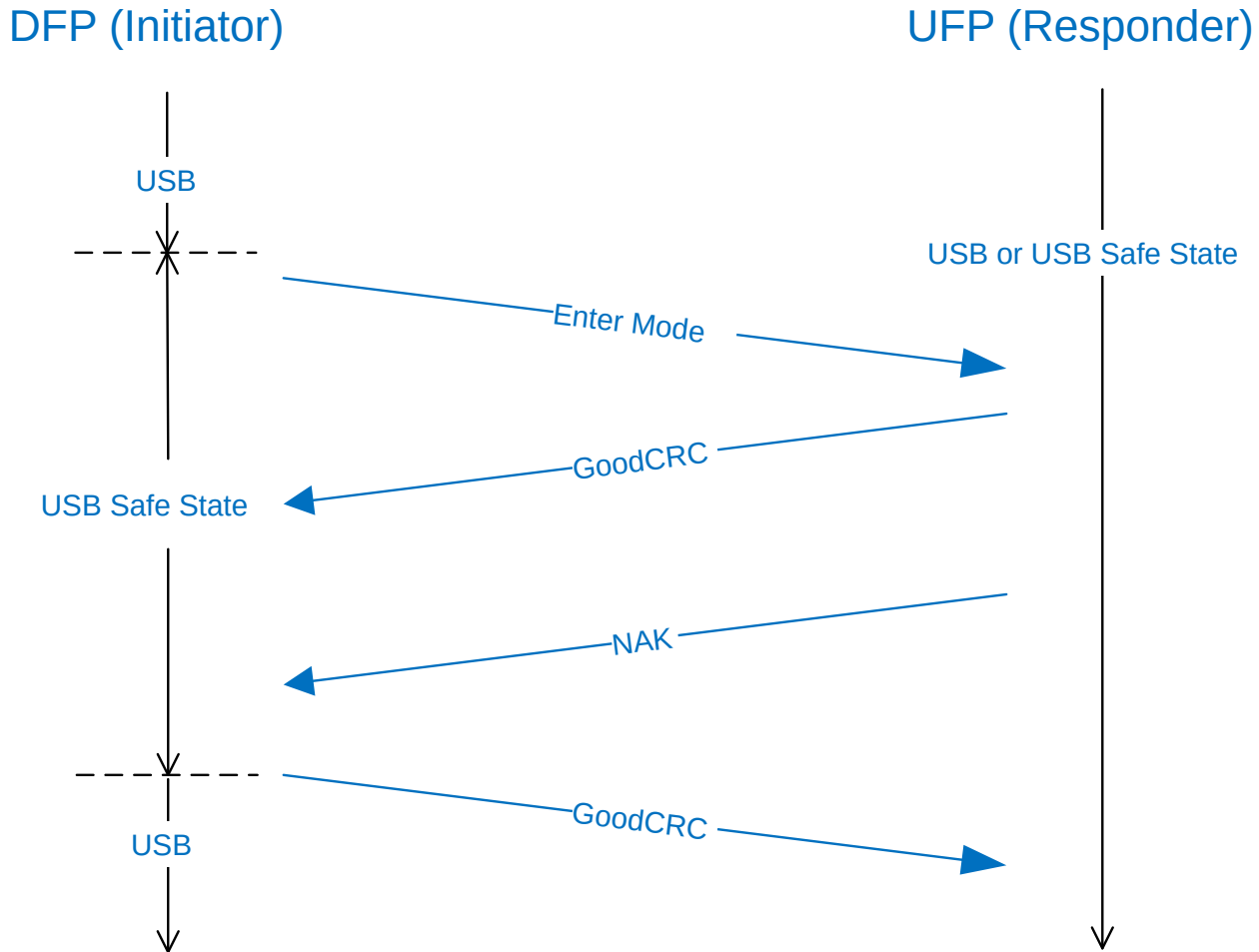


Figure 8.2. Unsuccessful Enter Mode sequence due to NAK

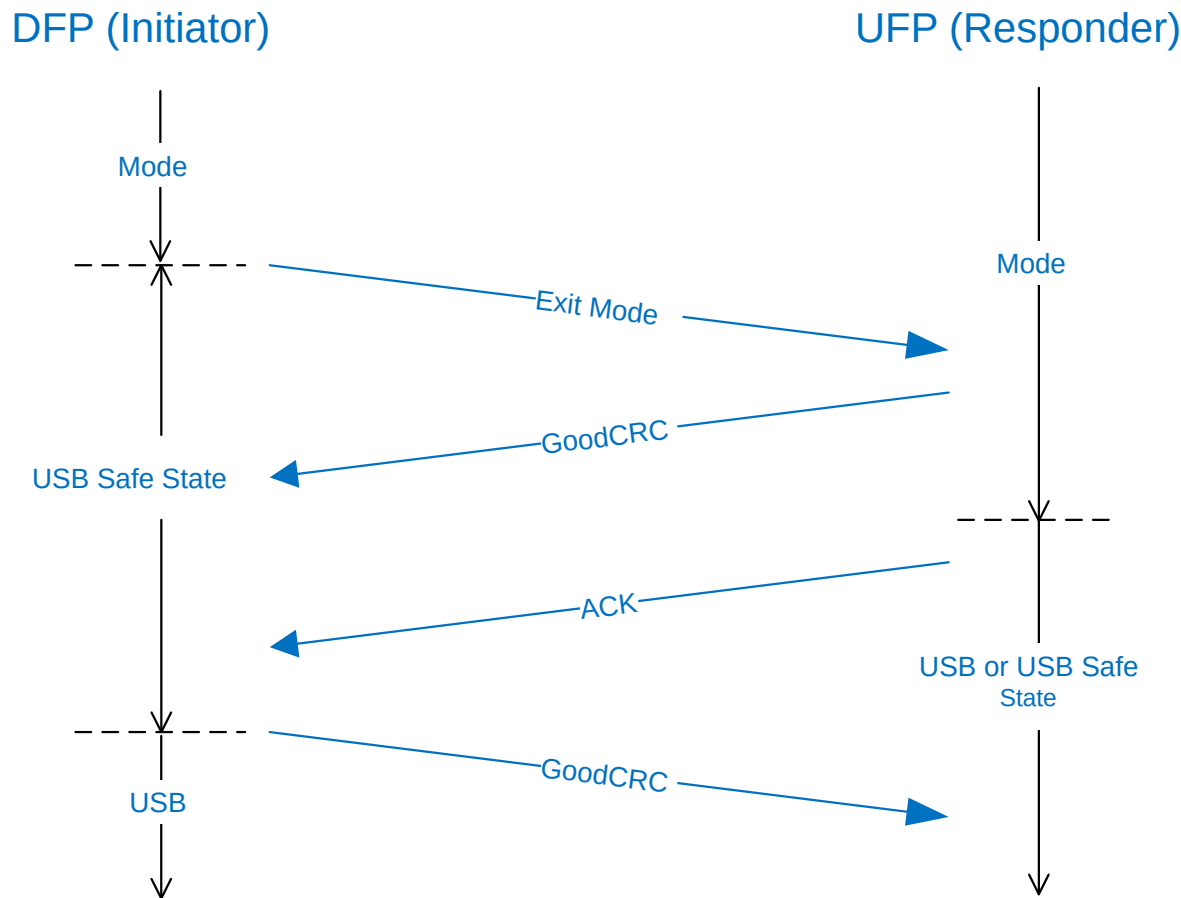
8.6.5. Exit Mode Command

Only the DFP **Shall** initiate the [Exit Mode](#) Process.

The Responder **Shall** exit its Active Mode before sending the response Message. The Initiator **Shall** exit its Active Mode when it receives the ACK. The Responder **Shall Not** return a BUSY acknowledgment and **Shall** only return a NAK acknowledgment to a Request not containing an Active Mode (i.e., **Invalid** object position). An Initiator which fails to receive an ACK within [tVDMWaitModeExit](#) or receives a NAK or BUSY response **Shall** exit its Active Mode.

[Figure 8.3](#), "[Exit Mode](#) sequence" shows the sequence of events during the transition between exiting an Active Mode and USB operation. It illustrates when the Responder's Alternate Mode changes and when the Initiator's Alternate Mode changes.

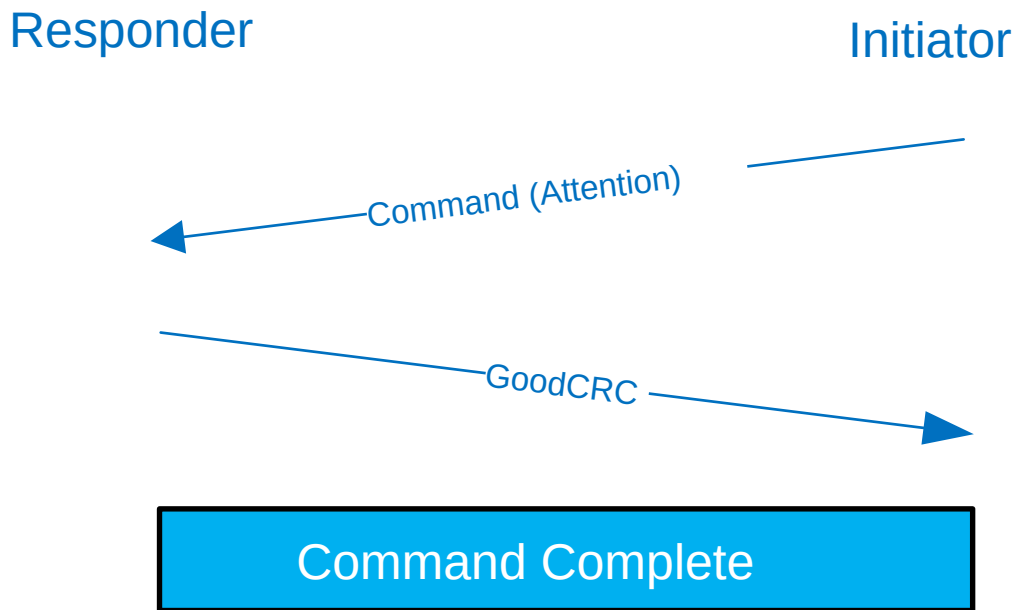
Figure 8.3. Exit Mode sequence



8.6.6. Attention

An Initiator **May** use the [Attention](#) Command to notify the Responder that it requires service.

A [Structured VDM Attention](#) AMS consists of a Command Request but no Command response. A [Structured VDM Attention](#) AMS is deemed to be completed when the [GoodCRC Message](#) has been successfully received by the Initiator in reply to its [Attention](#) Command Request.

Figure 8.4. Attention Command Request/response sequence

If Structured VDMs are supported, then no response **Shall** be made to an [Attention](#) Command.

If Structured VDMs are supported, but the [Structured VDM Attention](#) Command Request is an Unrecognized Message it **Shall** be **Ignored** (see [Table 8.1](#)).

8.7. Command Processes

8.7.1. Discovery Process

The Discovery Process has several phases. A Port **Shall** complete the Discovery Process to the extent that it requires the information contained in each Message for its normal operation. However, in each phase, if the Responder does not respond with [GoodCRC Message](#) or responds with an SVDM of Command Type NAK, or with a [Not_Supported Message](#), the Initiator **Shall Not** proceed to a phase of discovery contingent on the expected SVDM ACK.

8.7.1.1. Port Partner Discovery

1. The Initiator issues a [Discover Identity](#) REQ to determine the Responder's basic properties and Alternate Mode support. The Responder responds with [Discover Identity](#) ACK.
2. If the [Discover Identity](#) ACK sets Modal Operation Supported = 1 in the ID Header VDO, the Responder supports at least one Alternate Mode. The Initiator issues [Discover SVIDs](#) REQ to determine the list of SVIDs the Responder supports. If the Responder supports six or more SVIDs, the Initiator might issue multiple [Discover SVIDs](#) REQs.
3. The Initiator sends a [Discover Modes](#) REQ for any SVID supported by the Responder, and the Responder responds with a [Discover Modes](#) ACK containing Mode-specific information. The Initiator **May** initiate a [Discover Modes](#) REQ for any or all of the Responder's supported SVIDs.

8.7.1.2. Cable Plug Discovery

The Discovery Process with the Cable Plug as Responder is substantially similar to that with the Port Partner as Responder, with the following differences:

- The Initiator must be the VCONN Source to communicate with the Cable Plugs. (See [Section 7.13.](#))
- The Initiator **Shall Not** send discovery SVDM REQs to the Cable Plug at SOP". The Initiator **Shall** consider any responses to discovery SVDM REQs from the Cable Plug at SOP' to apply to the Cable Plug at SOP".
- The Initial Source/VCONN Source, if it communicates with the Cable Plug, **Shall** Initiate [Discover Identity](#) REQ to SOP' prior to the Initial Contract. (See [Section 8.6.1.2.1](#)).

8.7.2. Entering Alternate Modes

The result of the Discovery Process is that both the Initiator and Responder identify the Modes they mutually support. The Initiator (DFP), upon finding a suitable Alternate Mode, uses the [Enter Mode](#) Command to enable the Alternate Mode. Once the Alternate Mode is entered, the Device **Shall** remain in that Active Mode until the [Exit Mode](#) Command is successful (see [Section 8.6.5](#)).

A DFP **May** Initiate the [Enter Mode](#) Process with a Responder after it has successfully completed the Discovery Process with that Responder, subject to the specification of the Alternate Mode.

The Alternate Mode entry process with a Cable Plug as Responder is substantially similar to that with the Port Partner as Responder, with the following differences:

- The Initiator must be the VCONN Source to communicate with the Cable Plugs. (See [Section 7.13.](#))
- The specification for each Alternate Mode determines whether the DFP must complete the Enter Mode process on SOP' and SOP" and the timing of those processes relative to the Enter Mode process on SOP.

8.7.3. Exiting Alternate Modes

The Responder (UFP or Cable Plug) and Initiator continue using the Active Mode until the Active Mode is exited. Upon exiting an Alternate Mode, a Port or Cable Plug **Shall** exit any State entered via that Alternate Mode, including via Unstructured VDMs.

In a managed termination, using the [Exit Mode](#) Command, the Active Mode **Shall** be exited in a controlled manner as described in [Section 8.6.5](#) .

An unmanaged termination is triggered by one of the following conditions:

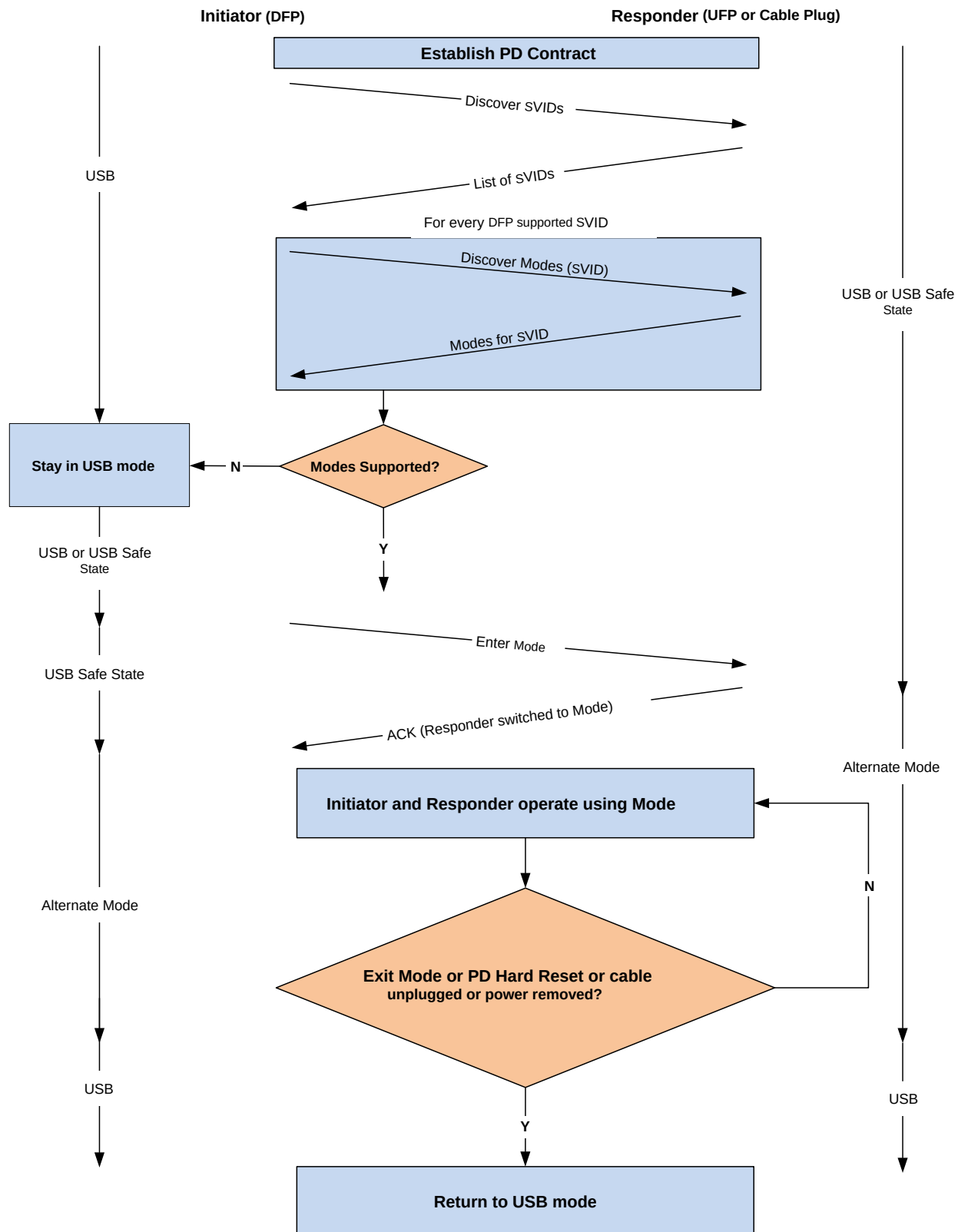
- [Data Reset](#)
- [Hard Reset](#)
- Error Recovery
- Detach

Upon an unmanaged termination, the Ports **Shall** exit all Active Modes but **Shall Not** transition through USB Safe State; the Cable Plugs **Shall** exit all Active Modes. The Cable Plugs **Shall** Exit any Active Modes upon a [Cable Reset](#).

Upon either type of termination, each Port **Shall** return to USB operation as defined in [USB-C] following an exit from an Alternate Mode.

The overall Message flow is illustrated in [Figure 8.5](#), "Enter/[Exit Mode](#) Process".

Figure 8.5. Enter/Exit Mode Process



The Initiator **Shall** return to USB Operation within [tVDMExitMode](#) of a disconnect, of Hard Reset Signaling being detected or Error Recovery.

The Responder **Shall** return to either USB operation or USB Safe State within [tVDMExitMode](#) of a disconnect, of [Hard Reset](#) Signaling being detected or Error Recovery.

Chapter 9. State Machine Diagrams

9.1. Protocol Layer State Diagrams

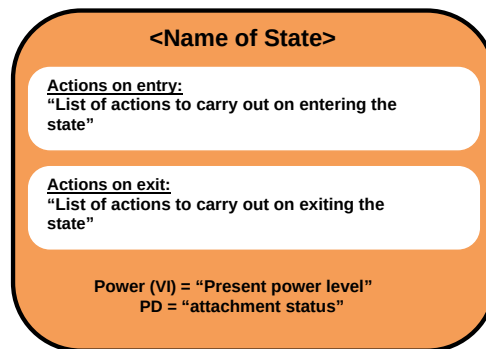
9.1.1. Introduction to Protocol Layer State Diagrams

These State diagrams define the operation of the Power Delivery Protocol Layer.

Note: These State diagrams are not intended to replace a well written and robust design.

[Figure 9.1](#) shows an outline of the states defined in the following sections. At the top there is the name of the State. This is followed by "Actions on entry" a list of actions carried out on entering the State and in some states "Actions on exit" a list of actions carried out on exiting the State.

Figure 9.1. Outline of States

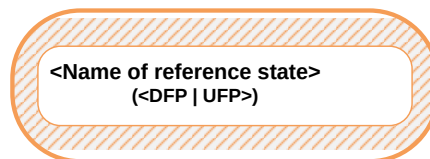


Transitions from one State to another are indicated by arrows with the conditions listed on the arrow. Where there are multiple conditions, these are Connected using either a logical OR "|" or a logical AND "&". The inverse of a condition is shown with a "NOT" in front of the condition.

In some cases, there are transitions which can occur from any State to a particular State. These are indicated by an arrow which is unconnected to a State at one end, but with the other end (the point) Connected to the final State.

In some State diagrams it is necessary to enter or exit from states in other diagrams. [Figure 9.2](#) indicates how such references are made. The reference is indicated with a hatched box. The box contains the name of the referenced State.

Figure 9.2. References to States



Timers are included in many of the states. Timers are initialized (set to their starting condition) and run (timer is counting) in the State it is referenced. As soon as the State is exited then the timer is no longer active. Timeouts of the timers are listed as conditions on State transitions.

Conditions listed on State transitions will come from one of three sources:

- Messages received from the PHY Layer.
- Events triggered within the Protocol Layer e.g., timer timeouts
- Message and related indications passed up to the Policy Engine from the Protocol Layer (Message sent; Message received etc.)

9.1.2. State Operation

The following section details Protocol Layer State Operation when sending and receiving SOP* Packets.

For each SOP' Communication being sent and received there **Shall** be separate Protocol Layer Transmission and Protocol Layer Reception and [Hard Reset](#) State Machine instances, with their own counter and timer instances. When Chunking is supported there **Shall** be separate Chunked Tx, Chunked Rx, and Chunked Message Router State Machine instances.

[Soft Reset](#) **Shall** only apply to the State Machine instances it is targeted at based on the type of SOP* Packet used to send the [Soft_Reset Message](#). The [Hard Reset](#) State Machine (including [Cable Reset](#)) **Shall** apply simultaneously to all Protocol Layer State Machine instances active in the DFP, UFP and Cable Plug (if present).

9.1.2.1. Protocol Layer Chunking

9.1.2.1.1. Architecture of Device Including Chunking Layer

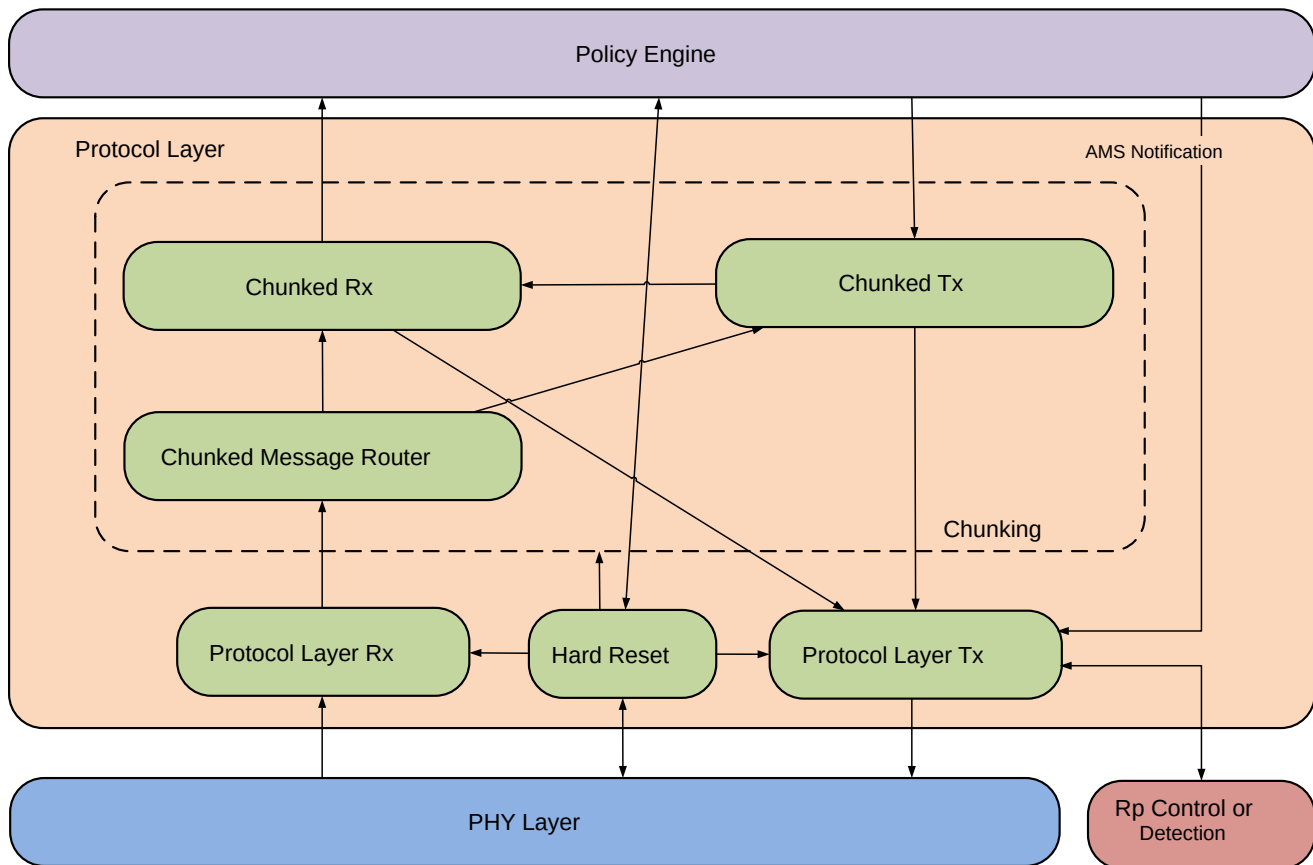
The Chunking component resides in the Protocol Layer between the Policy Engine and Protocol Tx/Rx. [Figure 9.3](#) illustrates the relationship between components.

The Chunking Layer comprises three related State machines:

- Chunked Rx.
- Chunked Tx.
- Chunked Message Router.

Note: The consequence of this architecture is that the Policy Engine deals entirely in Unchunked Messages. It will not receive (and might not respond to) a Message until all the related chunks have been collated.

If a PD Device or Cable Plug has no requirement to handle any Message requiring more than one Chunk of any [Extended Message](#), it **May** omit the Chunking Layer. In this case it **Shall** implement the ChunkingNotSupportedTimer to ensure compatible operation with partners which support Chunking (see [Section 7.31.15.1](#) and [Section 9.2.7](#)).

Figure 9.3. Chunking architecture Showing Message and Control Flow

9.1.2.1.1. Abort Mechanism

Long Chunked Messages bring with them the potential problem that they could prevent urgent Messages from being transmitted in a timely manner. An **Optional** Abort mechanism is provided to remedy this problem.

The Abort Flag referred to in the diagrams below **May** be set and examined by the Policy Engine. The specific means are left to the implementer.

9.1.2.1.1.2. Aborting Sending a Long-Chunked Message

A long-Chunked Message being sent **May** be aborted by setting the Abort Flag. The Message **Shall** be considered aborted when the Abort Flag is again cleared by the Chunked Tx State machine.

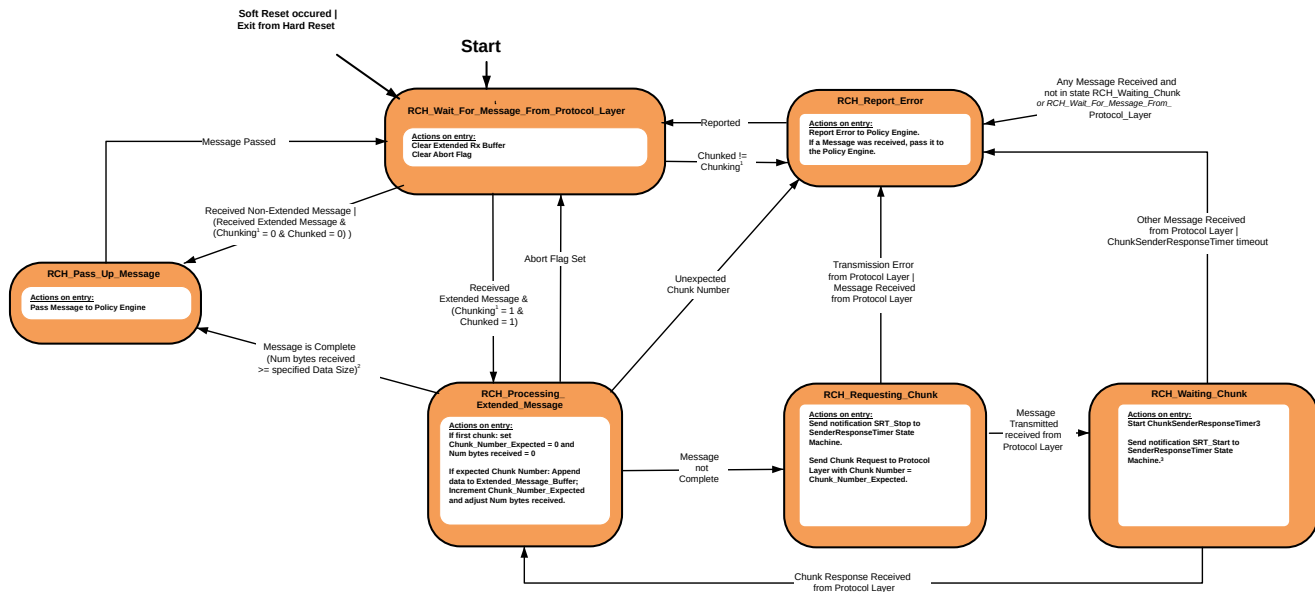
9.1.2.1.1.3. Aborting Receiving a Long-Chunked Message

If the Abort mechanism has been implemented, any Message sent while a Chunked Message receive is in progress will result in an error report being received by the Policy Engine, to indicate that the Message Request has been **Discarded**. If the Message was urgent the Policy Engine might set the Abort Flag, which will result in the incoming Chunked Message being aborted. The Abort Flag being cleared by the Chunked Rx State machine indicates that the urgent Message can now be sent.

9.1.2.1.2. Chunked Rx State Diagram

Figure 9.4 shows the State behavior for the Chunked Rx State Machine. This recognizes whether Chunked received Messages are involved and deals with requesting chunks when they are. It also performs validity checks on all Messages related to Chunking.

Figure 9.4. Chunked Rx State Diagram



1. Chunking is an internal State that is set to 1 if the 'Unchunked Extended Messages Supported' bit in either Source Capabilities or Request is 0. It defaults to 1 and is set after the first exchange of Source Capabilities and Request. It is also set to 1 for SOP' or SOP'' communication.
2. Additional bytes received over specified Data Size will be because of padding in the last Chunk.
3. This State is responsible for starting two timers of similar length. The implementor **Should** mitigate against more than one of these timers resulting in recovery action.

9.1.2.1.2.1. RCH_Wait_For_Message_From_Protocol_Layer State

The Chunked Rx State Machine **Shall** enter the RCH_Wait_For_Message_From_Protocol_Layer State:

- At startup.
- As a result of a [Soft Reset](#) occurring.
- On exit from a [Hard Reset](#).

On entry to the RCH_Wait_For_Message_From_Protocol_Layer State the Chunked Rx State machine clears the Extended Rx Buffer and clears the Abort Flag.

In the RCH_Wait_For_Message_From_Protocol_Layer State the Chunked Rx State machine waits until the Chunked

Message Router passes up a received Message.

The Chunked Rx State Machine **Shall** transition to the RCH_Pass_Up_Message State when:

- A non-[Extended Message](#) is passed up from the Chunked Message Router.
- An [Extended Message](#) is passed up from the Chunked Message Router, and the Policy Engine has determined that we are not doing Chunking, and the Message has its Chunked bit set to 0b.

The Chunked Rx State Machine **Shall** transition to the RCH_Processing_Extended_Message State when:

- An [Extended Message](#) is passed up from the Chunked Message Router, and the Policy Engine has determined that we are doing Chunking, and the Message has its Chunked bit set to 1b.

9.1.2.1.2.2. RCH_Pass_Up_Message State

On entry to the RCH_Pass_Up_Message State the Chunked Rx State machine **Shall** pass the received Message to the Policy Engine.

The Chunked Rx State Machine **Shall** transition to the RCH_Wait_For_Message_From_Protocol_Layer State when:

- The Message has been passed.

9.1.2.1.2.3. RCH_Processing_Extended_Message State

On entry to the RCH_Processing_Extended_Message State the Chunked Rx State machine **Shall**:

- If this is the first Chunk:
 - Set Chunk_Number_Expected = 0.
 - Set Num bytes received = 0.
- If Chunk contains the expected Chunk Number:
 - Append its data to the Extended_Message_Buffer.
 - Increment Chunk_Number_Expected.
 - Adjust Num bytes received.

The Chunked Rx State Machine **Shall** transition to the RCH_Pass_Up_Message State when:

- The Message is complete (i.e., Num bytes received \geq specified Data Size).

Note: The inequality allows for padding bytes in the last Chunk, which are not actually part of the Extended Message).

The Chunked Rx State Machine **Shall** transition to the RCH_Requesting_Chunk State when:

- The Message is not yet complete.

The Chunked Rx State Machine **Shall** transition to the RCH_Report_Error State when:

- An unexpected Chunk Number is received.

The Chunked Rx State Machine **Shall** transition to the RCH_Wait_For_Message_From_Protocol_Layer State when:

- The Abort Flag is set.

9.1.2.1.2.4. RCH_Requesting_Chunk State

On entry to the RCH_Requesting_Chunk State the Chunked Rx State machine **Shall**:

- Send notification SRT_Stop to SenderResponseTimer State machine (see [Section 9.2.2](#)).
- Send Chunk Request to Protocol Layer with Chunk Number = Chunk_Number_Expected. The Chunked Rx State Machine **Shall** transition to the RCH_Waiting_Chunk State when:
 - Message Transmitted is received from the Protocol Layer.

The Chunked Rx State Machine **Shall** transition to the RCH_Report_Error State when:

- Transmission Error is received from the Protocol Layer, or
- A Message is received from the Protocol Layer.

9.1.2.1.2.5. RCH_Waiting_Chunk State

On entry to the RCH_Waiting_Chunk State the Chunked Rx State machine **Shall**:

- Start the ChunkSenderResponseTimer.
- Send notification SRT_Start to SenderResponseTimer State machine (see [Section 9.2.2](#)).

The Chunked Rx State Machine **Shall** transition to the RCH_Processing_Extended_Message State when:

- A Chunk is received from the Protocol Layer.

The Chunked Rx State Machine **Shall** transition to the RCH_Report_Error State when:

- A Message, other than a Chunk, is received from the Protocol Layer, or
- The ChunkSenderResponseTimer expires.

9.1.2.1.2.6. RCH_Report_Error State

The Chunked Rx State Machine **Shall** enter the RCH_Report_Error State:

- When any Message is received and the Chunked Rx State Machine is not in one of the states RCH_Waiting_Chunk or RCH_Wait_For_Message_From_Protocol_Layer.

On entry to the RCH_Report_Error State the Chunked Rx State machine **Shall**:

- Report the error to the Policy Engine.
- If the State was entered because a Message was received, this Message **Shall** be passed to the Policy Engine.

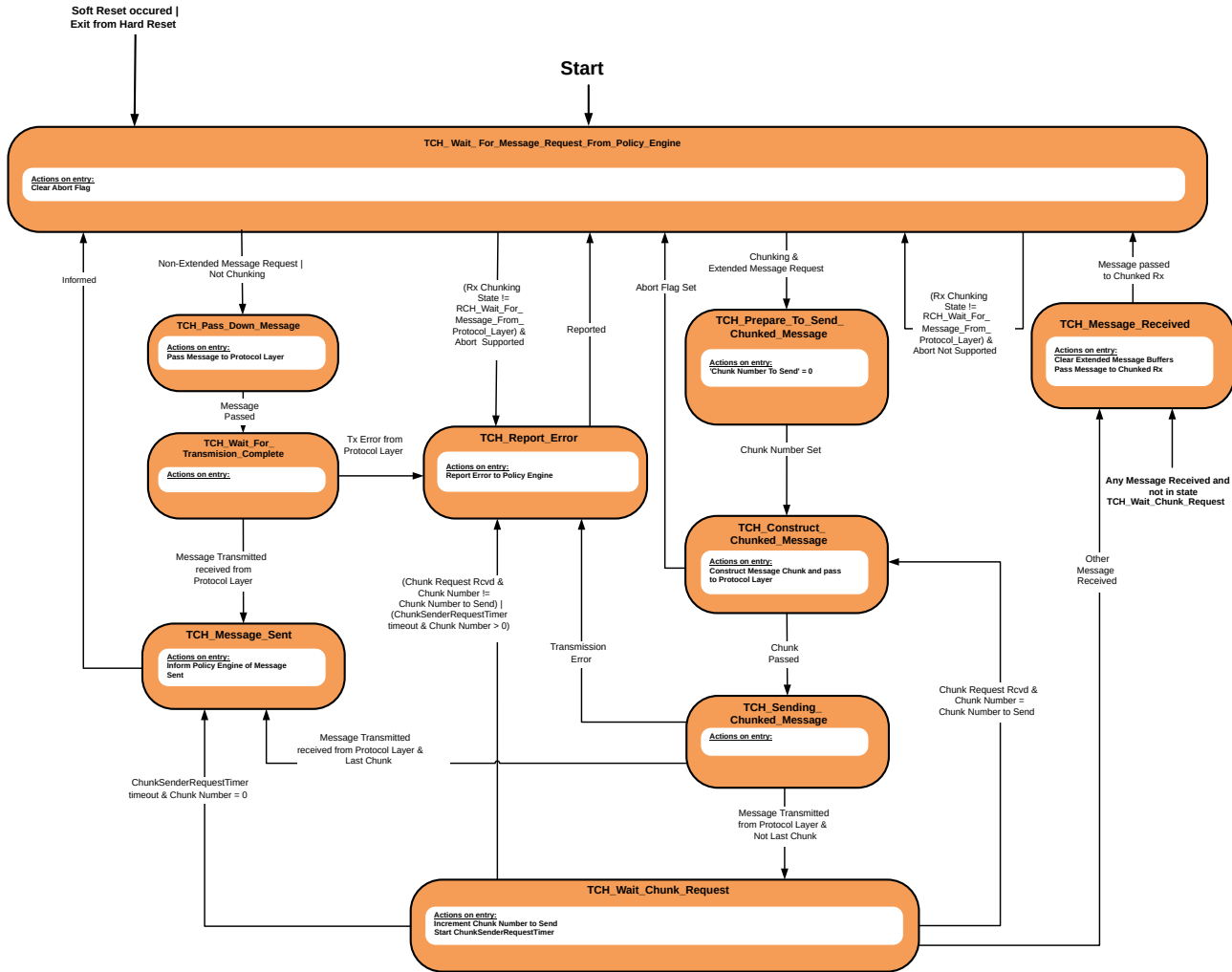
The Chunked Rx State Machine **Shall** transition to the RCH_Wait_For_Message_From_Protocol_Layer State when:

- The error has been reported.
- Any Message received was passed to the Policy Engine.

9.1.2.1.3. Chunked Tx State Diagram

- [Figure 9.5](#) shows the State behavior for the Chunked Tx State Machine. This recognizes whether Chunked transmitted Messages are involved and deals with sending chunks and waiting for Chunk requests when they are. It also performs validity checks on all related Messages related to Chunking.

Figure 9.5. Chunked Tx State Diagram



9.1.2.1.3.1. TCH_Wait_For_Message_Request_From_Policy_Engine State

The Chunked Tx State Machine **shall** enter the TCH_Wait_For_Message_Request_From_Policy_Engine State:

- At startup.
- As a result of a [Soft Reset](#) occurring.
- On exit from a [Hard Reset](#).

On entry to the TCH_Wait_For_Message_Request_From_Policy_Engine State the Chunked Tx State machine clears the Abort Flag.

In the TCH_Wait_For_Message_Request_From_Policy_Engine State the Chunked Tx State Machine waits until the Policy Engine sends it a Message Request.

The Chunked Tx State Machine **shall** transition to the TCH_Pass_Down_Message State when:

- A non-[Extended Message](#) Request is received from the Policy Engine, or
- A Message Request is received from the Policy Engine and the link is not Chunking.

The Chunked Tx State Machine **shall** transition to the TCH_Prepare_To_Send_Chunked_Message State when:

- An [Extended Message](#) Request is received from the Policy Engine, and the link is Chunking.

The Chunked Tx State Machine **Shall Discard** the Message Request and remain in the TCH_Wait_For_Message_Request_From_Policy_Engine State when:

- The Chunked Rx State is any other than RCH_Wait_For_Message_From_Protocol_Layer, and the Abort Flag has not been implemented.

The Chunked Tx State Machine **Shall Discard** the Message Request and enter the TCH_Report_Error State when:

- The Chunked Rx State is any other than RCH_Wait_For_Message_From_Protocol_Layer and the Abort Flag has been implemented.

9.1.2.1.3.2. TCH_Pass_Down_Message State

On entry to the TCH_Pass_Down_Message State the Chunked Tx State Machine **Shall** pass the Message to the Protocol Layer.

The Chunked Tx State Machine **Shall** transition to the TCH_Wait_For_Transmission_Complete State when:

- The Message has been passed to the Protocol Layer.

9.1.2.1.3.3. TCH_Wait_For_Transmission_Complete State

The Chunked Tx State Machine **Shall** transition to the TCH_Message_Sent State when:

- Message Transmitted has been received from the Protocol Layer.

The Chunked Tx State Machine **Shall** transition to the TCH_Report_Error State when:

- Transmission Error has been received from the Protocol Layer.

9.1.2.1.3.4. TCH_Message_Sent State

On entry to the TCH_Message_Sent State the Chunked Tx State Machine **Shall**:

- Inform the Policy Engine that the Message has been sent.

The Chunked Tx State Machine **Shall** transition to the TCH_Wait_For_Message_Request_From_Policy_Engine State when:

- The Policy Engine has been informed.

9.1.2.1.3.5. TCH_Prepare_To_Send_Chunked_Message State

On entry to the TCH_Prepare_To_Send_Chunked_Message State the Chunked Tx State Machine **Shall**:

- Set 'Chunk Number To Send' to zero.

The Chunked Tx State Machine **Shall** transition to the TCH_Construct_Chunked_Message State when:

- 'Chunk Number To Send' has been set to zero.

9.1.2.1.3.6. TCH_Construct_Chunked_Message State

On entry to the TCH_Construct_Chunked_Message State the Chunked Tx State Machine **Shall**:

- Construct a Message Chunk and pass it to the Protocol Layer.

The Chunked Tx State Machine **Shall** transition to the TCH_Sending_Chunked_Message State when:

- The Message Chunk has been passed to the Protocol Layer.

The Chunked Tx State Machine **Shall** transition to the TCH_Wait_For_Message_Request_From_Policy_Engine State when:

- The Abort Flag is set.

9.1.2.1.3.7. TCH_Sending_Chunked_Message State

The Chunked Tx State Machine **Shall** transition to the TCH_Wait_Chunk_Request State when:

- Message Transmitted is received from Protocol Layer and this was the last Chunk.

The Chunked Tx State Machine **Shall** transition to the TCH_Message_Sent State when:

- Message Transmitted is received from Protocol Layer and this was the last Chunk.

The Chunked Tx State Machine **Shall** transition to the TCH_Report_Error State when:

- Transmission Error has been received from the Protocol Layer.

9.1.2.1.3.8. TCH_Wait_Chunk_Request State

On entry to the TCH_Wait_Chunk_Request State the Chunked Tx State Machine **Shall**:

- Increment Chunk Number to Send.
- Start ChunkSenderRequestTimer.

The Chunked Tx State Machine **Shall** transition to the TCH_Report_Error State when:

- A Chunk Request has been received and the Chunk Number does not equal Chunk Number to Send or
- ChunkSenderRequestTimer has expired and Chunk Number is greater than zero.

The Chunked Tx State Machine **Shall** transition to the TCH_Message_Sent State when:

- ChunkSenderRequestTimer has expired and Chunk Number equals zero.

Note: This is the mechanism which allows the remote Port Partner or Cable Plug to omit the Chunking Layer. The Policy Engine will receive a Message Sent signal if the remote Port Partner or Cable Plug is present ([GoodCRC](#) Message received) but does not send a Chunk Request. After this the remote Port Partner will send a [Not_Supported Message](#), or the Cable Plug will **Ignore** the Chunked Message.

The Chunked Tx State Machine **Shall** transition to the TCH_Message_Received State when:

- Any other Message than Chunk Request is received.

9.1.2.1.3.9. TCH_Message_Received State

The Chunked Tx State Machine **Shall** enter the TCH_Message_Received State:

- When any Message is received, and the Chunked Tx State Machine is not in the TCH_Wait_Chunk_Request State.

On entry to the TCH_Message_Received State the Chunked Tx State Machine **Shall**:

- Clear the [Extended Message](#) Buffers.
- Pass the received Message to Chunked Rx Engine.

The Chunked Tx State Machine **Shall** transition to the TCH_Wait_For_Message_Request_From_Policy_Engine State when:

- The received Message has been passed to the Chunked Rx Engine.

9.1.2.1.3.10. TCH_Report_Error State

On entry to the TCH_Report_Error State the Chunked Tx State Machine **Shall**:

- Report the error to the Policy Engine.

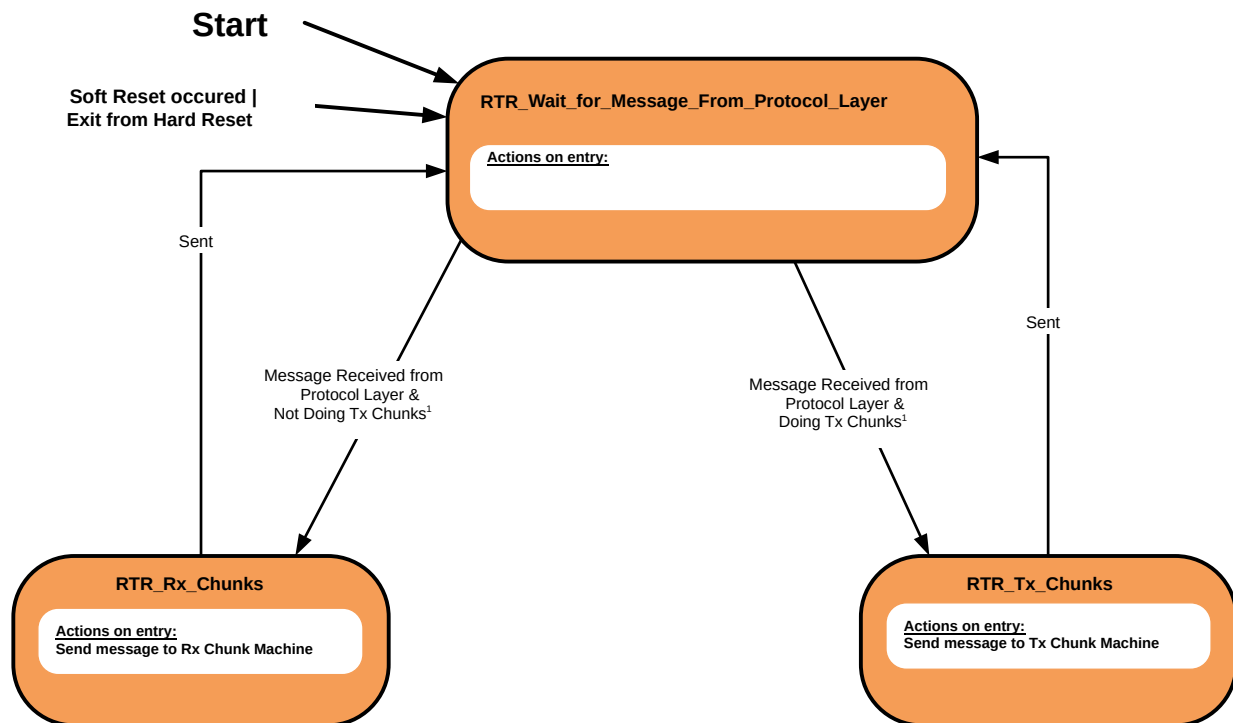
The Chunked Tx State Machine **Shall** transition to the TCH_Wait_For_Message_Request_From_Policy_Engine State when:

- The error has been reported.

9.1.2.1.4. Chunked Message Router State Diagram

[Figure 9.6](#) shows the State behavior for the Chunked Message Router. This determines to which State machine an incoming Message is routed to (Chunked Rx, Chunked Tx or direct to Policy Engine).

Figure 9.6. Chunked Message Router State Diagram



1. Doing Tx Chunks means that Chunked Tx State Machine is not in the TCH_Wait_For_Message_Request_From_Policy_Engine State.
2. Messages are taken to include notification about transmission success or otherwise of Messages.

9.1.2.1.4.1. RTR_Wait_for_Message_From_Protocol_Layer State

In the RTR_Wait_for_Message_From_Protocol_Layer State the Chunked Message Router waits until the Protocol Layer sends it a received Message.

The Chunked Message Router **Shall** transition to the RTR_Rx_Chunks State when:

- A Message is received from the Protocol Layer, and the combined Chunking is not doing Tx Chunks. The Chunked Message Router **Shall** transition to the RTR_Tx_Chunks State when:
 - A Message is received from the Protocol Layer, and the combined Chunking is doing Tx Chunks.

9.1.2.1.4.2. RTR_Rx_Chunks State

On entry to the RTR_Rx_Chunks State the Chunked Message Router **Shall**:

- Send the Message to the Chunked Rx State Machine.
- Transition to the RTR_Wait_for_Message_From_Protocol_Layer State.

9.1.2.1.4.3. RTR_Tx_Chunks State

On entry to the RTR_Tx_Chunks State the Chunked Message Router **Shall**:

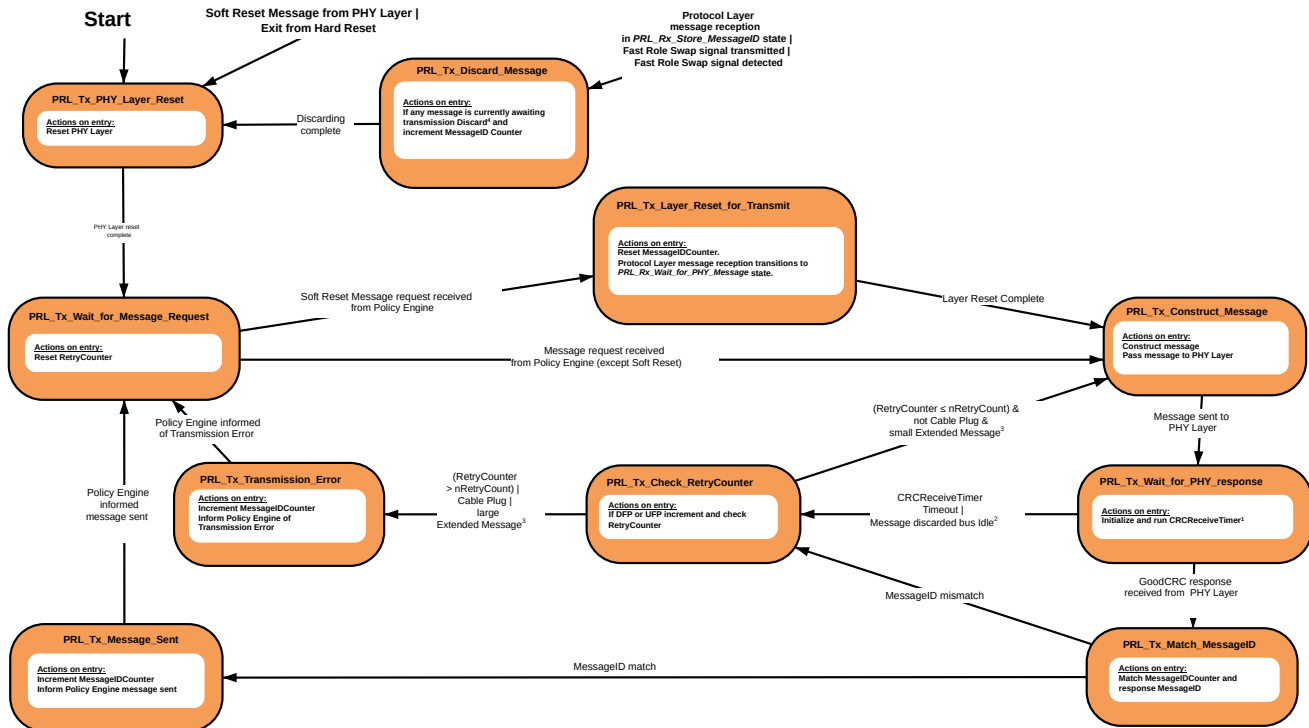
- Send the Message to the Chunked Tx State Machine.
- Transition to the RTR_Wait_for_Message_From_Protocol_Layer State.

9.1.2.2. Protocol Layer Message Transmission

9.1.2.2.1. Common Protocol Layer Message Transmission State Diagram

[Figure 9.7](#) shows the State behavior, common between the Source and the Sink, for the Protocol Layer when transmitting a Message.

Figure 9.7. Common Protocol Layer Message Transmission State Diagram



1. The CRCReceiveTimer is only started after the PHY has sent the Message. If the Message is not sent due to a busy channel, then the CRCReceiveTimer will not be started (see [Section 7.31.1](#)).

2. This indication is sent by the PHY Layer when a Message has been **Discarded** due to CC being busy, and after CC becomes Idle again (see [Section 5.2.2](#)). The CRCReceiveTimer is not running in this case since no Message has been sent.
3. A "small" [Extended Message](#) is either an [Extended Message](#) with Data Size ≤ MaxExtendedMsgLegacyLen bytes or an [Extended Message](#) with Data Size > MaxExtendedMsgLegacyLen bytes that has been Chunked. A "large" Extended Message is an [Extended Message](#) with Data Size > MaxExtendedMsgLegacyLen bytes that has not been Chunked.
4. See [Section 7.3](#) for details of when Messages are **Discarded**.

9.1.2.2.1.1. PRL_Tx_PHY_Layer_Reset State

The Protocol Layer **Shall** enter the PRL_Tx_PHY_Layer_Reset State:

- At startup.
- As a result of a [Soft Reset](#) Request being received by the PHY Layer.
- On exit from a [Hard Reset](#).

On entry to the PRL_Tx_PHY_Layer_Reset State the Protocol Layer **Shall** reset the PHY Layer (clear any outstanding messages and enable communications).

The Protocol Layer **Shall** transition to the PRL_Tx_Wait_for_Message_Request State when:

- When the PHY Layer reset is complete.

9.1.2.2.1.2. PRL_Tx_Wait_for_Message_Request State

In the PRL_Tx_Wait_for_Message_Request State the Protocol Layer waits until the Policy Engine directs it to send a Message.

- On entry to the PRL_Tx_Wait_for_Message_Request State the Protocol Layer **Shall** reset the RetryCounter.

The Protocol Layer **Shall** transition to the PRL_Tx_Construct_Message State when:

- A Message Request is received from the Policy Engine which is not a [Soft_Reset](#) Message. The Protocol Layer **Shall** transition to the PRL_Tx_Layer_Reset_for_Transmit State when:
 - A Message Request is received from the Policy Engine which is a [Soft_Reset](#) Message.

9.1.2.2.1.3. PRL_Tx_Layer_Reset_for_Transmit State

On entry to the PRL_Tx_Layer_Reset_for_Transmit State the Protocol Layer **Shall** reset the MessageIDCounter. The Protocol Layer **Shall** transition Protocol Layer Message reception to the PRL_Rx_Wait_for_PHY_Message State (see [Section 9.1.2.3.1](#)) in order to reset the stored MessageID.

The Protocol Layer **Shall** transition to the PRL_Tx_Construct_Message State when:

- The layer reset actions in this State have been completed.

9.1.2.2.1.4. PRL_Tx_Construct_Message State

On entry to the PRL_Tx_Construct_Message State the Protocol Layer **Shall** construct the Message requested by the Policy Engine, or resend a previously constructed Message, and then pass this Message to the PHY Layer. The Protocol Layer **Shall** transition to the PRL_Tx_Wait_for_PHY_Response State when:

- The Message has been sent to the PHY Layer.

9.1.2.2.1.5. PRL_Tx_Wait_for_PHY_Response State

On entry to the PRL_Tx_Wait_for_PHY_Response State, once the Message has been sent, the Protocol Layer **Shall** initialize and run the CRCReceiveTimer (see [Section 7.31.1](#)). The Protocol Layer **Shall** transition to the PRL_Tx_Match_MessageID State when:

- A [GoodCRC Message](#) response is received from the PHY Layer.

The Protocol Layer **Shall** transition to the PRL_Tx_Check_RetryCounter State when:

- The CRCReceiveTimer times out.
- Or the PHY Layer indicates that a Message has been **Discarded** due to the channel being busy but the channel is now Idle (see [Section 5.2.2](#)).

9.1.2.2.1.6. PRL_Tx_Match_MessageID State

On entry to the PRL_Tx_Match_MessageID State the Protocol Layer **Shall** compare the MessageIDCounter and the MessageID of the received [GoodCRC Message](#).

The Protocol Layer **Shall** transition to the PRL_Tx_Message_Sent State when:

- The MessageIDCounter and the MessageID of the received [GoodCRC Message](#) match. The Protocol Layer **Shall** transition to the PRL_Tx_Check_RetryCounter State when:
 - The MessageIDCounter and the MessageID of the received [GoodCRC Message](#) do not match.

9.1.2.2.1.7. PRL_Tx_Message_Sent State

On entry to the PRL_Tx_Message_Sent State the Protocol Layer **Shall** increment the MessageIDCounter and inform the Policy Engine that the Message has been sent.

The Protocol Layer **Shall** transition to the PRL_Tx_Wait_for_Message_Request State when:

- The Policy Engine has been informed that the Message has been sent.

9.1.2.2.1.8. PRL_Tx_Check_RetryCounter State

On entry to the PRL_Tx_Check_RetryCounter State the Protocol Layer in a DFP or UFP **Shall** increment the value of the RetryCounter and then check it in order to determine whether it is necessary to retry sending the Message.

Note: Cable Plugs do not retry Messages and so do not use the RetryCounter.

The Protocol Layer **Shall** transition to the PRL_Tx_Construct_Message State in order to retry Message sending when:

- $\text{RetryCounter} \leq \text{nRetryCount}$ and
- This is not a Cable Plug and
- This is an [Extended Message](#) with Data Size $\leq \text{MaxExtendedMsgLegacyLen}$ or
- This is an [Extended Message](#) that has been Chunked.

The Protocol Layer **Shall** transition to the PRL_Tx_Transmission_Error State when:

- RetryCounter > [nRetryCount](#) or
- This is a Cable Plug, which does not retry.
- This is an [Extended Message](#) with Data Size > MaxExtendedMsgLegacyLen that has not been Chunked.

9.1.2.2.1.9. PRL_Tx_Transmission_Error State

On entry to the PRL_Tx_Transmission_Error State the Protocol Layer **Shall** increment the MessageIDCounter and inform the Policy Engine of the transmission error.

The Protocol Layer **Shall** transition to the PRL_Tx_Wait_for_Message_Request State when:

- The Policy Engine has been informed of the transmission error.

PRL_Tx_Discard_Message State

Protocol Layer Message transmission **Shall** enter the PRL_Tx_Discard_Message State whenever:

- Protocol Layer Message reception receives an incoming Message or
- The [Fast Role Swap](#) Request is being transmitted (see [Section 10.3](#))
- The [Fast Role Swap](#) Request is detected (see [Section 10.3](#)).

On entry to the PRL_Tx_Discard_Message State, if there is a Message queued awaiting transmission, the Protocol Layer **Shall Discard** the Message according to the rules in [Section 5.2.2](#) and increment the MessageIDCounter.

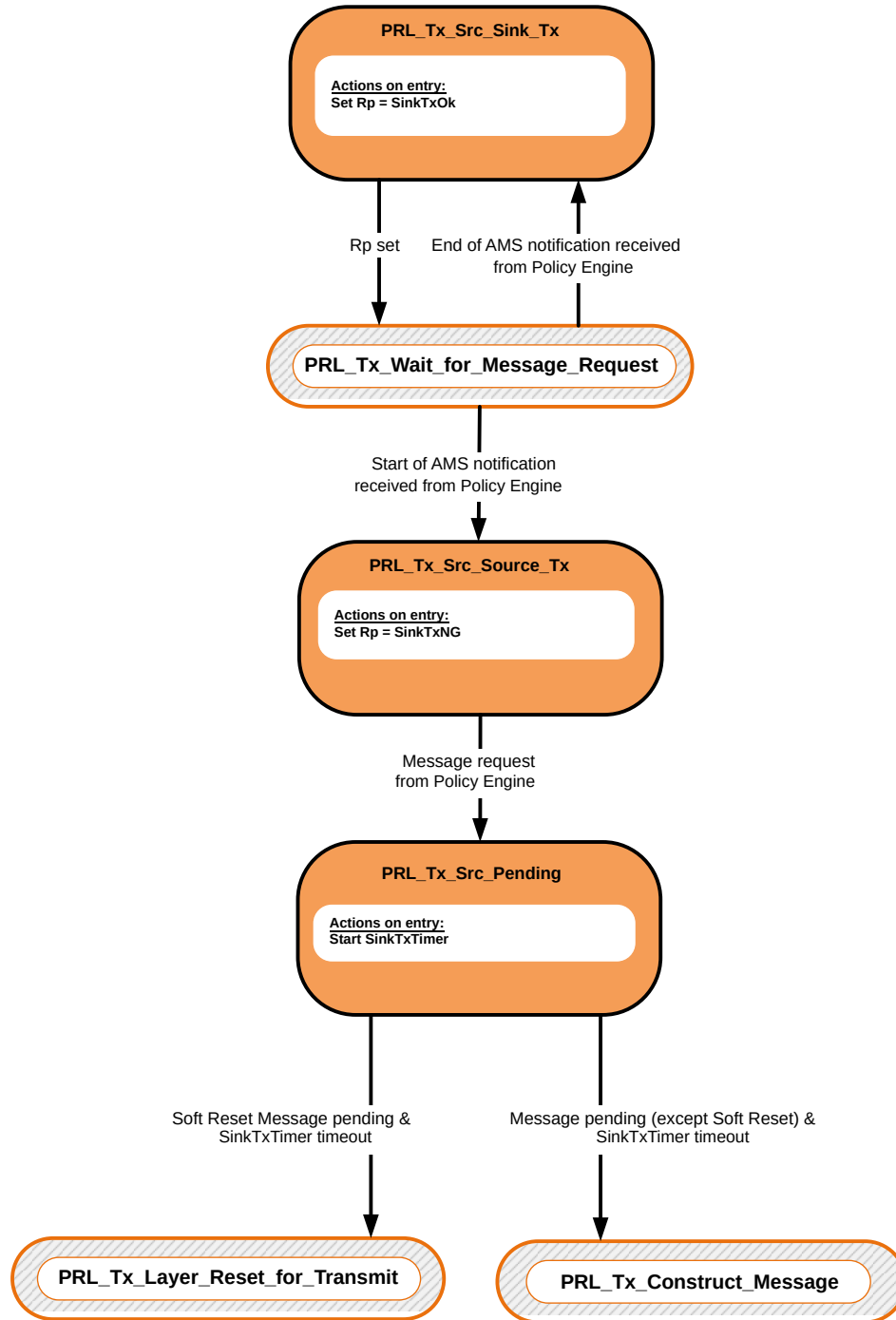
The Protocol Layer **Shall** transition to the PRL_Tx_PHY_Layer_Reset State when:

- Discarding is complete i.e., the Message queue is empty.

9.1.2.2.2. Source Protocol Layer Message Transmission State Diagram

[Figure 9.8](#) shows the State behavior for the Protocol Layer in a Source when transmitting a Message.

Figure 9.8. Source Protocol Layer Message Transmission State Diagram



9.1.2.2.2.1. PRL_Tx_Src_Sink_Tx State

In the PRL_Tx_Src_Sink_Tx State the Source sets Rp to SinkTxOK allowing the Sink to start an Atomic Message Sequence (AMS). The Protocol Layer in a Source **shall** transition from the PRL_Tx_Wait_for_Message_Request State to the PRL_Tx_Src_Sink_Tx State when:

- A notification is received from the Policy Engine that the end of an AMS has been reached.

On entry to the PRL_Tx_Src_Sink_Tx State the Protocol Layer **Shall** Request the PHY Layer to Rp to SinkTxOK. The Protocol Layer **Shall** transition to the PRL_Tx_Wait_for_Message_Request State when:

- Rp has been set.

9.1.2.2.2.2. PRL_Tx_Src_Source_Tx State

In the PRL_Tx_Src_Source_Tx State the Source sets Rp to SinkTxNG allowing the Source to start an Atomic Message Sequence (AMS). The Protocol Layer in a Source **Shall** transition from the PRL_Tx_Wait_for_Message_Request State to the PRL_Tx_Src_Source_Tx State when:

- A notification is received from the Policy Engine that an AMS will be starting.

On entry to the PRL_Tx_Src_Source_Tx State the Protocol Layer **Shall** set Rp to SinkTxNG. The Protocol Layer **Shall** transition to the PRL_Tx_Src_Pending State when:

- A Message Request is received from the Policy Engine.

9.1.2.2.2.3. PRL_Tx_Src_Pending State

In the PRL_Tx_Src_Pending State the Protocol Layer has a Message buffered ready for transmission. On entry to the PRL_Tx_Src_Pending State the SinkTxTimer **Shall** be initialized and run.

The Protocol Layer **Shall** transition to the PRL_Tx_Construct_Message State when:

- The pending Message Request from the Policy Engine is not a [Soft_Reset Message](#) and
- The SinkTxTimer times out.

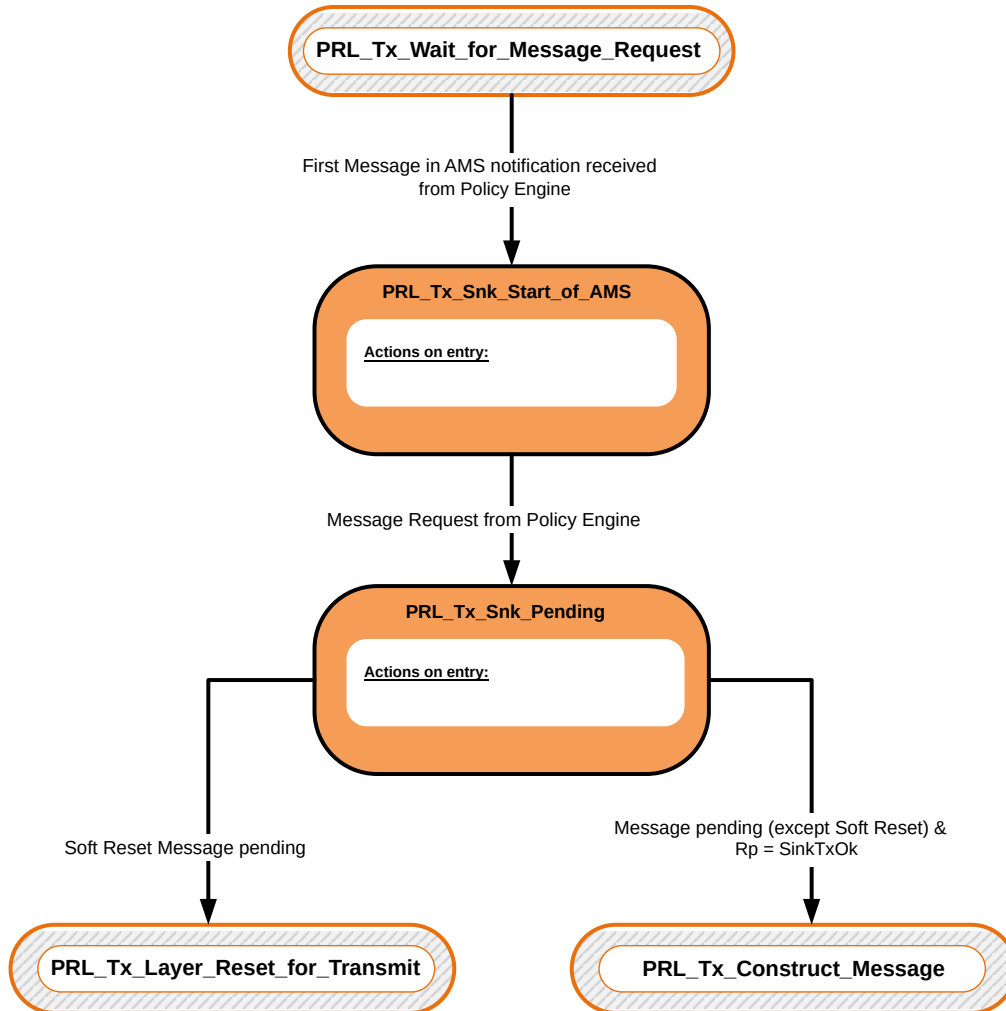
The Protocol Layer **Shall** transition to the PRL_Tx_Layer_Reset_for_Transmit State when:

- The pending Message Request from the Policy Engine is a [Soft_Reset Message](#) and
- The SinkTxTimer times out.

9.1.2.2.3. Sink Protocol Layer Message Transmission State Diagram

[Figure 9.9](#) shows the State behavior for the Protocol Layer in a Sink when transmitting a Message.

Figure 9.9. Sink Protocol Layer Message Transmission State Diagram



9.1.2.2.3.1. PRL_Tx_Snk_Start_of_AMS State

In the **PRL_Tx_Snk_Start_of_AMS** State the Protocol Layer waits for the first Message in a Sink initiated AMS. The Protocol Layer in a Sink **shall** transition from the **PRL_Tx_Wait_for_Message_Request** State to the **PRL_Tx_Snk_Start_of_AMS** State when:

- A notification is received from the Policy Engine that the next Message the Sink will send is the start of an AMS.

The Protocol Layer **shall** transition to the **PRL_Tx_Snk_Pending** State when:

- A Message Request is received from the Policy Engine.

9.1.2.2.3.2. PRL_Tx_Snk_Pending State

In the **PRL_Tx_Snk_Pending** State the Protocol Layer has the first Message in a Sink initiated AMS ready to send and is waiting for Rp to transition to SinkTxOK before sending the Message.

The Protocol Layer **shall** transition to the **PRL_Tx_Construct_Message** State when:

- A Message is Pending that is not a [Soft_Reset Message](#) and
- Rp is set to SinkTxOK.

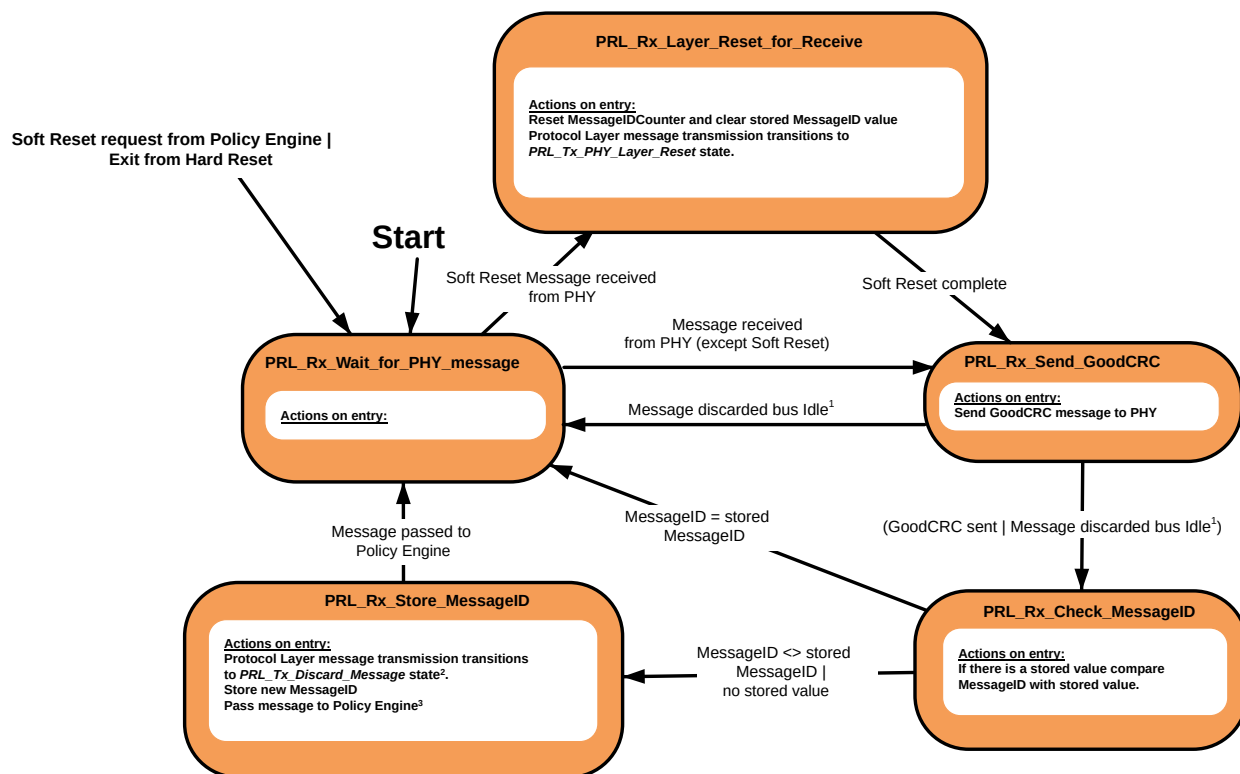
The Protocol Layer **Shall** transition to the PRL_Tx_Layer_Reset_for_Transmit State when:

- A [Soft_Reset Message](#) is pending.

9.1.2.3. Protocol Layer Message Reception

[Figure 9.10](#) shows the State behavior for the Protocol Layer when receiving a Message.

Figure 9.10. Protocol Layer Message reception



1. This indication is sent by the PHY when a Message has been **Discarded** due to CC being busy, and after CC becomes Idle again (see [Section 5.2.2](#)). Two alternate allowable transitions are shown.
2. In the case of a [Ping Message](#) being received, in order to maintain robust communications in the presence of collisions, the outgoing Message **Should Not** be **Discarded**.
3. See [Section 7.3](#) for details of when Messages are discarded.

9.1.2.3.1. PRL_Rx_Wait_for_PHY_Message State

The Protocol Layer **Shall** enter the PRL_Rx_Wait_for_PHY_Message State:

- At startup.
- As a result of a [Soft_Reset](#) Request from the Policy Engine.
- On exit from a [Hard_Reset](#).

In the PRL_Rx_Wait_for_PHY_Message State the Protocol Layer waits until the PHY Layer passes up a received Message.

The Protocol Layer **Shall** transition to the PRL_Rx_Send_GoodCRC State when:

- A Message is passed up from the PHY Layer.

The Protocol Layer **Shall** transition to the PRL_Rx_Layer_Reset_for_Receive State when:

- A [Soft_Reset Message](#) is received from the PHY Layer.

9.1.2.3.2. PRL_Rx_Layer_Reset_for_Receive State

On entry to the PRL_Rx_Layer_Reset_for_Receive State the Protocol Layer **Shall** reset the MessageIDCounter and clear the stored MessageID. The Protocol Layer **Shall** transition Protocol Layer Message transmission to the PRL_Tx_Wait_for_Message_Request State (see [Section 9.1.2.2.1.2](#)).

The Protocol Layer **Shall** transition to the PRL_Rx_Send_GoodCRC State when:

- The [Soft_Reset](#) actions in this State have been completed.

9.1.2.3.3. PRL_Rx_Send_GoodCRC State

On entry to the PRL_Rx_Send_GoodCRC State the Protocol Layer **Shall** construct a [GoodCRC Message](#) and Request the PHY Layer to transmit it. The Protocol Layer **Shall** transition to the PRL_Rx_Check_MessageID State when:

- The [GoodCRC Message](#) has been passed to the PHY Layer.

When the PHY Layer indicates that a Message has been **Discarded** due to CC being busy but CC is now Idle (see [Section 5.2.2](#)), the Protocol Layer **Shall** either:

- Transition to the PRL_Rx_Check_MessageID State or
- Transition to the PRL_Rx_Wait_for_PHY_Message State.

9.1.2.3.4. PRL_Rx_Check_MessageID State

On entry to the PRL_Rx_Check_MessageID State the Protocol Layer **Shall** compare the MessageID of the received Message with its stored value if a value has previously been stored.

The Protocol Layer **Shall** transition to the PRL_Rx_Wait_for_PHY_Message State when:

- The MessageID of the received Message equals the stored MessageID value since this is a Message retry which **Shall** be **Discarded**.

The Protocol Layer **Shall** transition to the PRL_Rx_Store_MessageID State when:

- The MessageID of the received Message does not equal the stored MessageID value since this is a new Message or
- This is the first received Message and no MessageID value is currently stored.

9.1.2.3.5. PRL_Rx_Store_MessageID State

On entry to the PRL_Rx_Store_MessageID State the Protocol Layer **Shall** transition Protocol Layer Message transmission to the PRL_Tx_Discard_Message State, replace the stored value of MessageID with the value of MessageID in the received Message and pass the Message up to the Policy Engine.

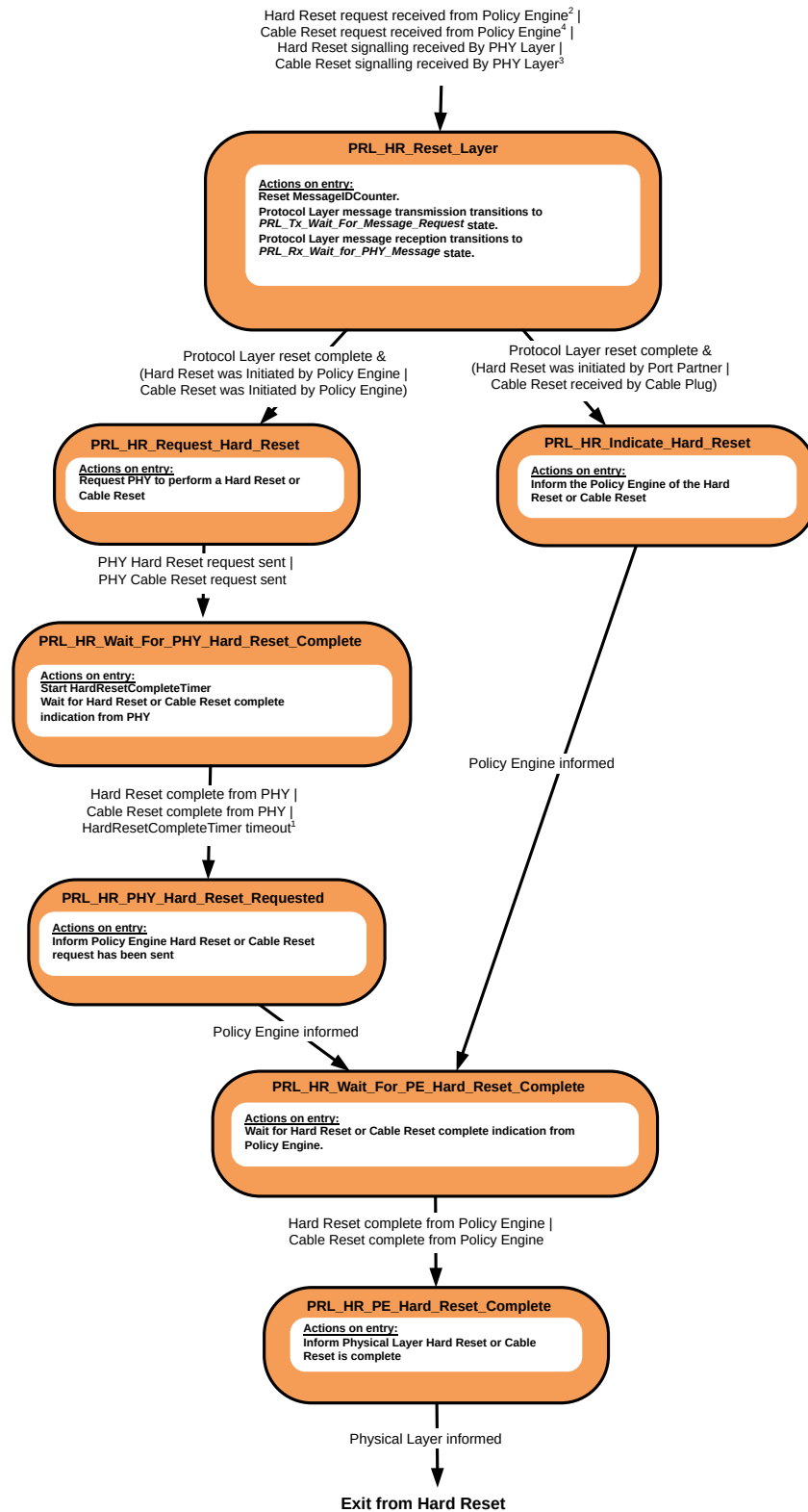
The Protocol Layer **Shall** transition to the PRL_Rx_Wait_for_PHY_Message State when:

- The Message has been passed up to the Policy Engine.

9.1.2.4. Hard Reset operation

[Figure 9.11](#) shows the State behavior for the Protocol Layer when receiving a [Hard Reset](#) or Cable Reset Request from the Policy Engine or [Hard Reset](#) Signaling or [Cable Reset](#) Signaling from the PHY Layer (see also [Section 7.1.3](#) and [Section 7.1.4](#)).

Figure 9.11. Hard/Cable Reset



1. If the HardResetCompleteTimer timeout occurs this means that the PHY is still waiting to send the [Hard Reset](#) due to a non-Idle channel. This condition will be cleared once the PE [Hard Reset](#) is completed.
2. Cable Plugs do not generate [Hard Reset](#) Signaling but are required to monitor for [Hard Reset](#) Signaling between the Port Partners and respond by resetting.
3. [Cable Reset](#) Signaling is only recognized by a Cable Plug.
4. [Cable Reset](#) Signaling cannot be generated by Cable Plugs.

9.1.2.4.1. PRL_HR_Reset_Layer State

The PRL_HR_Reset_Layer State defines the Mode of operation of both the Protocol Layer transmission and reception State machines during a [Hard Reset](#) or [Cable Reset](#). During Hard Reset no USB Power Delivery Protocol Messages are sent or received; only [Hard Reset](#) Signaling is present after which the communication channel is assumed to have been disabled by the PHY Layer until completion of the [Hard Reset](#). During [Cable Reset](#) no USB Power Delivery Protocol Messages are sent to or received by the Cable Plug but other USB Power Delivery communication **May** continue.

The Protocol Layer **Shall** enter the PRL_HR_Reset_Layer State from any other State when:

- A [Hard Reset](#) Request is received from the Policy Engine or
- [Hard Reset](#) Signaling is received from the PHY Layer or
- A [Cable Reset](#) Request is received from the Policy Engine or
- [Cable Reset](#) Signaling is received from the PHY Layer.

On entry to the PRL_HR_Reset_Layer State the Protocol Layer **Shall** reset the MessageIDCounter. It **Shall** also reset the states of the Protocol Layer transmission and reception State machines to their starting points. The Protocol Layer transmission State machine **Shall** transition to the PRL_Tx_Wait_for_Message_Request State. The Protocol Layer reception State machine **Shall** transition to the PRL_Rx_Wait_for_PHY_Message State.

The Protocol Layer **Shall** transition to the PRL_HR_Request_[Hard_Reset](#) State when:

- The Protocol Layer's reset is complete and
- The [Hard Reset](#) Request has originated from the Policy Engine or
- The [Cable Reset](#) Request has originated from the Policy Engine.

The Protocol Layer **Shall** transition to the PRL_HR_Indicate_[Hard_Reset](#) State when:

- The Protocol Layer's reset is complete and
- The [Hard Reset](#) Request has been passed up from the PHY Layer or
- A [Cable Reset](#) Request has been passed up from the PHY Layer (Cable Plug only).

9.1.2.4.2. PRL_HR_Indicate_Hard_Reset State

On entry to the PRL_HR_Indicate_[Hard_Reset](#) State the Protocol Layer **Shall** indicate to the Policy Engine that either [Hard Reset](#) Signaling or [Cable Reset](#) Signaling has been received.

The Protocol Layer **Shall** transition to the PRL_HR_Wait_for_PE_[Hard_Reset](#)_Complete State when:

- The indication to the Policy Engine has been sent.

9.1.2.4.3. PRL_HR_Request_Hard_Reset State

On entry to the PRL_HR_Request_[Hard_Reset](#) State the Protocol Layer **Shall** Request the PHY Layer to send either [Hard Reset](#) Signaling or [Cable Reset](#) Signaling.

The Protocol Layer **Shall** transition to the PRL_HR_Wait_for_PHY_Hard_Reset_Complete State when:

- The PHY Layer [Hard Reset](#) Signaling Request has been sent or
- The PHY Layer [Cable Reset](#) Signaling Request has been sent.

9.1.2.4.4. PRL_HR_Wait_for_PHY_Hard_Reset_Complete State

In the PRL_HR_Wait_for_PHY_Hard_Reset_Complete State the Protocol Layer **Shall** start the HardResetCompleteTimer and wait for the PHY Layer to indicate that the [Hard Reset](#) or [Cable Reset](#) has been completed.

The Protocol Layer **Shall** transition to the PRL_HR_PHY_Hard_Reset_Requested State when:

- A [Hard Reset](#) complete indication is received from the PHY Layer or
- A [Cable Reset](#) complete indication is received from the PHY Layer or
- The HardResetCompleteTimer times out.

9.1.2.4.5. PRL_HR_PHY_Hard_Reset_Requested State

On entry to the PRL_HR_PHY_Hard_Reset_Requested State the Protocol Layer **Shall** inform the Policy Engine that the PHY Layer has been requested to perform a [Hard Reset](#) or [Cable Reset](#).

The Protocol Layer **Shall** transition to the PRL_HR_Wait_for_PE_Hard_Reset_Complete State when:

- The Indication to the Policy Engine has been sent.

9.1.2.4.6. PRL_HR_Wait_for_PE_Hard_Reset_Complete State

In the PRL_HR_Wait_for_PE_Hard_Reset_Complete State the Protocol Layer **Shall** wait for the Policy Engine to indicate that the [Hard Reset](#) or [Cable Reset](#) has been completed.

The Protocol Layer **Shall** transition to the PRL_HR_PE_Hard_Reset_Complete State when:

- A [Hard Reset](#) complete indication is received from the Policy Engine or
- A [Cable Reset](#) complete indication is received from the Policy Engine.

9.1.2.4.7. PRL_HR_PE_Hard_Reset_Complete

On entry to the PRL_HR_PE_Hard_Reset_Complete State the Protocol Layer **Shall** inform the PHY Layer that the [Hard Reset](#) or [Cable Reset](#) is complete.

The Protocol Layer **Shall** exit from the [Hard Reset](#) and return to normal operation when:

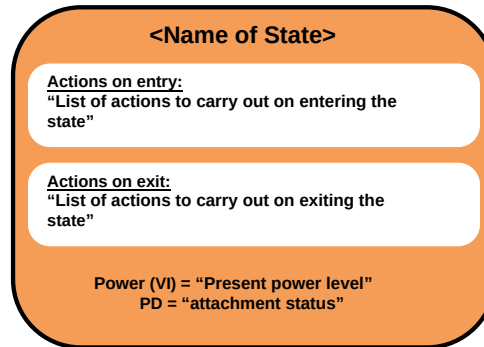
- The PHY Layer has been informed that the [Hard Reset](#) is complete so that it will re-enable the communications channel. If [Hard Reset](#) Signaling is still pending due to a non-Idle channel this **Shall** be cleared and not sent or the PHY Layer has been informed that the [Cable Reset](#) is complete.

9.2. Policy Engine Layer State Diagrams

9.2.1. Introduction to State diagrams

The State diagrams in this section define the operation of the Power Delivery Policy Engine.

Note: These State diagrams are not intended to replace a well written and robust design.

Figure 9.12. Outline of States

[Figure 9.12](#) shows an outline of the states defined in the following sections. At the top there is the name of the State. This is followed by "Actions on entry" a list of actions carried out on entering the State. If there are also "Actions on exit" a list of actions carried out on exiting the State, then these are listed as well; otherwise, this box is omitted from the State. At the bottom the status of PD is listed:

- "Power" which indicates the present output power for a Source Port or input power for a Sink Port.
- "PD" which indicates the present Attachment status either "Attached", "Detached", or "unknown".

Transitions from one State to another are indicated by arrows with the conditions listed on the arrow. Where there are multiple conditions, these are Connected using either a logical OR "|" or a logical AND "&".

In some cases, there are transitions which can occur from any State to a particular State. These are indicated by an arrow which is unconnected to a State at one end, but with the other end (the point) Connected to the final State. In some State diagrams it is necessary to enter or exit from states in other diagrams (e.g., Source Port or Sink Port State diagrams). [Figure 9.13](#) indicates how such references are made. The reference is indicated with a hatched box. The box contains the name of the State and whether the State is a DFP or UFP. It has also been necessary to indicate conditional entry to either Source Port or Sink Port State diagrams. This is achieved by the use of a bulleted list indicating the preconditions (see example in [Figure 9.14](#)). It is also possible that the entry and return states are the same. [Figure 9.15](#) indicates a State reference where each referenced State corresponds to either the entry State or the exit State.

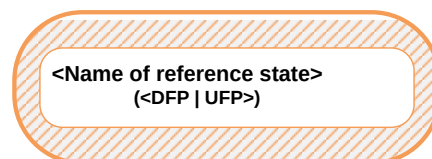
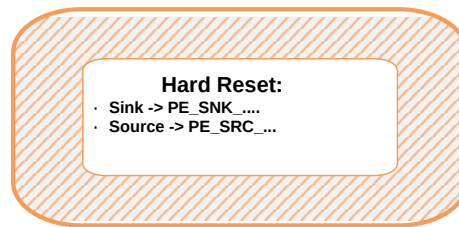
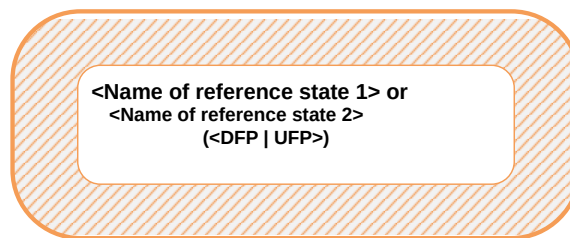
Figure 9.13. References to states

Figure 9.14. Example of State reference with conditions**Figure 9.15. Example of State reference with the same entry and exit**

Timers are included in many of the states. Timers are initialized (set to their starting condition) and run (timer is counting) in the particular State it is referenced. As soon as the State is exited then the timer is no longer active. Where the timers continue to run outside of the State (such as the NoResponseTimer), this is called out in the text. Timeouts of the timers are listed as conditions on State transitions.

The SenderResponseTimer is a special case, as it **May** be stopped and started from outside the states in which it is used. To allow this to be done without over-complicating the State diagrams, the SenderResponseTimer is described with its own State diagram ([Figure 9.16](#)). The control of this Timer is shared between the Policy Engine and the Chunking Layer.

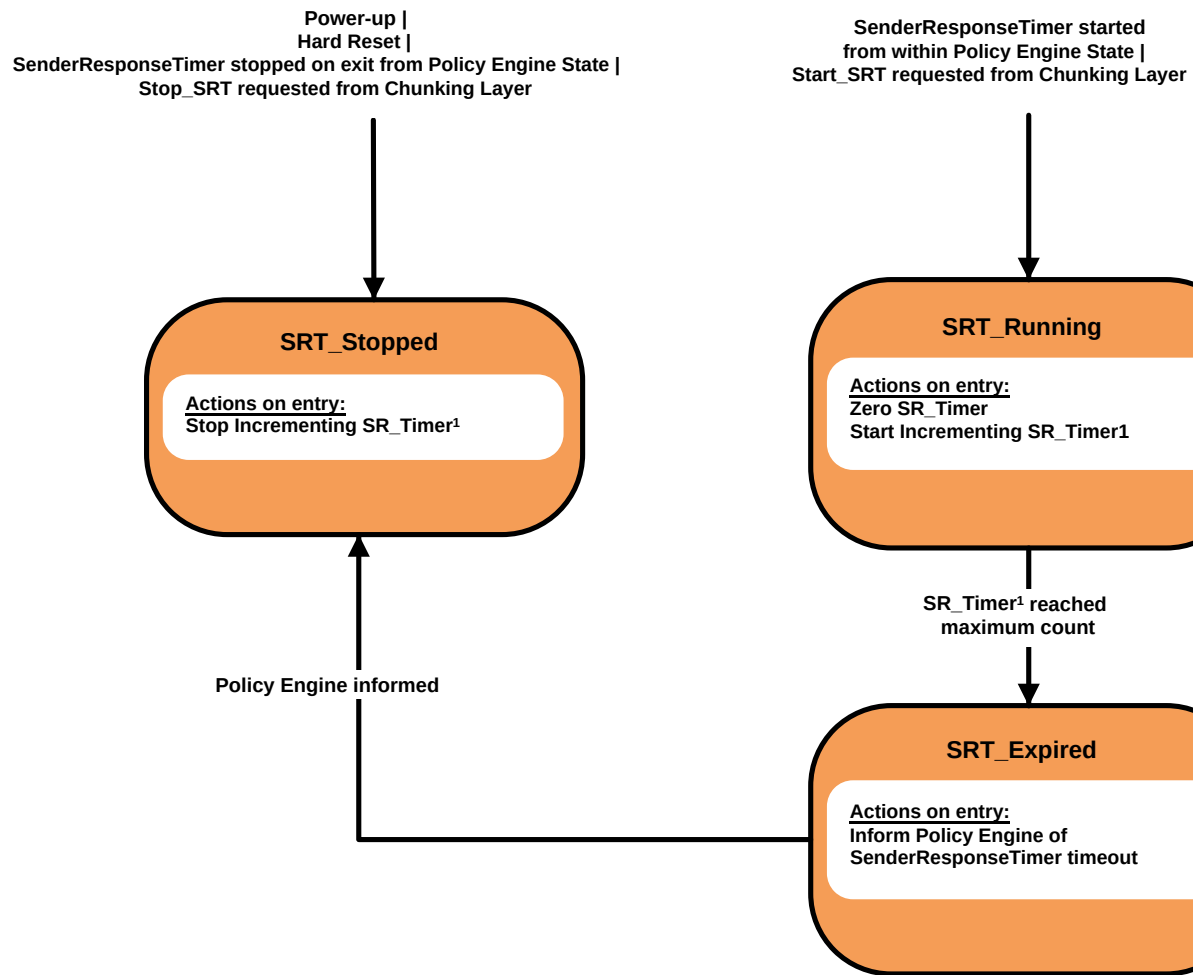
Conditions listed on State transitions will come from one of three sources and, when there is a conflict, **Should** be serviced in the following order:

1. Message and related indications passed up to the Policy Engine from the Protocol Layer (Message sent; Message received etc.).
2. Events triggered within the Policy Engine e.g., timer timeouts.
3. Information and requests coming from the Device Policy Manager relating either to Local Policy, or to other modules which the Device Policy Manager controls such as power supply and USB Type-C Port Control.

Note: The following State diagrams are not intended to cover all possible corner cases that could be encountered. For example, where an outgoing Message is **Discarded**, due to an incoming Message by the Protocol Layer it will be necessary for the higher layers of the system to handle a retry of the AMS that was being initiated, after first handling the incoming Message.

9.2.2. SenderResponseTimer State Diagram

This figure shows the State diagram for the Policy Engine in a Source Port or a Sink Port. The following sections describe operation in each of the states.

Figure 9.16. SenderResponseTimer Policy Engine State Diagram

1. The SR_Timer is regarded as the mechanism within the SenderResponseTimer State diagram that implements the SenderResponseTimer.

9.2.2.1. SRT_Stopped State

The SRT_Stopped State **shall** be the starting State for the SenderResponseTimer either on power up or after a [Hard Reset](#). On entry to this State the Policy Engine **shall** stop incrementing the SR_Timer. The Policy Engine **shall** transition to the SRT_Running State:

- When the SenderResponseTimer is started from within a Policy Engine State, or
- When a Start_SRT is requested from the Chunking Layer.

9.2.2.2. SRT_Running State

On entry to the SRT_Running State the SenderResponseTimer State machine **shall**:

- Set the SR_Timer to zero
- Start running SR_Timer.

The SenderResponseTimer State machine **Shall** transition to the SRT_Expired State:

- When the SR_Timer reaches its maximum count

The SenderResponseTimer State machine **Shall** transition to the SRT_Stopped State:

- When the SenderResponseTimer is stopped by exiting a Policy Engine State, or
- When a Stop_SRT is requested from the Chunking Layer

9.2.2.3. SRT_Expired State

On entry to the SRT_Running State the SenderResponseTimer State machine **Shall** Inform Policy Engine of SenderResponseTimer timeout.

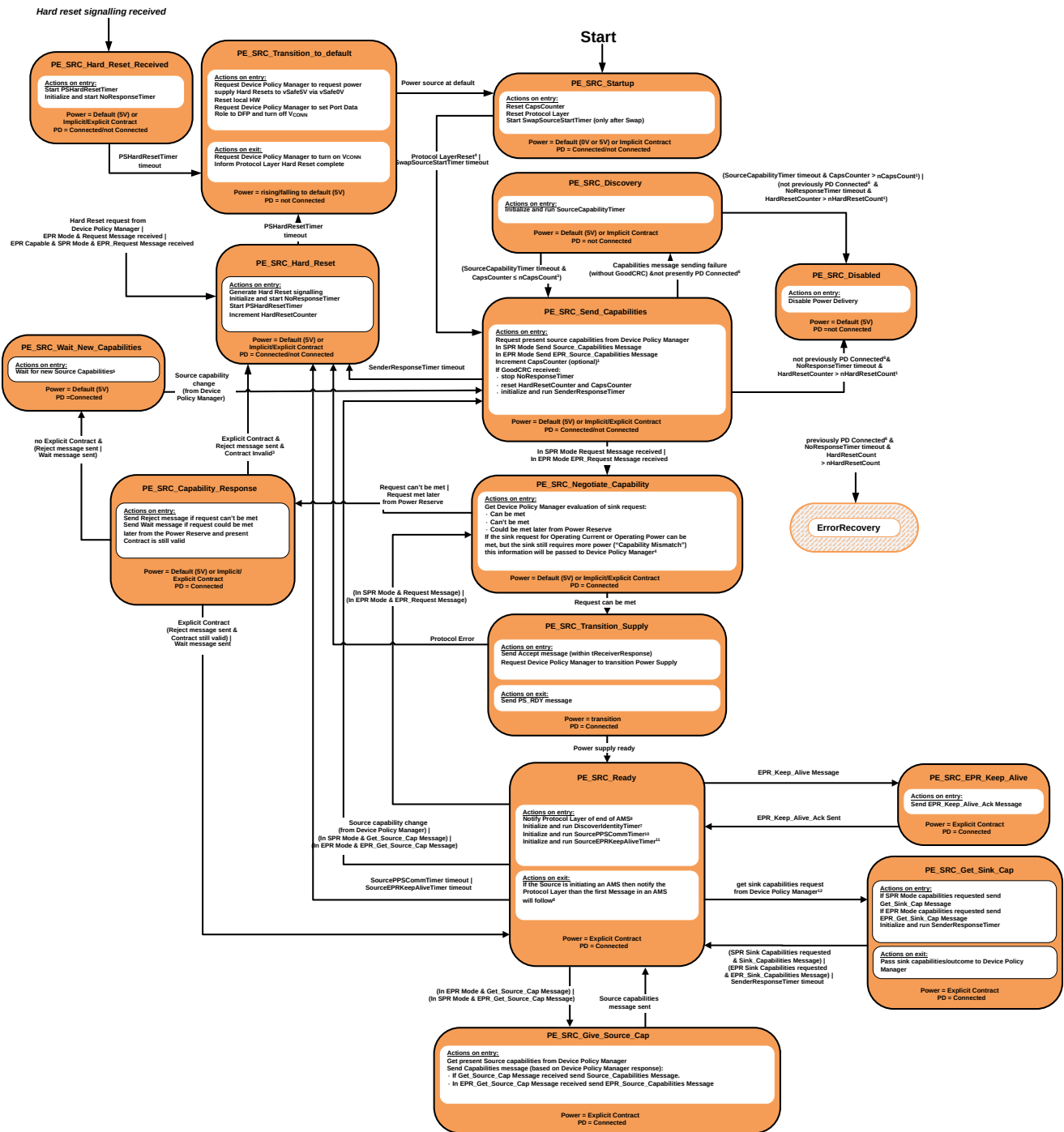
The Policy Engine **Shall** then transition to the SRT_Stopped State:

- When the Policy Engine has been informed.

9.2.3. Policy Engine Source Port State Diagram

This figure shows the State diagram for the Policy Engine in a Source Port. The following sections describe operation in each of the states.

Figure 9.17. Source Port State Diagram



1. Implementation of the CapsCounter is **Optional**. In the case where this is not implemented the Source **Shall** continue to send [Source Capabilities Messages](#) each time the SourceCapabilityTimer times out.
2. Since the Sink is required to make a **Valid Request** from the offered Capabilities the expected transition is via "Request can be met" unless the Source Capabilities have changed since the last offer.

3. "Contract **Invalid**" means that the previously Negotiated voltage and Current values are no longer included in the Source's new Capabilities. If the Sink fails to make a **Valid** Request in this case, then Power Delivery operation is no longer possible and Power Delivery Mode is exited with a [Hard Reset](#).
4. After a Power Swap the New Source is required to wait an additional tSwapSourceStart before sending a [Source_Capabilities Message](#). This delay is not required when first starting up a system.
5. PD Connected is defined as a situation when the Port Partners are actively communicating. The Port Partners remain PD Connected after a Swap until there is a transition to Disabled or the connector is able to identify a Detach
6. Port Partners are no longer PD Connected after a [Hard Reset](#), but consideration needs to be given as to whether there has been a PD Connection while the Ports have been Attached to prevent unnecessary USB Type-C Error Recovery.
7. The DiscoverIdentityTimer is run when this is a VCONN Source and a PD Connection with a Cable Plug needs to be established i.e. no [GoodCRC Message](#) has yet been received in response to a [Discover Identity](#) Command.
8. See [Section 5.2.2](#), [Section 7.31.13](#), and [Section 7.2](#).
9. In the PE_SRC_Wait_New_Capabilities State the Device Policy Manager **Should** either decide to send no further Source Capabilities or **Should** send a different set of Source Capabilities. Continuing to send the same set of Source Capabilities could result in a live lock situation.
10. The SourcePPSCommTimer is only initialized and run when the present Explicit Contract is for a PPS APDO. Sources that do not support PPS, do not need to implement the SourcePPSCommTimer.
11. The SourceEPRKeepAliveTimer is only initialized and run when the Source is in [EPR Mode](#); Sources that do not support [EPR Mode](#) do not need to implement the SourceEPRKeepAliveTimer.
12. Either SPR or EPR Sink Capabilities **May** be requested, regardless of whether or not the Source is currently operating in SPR or [EPR Mode](#).

9.2.3.1. PE_SRC_Startup State

PE_SRC_Startup **Shall** be the starting State for a Source Policy Engine either on power up or after a [Hard Reset](#). On entry to this State the Policy Engine **Shall** reset the CapsCounter and reset the Protocol Layer.

Note: Resetting the Protocol Layer will also reset the MessageIDCounter and stored MessageID.

The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State:

- When the Protocol Layer reset has completed if the PE_SRC_Startup State was entered due to the system first starting up.
- When the SwapSourceStartTimer times out if the PE_SRC_Startup State was entered as the result of a [Power Role Swap](#).

Note: Sources **Shall** remain in the PE_SRC_Startup State, without sending any [Source_Capabilities Messages](#) until a plug is Attached.

9.2.3.2. PE_SRC_Discovery State

On entry to the PE_SRC_Discovery State the Policy Engine **Shall** initialize and run the SourceCapabilityTimer in order to trigger sending a [Source_Capabilities Message](#).

The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State when:

- The SourceCapabilityTimer times out and CapsCounter \leq [nCapsCount](#). The Policy Engine **May** Optionally go to the PE_SRC_Disabled State when:
 - The Port Partners are not presently PD Connected
 - And the SourceCapabilityTimer times out
 - And CapsCounter $>$ [nCapsCount](#).

The Policy Engine **Shall** go to the PE_SRC_Disabled State when:

- The Port Partners have not been PD Connected (the Source Port remains Attached to a Port it has not had a PD Connection with during this Attachment)
- And the NoResponseTimer times out
- And the HardResetCounter $>$ [nHardResetCount](#).

Note: In the PE_SRC_Disabled State the Attached Device is assumed to be unresponsive. The Policy Engine operates as if the Device is Detached until such time as a Detach/Re-Attach is detected.

9.2.3.3. PE_SRC_Send_Capabilities State

Note: This State can be entered from the PE_SRC_Soft_Reset State.

On entry to the PE_SRC_Send_Capabilities State the Policy Engine **Shall** Request the present Port Capabilities from the Device Policy Manager. The Policy Engine **Shall** then Request the Protocol Layer to send a Capabilities Message containing these Capabilities. The Policy Engine **Shall** Request:

- A [Source_Capabilities Message](#) if the Source is in SPR Mode or
- An [EPR_Source_Capabilities Message](#) if the Source is in [EPR Mode](#). The Policy Engine **Shall** then increment the CapsCounter (if implemented). If a [GoodCRC Message](#) is received, then the Policy Engine **Shall**:
 - Stop the NoResponseTimer.
 - Reset the HardResetCounter and CapsCounter to zero.

Note: The HardResetCounter **Shall** only be set to zero in this State and at power up; its value **Shall** be maintained during a [Hard Reset](#).

- Initialize and run the SenderResponseTimer.

Once a [Source_Capabilities Message](#) has been received and acknowledged by a [GoodCRC Message](#), the Sink is required to then send a [Request Message](#) within tSenderResponse.

The Policy Engine **Shall** transition to the PE_SRC_Negotiate_Capability State when:

- A [Request Message](#) is received from the Sink and the Source is operating in SPR Mode or

An [EPR_Request Message](#) is received from the Sink and the Source is operating in [EPR Mode](#). The Policy Engine **Shall** transition to the PE_SRC_Discovery State when:

- The Protocol Layer indicates that the Message has not been sent and we are presently not Connected. This is part of the Capabilities sending process whereby successful Message sending indicates connection to a PD Sink Port.

The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:

- The SenderResponseTimer times out. In this case a transition back to USB Default Operation is required.

When the Port Partners have not been PD Connected (the Source Port remains Attached to a Port it has not had a PD Connection with during this Attachment), and the NoResponseTimer times out, and the HardResetCounter > [nHardResetCount](#), the Policy Engine **Shall** do one of the following:

- Transition to the PE_SRC_Discovery State.
- Transition to the PE_SRC_Disabled State.

Note: That in either case the Attached Device is assumed to be unresponsive. The Policy Engine **Should** operate as if the Device is Detached until such time as a Detach/Re-Attach is detected.

The Policy Engine **Shall** go to the ErrorRecovery State when:

- The Port Partners have previously been PD Connected (the Source Port remains Attached to a Port it has had a PD Connection with during this Attachment)
- And, the NoResponseTimer times out.
- And, the HardResetCounter > [nHardResetCount](#).

9.2.3.4. PE_SRC_Negotiate_Capability State

On entry to the PE_SRC_Negotiate_Capability State the Policy Engine **Shall** ask the Device Policy Manager to evaluate the Request from the Attached Sink. The response from the Device Policy Manager **Shall** be one of the following:

- The Request can be met.
- The Request cannot be met
- The Request could be met later from the Power Reserve.

The Policy Engine **Shall** transition to the PE_SRC_Transition_Supply State when:

- The Request can be met.

The Policy Engine **Shall** transition to the PE_SRC_Capability_Response State when:

- The Request cannot be met.
- Or the Request can be met later from the Power Reserve.

9.2.3.5. PE_SRC_Transition_Supply State

The Policy Engine **Shall** be in the PE_SRC_Transition_Supply State while the power supply is transitioning from one power to another. On entry to the PE_SRC_Transition_Supply State, the Policy Engine **Shall** Request the Protocol Layer to send an [Accept Message](#) and inform the Device Policy Manager that it **Shall** transition the power supply to the Requested power level.

Note: If the power supply is currently operating at the requested power no change will be necessary.

On exit from the PE_SRC_Transition_Supply State the Policy Engine **Shall** Request the Protocol Layer to send a [PS_RDY Message](#).

The Policy Engine **Shall** transition to the PE_SRC_Ready State when:

The Device Policy Manager informs the Policy Engine that the power supply is ready. The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:

- A Protocol Error occurs.

9.2.3.6. PE_SRC_Ready State

In the PE_SRC_Ready State the PD Source **Shall** be operating at a stable power with no ongoing Negotiation. It **Shall** respond to requests from the Sink, events from the Device Policy Manager. On entry to the PE_SRC_Ready State the Source Shall notify the Protocol Layer of the end of the Atomic Message Sequence (AMS). If the transition into PE_SRC_Ready is the result of Protocol Error that has not caused a [Soft Reset](#) (see [Section 9.2.5.1](#)) then the notification to the Protocol Layer of the end of the AMS **Shall Not** be sent since there is a Message to be processed.

On entry to the PE_SRC_Ready State if this is a VCONN Source which needs to establish communication with a Cable Plug, the Policy Engine **Shall**:

- Initialize and run the DiscoverIdentityTimer (no [GoodCRC Message](#) response yet received to [Discover Identity](#) Message).

On entry to the PE_SRC_Ready State if the current Explicit Contract is for a PPS APDO, then the Policy Engine

Shall do the following:

- Initialize and run the SourcePPSCCommTimer.

On entry to the PE_SRC_Ready State if the current Explicit Contract is for EPR Mode, then the Policy Engine **Shall** do the following:

- Initialize and run the SourceEPRKeepAliveTimer.

On exit from the PE_SRC_Ready, if the Source is initiating an AMS, then the Policy Engine **Shall** notify the Protocol Layer that the first Message in an AMS will follow.

The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State when:

- The Device Policy Manager indicates that Source Capabilities have changed or
- A [Get_Source_Cap Message](#) is received, and the Source is in SPR Mode or

An [EPR_Get_Source_Cap Message](#) is received, and the Source is in [EPR Mode](#). The Policy Engine **Shall** transition to the PE_SRC_Negotiate_Capability State when:

- A [Request Message](#) is received, and the Source is in SPR Mode or

An [EPR_Request Message](#) is received, and the Source is in [EPR Mode](#). The Policy Engine **Shall** transition to the PE_SRC_Get_Sink_Cap State when:

- The Device Policy Manager asks for the Sink Capabilities.

The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:

- The Source is operating as a PPS and the SourcePPSCCommTimer Timer times-out or

The Source is in [EPR Mode](#) and the SourceEPRKeepAliveTimer Timer times-out. The Policy Engine **Shall** transition to the PE_SRC_EPR_Keep_Alive State when:

- An [EPR_Keep_Alive Message](#) is received.

The Policy Engine **Shall** transition to the PE_SRC_Give_Source_Cap State when:

- In [EPR Mode](#) and a [Get_Source_Cap Message](#) is received or
- In SPR Mode and an [EPR_Get_Source_Cap Message](#) is received.

If a transition into the PE_SRC_Ready State will result in an immediate transition out of the PE_SRC_Ready State within tSrcHoldsBus e.g. it is due to a Protocol Error that has not resulted in a [Soft Reset](#), then the notifications of the end of AMS and first Message in an AMS **May Not** be sent to avoid changing the Rp value unnecessarily.

9.2.3.7. PE_SRC_Disabled State

In the PE_SRC_Disabled State the PD Source supplies default power and is unresponsive to USB Power Delivery messaging, but not to [Hard Reset](#) Signaling.

9.2.3.8. PE_SRC_Capability_Response State

The Policy Engine **Shall** enter the PE_SRC_Capability_Response State if there is a Request received from the Sink that cannot be met based on the present Capabilities. When the present Explicit Contract is not within the present Capabilities it is regarded as **Invalid** and a [Hard Reset](#) will be triggered.

On entry to the PE_SRC_Hard_Reset State the Policy Engine **Shall** Request the Protocol Layer to send one of the following:

- [Reject Message](#) - if the Request cannot be met or the present Explicit Contract is **Invalid**.
- [Wait Message](#) - if the Request could be met later from the Power Reserve. A [Wait Message](#) **Shall Not** be sent if the present Explicit Contract is **Invalid**.

The Policy Engine **Shall** transition to the PE_SRC_Ready State when:

- There is an Explicit Contract and
- A [Reject Message](#) has been sent and the present Explicit Contract is still **Valid** or
- A [Wait Message](#) has been sent.

The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:

- There is an Explicit Contract and
- The [Reject Message](#) has been sent and the present Explicit Contract is **Invalid** (i.e., the Sink had to Request a new value so instead we will return to USB Default Operation).

The Policy Engine **Shall** transition to the PE_SRC_Wait_New_Capabilities State when:

- There is no Explicit Contract and
- A [Reject Message](#) has been sent or
- A [Wait Message](#) has been sent.

9.2.3.9. PE_SRC_Hard_Reset State

The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State from any State when:

- [Hard Reset](#) Request from Device Policy Manager or
- In [EPR Mode](#) and a [Request Message](#) is received or

- PR Capable and in SPR Mode and an [EPR_Request Message](#) is received.

On entry to the PE_SRC_Hard_Reset State the Policy Engine **Shall**:

- Request the generation of [Hard Reset](#) Signaling by the PHY Layer which might take up to tHardResetComplete.
- initialize and run the NoResponseTimer.

Note: The NoResponseTimer **Shall** continue to run in every State until it is stopped or times out.

The Policy Engine **Shall** transition to the PE_SRC_Transition_to_default State when:

- The PSHardResetTimer times out.

9.2.3.10. PE_SRC_Hard_Reset_Received State

The Policy Engine **Shall** transition from any State to the PE_SRC_Hard_Reset_Received State when:

- [Hard Reset](#) Signaling is detected.

On entry to the PE_SRC_Hard_Reset_Received State the Policy Engine **Shall**:

- initialize and run the PSHardResetTimer
- initialize and run the NoResponseTimer.

Note: The NoResponseTimer **Shall** continue to run in every State until it is stopped or times out. The Policy Engine **Shall** transition to the PE_SRC_Transition_to_default State when:

- The PSHardResetTimer times out.

9.2.3.11. PE_SRC_Transition_to_default State

On entry to the PE_SRC_Transition_to_default State the Policy Engine **Shall**:

- Indicate to the Device Policy Manager that the power supply **Shall** [Hard Reset](#) (see [Section 4.4](#)).
- Request a reset of the local hardware Request the Device Policy Manager to set the Port Data Role to DFP and turn off VCONN. On exit from the PE_SRC_Transition_to_default State the Policy Engine **Shall**:
- Request the Device Policy Manager to turn on VCONN inform the Protocol Layer that the [Hard Reset](#) is complete. The Policy Engine **Shall** transition to the PE_SRC_Startup State when:
- The Device Policy Manager indicates that the power supply has reached the default level.

9.2.3.12. PE_SRC_Get_Sink_Cap State

In this State the Policy Engine, due to a Request from the Device Policy Manager, **Shall** Request the Capabilities from the Attached Sink. On entry to the PE_SRC_Get_Sink_Cap State the Policy Engine **Shall** Request the Protocol Layer to send a [Get_Sink_Cap Message](#) in order to retrieve the Sink Capabilities. The Policy Engine **Shall** send:

- A [Get_Sink_Cap Message](#) when the Device Policy Manager requests SPR Capabilities or

An [EPR_Get_Sink_Cap Message](#) when the Device Policy Manager requests EPR Capabilities. The Policy Engine **Shall** then start the SenderResponseTimer. On exit from the PE_SRC_Get_Sink_Cap State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout).

The Policy Engine **Shall** transition to the PE_SRC_Ready State when:

- SPR Sink Capabilities were requested and a [Sink_Capabilities Message](#) is received or
- EPR Sink Capabilities were requested and an [EPR_Sink_Capabilities Message](#) is received or
- The SenderResponseTimer times out.

9.2.3.13. PE_SRC_Wait_New_Capabilities State

In this State the Policy Engine has been unable to Negotiate an Explicit Contract and is waiting for new Capabilities from the Device Policy Manager.

The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State when:

- The Device Policy Manager indicates that Source Capabilities have changed.

9.2.3.14. PE_SRC_EPR_Keep_Alive State

On entry to the PE_SRC_EPR_Keep_Alive State the Policy Engine **Shall** send a [EPR_Keep_Alive_Ack Message](#). The Policy Engine **Shall** transition to the PE_SRC_Ready State when:

- The [EPR_Keep_Alive_Ack Message](#) has been sent.

9.2.3.15. PE_SRC_Give_Source_Cap State

On entry to the PE_SRC_Give_Source_Cap State the Policy Engine **Shall** Request the Device Policy Manager for the current system Capabilities.

The Policy Engine **Shall** then Request the Protocol Layer to send a Source Capabilities Message containing these Capabilities.

The Policy Engine **Shall** send:

- A [Source_Capabilities Message](#) when a [Get_Source_Cap Message](#) is received or

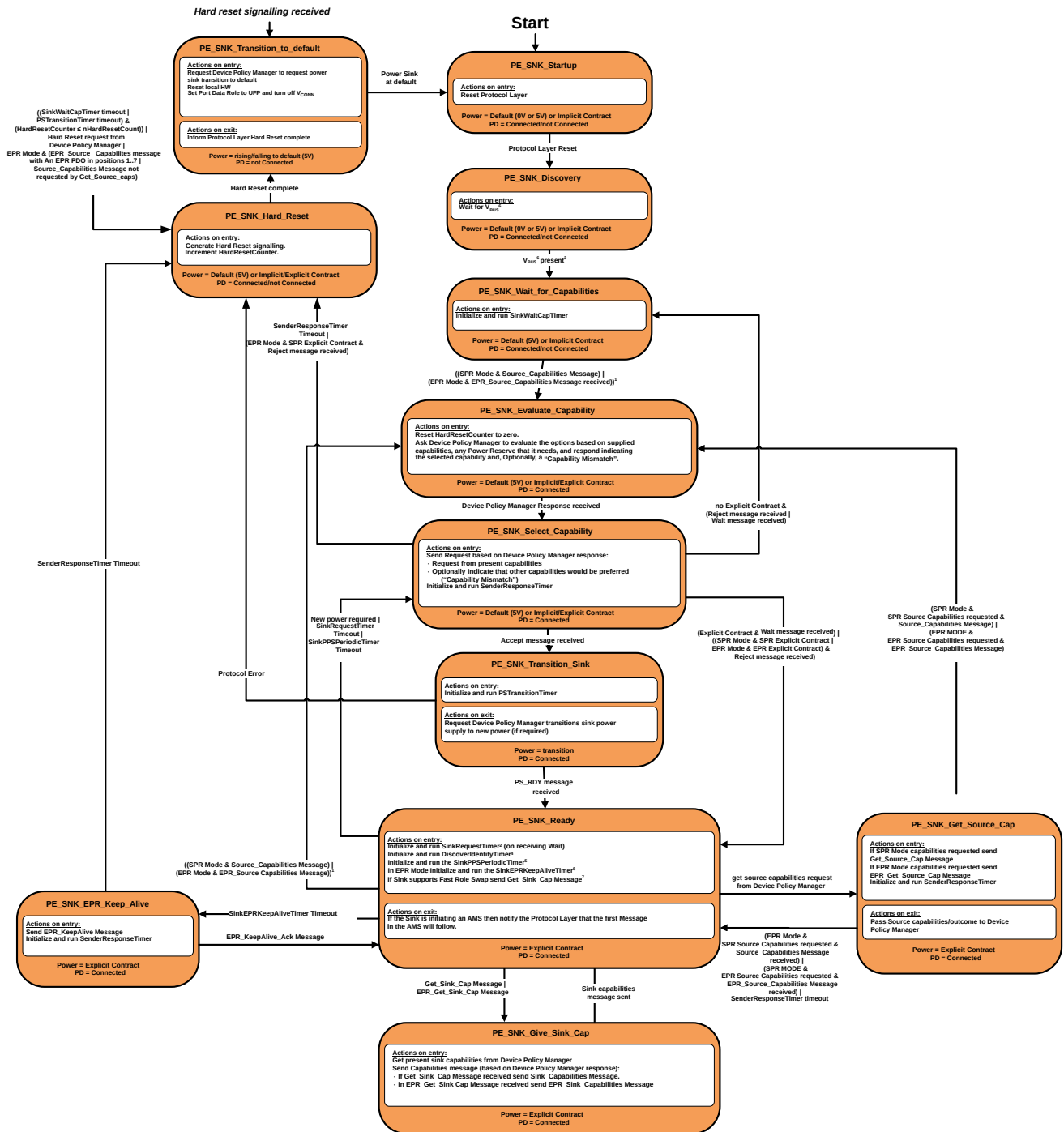
An [EPR_Source_Capabilities Message](#) when a [EPR_Get_Source_Cap Message](#) is received. The Policy Engine **Shall** transition to the PE_SNK_Ready State when:

- The Source Capabilities Message has been successfully sent.

9.2.4. Policy Engine Sink Port State Diagram

This figure shows the State diagram for the Policy Engine in a Sink Port. The following sections describe operation in each of the states.

Figure 9.18. Sink Port State Diagram



1. Source Capabilities Messages received in States other than PE_SNK_Wait_for_Capabilities, PE_SNK_Ready or PE_SNK_Get_Source_Cap constitute a Protocol Error.
2. The SinkRequestTimer **Should Not** be stopped if a [Ping \(Deprecated\)](#) Message is received in the PE_SNK_Ready State since it represents the maximum time between requests after a [Wait Message](#) which is not reset by a [Ping \(Deprecated\)](#) Message.

3. During a [Hard Reset](#) the Source voltage will transition to [vSafe0V](#) and then transition to [vSafe5V](#). Sinks need to ensure that VBUS present is not indicated until after the Source has completed the [Hard Reset](#) process by detecting both of these transitions.
4. The DiscoverIdentityTimer is run when this is a VCONN Source and a PD Connection with a Cable Plug needs to be established i.e., no [GoodCRC Message](#) has yet been received in response to a [Discover Identity](#) Command.
5. The SinkPPSPeriodicTimer is only initialized and run when the present Explicit Contract is for a PPS APDO. Sinks that do not support PPS do not need to implement the SinkPPSPeriodicTimer.
6. A Sink that is a VPD **May** use VCONN as a proxy for VBUS.
7. To be sent once, and only required if [Fast Role Swap](#) is supported by the Sink.

9.2.4.1. PE_SNK_Startup State

PE_SNK_Startup **Shall** be the starting State for a Sink Policy Engine either on power up or after a [Hard Reset](#). On entry to this State the Policy Engine **Shall** reset the Protocol Layer.

Note: Resetting the Protocol Layer will also reset the MessageIDCounter and stored MessageID (see [Section 9.1.2.3](#)). Once the reset process completes, the Policy Engine **Shall** transition to the PE_SNK_Discovery State.

9.2.4.2. PE_SNK_Discovery State

In the PE_SNK_Discovery State the Sink Policy Engine waits for VBUS to be present. The Policy Engine **Shall** transition to the PE_SNK_Wait_for_Capabilities State when:

- The Device Policy Manager indicates that VBUS has been detected.

9.2.4.3. PE_SNK_Wait_for_Capabilities State

On entry to the PE_SNK_Wait_for_Capabilities State the Policy Engine **Shall** initialize and start the SinkWaitCap-Timer.

The Policy Engine **Shall** transition to the PE_SNK_Evaluate_Capability State when:

- The Sink is in SPR Mode and a [Source_Capabilities Message](#) is received or
- The Sink is in [EPR Mode](#) and an [EPR_Source_Capabilities Message](#) is received.

9.2.4.4. PE_SNK_Evaluate_Capability State

The PE_SNK_Evaluate_Capability State is first entered when the Sink receives its first [Source_Capabilities Message](#) from the Source. At this point the Sink knows that it is Attached to and communicating with a PD Capable Source.

On entry to the PE_SNK_Evaluate_Capability State the Policy Engine **Shall** Request the Device Policy Manager to evaluate the supplied Source Capabilities based on Local Policy. The Device Policy Manager **Shall** indicate to the Policy Engine the new power level required, selected from the present offered Capabilities. The Device Policy Manager **Shall** also indicate to the Policy Engine a Capabilities Mismatch if the offered power does not meet the Device's requirements.

The Policy Engine **Shall** transition to the PE_SNK_Select_Capability State when:

- A response is received from the Device Policy Manager.

9.2.4.5. PE_SNK_Select_Capability State

On entry to the PE_SNK_Select_Capability State the Policy Engine **Shall** Request the Protocol Layer to send a response Message, based on the evaluation from the Device Policy Manager. The Message **Shall** be one of the following:

- A Request from the offered Source Capabilities.
- A Request from the offered Source Capabilities with an indication that another power level would be preferred (Capability Mismatch bit set).

When in SPR Mode a [Request Message](#) **Shall** be sent.

When in [EPR Mode](#) an [EPR_Request Message](#) **Shall** be sent.

The Policy Engine **Shall** initialize and run the SenderResponseTimer.

The Policy Engine **Shall** transition to the PE_SNK_Transition_Sink State when:

- An [Accept Message](#) is received from the Source.

The Policy Engine **Shall** transition to the PE_SNK_Wait_for_Capabilities State when:

- There is no Explicit Contract in place and
- A [Reject Message](#) is received from the Source or
- A [Wait Message](#) is received from the Source.

The Policy Engine **Shall** transition to the PE_SNK_Ready State when:

- There is an SPR Explicit Contract in SPR Mode or an EPR Explicit Contract in [EPR Mode](#) and a [Reject Message](#) is received from the Source or
- There is an Explicit Contract in place and a [Wait Message](#) is received from the Source.

The Policy Engine **Shall** transition to the PE_SNK_Hard_Reset State when:

- A SenderResponseTimer timeout occurs or
- An EPR Explicit Contract has not been established in [EPR Mode](#) and a [Reject Message](#) is received from the Source.

9.2.4.6. PE_SNK_Transition_Sink State

On entry to the PE_SNK_Transition_Sink State the Policy Engine **Shall** initialize and run the PSTransitionTimer (timeout will lead to a [Hard Reset](#) see [Section 9.2.4.8](#) and **Shall** then Request the Device Policy Manager to transition the Sink's power supply to the new power level.

Note: If there is no power level change the Device Policy Manager **Should Not** affect any change to the power supply.

On exit from the PE_SNK_Transition_Sink State the Policy Engine **Shall** Request the Device Policy Manager to transition the Sink's power supply to the new power level.

The Policy Engine **Shall** transition to the PE_SNK_Ready State when:

- A [PS_RDY Message](#) is received from the Source.

The Policy Engine **Shall** transition to the PE_SNK_Hard_Reset State when:

- A Protocol Error occurs.

9.2.4.7. PE_SNK_Ready State

- The Device Policy Manager requests an update of the remote Source Capabilities. The Policy Engine **Shall** transition to the PE_SNK_EPR_Keep_Alive State when:
 - The SinkEPRKeepAliveTimer timeouts out.

9.2.4.8. PE_SNK_Hard_Reset State

The Policy Engine **Shall** transition to the PE_SNK_Hard_Reset State from any State when:

- (PSTransitionTimer times out) and
- (HardResetCounter ≤ [nHardResetCount](#)))
- [Hard Reset](#) Request from Device Policy Manager or
- In [EPR Mode](#) and
 - An [EPR_Source_Capabilities Message](#) is received with an EPR PDO or APDO in object positions 1...7 or
 - A [Source_Capabilities Message](#) is received that has not been requested using a [Get_Source_Cap](#) Message.
- The Policy Engine **May** transition to the PE_SNK_Hard_Reset State from any State when:
 - SinkWaitCapTimer times out.

Note: If the SinkWaitCapTimer times out and the HardResetCounter is greater than [nHardResetCount](#) the Sink **Shall** assume that the Source is non-responsive.

Note: The HardResetCounter is reset on a power cycle or Detach.

On entry to the PE_SNK_Hard_Reset State the Policy Engine **Shall** Request the generation of [Hard Reset](#) Signaling by the PHY Layer and increment the HardResetCounter.

The Policy Engine **Shall** transition to the PE_SNK_Transition_to_default State when:

- The [Hard Reset](#) is complete.

9.2.4.9. PE_SNK_Transition_to_default State

The Policy Engine **Shall** transition from any State to PE_SNK_Transition_to_default State when:

- [Hard Reset](#) Signaling is detected.

When [Hard Reset](#) Signaling is received or transmitted then the Policy Engine **Shall** transition from any State to PE_SNK_Transition_to_default. This State can also be entered from the PE_SNK_Hard_Reset State. On entry to the PE_SNK_Transition_to_default State the Policy Engine **Shall**:

- Indicate to the Device Policy Manager that the Sink **Shall** transition to default
- Request a reset of the local hardware

- Request the Device Policy Manager that the Port Data Role is set to UFP.

The Policy Engine **Shall** transition to the PE_SNK_Startup State when:

- The Device Policy Manager indicates that the Sink has reached the default level.

9.2.4.10. PE_SNK_Give_Sink_Cap State

On entry to the PE_SNK_Give_Sink_Cap State the Policy Engine **Shall** Request the Device Policy Manager for the current system Capabilities. The Policy Engine **Shall** then Request the Protocol Layer to send a [Sink_Capabilities Message](#) containing these Capabilities. The Policy Engine **Shall** send:

- A [Sink_Capabilities Message](#) when a [Get_Sink_Cap Message](#) is received or
- An [EPR_Sink_Capabilities Message](#) when a [EPR_Get_Sink_Cap Message](#) is received. The Policy Engine **Shall** transition to the PE_SNK_Ready State when:
 - The [Sink_Capabilities Message](#) has been successfully sent.

9.2.4.11. PE_SNK_EPR_Keep_Alive

- On entry to the PE_SNK_EPR_Keep_Alive State the Policy Engine **Shall** send an [EPR_Keep_Alive Message](#) and initialize and run the SenderResponseTimer.

The Policy Engine **Shall** transition to the PE_SNK_Ready State when:

- A [EPR_Keep_Alive_Ack Message](#) is received.

The Policy Engine **Shall** transition to the PE_SNK_Hard_Reset State when:

- The SenderResponseTimer times out.

9.2.4.12. PE_SNK_Get_Source_Cap State

- On entry to the PE_SNK_Get_Source_Cap State the Policy Engine **Shall** Request the Protocol Layer to send a get Source Capabilities Message in order to retrieve the Source Capabilities. The Policy Engine **Shall** send:
 - A [Get_Source_Cap Message](#) when the Device Policy Manager requests SPR Capabilities or
- An [EPR_Get_Source_Cap Message](#) when the Device Policy Manager requests EPR Capabilities. The Policy Engine **Shall** then start the SenderResponseTimer.

On exit from the PE_SNK_Get_Source_Cap State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout).

The Policy Engine **Shall** transition to the PE_SNK_Ready State when:

- In [EPR Mode](#) and SPR Source Capabilities were requested and a [Source_Capabilities Message](#) is received or
- In SPR Mode and EPR Source Capabilities were requested and an [EPR_Source_Capabilities Message](#) is received or
- The SenderResponseTimer times out.

The Policy Engine **Shall** transition to the PE_SNK_Evaluate_Capability State when:

- In SPR Mode and SPR Source Capabilities were requested and a [Source_Capabilities Message](#) is received or

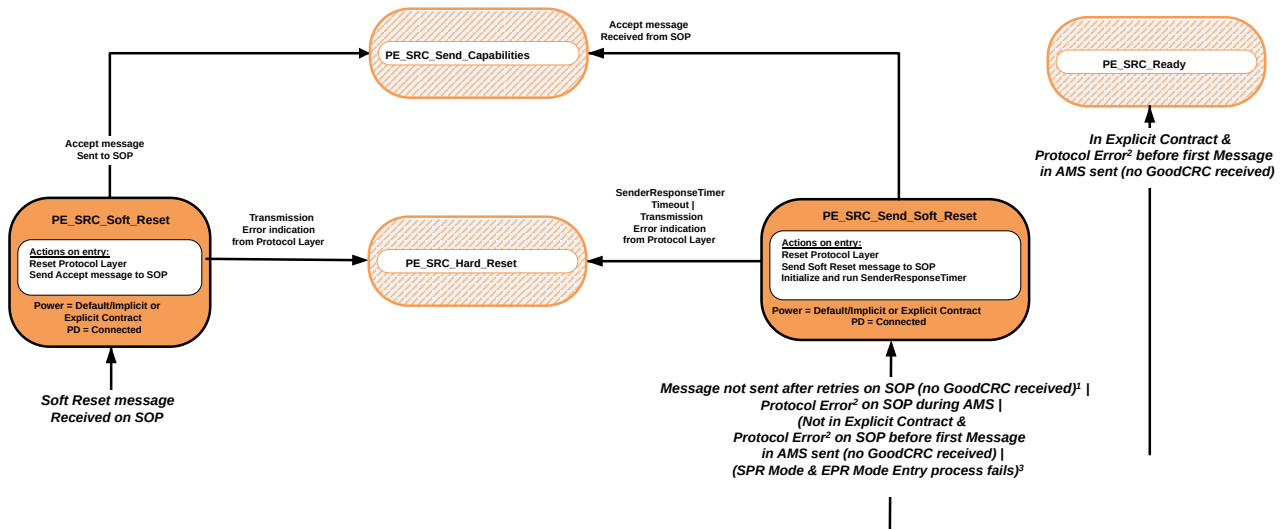
- In [EPR Mode](#) and EPR Source Capabilities were requested and an [EPR_Source_Capabilities Message](#) is received.

9.2.5. SOP Soft Reset and Protocol Error State Diagrams

9.2.5.1. SOP Source Port Soft Reset and Protocol Error State Diagram

This figure shows the State diagram for the Policy Engine in a Source Port when performing a [Soft Reset](#) of its Port Partner i.e., using SOP. The following sections describe operation in each of the states.

Figure 9.19. SOP Source Port Soft Reset and Protocol Error State Diagram



1. Excludes the [Soft_Reset Message](#) itself.
2. An Unrecognized or Unsupported Message received on SOP will result in a [Not_Supported Message](#) response being generated on SOP (see [Section 9.2.7.1](#)).
3. See [Section 7.30.1](#) for the conditions when a [Soft_Reset Message](#) **Shall** be sent by the Source during the [EPR Mode](#) Entry process.

9.2.5.1.1. PE_SRC_Send_Soft_Reset State

The PE_SRC_Send_Soft_Reset State **Shall** be entered from any State when:

- A Protocol Error on SOP is detected by the Protocol Layer during an AMS (see [Section 7.1.1](#)) or
- A Message has not been sent after retries to the Sink or
- When not in an Explicit Contract and Protocol Errors occurred on SOP during any AMS where the first Message in the AMS has not yet been sent i.e., an Unexpected Message is received instead of the expected [GoodCRC Message](#) response or
- When in SPR Mode and the [EPR Mode](#) Entry process fails. The main exceptions to this rule are when:
 - The Source is in the PE_SRC_Send_Capabilities State, there is a [Source_Capabilities Message](#) sending failure on SOP (without a [GoodCRC Message](#)) and the Source is not presently Attached (as indicated in [Figure 9.17](#)). In this case, the PE_SRC_Discovery State is entered (see [Section 9.2.3.2](#)).

- When the voltage is in transition due to a new Explicit Contract being Negotiated (see [Section 9.2.3](#)). In this case [Hard Reset](#) Signaling will be generated.
- During a [Power Role Swap](#) when the power supply is in transition (see [Section 9.2.20.3](#) and [Section 9.2.20.4](#)). In this case USB Type-C Error Recovery will be triggered directly.
- During a [Data Role Swap](#) when there is a mismatch in the Port Data Role field (see [Table 6.9](#)). In this case USB Type-C Error Recovery will be triggered directly.

Protocol Errors occurring in the following situations **Shall Not** lead to a [Soft Reset](#), but **Shall** result in a transition to the PE_SRC_Ready State where the Message received will be handled as if it had been received in the PE_SRC_Ready State:

- When in an Explicit Contract and Protocol Errors occurred on SOP during any AMS where the first Message in the AMS has not yet been sent i.e., an Unexpected Message is received instead of the expected [GoodCRC](#) Message response.

On entry to the PE_SRC_Send_[Soft_Reset](#) State the Policy Engine **Shall** Request the SOP Protocol Layer to perform a [Soft Reset](#), then **Shall** send a [Soft_Reset Message](#) to the Sink on SOP, and initialize and run the SenderResponseTimer.

The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State when:

- An [Accept Message](#) has been received on SOP.

The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:

- A SenderResponseTimer timeout occurs.
- Or the Protocol Layer indicates that a transmission error has occurred.

9.2.5.1.2. PE_SRC_Soft_Reset State

- The PE_SRC_Soft_Reset State **Shall** be entered from any State when a [Soft_Reset Message](#) is received on SOP from the Protocol Layer.

On entry to the PE_SRC_Soft_Reset State the Policy Engine **Shall** reset the SOP Protocol Layer and **Shall** then Request the Protocol Layer to send an [Accept Message](#) on SOP.

The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State (see [Section 9.2.3.3](#)) when:

- The [Accept Message](#) has been sent on SOP.

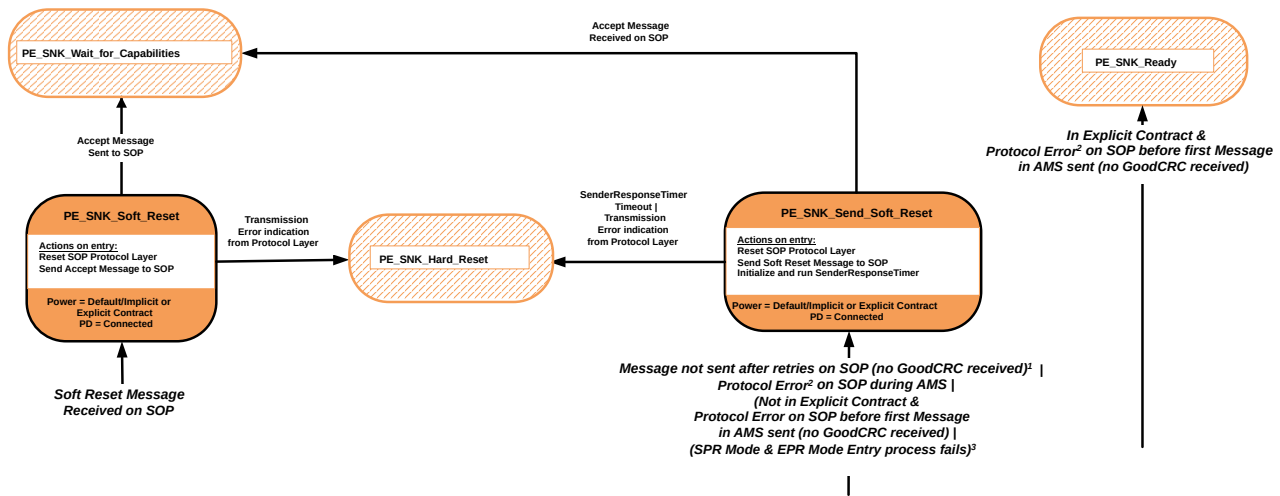
The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:

- The Protocol Layer indicates that a transmission error has occurred.

9.2.5.2. SOP Sink Port Soft Reset and Protocol Error State Diagram

This figure shows the State diagram for the Policy Engine in a Sink Port when performing a [Soft Reset](#) of its Port Partner i.e., using SOP. The following sections describe operation in each of the states.

Figure 9.20. Sink Port Soft Reset and Protocol Error Diagram



1. Excludes the [Soft_Reset Message](#) itself.
2. An Unrecognized or Unsupported Message received on SOP will result in a [Not_Supported Message](#) response being generated on SOP (see [Section 9.2.7.2](#)).
3. See [Section 7.30.1](#) for the conditions when a [Soft_Reset Message](#) **Shall** be sent by the Source during the [EPR Mode](#) Entry process.

9.2.5.2.1. PE_SNK_Send_Soft_Reset State

The PE_SNK_Send_Soft_Reset State **Shall** be entered from any State when:

- A Protocol Error on SOP is detected by the Protocol Layer during an AMS (see [Section 7.1.1](#)) or
- A Message has not been sent after retries to the Sink or
- When not in an Explicit Contract and Protocol Errors occurred on SOP during any AMS where the first Message in the AMS has not yet been sent i.e., an Unexpected Message is received instead of the expected [GoodCRC Message](#) response.
- When in SPR Mode and the [EPR Mode](#) Entry process fails. The main exceptions to this rule are when:
 - When the voltage is in transition due to a new Explicit Contract being Negotiated (see [Section 9.2.4](#)). In this case a [Hard Reset](#) will be generated.
 - During a [Power Role Swap](#) when the power supply is in transition (see [Section 9.2.20.3](#) and [Section 9.2.20.4](#)). In this case a [Hard Reset](#) will be triggered directly.
 - During a [Data Role Swap](#) when the DFP/UFP Data Roles are changing. In this case USB Type-C Error Recovery will be triggered directly.

Note: Protocol Errors occurring in the following situations **Shall Not** lead to a [Soft Reset](#), but **Shall** result in a transition to the PE_SNK_Ready State where the Message received will be handled as if it had been received in the PE_SNK_Ready State:

- When in an Explicit Contract and Protocol Errors occurred on SOP during any AMS where the first Message in the AMS has not yet been sent i.e., an Unexpected Message is received instead of the expected [GoodCRC](#) Message response.

On entry to the PE_SNK_Send_[Soft_Reset](#) State the Policy Engine **Shall** Request the SOP Protocol Layer to perform a [Soft_Reset](#), then **Shall** send a [Soft_Reset Message](#) on SOP to the Source, and initialize and run the SenderResponseTimer.

The Policy Engine **Shall** transition to the PE_SNK_Wait_for_Capabilities State when:

- An [Accept Message](#) has been received on SOP.

The Policy Engine **Shall** transition to the PE_SNK_Hard_Reset State when:

- A SenderResponseTimer timeout occurs.
- Or the Protocol Layer indicates that a transmission error has occurred.

9.2.5.2.2. PE_SNK_Soft_Reset State

- The PE_SNK_Soft_Reset State **Shall** be entered from any State when a [Soft_Reset Message](#) is received on SOP from the Protocol Layer.

On entry to the PE_SNK_Soft_Reset State the Policy Engine **Shall** reset the SOP Protocol Layer and **Shall** then Request the Protocol Layer to send an [Accept Message](#) on SOP. The Policy Engine **Shall** transition to the PE_SNK_Wait_for_Capabilities State when:

- The [Accept Message](#) has been sent on SOP.

The Policy Engine **Shall** transition to the PE_SNK_Hard_Reset State when:

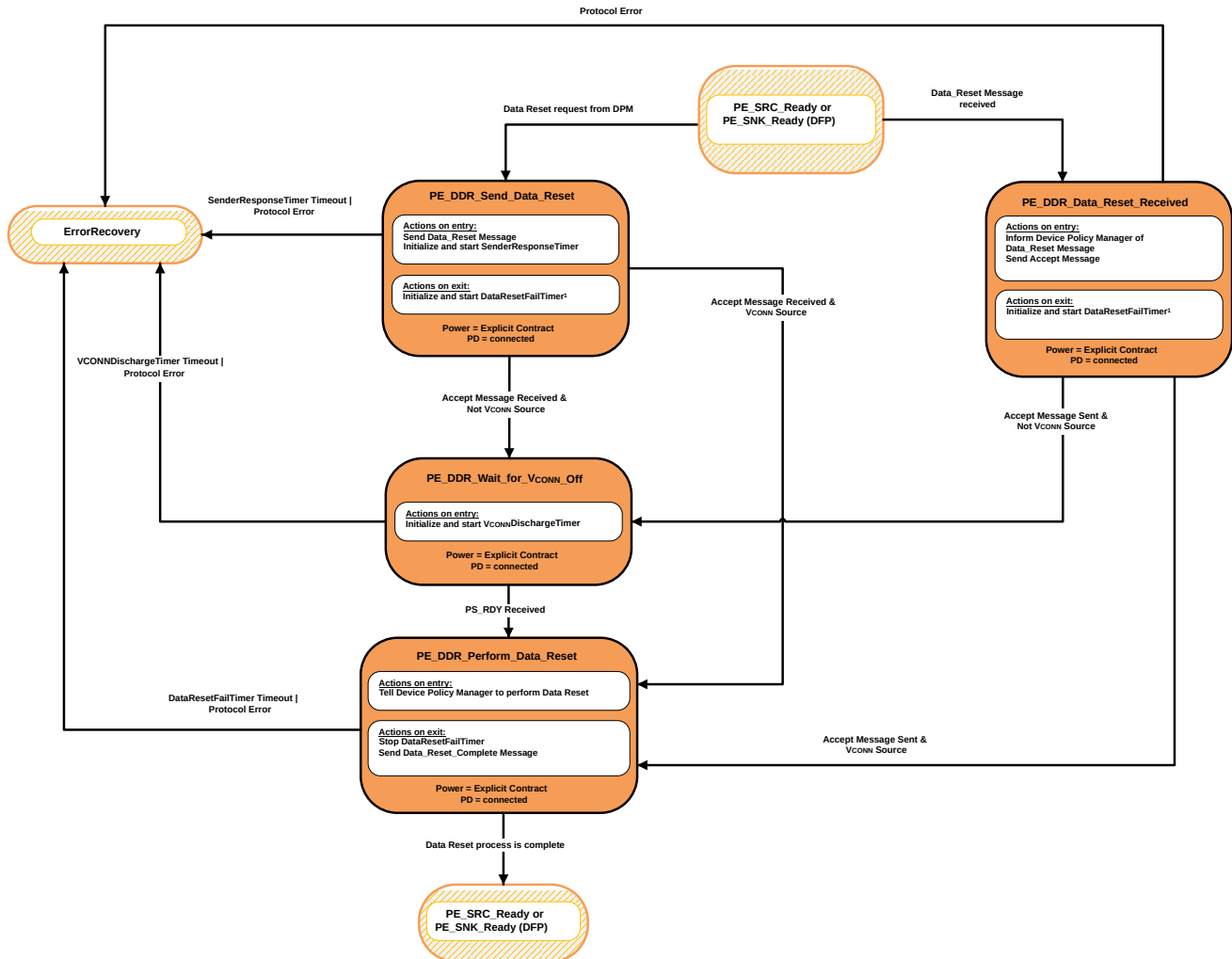
- The Protocol Layer indicates that a transmission error has occurred.

9.2.6. Data Reset State Diagrams

9.2.6.1. DFP Data_Reset Message State Diagrams

[Figure 9.21](#) shows the State diagram for a [Data_Reset Message](#) sent or received by a DFP.

Figure 9.21. DFP Data_Reset Message State Diagram



- Note: The DataResetFailTimer **Shall** continue to run in every State until it is stopped or times out.

9.2.6.1.1. PE_DDR_Send_Data_Reset State

The PE_DDR_Send_Data_Reset State **Shall** be entered from the PE_SRC_Ready or PE_SNK_Ready State when requested by the Device Policy Manager.

On entry to the PE_DDR_Send_Data_Reset State the Policy Engine **Shall** Request the Protocol Layer to send a [Data_Reset Message](#) and then initialize and start the SenderResponseTimer.

On exit from the PE_DDR_Send_Data_Reset State the Policy Engine **Shall** initialize and start the DataResetFail-Timer.

The Policy Engine **Shall** transition to the PE_DDR_Perform_Data_Reset State when:

- An [Accept Message](#) has been received and
- The DFP is presently the VCONN Source.

The Policy Engine **Shall** transition to the PE_DDR_Wait_For_VCONN_Off State when:

- An [Accept Message](#) has been received and
- The DFP is not presently the VCONN Source.

The Policy Engine **Shall** transition to ErrorRecovery when:

- A SenderResponseTimer timeout occurs or
- A Protocol Error occurs.

9.2.6.1.2. PE_DDR_Data_Reset_Received State

The PE_DDR_Data_Reset_Received State **Shall** be entered from the PE_SRC_Ready or PE_SNK_Ready State when a [Data_Reset Message](#) is received.

On entry to the PE_DDR_Data_Reset_Received State the Policy Engine **Shall** inform the Device Policy Manager and then **Shall** send an [Accept Message](#). On exit from the PE_DDR_Data_Reset_Received State, the Policy Engine **Shall** initialize and start the DataResetFailTimer.

The Policy Engine **Shall** transition to the PE_DDR_Perform_Data_Reset State when:

- An [Accept Message](#) has been sent and
- The DFP is presently the VCONN Source.

The Policy Engine **Shall** transition to the PE_DDR_Wait_For_VCONN_Off State when:

- An [Accept Message](#) has been sent and
- The DFP is not presently the VCONN Source.

The Policy Engine **Shall** transition to ErrorRecovery when:

- A Protocol Error occurs.

9.2.6.1.3. PE_DDR_Wait_For_VCONN_Off State

On entry to the PE_DDR_Wait_For_VCONN_Off State the Policy Engine **Shall** initialize and start the VCONNDIS-chargeTimer.

The Policy Engine **Shall** transition to the PE_DDR_Perform_Data_Reset State when:

- A [PS_RDY Message](#) is received.

The Policy Engine **Shall** transition to ErrorRecovery when:

- The VCONNDISchargeTimer has timed out or
- A Protocol Error occurs.

9.2.6.1.4. PE_DDR_Perform_Data_Reset State

On entry to the PE_DDR_Perform_Data_Reset State the Policy Engine **Shall** Request the Device Policy Manager to complete the [Data_Reset](#) process as defined in [Section 7.1.2](#). On exit from the PE_DDR_Perform_Data_Reset State the Policy Engine **Shall** stop the DataResetFailTimer and send a [Data_Reset_Complete Message](#).

The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State depending on the DFP's Power Role when:

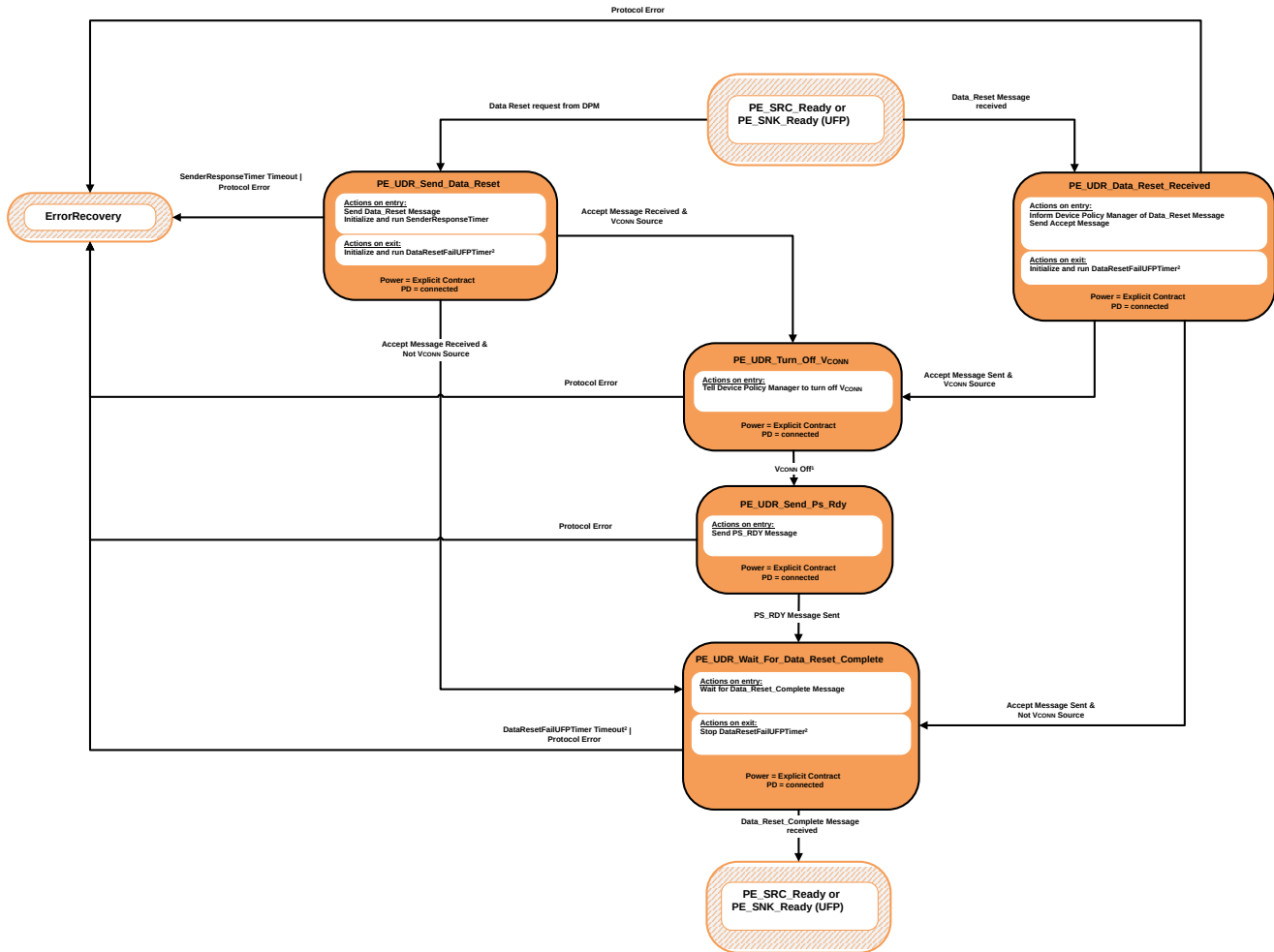
- The DPM indicates that [Data_Reset](#) process is complete (see [Section 7.1.2](#)). The Policy Engine **Shall** transition to ErrorRecovery when:

- The DataResetFailTimer times out
- A Protocol Error occurs.

9.2.6.2. UFP Data_Reset Message State Diagrams

Figure 9.22 shows the State diagram for a [Data_Reset Message](#) sent or received by a UFP.

Figure 9.22. UFP Data_Reset Message State Diagram



1. VCONN **Shall** be fully discharged see [Section 4.6](#).
2. Note: The DataResetFailUFPTimer **Shall** continue to run in every State until it is stopped or times out.

9.2.6.2.1. PE_UDR_Send_Data_Reset State

The PE_UDR_Send_Data_Reset State **Shall** be entered from the PE_SRC_Ready or PE_SNK_Ready State when requested by the Device Policy Manager.

On entry to the PE_UDR_Send_Data_Reset State the Policy Engine **Shall** Request the Protocol Layer to send a [Data_Reset Message](#) and then initialize and run the SenderResponseTimer. On exit from the PE_UDR_Send_Data_Reset State the Policy Engine **Shall** initialize and run the DataResetFailUFPTimer.

The Policy Engine **Shall** transition to the PE_UDR_Turn_Off_VCONN State when:

- An [Accept Message](#) has been received and
- The UFP is presently the VCONN Source.

The Policy Engine **Shall** transition to the PE_UDR_Wait_For_Data_Reset_Complete State when:

- An [Accept Message](#) has been received and
- The UFP is not presently the VCONN Source.

The Policy Engine **Shall** transition to ErrorRecovery when:

- The SenderResponseTimer has timed out or
- A Protocol Error occurs.

9.2.6.2.2. PE_UDR_Data_Reset_Received State

The PE_UDR_Data_Reset_Received State **Shall** be entered from either the PE_SRC_Ready or PE_SNK_Ready State when a [Data_Reset Message](#) is received.

On entry to the PE_UDR_Data_Reset_Received State the Policy Engine **Shall** inform the Device Policy Manager and then **Shall** send an [Accept Message](#). On exit from the PE_UDR_Data_Reset_Received State the Policy Engine **Shall** initialize and run the DataResetFailUFPTimer.

The Policy Engine **Shall** transition to the PE_UDR_Turn_Off_VCONN State when:

- An [Accept Message](#) has been sent and
- The UFP is presently the VCONN Source.

The Policy Engine **Shall** transition to the PE_UDR_Wait_For_Data_Reset_Complete State when:

- An [Accept Message](#) has been sent and
- The UFP is not presently the VCONN Source.

The Policy Engine **Shall** transition to ErrorRecovery when:

- A Protocol Error occurs.

9.2.6.2.3. PE_UDR_Turn_Off_VCONN State

On entry to the PE_UDR_Turn_Off_VCONN State the Policy Engine **Shall** Request the Device Policy Manager to turn off VCONN.

The Policy Engine **Shall** transition to the PE_UDR_Send_Ps_Rdy State when:

- The DPM indicates that VCONN has been turned off (VCONN below vRaReconnect see [USB-C]). The Policy Engine **Shall** transition to ErrorRecovery when:
 - A Protocol Error occurs.

9.2.6.2.4. PE_UDR_Send_Ps_Rdy State

- On entry to the PE_UDR_Send_Ps_Rdy State the Policy Engine **Shall** send a [PS_RDY Message](#). The Policy Engine **Shall** transition to the PE_UDR_Wait_For_Data_Reset_Complete State when:

- The [PS_RDY Message](#) has been sent.

The Policy Engine **Shall** transition to ErrorRecovery when:

- A Protocol Error occurs.

9.2.6.2.5. PE_UDR_Wait_For_Data_Reset_Complete State

- On entry to the PE_UDR_Wait_For_Data_Reset_Complete State the Policy Engine **Shall** wait for the [Data_Reset_Complete Message](#).

On exit from the PE_UDR_Wait_For_Data_Reset_Complete State the Policy Engine **Shall** stop the DataResetFailUFPTimer.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State depending on the UFP's Power Role when:

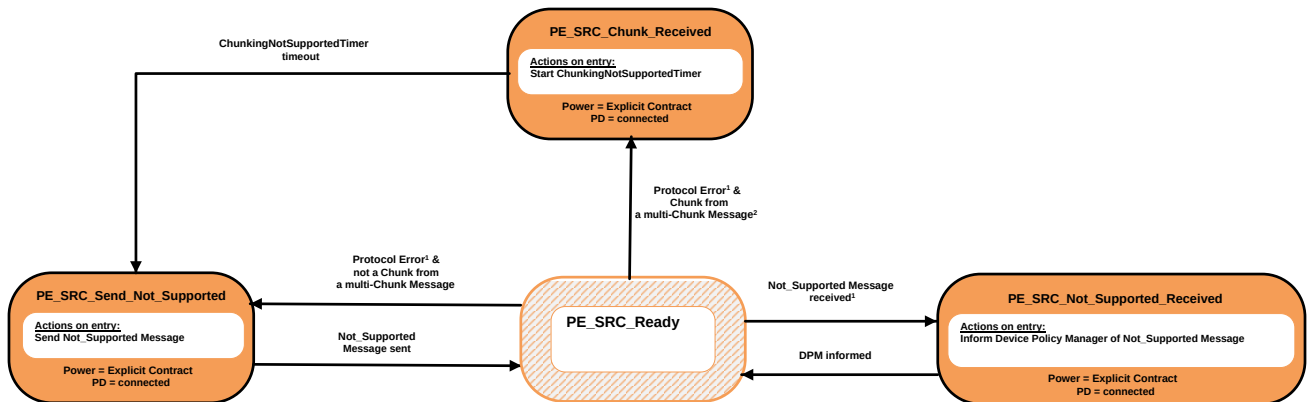
- The [Data_Reset_Complete Message](#) is received. The Policy Engine **Shall** transition to ErrorRecovery when:
 - The DataResetFailUFPTimer times out or
 - A Protocol Error occurs.

9.2.7. Not Supported Message State Diagrams

9.2.7.1. Source Port Not Supported Message State Diagram

[Figure 9.23](#) shows the State diagram for a [Not_Supported Message](#) sent or received by a Source Port.

Figure 9.23. Source Port Not Supported Message State Diagram



1. Transition as a result of an Unsupported Message being received in the PE_SRC_Ready State directly (see also [Section 9.2.5.1](#)).
2. Transition can only occur where a manufacturer has opted not to implement a Chunking State machine (see [Section 9.1.2.1](#)) and is communicating with a system which is attempting to send it Chunks.

9.2.7.1.1. PE_SRC_Send_Not_Supported State

The PE_SRC_Send_Not_Supported State **Shall** be entered from the PE_SRC_Ready State either as the result of a Protocol Error received during an interruptible AMS or as a result of an Unsupported Message being received in

the PE_SRC_Ready State directly except for the first Chunk in a multi-Chunk Message (see also [Section 9.1.2.1](#) and [Section 9.2.5.1](#)).

On entry to the PE_SRC_Send_Not_Supported State (from the PE_SRC_Ready State) the Policy Engine **Shall** Request the Protocol Layer to send a [Not_Supported](#) Message.

The Policy Engine **Shall** transition back to the previous State (PE_SRC_Ready see [Figure 9.17](#)) when:

- The [Not_Supported Message](#) has been successfully sent.

9.2.7.1.2. PE_SRC_Not_Supported_Received State

The PE_SRC_Not_Supported_Received State **Shall** be entered from the PE_SRC_Ready State when a [Not_Supported Message](#) is received.

On entry to the PE_SRC_Not_Supported_Received State the Policy Engine **Shall** inform the Device Policy Manager.

The Policy Engine **Shall** transition back to the previous State (PE_SRC_Ready see [Figure 9.17](#)) when:

- The Device Policy Manager has been informed.

9.2.7.1.3. PE_SRC_Chunk_Received State

The PE_SRC_Chunk_Received State **Shall** be entered from the PE_SRC_Ready State as a result of an Unsupported Message being received in the PE_SRC_Ready State directly where the Message is a Chunk in a multi-Chunk Message (see also [Section 9.1.2.1](#) and [Section 9.2.5.1](#)).

On entry to the PE_SRC_Chunk_Received State (from the PE_SRC_Ready State) the Policy Engine **Shall** initialize and run the ChunkingNotSupportedTimer.

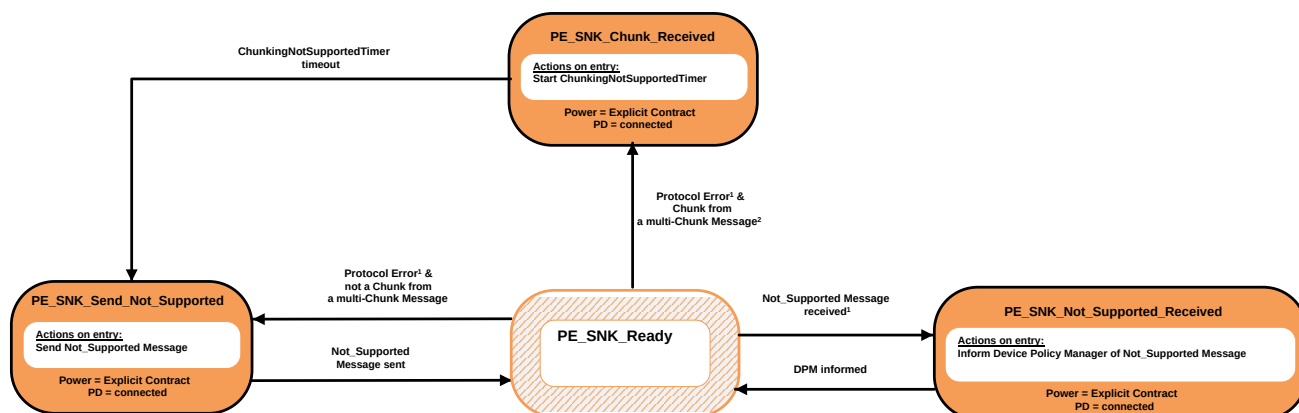
The Policy Engine **Shall** transition to PE_SRC_Send_Not_Supported when:

- The ChunkingNotSupportedTimer has timed out.

9.2.7.2. Sink Port Not Supported Message State Diagram

[Figure 9.24](#) shows the State diagram for a [Not_Supported Message](#) sent or received by a Sink Port.

Figure 9.24. Sink Port Not Supported Message State Diagram



1. Transition as a result of an Unsupported Message being received in the PE_SNK_Ready State directly (see also [Section 9.2.5.2](#)).

2. Transition can only occur where a manufacturer has opted not to implement a Chunking State machine (see [Section 9.1.2.1](#)) and is communicating with a system which is attempting to send it Chunks.

9.2.7.2.1. PE_SNK_Send_Not_Supported State

The PE_SNK_Send_Not_Supported State **shall** be entered from the PE_SNK_Ready State either as the result of a Protocol Error received during an interruptible AMS or as a result of an Unsupported Message being received in the PE_SNK_Ready State directly except for the first Chunk in a multi-Chunk Message (see also [Section 9.1.2.1](#) and [Section 9.2.5.1](#)).

On entry to the PE_SNK_Send_Not_Supported State (from the PE_SNK_Ready State) the Policy Engine **shall** Request the Protocol Layer to send a [Not_Supported](#) Message.

The Policy Engine **shall** transition back to the previous State (PE_SNK_Ready see [Figure 9.18](#)) when:

- The [Not_Supported Message](#) has been successfully sent.

9.2.7.2.2. PE_SNK_Not_Supported_Received State

The PE_SNK_Not_Supported_Received State **shall** be entered from the PE_SNK_Ready State when a [Not_Supported Message](#) is received.

On entry to the PE_SNK_Not_Supported_Received State the Policy Engine **shall** inform the Device Policy Manager.

The Policy Engine **shall** transition back to the previous State (PE_SNK_Ready see [Figure 9.18](#)) when:

- The Device Policy Manager has been informed.

9.2.7.2.3. PE_SNK_Chunk_Received State

The PE_SNK_Chunk_Received State **shall** be entered from the PE_SNK_Ready State as a result of an Unsupported Message being received in the PE_SNK_Ready State directly where the Message is a Chunk in a multi-Chunk Message (see also [Section 9.1.2.1](#) and [Section 9.2.5.1](#)).

On entry to the PE_SNK_Chunk_Received State (from the PE_SNK_Ready State) the Policy Engine **shall** initialize and run the ChunkingNotSupportedTimer.

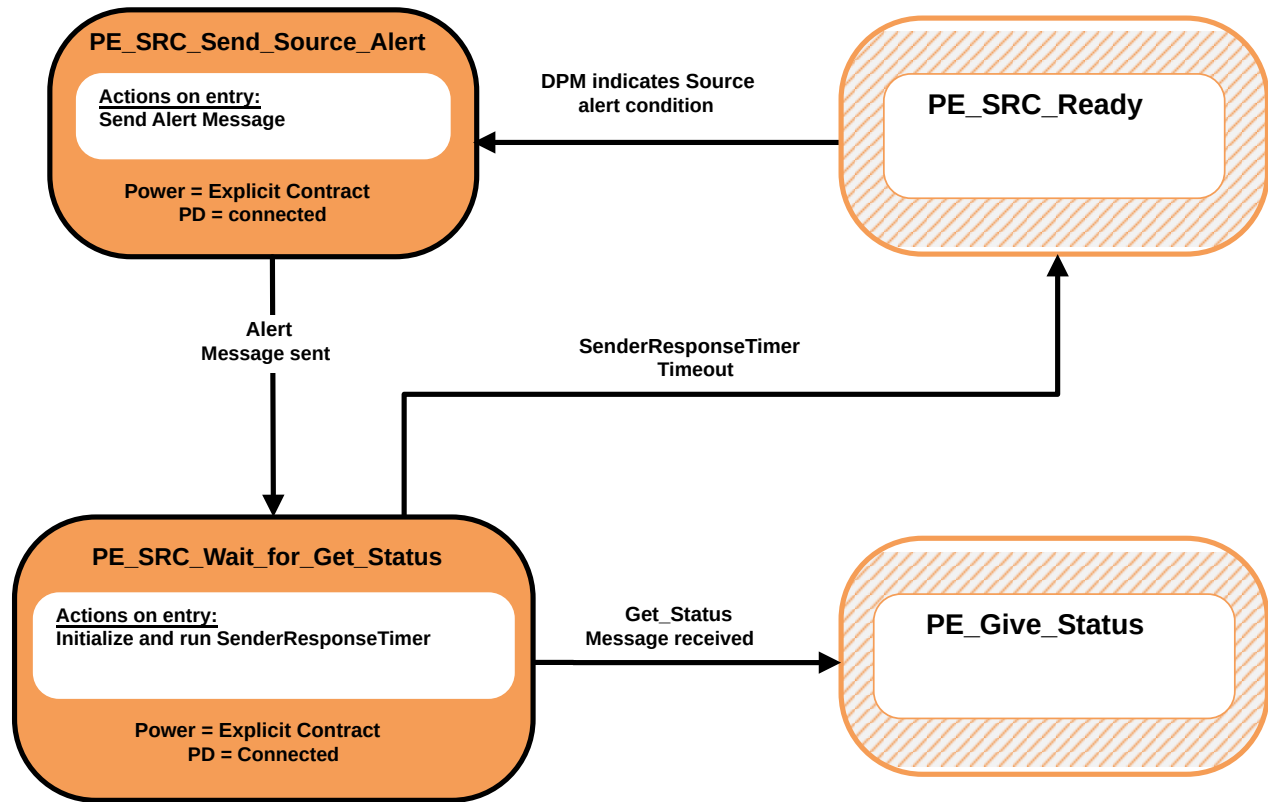
The Policy Engine **shall** transition to PE_SNK_Send_Not_Supported when:

- The ChunkingNotSupportedTimer has timed out.

9.2.8. Alert State Diagrams

9.2.8.1. Source Port Source Alert State Diagram

[Figure 9.25](#) shows the State diagram for an [Alert Message](#) sent by a Source Port.

Figure 9.25. Source Port Source Alert State Diagram**9.2.8.1.1. PE_SRC_Send_Source_Alert State**

The PE_SRC_Send_Source_Alert State **Shall** be entered from the PE_SRC_Ready State when the Device Policy Manager indicates that there is a Source alert condition to be reported.

On entry to the PE_SRC_Send_Source_Alert State the Policy Engine **Shall** Request the Protocol Layer to send an [Alert Message](#).

The Policy Engine **Shall** transition to the PE_SRC_Wait_for_Get_Status State when:

- The [Alert Message](#) has been successfully sent.

9.2.8.1.2. PE_SRC_Wait_for_Get_Status State

On entry to the PE_SRC_Wait_for_Get_Status State the Policy Engine **Shall** initialize and run the SenderResponseTimer.

The Policy Engine **Shall** transition back to the PE_Give_Status State (see [Figure 9.36](#)) when:

- A [Get_Status Message](#) is received.

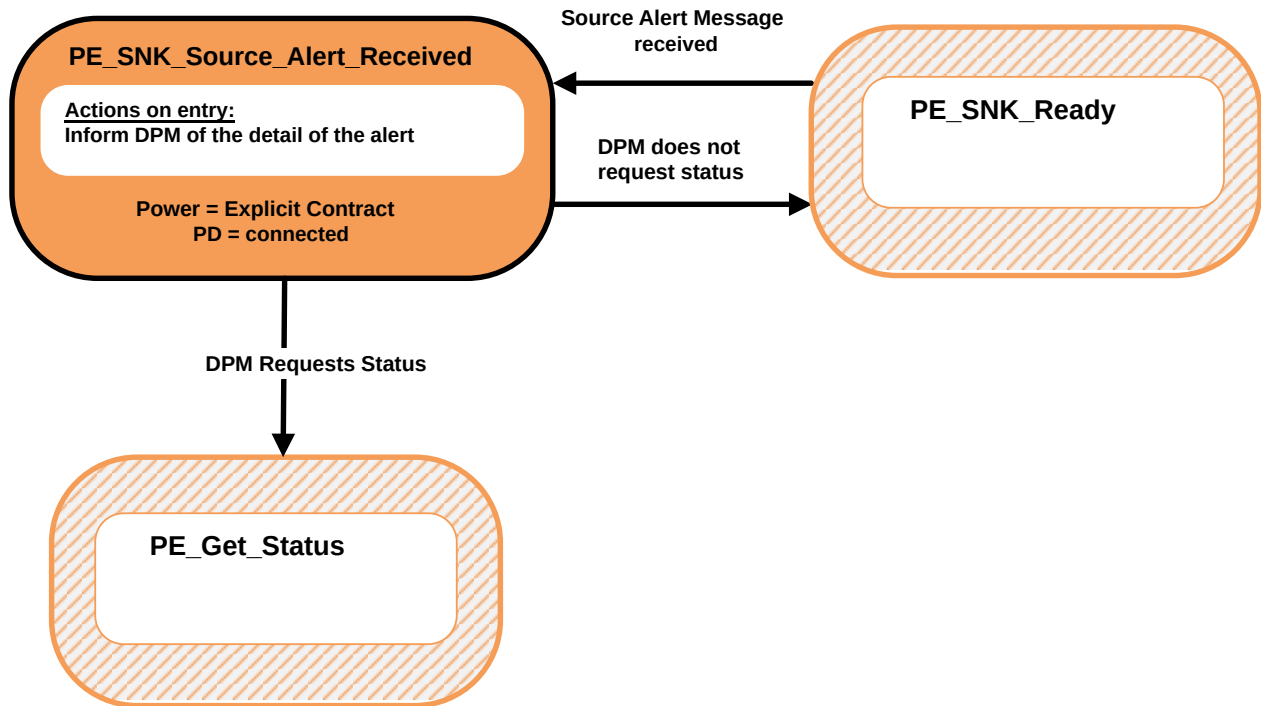
The Policy Engine **Shall** transition back to PE_SRC_Ready (see [Figure 9.17](#)) when:

- The SenderResponseTimer times out.

9.2.8.2. Sink Port Source Alert State Diagram

Figure 9.26 shows the State diagram for an [Alert Message](#) received by a Sink Port.

Figure 9.26. Sink Port Source Alert State Diagram



9.2.8.2.1. PE_SNK_Source_Alert_Received State

The **PE_SNK_Source_Alert_Received** State **Shall** be entered from the **PE_SNK_Ready** State when an [Alert Message](#) is received.

On entry to the **PE_SNK_Source_Alert_Received** State the Policy Engine **Shall** inform the Device Policy Manager of the details of the Source alert.

The Policy Engine **Shall** transition to the **PE_Get_Status** State (see [Figure 9.35](#)) when:

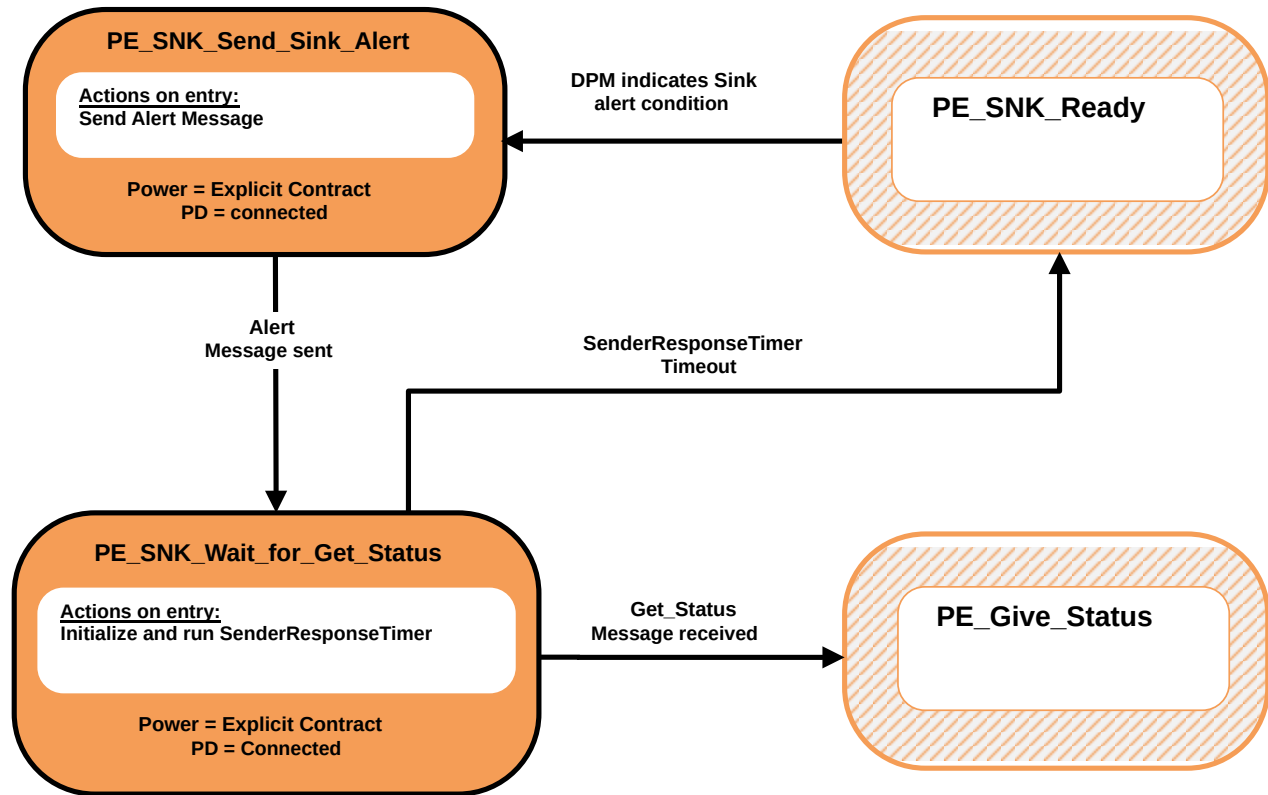
- The DPM requests status.

The Policy Engine **Shall** transition back to the **PE_SNK_Ready** State (see [Figure 9.18](#)) when:

- The DPM does not Request status.

9.2.8.3. Sink Port Sink Alert State Diagram

Figure 9.27 shows the State diagram for an [Alert Message](#) sent by a Sink Port.

Figure 9.27. Sink Port Sink Alert State Diagram**9.2.8.3.1. PE_SNK_Send_Sink_Alert State**

The PE_SNK_Send_Sink_Alert State **Shall** be entered from the PE_SNK_Ready State when the Device Policy Manager indicates that there is a Source alert condition to be reported.

On entry to the PE_SNK_Send_Sink_Alert State the Policy Engine **Shall** Request the Protocol Layer to send an [Alert Message](#).

The Policy Engine **Shall** transition to the PE_SNK_Wait_for_Get_Status State when:

- The [Alert Message](#) has been successfully sent.

9.2.8.3.2. PE_SNK_Wait_for_Get_Status State

On entry to the PE_SNK_Wait_for_Get_Status State the Policy Engine **Shall** initialize and run the SenderResponseTimer.

The Policy Engine **Shall** transition back to the PE_Give_Status State (see [Figure 9.36](#)) when:

- A [Get_Status Message](#) is received.

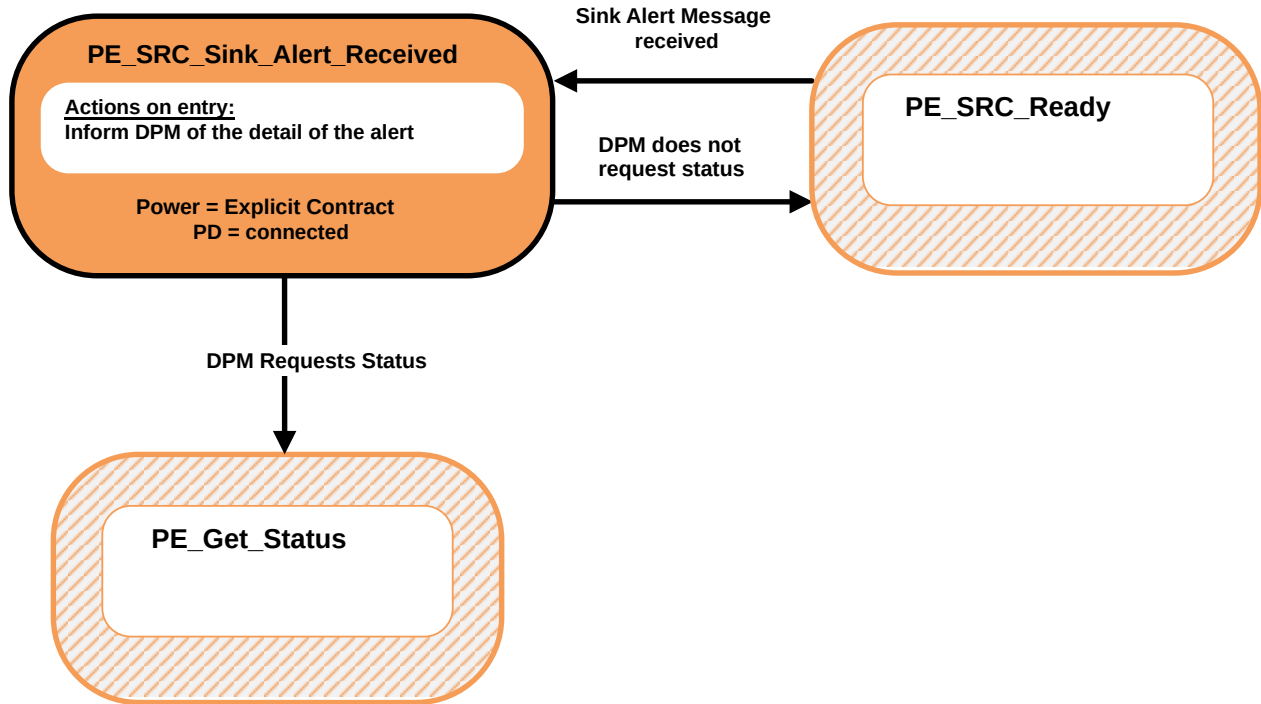
The Policy Engine **Shall** transition back to the PE_SNK_Ready (see [Figure 9.18](#)) when:

- The SenderResponseTimer times out.

9.2.8.4. Source Port Sink Alert State Diagram

Figure 9.28 shows the State diagram for an [Alert Message](#) received by a Source Port.

Figure 9.28. Source Port Sink Alert State Diagram



9.2.8.4.1. PE_SRC_Sink_Alert_Received State

The **PE_SRC_Sink_Alert_Received** State **Shall** be entered from the **PE_SRC_Ready** State when an [Alert Message](#) is received.

On entry to the **PE_SRC_Sink_Alert_Received** State the Policy Engine **Shall** inform the Device Policy Manager of the details of the Source alert.

The Policy Engine **Shall** transition to the **PE_Get_Status** State (see [Figure 9.35](#)) when:

- The DPM requests status.

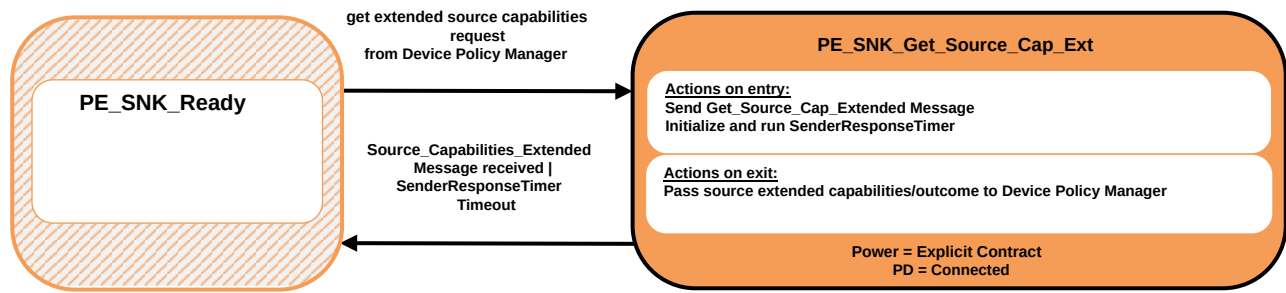
The Policy Engine **Shall** transition back to the **PE_SRC_Ready** (see [Figure 9.17](#)) when:

- The DPM does not Request status.

9.2.9. Source/Sink Capabilities Extended State Diagrams

9.2.9.1. Sink Port Get Source Capabilities Extended State Diagram

[Figure 9.29](#) shows the State diagram for a Sink on receiving a Request from the Device Policy Manager to get the Port Partner's extended Source Capabilities. See also [Section 7.19.2](#).

Figure 9.29. Sink Port Get Source Capabilities Extended State Diagram

9.2.9.1.1. PE_SNK_Get_Source_Cap_Ext State

The Policy Engine **Shall** transition to the PE_SNK_Get_Source_Cap_Ext State, from the PE_SNK_Ready State, due to a Request to get the remote extended Source Capabilities from the Device Policy Manager.

On entry to the PE_SNK_Get_Source_Cap_Ext State the Policy Engine **Shall** send a [Get_Source_Cap_Extended Message](#) and initialize and run the SenderResponseTimer.

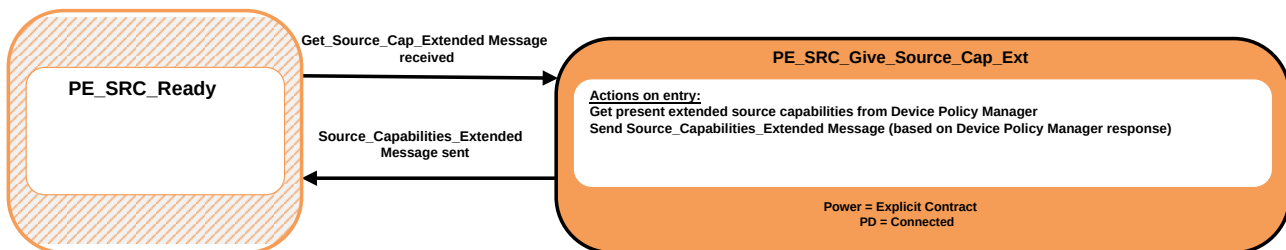
On exit from the PE_SNK_Get_Source_Cap_Ext State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout).

The Policy Engine **Shall** transition back to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

- A [Source_Capabilities_Extended Message](#) is received
- Or SenderResponseTimer times out.

9.2.9.2. Source Give Source Capabilities Extended State Diagram

[Figure 9.30](#) shows the State diagram for a Source on receiving a [Get_Source_Cap_Extended Message](#). See also [Section 7.19.2](#).

Figure 9.30. Source Give Source Capabilities Extended State Diagram

9.2.9.2.1. PE_SRC_Give_Source_Cap_Ext State

The Policy Engine **Shall** transition to the PE_SRC_Give_Source_Cap_Ext State, from the PE_SRC_Ready State, when a [Get_Source_Cap_Extended Message](#) is received.

On entry to the PE_SRC_Give_Source_Cap_Ext State the Policy Engine **Shall** Request the present extended Source Capabilities from the Device Policy Manager and then send a [Source_Capabilities_Extended Message](#) based on these Capabilities.

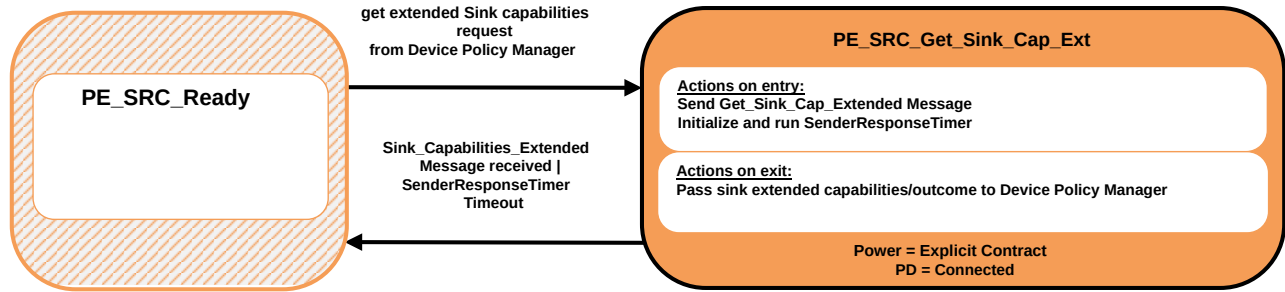
The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#)) when:

- The [Source_Capabilities_Extended Message](#) has been successfully sent.

9.2.9.3. Source Port Get Sink Capabilities Extended State Diagram

[Figure 9.31](#) shows the State diagram for a Source on receiving a Request from the Device Policy Manager to get the Port Partner's extended Sink Capabilities. See also [Section 6.5.16](#).

Figure 9.31. Source Port Get Sink Capabilities Extended State Diagram



9.2.9.3.1. PE_SRC_Get_Sink_Cap_Ext State

The Policy Engine **Shall** transition to the PE_SRC_Get_Sink_Cap_Ext State, from the PE_SRC_Ready State, due to a Request to get the remote extended Source Capabilities from the Device Policy Manager.

On entry to the PE_SRC_Get_Sink_Cap_Ext State the Policy Engine **Shall** send a [Get_Sink_Cap_Extended Message](#) and initialize and run the SenderResponseTimer.

On exit from the PE_SRC_Get_Sink_Cap_Ext State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout).

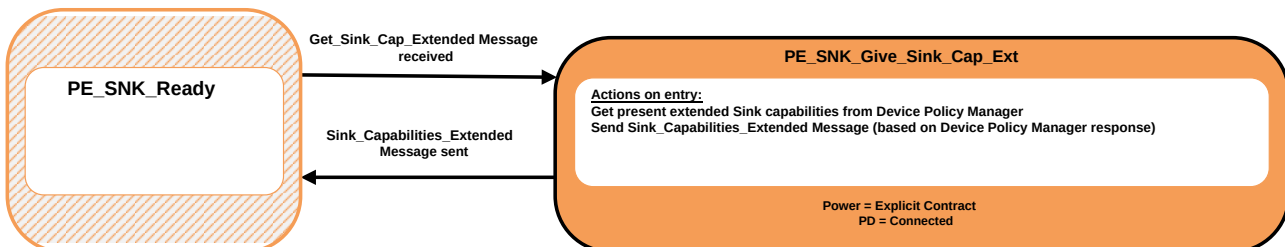
The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#)) when:

- A [Sink_Capabilities_Extended Message](#) is received
- Or SenderResponseTimer times out.

9.2.9.4. Sink Give Sink Capabilities Extended State Diagram

[Figure 9.32](#) shows the State diagram for a Source on receiving a [Get_Sink_Cap_Extended Message](#). See also [Section 6.5.16](#).

Figure 9.32. Sink Give Sink Capabilities Extended State Diagram



9.2.9.4.1. PE_SNK_Give_Sink_Cap_Ext State

The Policy Engine **Shall** transition to the PE_SNK_Give_Sink_Cap_Ext State, from the PE_SNK_Ready State, when a [Get_Sink_Cap_Extended Message](#) is received.

On entry to the PE_SNK_Give_Sink_Cap_Ext State the Policy Engine **Shall** Request the present extended Source Capabilities from the Device Policy Manager and then send a [Sink_Capabilities_Extended Message](#) based on these Capabilities.

The Policy Engine **Shall** transition back to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

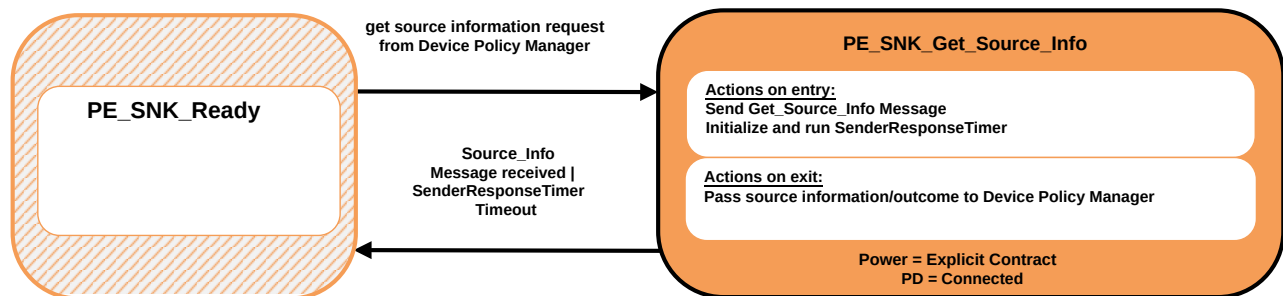
- The [Sink_Capabilities_Extended Message](#) has been successfully sent.

9.2.10. Source Information State Diagrams

9.2.10.1. Sink Port Get Source Information State Diagram

[Figure 9.33](#) shows the State diagram for a Sink on receiving a Request from the Device Policy Manager to get the Port Partner's Source information. See also [Section 6.3.23](#) and [Section 6.4.10](#).

Figure 9.33. Sink Port Get Source Information State Diagram



9.2.10.1.1. PE_SNK_Get_Source_Info State

The Policy Engine **Shall** transition to the PE_SNK_Get_Source_Info State, from the PE_SNK_Ready State, due to a Request to get the remote Source information from the Device Policy Manager.

On entry to the PE_SNK_Get_Source_Info State the Policy Engine **Shall** send a [Get_Source_Info Message](#) and initialize and run the SenderResponseTimer.

On exit from the PE_SNK_Get_Source_Info State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (information or response timeout).

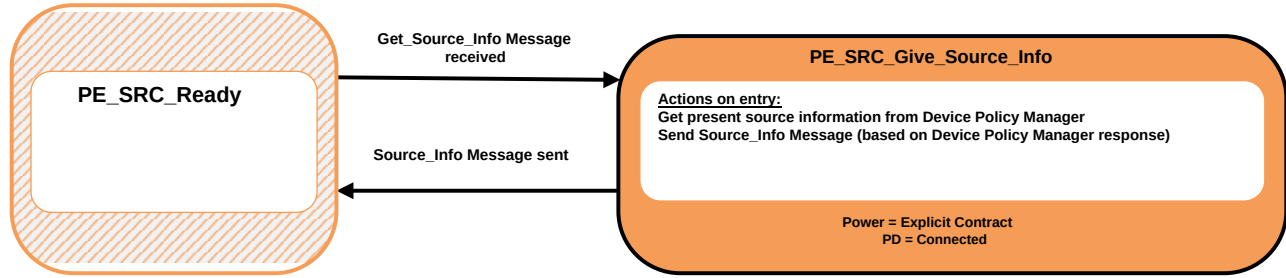
The Policy Engine **Shall** transition back to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

- A [Source_Info Message](#) is received
- Or SenderResponseTimer times out.

9.2.10.2. Source Give Source Information State Diagram

[Figure 9.34](#) shows the State diagram for a Source on receiving a [Get_Source_Info Message](#). See also [Section 6.3.23](#) and [Section 6.4.10](#).

Figure 9.34. Source Give Source Information State Diagram



9.2.10.2.1. PE_SRC_Give_Source_Info State

The Policy Engine **Shall** transition to the PE_SRC_Give_Source_Info State, from the PE_SRC_Ready State, when a [Get_Source_Info Message](#) is received.

On entry to the PE_SRC_Give_Source_Info State the Policy Engine **Shall** Request the present Source information from the Device Policy Manager and then send a [Source_Info Message](#) based on this information.

The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#)) when:

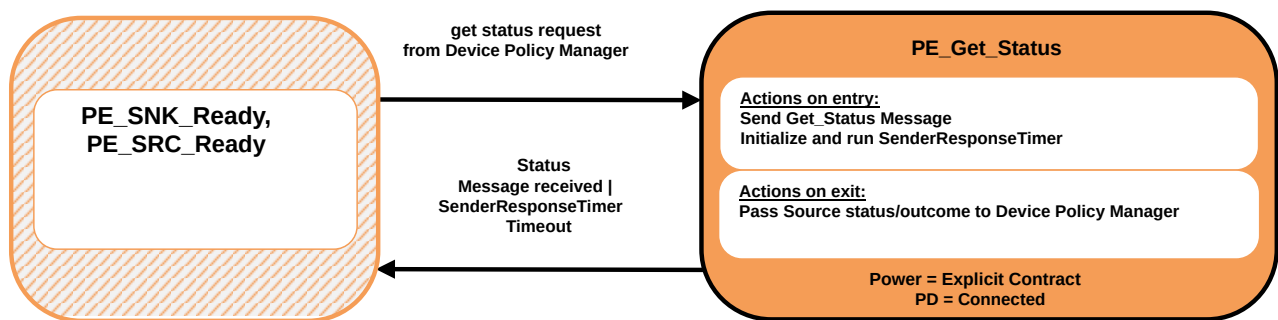
- The [Source_Info Message](#) has been successfully sent.

9.2.11. Status State Diagrams

9.2.11.1. Get Status State Diagram

[Figure 9.35](#) shows the State diagram for a Port on receiving a Request from the Device Policy Manager to get the Port Partner or Cable Plug's Status. See also [Section 6.5.3](#).

Figure 9.35. Get Status State Diagram



9.2.11.1.1. PE_Get_Status State

The Policy Engine **Shall** transition to the PE_Get_Status State, from the PE_SRC_Ready or PE_SNK_Ready States, due to a Request to get the Port Partner or Cable Plug's status from the Device Policy Manager.

On entry to the PE_Get_Status State the Policy Engine **Shall** send a [Get_Status Message](#) and initialize and run the SenderResponseTimer.

On exit from the PE_Get_Status State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (status or response timeout).

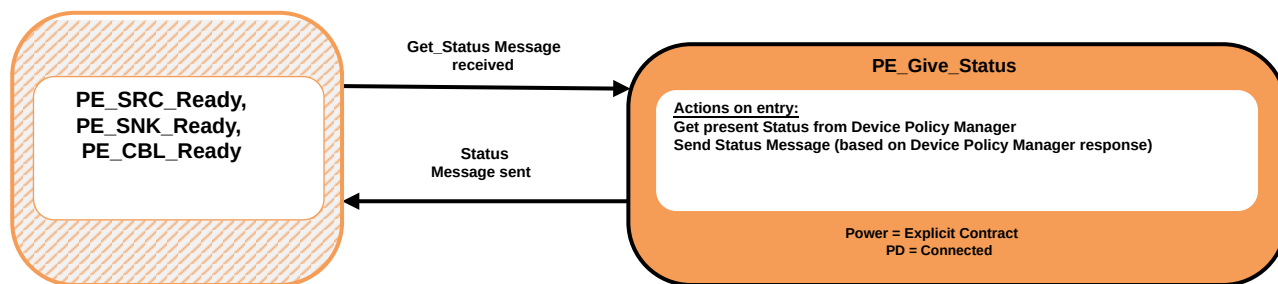
The Policy Engine **Shall** transition back to the PE_SRC_Ready or PE_SNK_Ready States as appropriate (see [Figure 9.17](#) or [Figure 9.18](#)) when:

- A [Status Message](#) is received
- Or SenderResponseTimer times out.

9.2.11.2. Give Status State Diagram

[Figure 9.36](#) shows the State diagram for a Source on receiving a [Get_Status Message](#). See also [Section 6.5.3](#).

Figure 9.36. Give Status State Diagram



9.2.11.2.1. PE_Give_Status State

The Policy Engine **Shall** transition to the PE_Give_Status State, from the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready States, when a [Get_Status Message](#) is received.

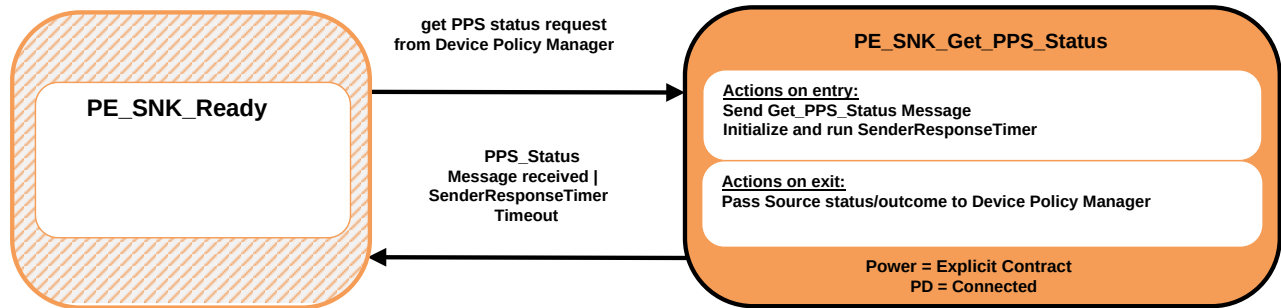
On entry to the PE_Give_Status State the Policy Engine **Shall** Request the present Source status from the Device Policy Manager and then send a Status Message based on these Capabilities.

The Policy Engine **Shall** transition back to the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready States as appropriate (see [Figure 9.17](#) , [Figure 9.18](#) and [Figure 9.88](#)) when:

- The [Status Message](#) has been successfully sent.

9.2.11.3. Sink Port Get Source PPS Status State Diagram

[Figure 9.37](#) shows the State diagram for a Sink on receiving a Request from the Device Policy Manager to get the Port Partner's Source status when operating as a PPS. See also [Section 6.3.20](#).

Figure 9.37. Sink Port Get Source PPS Status State Diagram**9.2.11.3.1. PE_SNK_Get_PPS_Status State**

The Policy Engine **Shall** transition to the PE_SNK_Get_PPS_Status State, from the PE_SNK_Ready State, due to a Request to get the remote Source PPS status from the Device Policy Manager.

On entry to the PE_SNK_Get_PPS_Status State the Policy Engine **Shall** send a [Get_PPS_Status Message](#) and initialize and run the SenderResponseTimer.

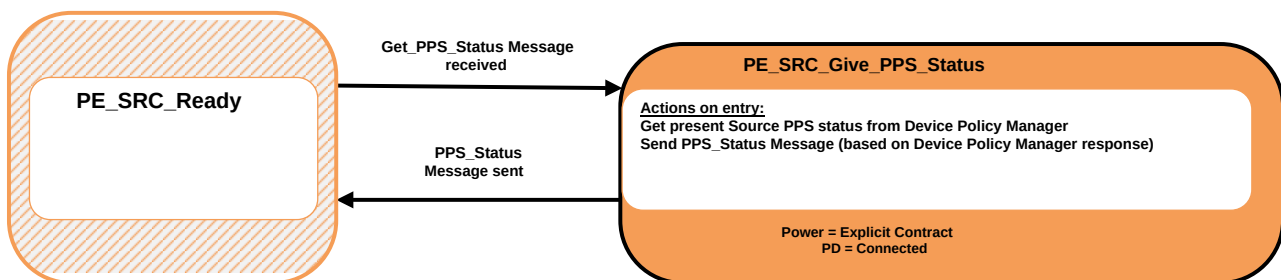
On exit from the PE_SNK_Get_PPS_Status State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (status or response timeout).

The Policy Engine **Shall** transition back to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

- A [PPS_Status Message](#) is received
- Or SenderResponseTimer times out.

9.2.11.4. Source Give Source PPS Status State Diagram

[Figure 9.38](#) shows the State diagram for a Source on receiving a [Get_PPS_Status Message](#). See also [Section 6.3.20](#).

Figure 9.38. Source Give Source PPS Status State Diagram**9.2.11.4.1. PE_SRC_Give_PPS_Status State**

The Policy Engine **Shall** transition to the PE_SRC_Give_PPS_Status State, from the PE_SRC_Ready State, when a [Get_PPS_Status Message](#) is received.

On entry to the PE_SRC_Give_PPS_Status State the Policy Engine **Shall** Request the present Source PPS status from the Device Policy Manager and then send a [PPS_Status Message](#) based on these Capabilities.

The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#)) when:

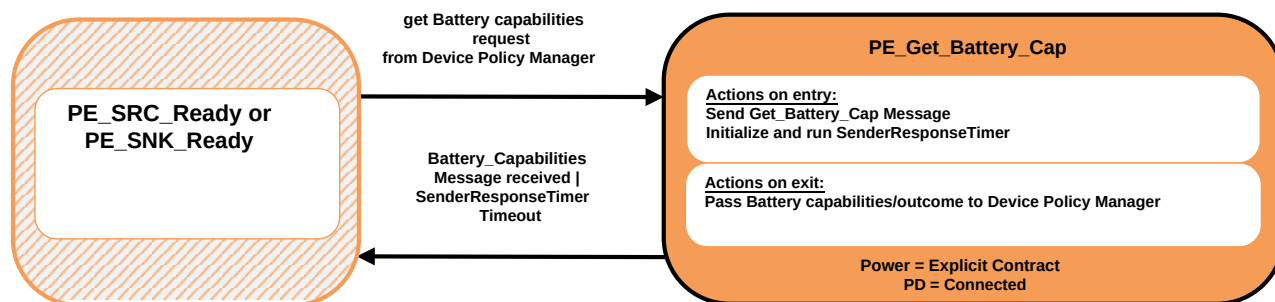
- The [PPS_Status Message](#) has been successfully sent.

9.2.12. Battery Capabilities State Diagrams

9.2.12.1. Get Battery Capabilities State Diagram

[Figure 9.39](#) shows the State diagram for a Source or Sink on receiving a Request from the Device Policy Manager to get the Port Partner's [Battery Capabilities](#) for a specified Battery. See also [Section 6.5.6](#).

Figure 9.39. Get Battery Capabilities State Diagram



9.2.12.1.1. PE_Get_Battery_Cap State

The Policy Engine **Shall** transition to the PE_Get_Battery_Cap State, from either the PE_SRC_Ready or PE_SNK_Ready State, due to a Request to get the remote [Battery Capabilities](#), for a specified Battery, from the Device Policy Manager.

On entry to the PE_Get_Battery_Cap State the Policy Engine **Shall** send a [Get_Battery_Cap Message](#) and initialize and run the SenderResponseTimer.

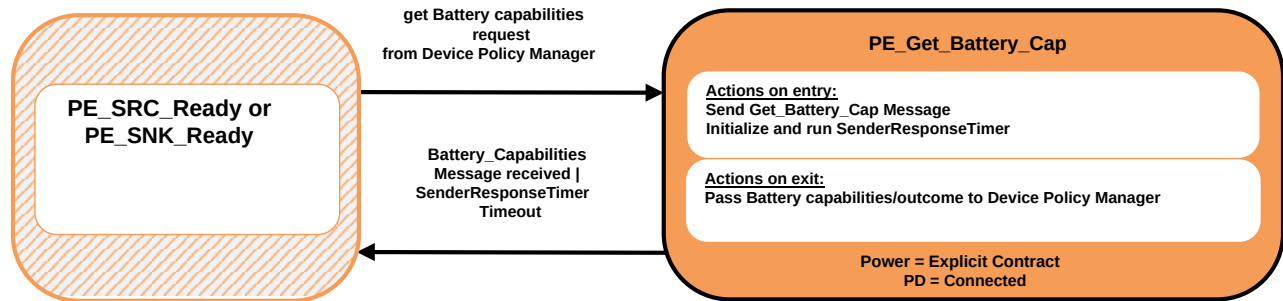
On exit from the PE_Get_Battery_Cap State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout).

The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) [Figure 9.18](#)) when:

- A [Battery_Capabilities Message](#) is received
- Or SenderResponseTimer times out.

9.2.12.2. Give Battery Capabilities State Diagram

[Figure 9.40](#) shows the State diagram for a Source or Sink on receiving a [Get_Battery_Cap](#) Message. See also [Section 6.5.6](#).

Figure 9.40. Give Battery Capabilities State Diagram

9.2.12.2.1. PE_Give_Battery_Cap State

The Policy Engine **Shall** transition to the PE_Give_Battery_Cap State, from either the PE_SRC_Ready or PE_SNK_Ready State, when a [Get_Battery_Cap Message](#) is received.

On entry to the PE_Give_Battery_Cap State the Policy Engine **Shall** Request the present [Battery Capabilities](#), for the requested Battery, from the Device Policy Manager and then send a [Battery_Capabilities Message](#) based on these Capabilities.

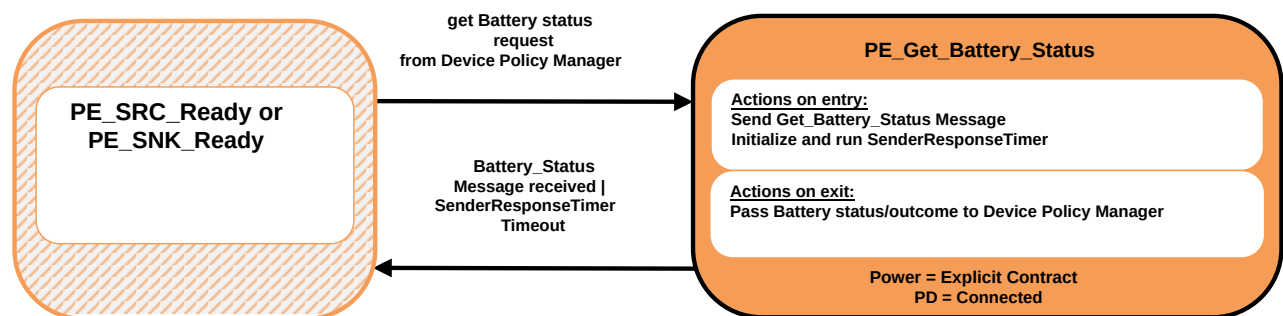
The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- The [Battery_Capabilities Message](#) has been successfully sent.

9.2.13. Battery Status State Diagrams

9.2.13.1. Get Battery Status State Diagram

[Figure 9.41](#) shows the State diagram for a Source or Sink on receiving a Request from the Device Policy Manager to get the Port Partner's Battery status for a specified Battery. See also [Section 6.5.5](#).

Figure 9.41. Get Battery Status State Diagram

9.2.13.1.1. PE_Get_Battery_Status State

The Policy Engine **Shall** transition to the PE_Get_Battery_Status State, from either the PE_SRC_Ready or PE_SNK_Ready State, due to a Request to get the remote Battery status, for a specified Battery, from the Device Policy Manager.

On entry to the PE_Get_Battery_Status State the Policy Engine **Shall** send a [Get_Battery_Status Message](#) and initialize and run the SenderResponseTimer.

On exit from the PE_Get_Battery_Status State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (status or response timeout).

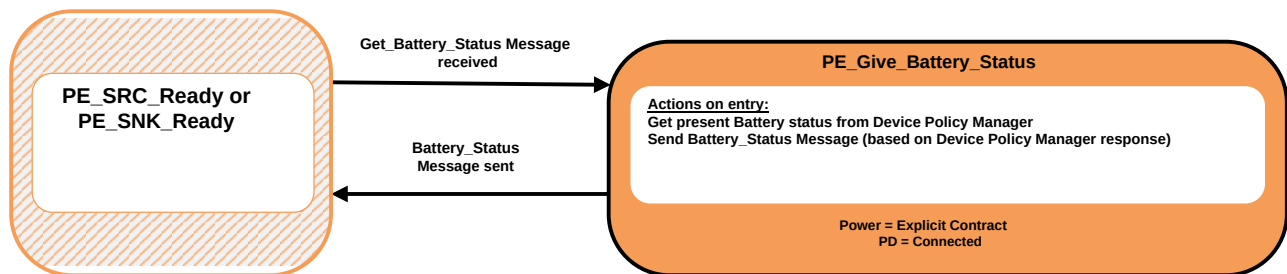
The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- A [Battery_Status Message](#) is received
- Or SenderResponseTimer times out.

9.2.13.2. Give Battery Status State Diagram

[Figure 9.42](#) shows the State diagram for a Source or Sink on receiving a [Get_Battery_Status Message](#). See also [Section 6.5.5](#).

Figure 9.42. Give Battery Status State Diagram



9.2.13.2.1. PE_Give_Battery_Status State

The Policy Engine **Shall** transition to the PE_Give_Battery_Status State, from either the PE_SRC_Ready or PE_SNK_Ready State, when a [Get_Battery_Status Message](#) is received.

On entry to the PE_Give_Battery_Status State the Policy Engine **Shall** Request the present Battery status, for the requested Battery, from the Device Policy Manager and then send a [Battery_Status Message](#) based on this status.

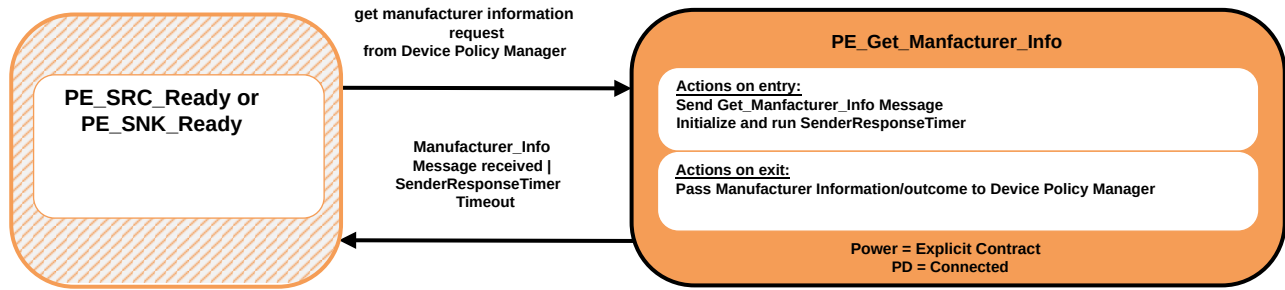
The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- The [Battery_Status Message](#) has been successfully sent.

9.2.14. Manufacturer Information State Diagrams

9.2.14.1. Get Manufacturer Information State Diagram

[Figure 9.43](#) shows the State diagram for a Source or Sink on receiving a Request from the Device Policy Manager to get the Port Partner or Cable Plug's Manufacturer Information. See also [Section 6.5.7](#).

Figure 9.43. Get Manufacturer Information State Diagram**9.2.14.1.1. PE_Get_Manufacturer_Info State**

The Policy Engine **Shall** transition to the PE_Get_Manufacturer_Info State, from either the PE_SRC_Ready or PE_SNK_Ready State, due to a Request to get the remote [Manufacturer Information](#) from the Device Policy Manager.

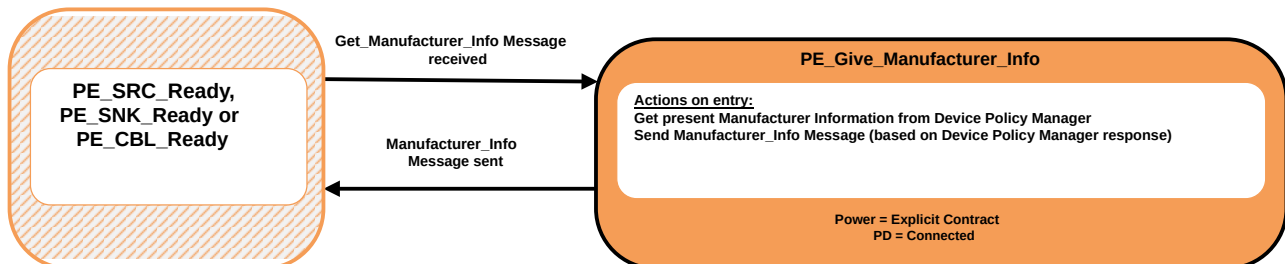
On entry to the PE_Get_Manufacturer_Info State the Policy Engine **Shall** send a [Get_Manufacturer_Info Message](#) and initialize and run the SenderResponseTimer.

On exit from the PE_Get_Manufacturer_Info State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (information or response timeout). The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- A [Manufacturer_Info Message](#) is received
- Or SenderResponseTimer times out.

9.2.14.2. Give Manufacturer Information State Diagram

[Figure 9.44](#) shows the State diagram for a Source, Sink or Cable Plug on receiving a [Get_Manufacturer_Info Message](#). See also [Section 6.5.7](#).

Figure 9.44. Give Manufacturer Information State Diagram**9.2.14.2.1. PE_Give_Manufacturer_Info State**

The Policy Engine **Shall** transition to the PE_Give_Manufacturer_Info State, from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State, when a [Get_Manufacturer_Info Message](#) is received.

On entry to the PE_Give_Manufacturer_Info State the Policy Engine **Shall** Request the manufacturer information from the Device Policy Manager and then send a [Manufacturer_Info Message](#) based on this status.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State as appropriate (see [Figure 9.17](#) , [Figure 9.18](#) and [Figure 9.88](#)) when:

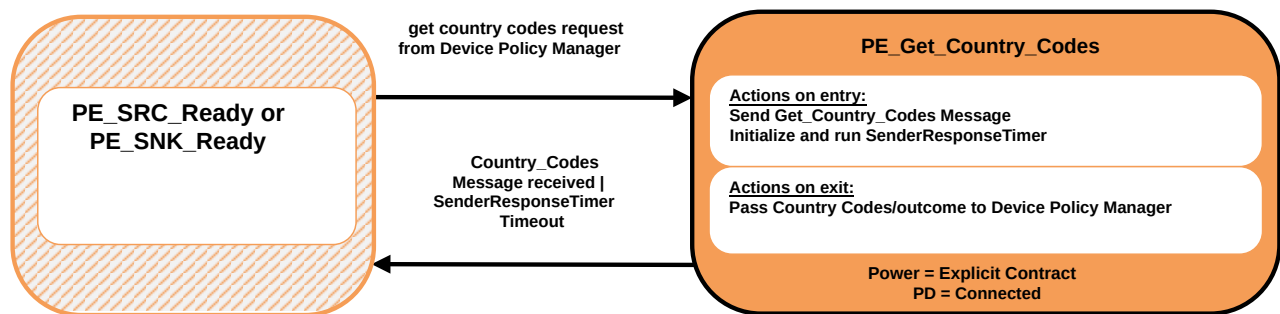
- The [Manufacturer_Info Message](#) has been successfully sent.

9.2.15. Country Codes and Information State Diagrams

9.2.15.1. Get Country Codes State Diagram

[Figure 9.45](#) shows the State diagram for a Source or Sink on receiving a Request from the Device Policy Manager to get the Port Partner or Cable Plug's [Country Codes](#). See also [Section 6.3.21](#).

Figure 9.45. Get Country Codes State Diagram



9.2.15.1.1. PE_Get_Country_Codes State

The Policy Engine **Shall** transition to the PE_Get_Country_Codes State, from either the PE_SRC_Ready or PE_SNK_Ready State, due to a Request to get the remote [Country Codes](#) from the Device Policy Manager.

On entry to the PE_Get_Country_Codes State the Policy Engine **Shall** send a [Get_Country_Codes Message](#) and initialize and run the SenderResponseTimer.

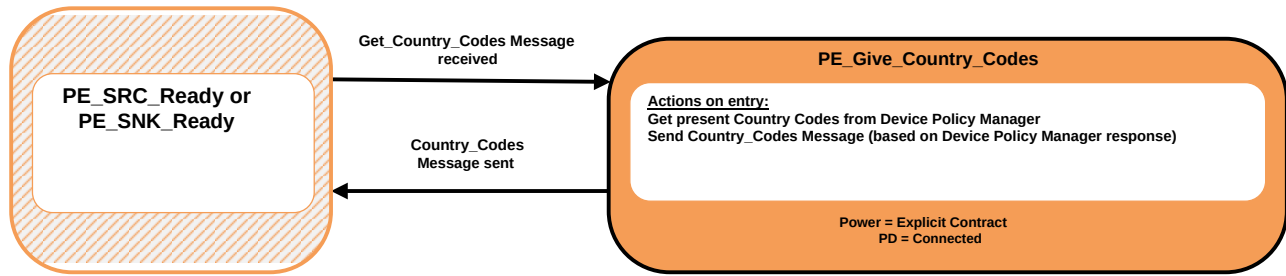
On exit from the PE_Get_Country_Codes State the Policy Engine **Shall** inform the Device Policy Manager of the outcome ([Country Codes](#) or response timeout).

The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- A [Country_Codes Message](#) is received
- Or SenderResponseTimer times out.

9.2.15.2. Give Country Codes State Diagram

[Figure 9.46](#) shows the State diagram for a Source or Sink on receiving a [Get_Country_Codes](#) Message. See also [Section 6.3.21](#).

Figure 9.46. Give Country Codes State Diagram**9.2.15.2.1. PE_Give_Country_Codes State**

The Policy Engine **Shall** transition to the **PE_Give_Country_Codes** State, from either the **PE_SRC_Ready** or **PE_SNK_Ready** State, when a [Get_Country_Codes Message](#) is received.

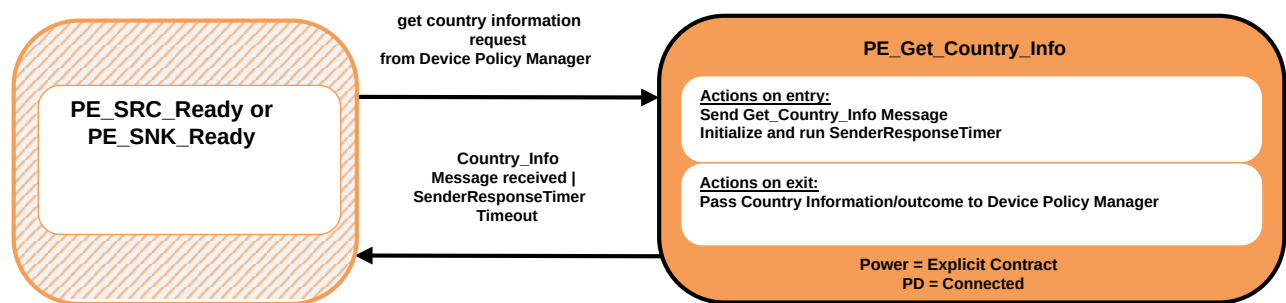
On entry to the **PE_Give_Country_Codes** State the Policy Engine **Shall** Request the country codes from the Device Policy Manager and then send a [Country_Codes](#) Message containing these codes.

The Policy Engine **Shall** transition back to either the **PE_SRC_Ready** or **PE_SNK_Ready** State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- The [Country_Codes Message](#) has been successfully sent.

9.2.15.3. Get Country Information State Diagram

[Figure 9.47](#) shows the State diagram for a Source or Sink on receiving a Request from the Device Policy Manager to get the Port Partner or Cable Plug's [Country Information](#). See also [Section 6.4.6](#).

Figure 9.47. Get Country Information State Diagram**9.2.15.3.1. PE_Get_Country_Info State**

The Policy Engine **Shall** transition to the **PE_Get_Country_Info** State, from either the **PE_SRC_Ready** or **PE_SNK_Ready** State, due to a Request to get the remote [Manufacturer Information](#) from the Device Policy Manager.

On entry to the **PE_Get_Country_Info** State the Policy Engine **Shall** send a [Get_Manufacturer_Info Message](#) and initialize and run the **SenderResponseTimer**.

On exit from the PE_Get_Country_Info State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (country information or response timeout).

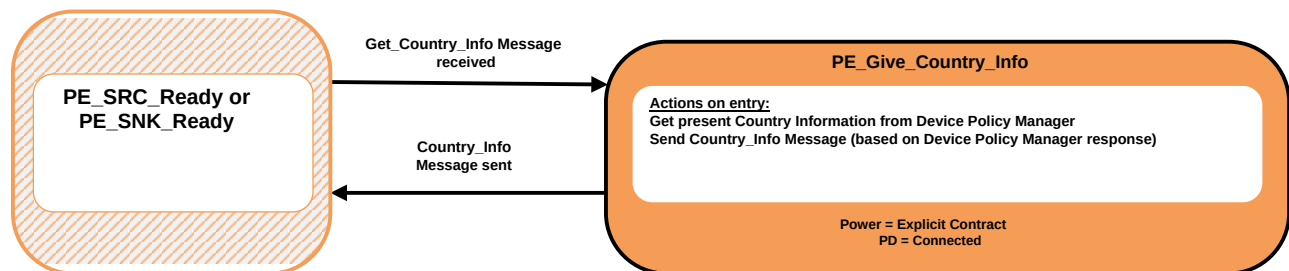
The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- A [Country_Info Message](#) is received
- Or SenderResponseTimer times out.

9.2.15.4. Give Country Information State Diagram

[Figure 9.48](#) shows the State diagram for a Source or Sink on receiving a [Get_Country_Info](#) Message. See also [Section 6.4.6](#).

Figure 9.48. Give Country Information State Diagram



9.2.15.4.1. PE_Give_Country_Info State

The Policy Engine **Shall** transition to the PE_Give_Country_Info State, from either the PE_SRC_Ready or PE_SNK_Ready State, when a [Get_Country_Info Message](#) is received.

On entry to the PE_Give_Country_Info State the Policy Engine **Shall** Request the country information from the Device Policy Manager and then send a [Country_Info Message](#) containing this country information.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

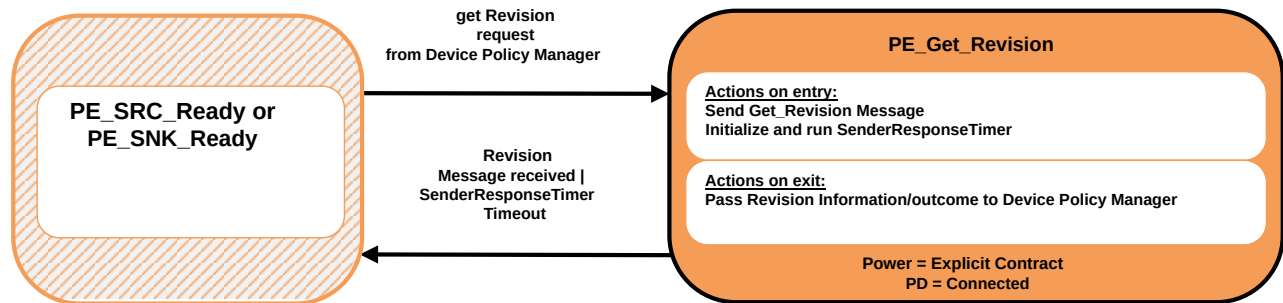
- The [Country_Info Message](#) has been successfully sent.

9.2.16. Revision State Diagrams

9.2.16.1. Get Revision State Diagram

[Figure 9.49](#) shows the State diagram for a Source or Sink on receiving a Request from the Device Policy Manager to get the Port Partner or Cable Plug's [Revision Information](#). See also [Section 6.3.24](#) and [Section 6.4.11](#).

Figure 9.49. Get Revision State Diagram



9.2.16.1.1. PE_Get_Revision State

The Policy Engine **Shall** transition to the PE_Get_Revision State, from either the PE_SRC_Ready or PE_SNK_Ready State, due to a Request to get the remote [Revision Information](#) from the Device Policy Manager.

On entry to the PE_Get_Revision State the Policy Engine **Shall** send a [Get_Revision Message](#) and initialize and run the SenderResponseTimer.

On exit from the PE_Get_Revision State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Revision information or response timeout).

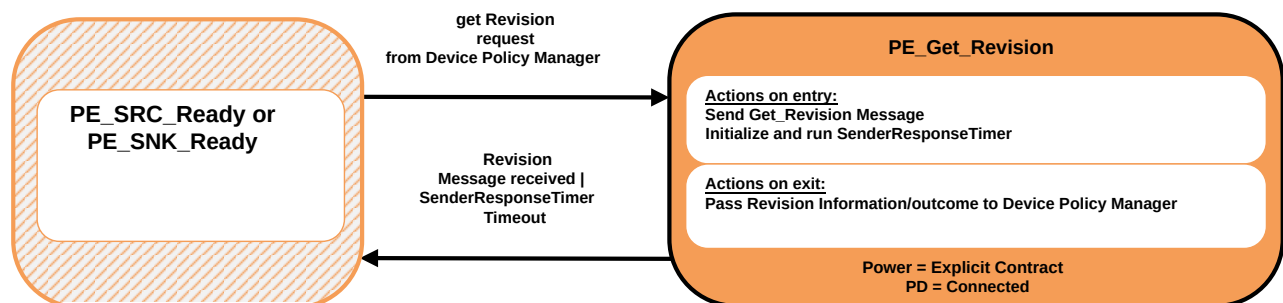
The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- A [Revision Message](#) is received
- Or SenderResponseTimer times out.

9.2.16.2. Give Revision State Diagram

[Figure 9.50](#) shows the State diagram for a Source, Sink or Cable Plug on receiving a [Get_Revision Message](#). See also [Section 6.3.24](#) and [Section 6.4.11](#).

Figure 9.50. Give Revision State Diagram



9.2.16.2.1. PE_Give_Revision State

The Policy Engine **Shall** transition to the PE_Give_Revision State, from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State, when a [Get_Revision](#) Message is received.

On entry to the PE_Give_Revision State the Policy Engine **Shall** Request the Revision information from the Device Policy Manager and then send a Revision Message based on this information.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State as appropriate (see [Figure 9.17](#) , [Figure 9.18](#) and [Figure 9.88](#)) when:

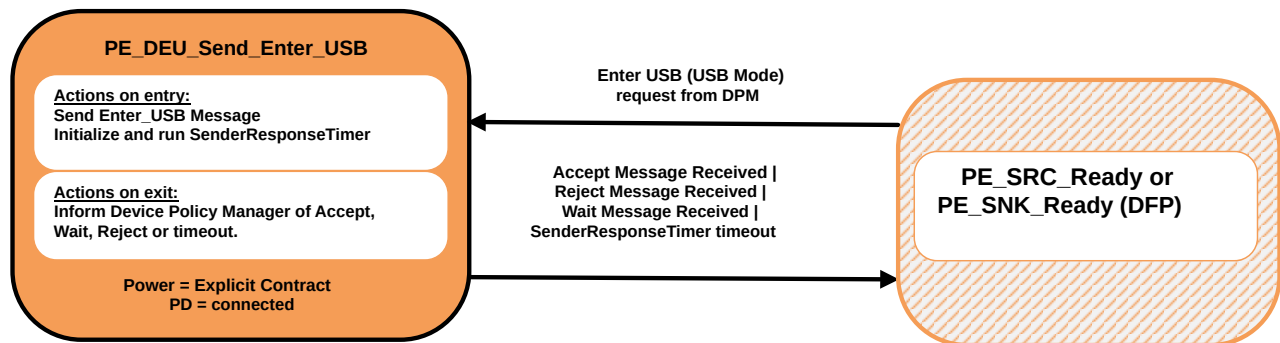
- The [Revision Message](#) has been successfully sent.

9.2.17. Enter_USB Message State Diagrams

9.2.17.1. DFP Enter_USB Message State Diagrams

[Figure 9.51](#) shows the State diagram for an [Enter_USB Message](#) sent by a DFP.

Figure 9.51. DFP Enter_USB Message State Diagram



9.2.17.1.1. PE_DEU_Send_Enter_USB State

The PE_DEU_Send_Enter_USB State **Shall** be entered from the PE_SRC_Ready or PE_SNK_Ready State when requested by the Device Policy Manager and the Port is operating as a DFP. On entry to the PE_DEU_Send_Enter_USB State the Policy Engine **Shall** Request the Protocol Layer to send an [Enter_USB Message](#) and then initialize and run the SenderResponseTimer.

On exit from the PE_DEU_Send_Enter_USB State the Policy Engine **Shall** inform the Device Policy Manager of the outcome: [Accept Message](#) received, [Reject Message](#) received, SenderResponseTimer timeout.

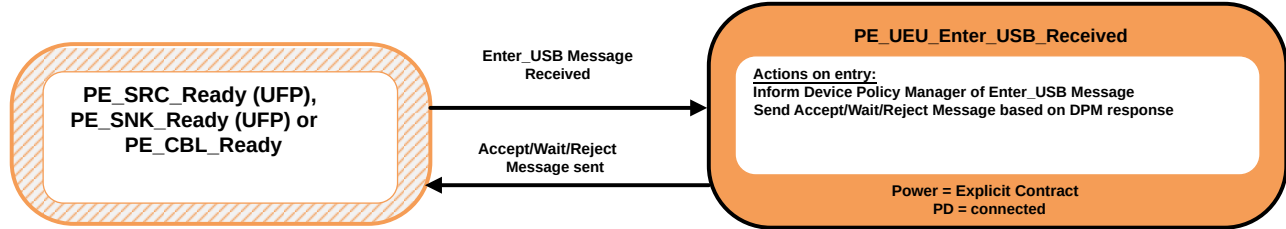
The Policy Engine **Shall** transition back to the PE_SRC_Ready or PE_SNK_Ready State depending on the Ports Power Role when:

- An [Accept Message](#) has been received or
- A [Wait Message](#) has been received or
- A [Reject Message](#) has been received
- There is a SenderResponseTimer timeout.

9.2.17.2. UFP or Cable Plug Enter_USB Message State Diagrams

[Figure 9.52](#) shows the State diagram for an [Enter_USB Message](#) received by a UFP or Cable Plug.

Figure 9.52. UFP Enter_USB Message State Diagram



9.2.17.2.1. PE_UEU_Enter_USB_Received State

The PE_UEU_Enter_USB_Received State **Shall** be entered from the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State as appropriate (see [Figure 9.17](#), [Figure 9.18](#) and [Figure 9.88](#)) when an [Enter_USB Message](#) is received and the Port is operating as a UFP or is a Cable Plug.

On entry to the PE_UEU_Enter_USB_Received State the Policy Engine **Shall** inform the Device Policy Manager. The Device Policy Manager responds with an indication of whether the [Enter_USB Message](#) is to be accepted or rejected. The Policy Engine **Shall** send either an [Accept Message](#), a [Wait Message](#) or a [Reject Message](#) as appropriate.

The Policy Engine **Shall** transition back to the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State as appropriate when:

- Either an [Accept Message](#), a [Wait Message](#) or a [Reject Message](#) has been sent.

9.2.18. Security State Diagrams

9.2.18.1. Send Security Request State Diagram

[Figure 9.53](#) shows the State diagram for a Source or Sink on receiving a Request from the Device Policy Manager to send a security Request.

Figure 9.53. Send security Request State Diagram



9.2.18.1.1. PE_Send_Security_Request State

The Policy Engine **Shall** transition to the PE_Send_Security_Request State, from either the PE_SRC_Ready or PE_SNK_Ready State, due to a Request to send a security Request from the Device Policy Manager.

On entry to the PE_Send_Security_Request State the Policy Engine **Shall** send a [Security_Request Message](#).

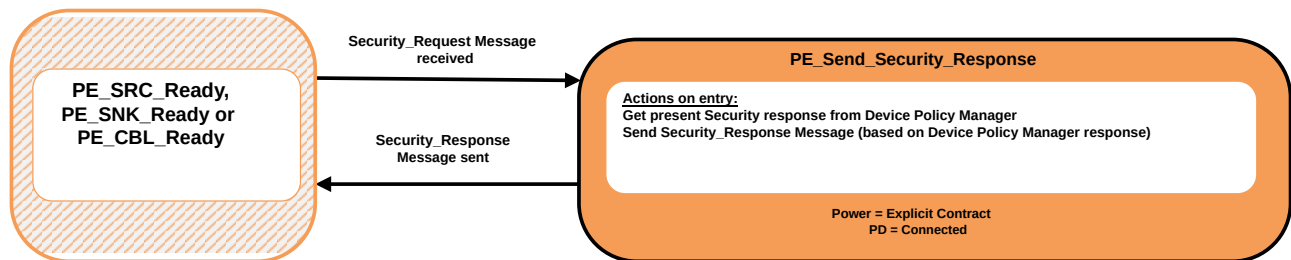
The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- The [Security_Request Message](#) has been sent.

9.2.18.2. Send Security Response State Diagram

[Figure 9.54](#) shows the State diagram for a Source, Sink or Cable Plug on receiving a [Security_Request Message](#). See also [Section 6.5.9](#).

Figure 9.54. Send security response State Diagram



9.2.18.2.1. PE_Send_Security_Response State

The Policy Engine **Shall** transition to the PE_Send_Security_Response State, from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State, when a [Security_Request Message](#) is received.

On entry to the PE_Send_Security_Response State the Policy Engine **Shall** Request the appropriate response from the Device Policy Manager and then send a [Security_Response Message](#) based on this status.

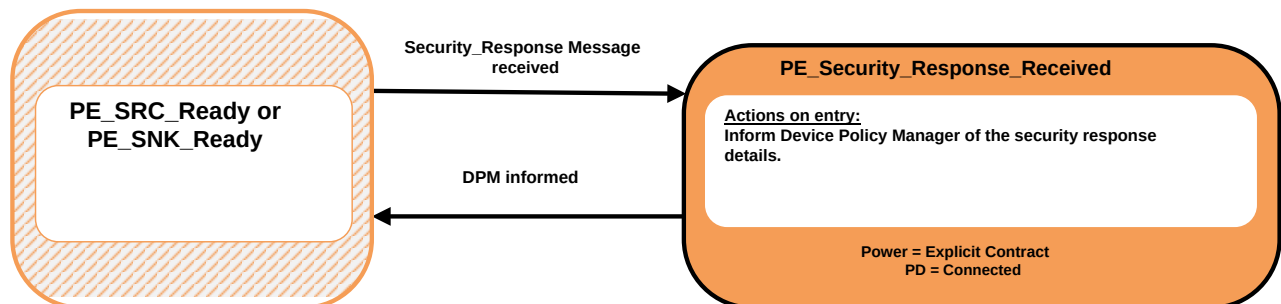
The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State as appropriate (see [Figure 9.17](#) , [Figure 9.18](#) and [Figure 9.88](#)) when:

- The [Security_Response Message](#) has been successfully sent.

9.2.18.3. Security Response Received State Diagram

[Figure 9.55](#) shows the State diagram for a Source or Sink on receiving a [Security_Response](#) Message. See also [Section 6.5.10](#).

Figure 9.55. Security response received State Diagram



9.2.18.3.1. PE_Security_Response_Received State

The Policy Engine **Shall** transition to the PE_Security_Response_Received State, from either the PE_SRC_Ready or PE_SNK_Ready when a [Security_Response](#) Message is received.

On entry to the PE_Security_Response_Received State the Policy Engine **Shall** inform the Device Policy Manager of the details of the security response.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) , [Figure 9.18](#) and [Figure 9.88](#)) when:

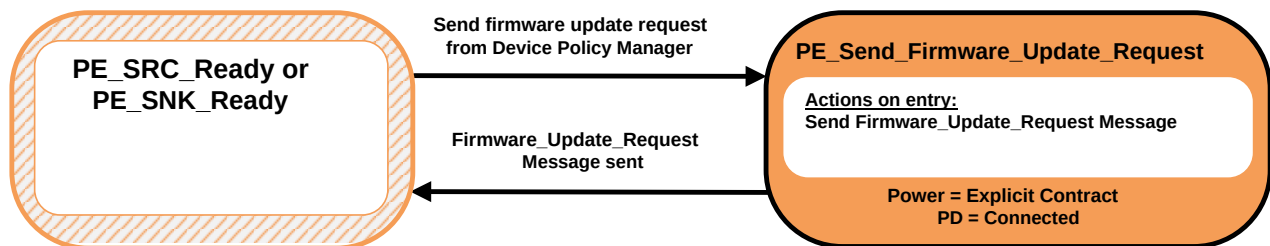
- The Device Policy Manager has been informed.

9.2.19. Firmware Update State Diagrams

9.2.19.1. Send Firmware Update Request State Diagram

[Figure 9.56](#) shows the State diagram for a Source or Sink on receiving a Request from the Device Policy Manager to send a firmware update Request.

Figure 9.56. Send firmware update Request State Diagram



9.2.19.1.1. PE_Send_Firmware_Update_Request State

The Policy Engine **Shall** transition to the PE_Send_Firmware_Update_Request State, from either the PE_SRC_Ready or PE_SNK_Ready State, due to a Request to send a firmware update Request from the Device Policy Manager.

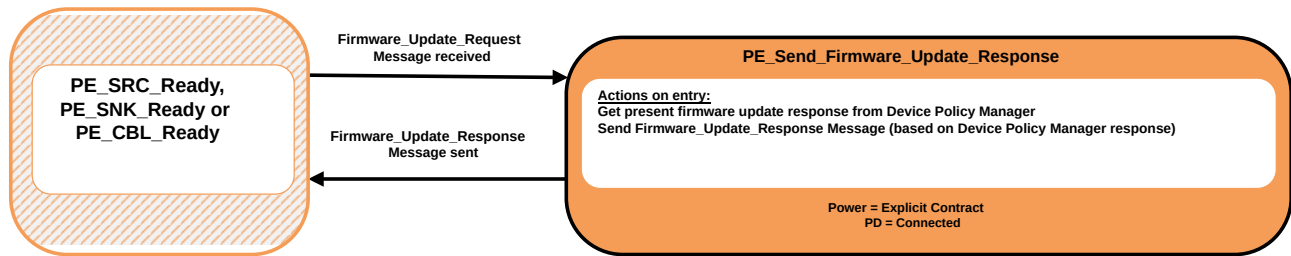
On entry to the PE_Send_Firmware_Update_Request State the Policy Engine **Shall** send a [Firmware_Update_Request Message](#).

The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) and [Figure 9.18](#)) when:

- The [Firmware_Update_Request Message](#) has been sent.

9.2.19.2. Send Firmware Update Response State Diagram

[Figure 9.57](#) shows the State diagram for a Source, Sink or Cable Plug on receiving a [Firmware_Update_Request Message](#). See also [Section 6.5.11](#).

Figure 9.57. Send firmware update response State Diagram

9.2.19.2.1. PE_Send_Firmware_Update_Response State

The Policy Engine **Shall** transition to the PE_Send_Firmware_Update_Response State, from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State, when a [Firmware_Update_Request Message](#) is received.

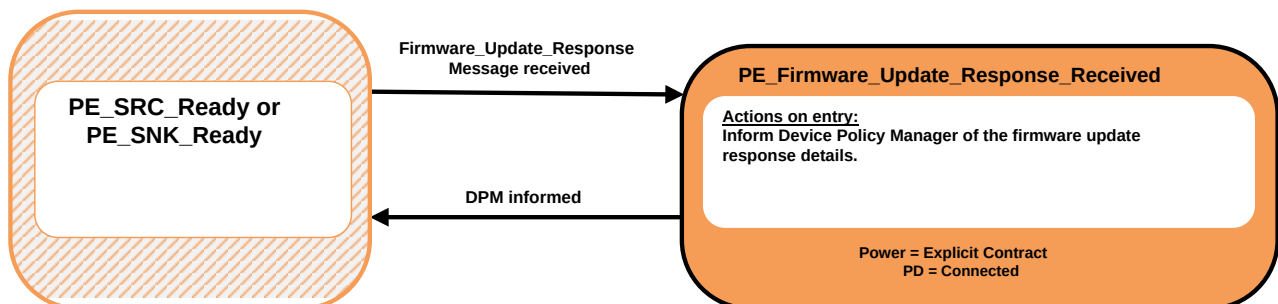
On entry to the PE_Send_Firmware_Update_Response State the Policy Engine **Shall** Request the appropriate response from the Device Policy Manager and then send a [Firmware_Update_Response Message](#) based on this status.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State as appropriate (see [Figure 9.17](#) , [Figure 9.18](#) and [Figure 9.88](#)) when:

- The [Firmware_Update_Response Message](#) has been successfully sent.

9.2.19.3. Firmware Update Response Received State Diagram

[Figure 9.58](#) shows the State diagram for a Source or Sink on receiving a [Firmware_Update_Response Message](#). See also [Section 6.5.12](#).

Figure 9.58. Firmware update response received State Diagram

9.2.19.3.1. PE_Firmware_Update_Response_Received State

The Policy Engine **Shall** transition to the PE_Firmware_Update_Response_Received State, from either the PE_SRC_Ready or PE_SNK_Ready when a [Firmware_Update_Response Message](#) is received.

On entry to the PE_Firmware_Update_Response_Received State the Policy Engine **Shall** inform the Device Policy Manager of the details of the firmware update response.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready or PE_SNK_Ready State as appropriate (see [Figure 9.17](#) , [Figure 9.18](#) and [Figure 9.88](#)) when:

- The Device Policy Manager has been informed.

9.2.20. Dual-Role Port State Diagrams

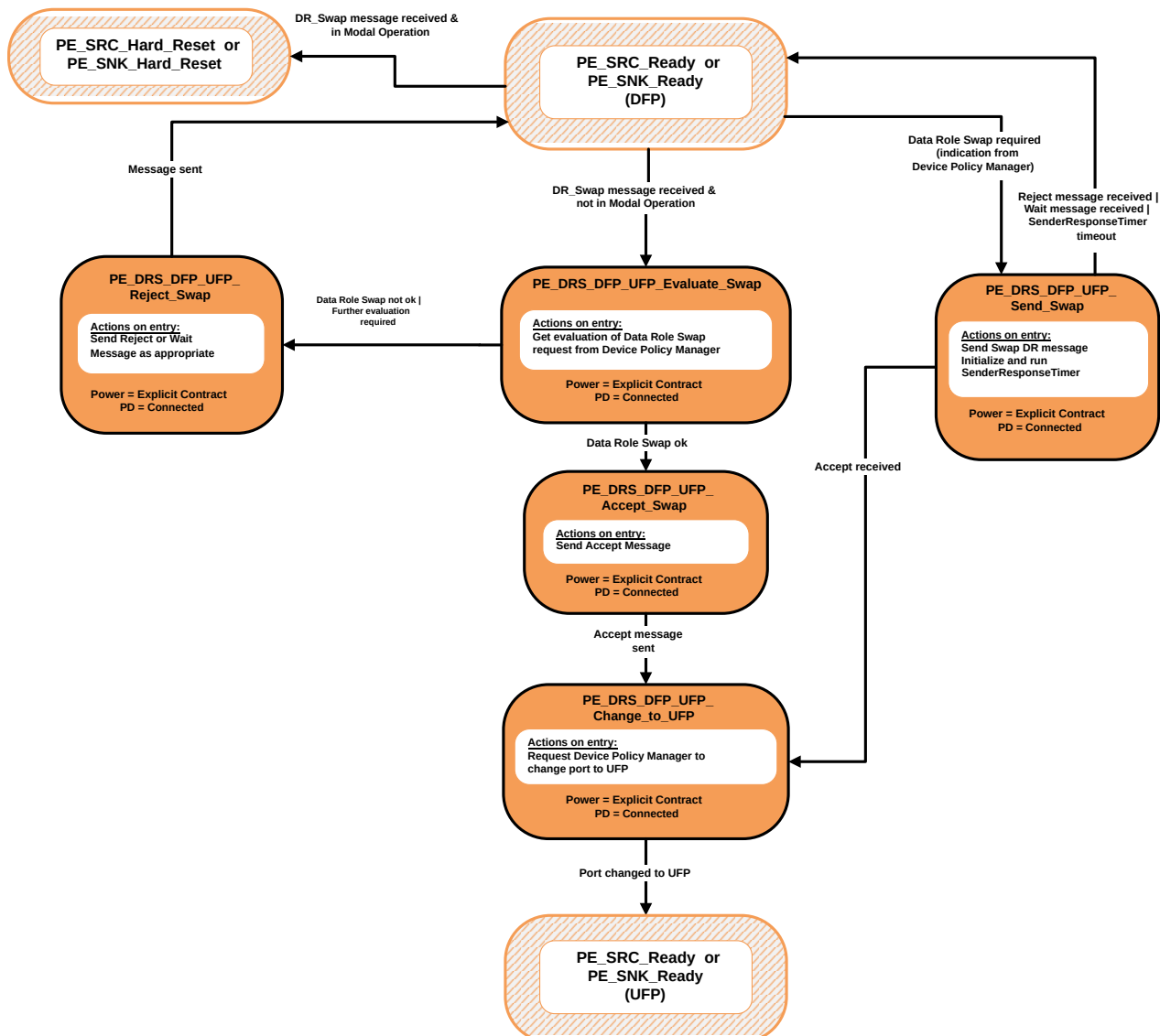
Dual-Role Ports that combine Source and Sink functionality **shall** comprise Source and Sink Policy Engine State machines. In addition they **shall** have the capability to perform a [Power Role Swap](#) from the PE_SRC_Ready or PE_SNK_Ready states and **shall** return to USB Default Operation on a [Hard Reset](#).

The State Diagrams in this section **shall** apply to every [USB-C] DRP.

9.2.20.1. DFP to UFP Data Role Swap State Diagram

[Figure 9.59](#) shows the additional State diagram required to perform a [Data Role Swap](#) from DFP to UFP operation and the changes that **shall** be followed for error and [Hard Reset](#) handling.

Figure 9.59. DFP to UFP Data Role Swap State Diagram



9.2.20.1.1. PE_SRC_Ready or PE_SNK_Ready State

The [Data Role Swap](#) process **Shall** start only from either the PE_SRC_Ready or PE_SNK_Ready State where power is stable.

The Policy Engine **Shall** transition to the PE_DRS_DFP_UFP_Evaluate_Swap State when:

- A [DR_Swap Message](#) is received and
- There are no Active Modes (not in Modal Operation).

The Policy Engine **Shall** transition to either the PE_SRC_Hard_Reset or PE_SNK_Hard_Reset states when:

- A [DR_Swap Message](#) is received and
- There are one or more Active Modes (Modal Operation).

The Policy Engine **Shall** transition to the PE_DRS_DFP_UFP_Send_Swap State when:

- The Device Policy Manager indicates that a [Data Role Swap](#) is required.

9.2.20.1.2. PE_DRS_DFP_UFP_Evaluate_Swap State

On entry to the PE_DRS_DFP_UFP_Evaluate_Swap State the Policy Engine **Shall** ask the Device Policy Manager whether a [Data Role Swap](#) can be made.

The Policy Engine **Shall** transition to the PE_DRS_DFP_UFP_Accept_Swap State when:

- The Device Policy Manager indicates that a [Data Role Swap](#) is OK.

The Policy Engine **Shall** transition to the PE_DRS_DFP_UFP_Reject_Swap State when:

- The Device Policy Manager indicates that a [Data Role Swap](#) is not OK.
- Or further evaluation of the [Data Role Swap](#) Request is needed.

9.2.20.1.3. PE_DRS_DFP_UFP_Accept_Swap State

On entry to the PE_DRS_DFP_UFP_Accept_Swap State the Policy Engine **Shall** Request the Protocol Layer to send an [Accept Message](#).

The Policy Engine **Shall** transition to the PE_DRS_DFP_UFP_Change_to_UFP State when:

- The [Accept Message](#) has been sent.

9.2.20.1.4. PE_DRS_DFP_UFP_Change_to_UFP State

On entry to the PE_DRS_DFP_UFP_Change_to_UFP State the Policy Engine **Shall** Request the Device Policy Manager to change the Port from a DFP to a UFP.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

- The Device Policy Manager indicates that the Port has been changed to a UFP.

9.2.20.1.5. PE_DRS_DFP_UFP_Send_Swap State

On entry to the PE_DRS_DFP_UFP_Send_Swap State the Policy Engine **Shall** Request the Protocol Layer to send a [DR_Swap Message](#) and **Shall** start the SenderResponseTimer.

On exit from the PE_DRS_DFP_UFP_Send_Swap State the Policy Engine **Shall** stop the SenderResponseTimer.

The Policy Engine **Shall** continue as a DFP and **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

- A [Reject Message](#) is received.
- Or a [Wait Message](#) is received.
- Or the SenderResponseTimer times out.

The Policy Engine **Shall** transition to the PE_DRS_DFP_UFP_Change_to_UFP State when:

- An [Accept Message](#) is received.

9.2.20.1.6. PE_DRS_DFP_UFP_Reject_Swap State

On entry to the PE_DRS_DFP_UFP_Reject_Swap State the Policy Engine **Shall** Request the Protocol Layer to send:

- A [Reject Message](#) if the Device is unable to perform a [Data Role Swap](#) at this time.
- A [Wait Message](#) if further evaluation of the [Data Role Swap](#) Request is required.

Note: In this case it is expected that one of the Port Partners will send a [DR_Swap Message](#) at a later time (see [Section 7.12.2](#)).

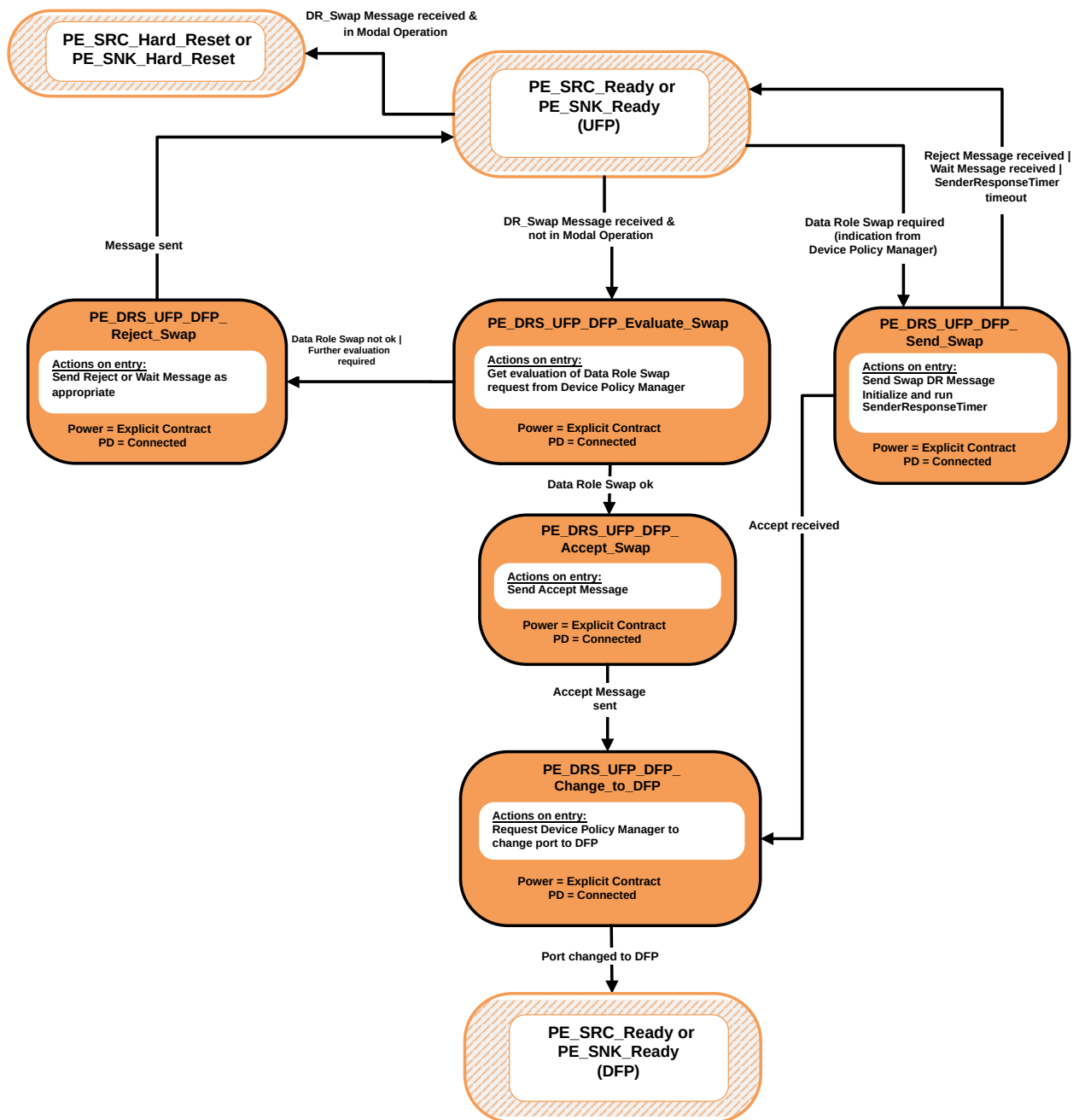
The Policy Engine **Shall** continue as a DFP and **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

- The [Reject](#) or [Wait Message](#) has been sent.

9.2.20.2. UFP to DFP Data Role Swap State Diagram

[Figure 9.60](#) shows the additional State diagram required to perform a [Data Role Swap](#) from DRP UFP to DFP operation and the changes that **Shall** be followed for error and Hard Reset handling.

Figure 9.60. UFP to DFP Data Role Swap State Diagram



9.2.20.2.1. PE_SRC_Ready or PE_SNK_Ready State

The [Data Role Swap](#) process **shall** start only from the either the PE_SRC_Ready or PE_SNK_Ready State where power is stable.

The Policy Engine **shall** transition to the PE_DRS_UFP_DFP_Evaluate_Swap State when:

- A [DR_Swap Message](#) is received and
- There are no Active Modes (not in Modal Operation).

The Policy Engine **Shall** transition to either the PE_SRC_Hard_Reset or PE_SNK_Hard_Reset states when:

- A [DR_Swap Message](#) is received and
- There are one or more Active Modes (Modal Operation).

The Policy Engine **Shall** transition to the PE_DRS_UFP_DFP_Send_Swap State when:

- The Device Policy Manager indicates that a [Data Role Swap](#) is required.

9.2.20.2.2. PE_DRS_UFP_DFP_Evaluate_Swap State

On entry to the PE_DRS_UFP_DFP_Evaluate_Swap State the Policy Engine **Shall** ask the Device Policy Manager whether a [Data Role Swap](#) can be made.

The Policy Engine **Shall** transition to the PE_DRS_UFP_DFP_Accept_Swap State when:

- The Device Policy Manager indicates that a [Data Role Swap](#) is OK.

The Policy Engine **Shall** transition to the PE_DRS_UFP_DFP_Reject_Swap State when:

- The Device Policy Manager indicates that a [Data Role Swap](#) is not OK.
- Or further evaluation of the [Data Role Swap](#) Request is needed.

9.2.20.2.3. PE_DRS_UFP_DFP_Accept_Swap State

On entry to the PE_DRS_UFP_DFP_Accept_Swap State the Policy Engine **Shall** Request the Protocol Layer to send an [Accept Message](#).

The Policy Engine **Shall** transition to the PE_DRS_UFP_DFP_Change_to_DFP State when:

- The [Accept Message](#) has been sent.

9.2.20.2.4. PE_DRS_UFP_DFP_Change_to_DFP State

On entry to the PE_DRS_UFP_DFP_Change_to_DFP State the Policy Engine **Shall** Request the Device Policy Manager to change the Port from a UFP to a DFP.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

- The Device Policy Manager indicates that the Port has been changed to a DFP.

9.2.20.2.5. PE_DRS_UFP_DFP_Send_Swap State

On entry to the PE_DRS_UFP_DFP_Send_Swap State the Policy Engine **Shall** Request the Protocol Layer to send a [DR_Swap Message](#) and **Shall** start the SenderResponseTimer.

On exit from the PE_DRS_UFP_DFP_Send_Swap State the Policy Engine **Shall** stop the SenderResponseTimer.

The Policy Engine **Shall** continue as a UFP and **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

- A [Reject Message](#) is received.
- Or a [Wait Message](#) is received.

- Or the `SenderResponseTimer` times out.

The Policy Engine **Shall** transition to the `PE_DRS_UFP_DFP_Change_to_DFP` State when:

- An [Accept Message](#) is received.

9.2.20.2.6. PE_DRS_UFP_DFP_Reject_Swap State

On entry to the `PE_DRS_UFP_DFP_Reject_Swap` State the Policy Engine **Shall** Request the Protocol Layer to send:

- A [Reject Message](#) if the Device is unable to perform a [Data Role Swap](#) at this time.
- A [Wait Message](#) if further evaluation of the [Data Role Swap](#) Request is required.

Note: In this case it is expected that one of the Port Partners will send a [DR_Swap Message](#) at a later time (see [Section 7.12.2](#)).

The Policy Engine **Shall** continue as a UFP and **Shall** transition to the either the `PE_SRC_Ready` or `PE_SNK_Ready` State when:

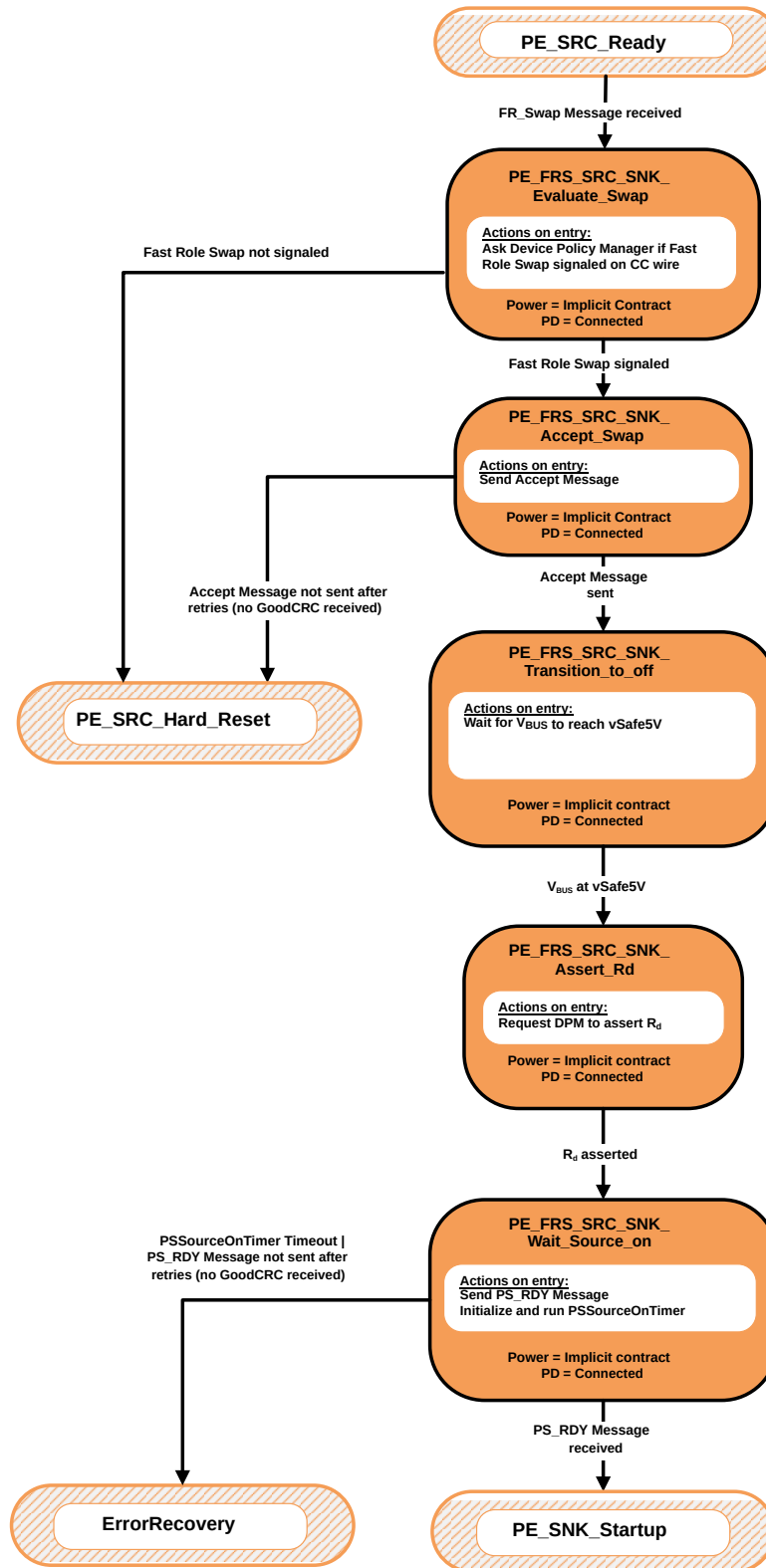
- The [Reject](#) or [Wait Message](#) has been sent.

9.2.20.3. Policy Engine in Source to Sink Power Role Swap State Diagram

Dual-Role Ports that combine Source and Sink functionality **Shall** comprise Source and Sink Policy Engine State machines. In addition, they **Shall** have the capability to do a [Power Role Swap](#) from the `PE_SRC_Ready` State and **Shall** return to USB Default Operation on a [Hard Reset](#).

[Figure 9.61](#) shows the additional State diagram required to perform a [Power Role Swap](#) from Source to Sink Power Roles and the changes that **Shall** be followed for error handling.

Figure 9.61. Dual-Role Port in Source to Sink Power Role Swap State Diagram



9.2.20.3.1. PE_SRC_Ready State

The [Power Role Swap](#) process **Shall** start only from the PE_SRC_Ready State where power is stable. The Policy Engine **Shall** transition to the PE_PRS_SRC_SNK_Evaluate_Swap State when:

- A [PR_Swap Message](#) is received.

The Policy Engine **Shall** transition to the PE_PRS_SRC_SNK_Send_Swap State when:

- The Device Policy Manager indicates that a [Power Role Swap](#) is required.

9.2.20.3.2. PE_PRS_SRC_SNK_Evaluate_Swap State

On entry to the PE_PRS_SRC_SNK_Evaluate_Swap State the Policy Engine **Shall** ask the Device Policy Manager whether a [Power Role Swap](#) can be made.

The Policy Engine **Shall** transition to the PE_PRS_SRC_SNK_Accept_Swap State when:

- The Device Policy Manager indicates that a [Power Role Swap](#) is OK.

The Policy Engine **Shall** transition to the PE_PRS_SRC_SNK_Reject_Swap State when:

- The Device Policy Manager indicates that a [Power Role Swap](#) is not OK.
- Or further evaluation of the [Power Role Swap](#) Request is needed.

9.2.20.3.3. PE_PRS_SRC_SNK_Accept_Swap State

On entry to the PE_PRS_SRC_SNK_Accept_Swap State the Policy Engine **Shall** Request the Protocol Layer to send an [Accept Message](#).

The Policy Engine **Shall** transition to the PE_PRS_SRC_SNK_Transition_to_off State when:

- The [Accept Message](#) has been sent.

9.2.20.3.4. PE_PRS_SRC_SNK_Transition_to_off State

On entry to the PE_PRS_SRC_SNK_Transition_to_off State the Policy Engine **Shall** Request the Device Policy Manager to turn off the Source.

The Policy Engine **Shall** transition to the PE_PRS_SRC_SNK_Assert_Rd State when:

- The Device Policy Manager indicates that the Source has been turned off.

9.2.20.3.5. PE_PRS_SRC_SNK_Assert_Rd State

On entry to the PE_PRS_SRC_SNK_Assert_Rd State the Policy Engine **Shall** Request the Device Policy Manager to change the resistor asserted on the CC wire from Rp to Rd.

The Policy Engine **Shall** transition to the PE_PRS_SRC_SNK_Wait_Source_on State when:

- The Device Policy Manager indicates that Rd is asserted.

9.2.20.3.6. PE_PRS_SRC_SNK_Wait_Source_on State

On entry to the PE_PRS_SRC_SNK_Wait_Source_on State the Policy Engine **Shall** Request the Protocol Layer to send a [PS_RDY Message](#) and **Shall** start the PSSourceOnTimer.

On exit from the Source off State the Policy Engine **Shall** stop the PSSourceOnTimer. The Policy Engine **Shall** transition to the PE_SNK_Startup when:

- A [PS_RDY Message](#) is received indicating that the remote Source is now supplying power. The Policy Engine **Shall** transition to the ErrorRecovery State when:
 - The PSSourceOnTimer times out or
 - The [PS_RDY Message](#) is not sent after retries (a [GoodCRC Message](#) has not been received).

Note: A [Soft Reset](#) **Shall Not** be initiated in this case.

9.2.20.3.7. PE_PRS_SRC_SNK_Send_Swap State

- On entry to the PE_PRS_SRC_SNK_Send_Swap State the Policy Engine **Shall** Request the Protocol Layer to send a [PR_Swap Message](#) and **Shall** start the SenderResponseTimer.

On exit from the PE_PRS_SRC_SNK_Send_Swap State the Policy Engine **Shall** stop the SenderResponseTimer. The Policy Engine **Shall** transition to the PE_SRC_Ready State when:

- A [Reject Message](#) is received.
- Or a [Wait Message](#) is received.
- Or the SenderResponseTimer times out.

The Policy Engine **Shall** transition to the PE_PRS_SRC_SNK_Transition_to_off State when:

- An [Accept Message](#) is received.

9.2.20.3.8. PE_PRS_SRC_SNK_Reject_Swap State

- On entry to the PE_PRS_SRC_SNK_Reject_Swap State the Policy Engine **Shall** Request the Protocol Layer to send:
 - A [Reject Message](#) if the Device is unable to perform a Power Role Swap at this time.
 - A [Wait Message](#) if further evaluation of the [Power Role Swap](#) Request is required.

Note: In this case it is expected that one of the Port Partners will send a [PR_Swap Message](#) at a later time (see [Section 7.10.1](#)).

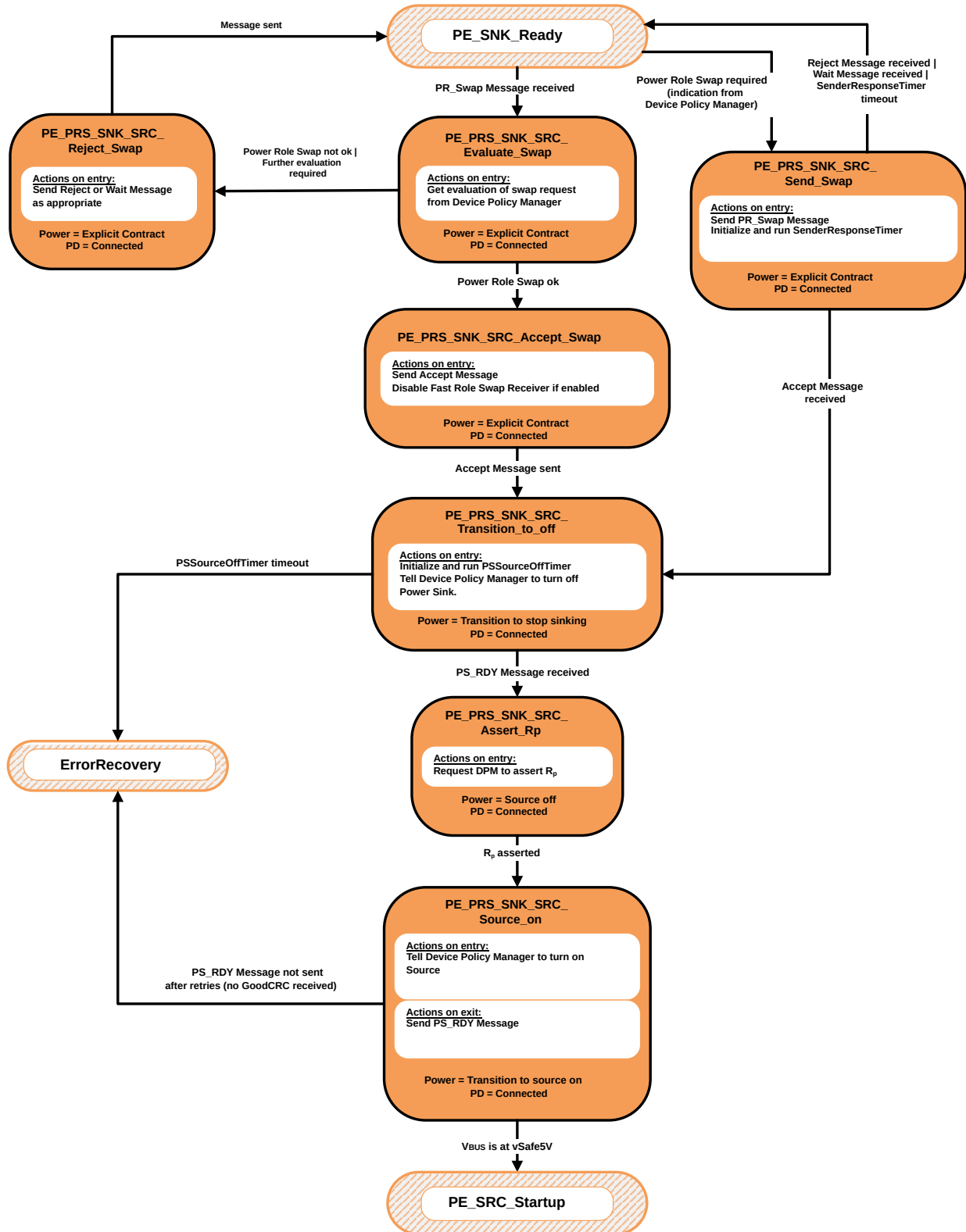
The Policy Engine **Shall** transition to the PE_SRC_Ready when:

- The [Reject](#) or [Wait Message](#) has been sent.

9.2.20.4. Policy Engine in Sink to Source Power Role Swap State Diagram

Dual-Role Ports that combine Sink and Source functionality **Shall** comprise Sink and Source Policy Engine State machines. In addition, they **Shall** have the capability to do a [Power Role Swap](#) from the PE_SNK_Ready State and **Shall** return to USB Default Operation on a [Hard Reset](#).

[Figure 9.62](#) shows the additional State diagram required to perform a [Power Role Swap](#) from Sink to Source Power Roles and the changes that **Shall** be followed for error handling.

Figure 9.62. Dual-role Port in Sink to Source Power Role Swap State Diagram

9.2.20.4.1. PE_SNK_Ready State

The [Power Role Swap](#) process **Shall** start only from the PE_SNK_Ready State where power is stable. The Policy Engine **Shall** transition to the PE_PRS_SNK_SRC_Evaluate_Swap State when:

- A [PR_Swap Message](#) is received.

The Policy Engine **Shall** transition to the PE_PRS_SNK_SRC_Send_Swap State when:

- The Device Policy Manager indicates that a [Power Role Swap](#) is required.

9.2.20.4.2. PE_PRS_SNK_SRC_Evaluate_Swap State

On entry to the PE_PRS_SNK_SRC_Send_Swap State the Policy Engine **Shall** ask the Device Policy Manager whether a [Power Role Swap](#) can be made.

The Policy Engine **Shall** transition to the PE_PRS_SNK_SRC_Accept_Swap State when:

- The Device Policy Manager indicates that a [Power Role Swap](#) is OK.

The Policy Engine **Shall** transition to the PE_PRS_SNK_SRC_Reject_Swap State when:

- The Device Policy Manager indicates that a [Power Role Swap](#) is not OK.

9.2.20.4.3. PE_PRS_SNK_SRC_Accept_Swap State

On entry to the PE_PRS_SNK_SRC_Accept_Swap State the Policy Engine **Shall** Request the Protocol Layer to send an

[Accept Message](#) and **Shall** disable the [Fast Role Swap](#) receiver if this is enabled.

The Policy Engine **Shall** transition to the PE_PRS_SNK_SRC_Transition_to_off State when:

- The [Accept Message](#) has been sent.

9.2.20.4.4. PE_PRS_SNK_SRC_Transition_to_off State

On entry to the PE_PRS_SNK_SRC_Transition_to_off State the Policy Engine **Shall** initialize and run the PSSourceOffTimer and then Request the Device Policy Manager to turn off the Sink. The Policy Engine **Shall** transition to the ErrorRecovery State when:

- The PSSourceOffTimer times out.

The Policy Engine **Shall** transition to the PE_PRS_SNK_SRC_Assert_Rp State when:

- A [PS_RDY Message](#) is received.

9.2.20.4.5. PE_PRS_SNK_SRC_Assert_Rp State

On entry to the PE_PRS_SNK_SRC_Assert_Rp State the Policy Engine **Shall** Request the Device Policy Manager to change the resistor asserted on the CC wire from Rd to Rp.

The Policy Engine **Shall** transition to the PE_PRS_SNK_SRC_Source_on State when:

- The Device Policy Manager indicates that Rd is asserted.

9.2.20.4.6. PE_PRS_SNK_SRC_Source_on State

On entry to the PE_PRS_SNK_SRC_Source_on State the Policy Engine **Shall** Request the Device Policy Manager to turn on the Source.

On exit from the PE_PRS_SNK_SRC_Source_on State the Policy Engine **Shall** send a [PS_RDY Message](#). The Policy Engine **Shall** transition to the PE_SRC_Startup State when:

- The Source Port VBUS is at [vSafe5V](#).

The Policy Engine **Shall** transition to the ErrorRecovery State when:

- The [PS_RDY Message](#) is not sent after retries (a [GoodCRC Message](#) has not been received). A [Soft Reset](#) **Shall Not** be initiated in this case.

9.2.20.4.7. PE_PRS_SNK_SRC_Send_Swap State

On entry to the PE_PRS_SNK_SRC_Send_Swap State the Policy Engine **Shall** Request the Protocol Layer to send a [PR_Swap Message](#) and **Shall** initialize and run the SenderResponseTimer. The Policy Engine **Shall** transition to the PE_SNK_Ready State when:

- A [Reject Message](#) is received.
- Or a [Wait Message](#) is received.
- Or the SenderResponseTimer times out.

The Policy Engine **Shall** transition to the PE_PRS_SNK_SRC_Transition_to_off State when:

- An [Accept Message](#) is received.

9.2.20.4.8. PE_PRS_SNK_SRC_Reject_Swap State

On entry to the PE_PRS_SNK_SRC_Reject_Swap State the Policy Engine **Shall** Request the Protocol Layer to send:

- A [Reject Message](#) if the Device is unable to perform a [Power Role Swap](#) at this time.
- A [Wait Message](#) if further evaluation of the [Power Role Swap](#) Request is required.

Note: In this case it is expected that one of the Port Partners will send a [PR_Swap Message](#) at a later time (see [Section 7.10.1](#)). The Policy Engine **Shall** transition to the PE_SNK_Ready State when:

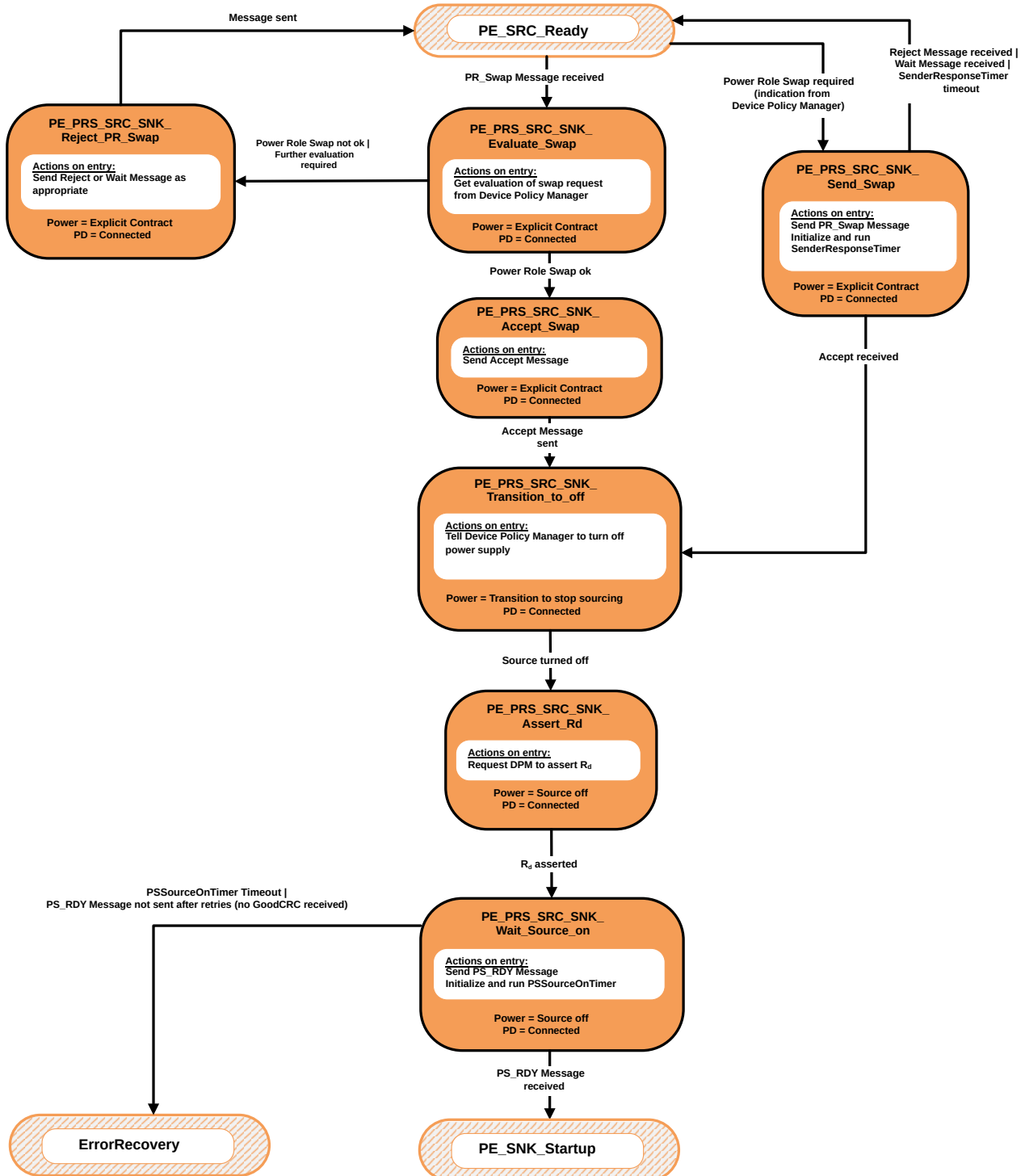
- The [Reject](#) or [Wait Message](#) has been sent.

9.2.20.5. Policy Engine in Source to Sink Fast Role Swap State Diagram

Dual-Role Ports that combine Source and Sink functionality **Shall** comprise Source and Sink Policy Engine State machines. In addition, they **Shall** have the capability to do a [Power Role Swap](#) from the PE_SRC_Ready State and **Shall** return to USB Default Operation on a [Hard Reset](#).

[Figure 9.63](#) shows the additional State diagram required to perform a [Power Role Swap](#) from Source to Sink Power Roles and the changes that **Shall** be followed for error handling.

Figure 9.63. Dual-Role Port in Source to Sink Power Role Swap State Diagram



9.2.20.5.1. PE_SRC_Ready State

The [Fast Role Swap](#) process **Shall** start only from the PE_SRC_Ready State where power is stable. The Policy Engine **Shall** transition to the PE_FRS_SRC_SNK_Evaluate_Swap State when:

- An [FR_Swap Message](#) is received.

9.2.20.5.2. PE_FRS_SRC_SNK_Evaluate_Swap State

On entry to the PE_FRS_SRC_SNK_Evaluate_Swap State the Policy Engine **Shall** ask the Device Policy Manager whether [Fast Role Swap](#) has been signaled on the CC wire.

The Policy Engine **Shall** transition to the PE_FRS_SRC_SNK_Accept_Swap State when:

- The Device Policy Manager indicates that a [Fast Role Swap](#) has been signaled. The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:
 - The Device Policy Manager indicates that a [Fast Role Swap](#) is not being signaled.

9.2.20.5.3. PE_FRS_SRC_SNK_Accept_Swap State

On entry to the PE_FRS_SRC_SNK_Accept_Swap State the Policy Engine **Shall** Request the Protocol Layer to send an [Accept Message](#).

The Policy Engine **Shall** transition to the PE_FRS_SRC_SNK_Transition_to_off State when:

- The [Accept Message](#) has been sent.

The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:

- The [Accept Message](#) is not sent after retries (a [GoodCRC Message](#) has not been received).

Note: A [Soft Reset](#) **Shall Not** be initiated in this case.

9.2.20.5.4. PE_FRS_SRC_SNK_Transition_to_off State

On entry to the PE_FRS_SRC_SNK_Transition_to_off State the Policy Engine **Shall** wait until VBUS has discharged to [vSafe5V](#).

The Policy Engine **Shall** transition to the PE_FRS_SRC_SNK_Assert_Rd State when:

- The Device Policy Manager indicates that VBUS has discharged to [vSafe5V](#).

9.2.20.5.5. PE_FRS_SRC_SNK_Assert_Rd State

On entry to the PE_FRS_SRC_SNK_Assert_Rd State the Policy Engine **Shall** Request the Device Policy Manager to change the resistor asserted on the CC wire from Rp to Rd.

The Policy Engine **Shall** transition to the PE_FRS_SRC_SNK_Wait_Source_on State when:

- The Device Policy Manager indicates that Rd is asserted.

9.2.20.5.6. PE_FRS_SRC_SNK_Wait_Source_on State

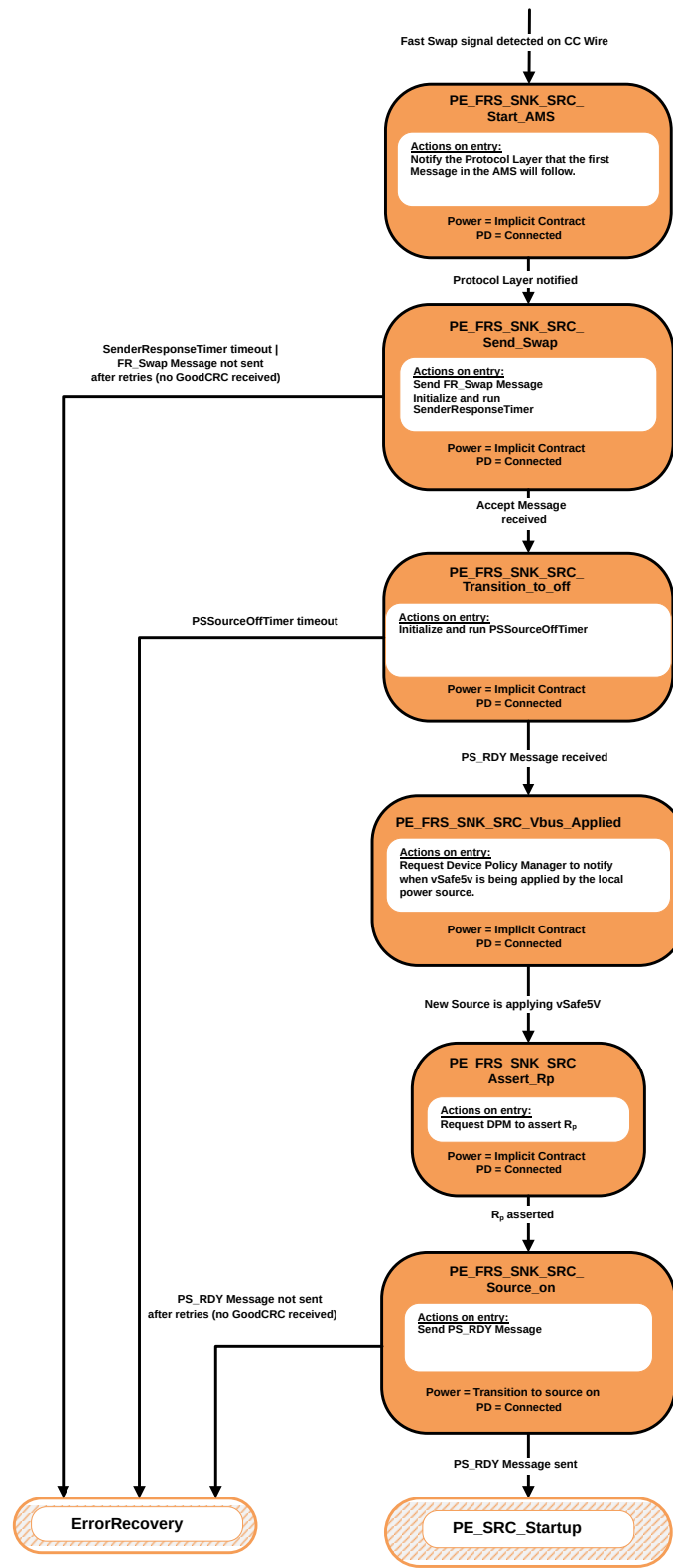
- A [PS_RDY Message](#) is received indicating that the New Source is now applying Rp. The Policy Engine **Shall** transition to the ErrorRecovery State when:
 - The PSSourceOnTimer times out or
 - The [PS_RDY Message](#) is not sent after retries (a [GoodCRC Message](#) has not been received).

Note: A [Soft Reset](#) **Shall Not** be initiated in this case.

9.2.20.6. Policy Engine in Sink to Source Fast Role Swap State Diagram

Dual-Role Ports that combine Sink and Source functionality **Shall** comprise Sink and Source Policy Engine State machines. In addition, they **Should** have the capability to do a [Fast Role Swap](#) from the PE_SNK_Ready State and **Shall** return to USB Default Operation on a [Hard Reset](#).

[Figure 9.64](#) shows the additional State diagram required to perform a [Fast Role Swap](#) from Sink to Source Power Roles and the changes that **Shall** be followed for error handling.

Figure 9.64. Dual-role Port in Sink to Source Fast Role Swap State Diagram

9.2.20.6.1. PE_FRS_SNK_SRC_Start_AMS State

The Policy Engine **Shall** transition to the PE_FRS_SNK_SRC_Start_AMS State from any other State provided there is an Explicit Contract in place when:

- The Sink Capabilities received from the Initial Source by the Policy Engine has at least one of the [Fast Role Swap](#) bits set.
- The system has sufficient reserve power to provide the requested current to the Initial Source, as requested in the [Fast Role Swap](#) bits in the Sink Capabilities, and is willing to dedicate it to the Port
- The Device Policy Manager indicates that a [Fast Role Swap](#) signal has been detected on the CC wire.

On entry to the PE_FRS_SNK_SRC_Start_AMS State the Policy Engine **Shall** notify the Protocol Layer that the first Message in an AMS will follow.

The Policy Engine **Shall** transition to the PE_FRS_SNK_SRC_Send_Swap State when:

- The Protocol Layer has been notified.

9.2.20.6.2. PE_FRS_SNK_SRC_Send_Swap State

On entry to the PE_FRS_SNK_SRC_Send_Swap State the Policy Engine **Shall** Request the Protocol Layer to send an [FR_Swap Message](#) and **Shall** initialize and run the SenderResponseTimer.

The Policy Engine **Shall** transition to the PE_FRS_SNK_SRC_Transition_to_off State when:

- An [Accept Message](#) is received.

The Policy Engine **Shall** transition to the ErrorRecovery State when:

- The SenderResponseTimer times out or
- The [FR_Swap Message](#) is not sent after retries (a [GoodCRC Message](#) has not been received). A [Soft Reset](#) **Shall Not** be initiated in this case.

9.2.20.6.3. PE_FRS_SNK_SRC_Transition_to_off State

On entry to the PE_FRS_SNK_SRC_Transition_to_off State the Policy Engine **Shall** initialize and run the PSSourceOffTimer and then Request the Device Policy Manager to turn off the Sink. The Policy Engine **Shall** transition to the ErrorRecovery State when:

- The PSSourceOffTimer times out.

The Policy Engine **Shall** transition to the PE_FRS_SNK_SRC_VBUS_Applied State when:

- A [PS_RDY Message](#) is received.

9.2.20.6.4. PE_FRS_SNK_SRC_VBUS_Applied State

On entry to the PE_FRS_SNK_SRC_VBUS_Applied State the Policy Engine waits for a notification from the Device Policy Manager that the local power Source has applied [vSafe5V](#) to VBUS (see [Section 10.3](#)).

Note: This could have already been applied prior to entering this State or could be applied while waiting in this State.

The Policy Engine **Shall** transition to the PE_FRS_SNK_SRC_Assert_Rp State when:

- The Device Policy Manager indicates that [vSafe5V](#) is being applied.

9.2.20.6.5. PE_FRS_SNK_SRC_Assert_Rp State

On entry to the PE_FRS_SNK_SRC_Assert_Rp State the Policy Engine **Shall** Request the Device Policy Manager to change the resistor asserted on the CC wire from Rd to Rp .

The Policy Engine **Shall** transition to the PE_FRS_SNK_SRC_Source_on State when:

- The Device Policy Manager indicates that Rp is asserted.

9.2.20.6.6. PE_FRS_SNK_SRC_Source_on State

On entry to the PE_FRS_SNK_SRC_Source_on State the Policy Engine **Shall** Request the Device Policy Manager to turn on the Source.

On exit from the PE_FRS_SNK_SRC_Source_on State the Policy Engine **Shall** send a [PS_RDY Message](#). The Policy Engine **Shall** transition to the PE_SRC_Startup State when:

- The [PS_RDY Message](#) has been sent.

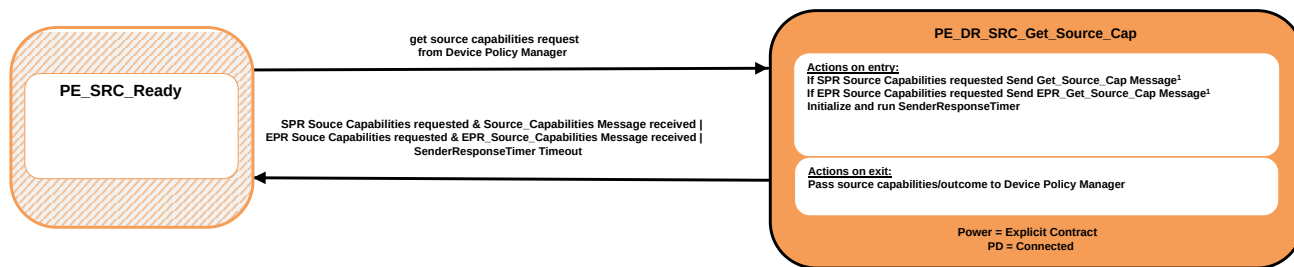
The Policy Engine **Shall** transition to the ErrorRecovery State when:

- The [PS_RDY Message](#) is not sent after retries (a [GoodCRC Message](#) has not been received). A [Soft Reset](#) **Shall Not** be initiated in this case.

9.2.20.7. Dual-Role (Source Port) Get Source Capabilities State Diagram

[Figure 9.65](#) shows the State diagram for a Dual-Role Device, presently operating as a Source, on receiving a Request from the Device Policy Manager to get the Port Partner's Source Capabilities. See also [Section 6.4.1.1](#).

Figure 9.65. Dual-Role (Source) Get Source Capabilities diagram



1. Either SPR or EPR Source Capabilities **May** be requested, regardless of whether or not the Source is currently operating in SPR or [EPR Mode](#)..

9.2.20.7.1. PE_DR_SRC_Get_Source_Cap State

The Policy Engine **Shall** transition to the PE_DR_SRC_Get_Source_Cap State, from the PE_SRC_Ready State, due to a Request to get the remote Source Capabilities from the Device Policy Manager.

- On entry to the PE_DR_SRC_Get_Source_Cap State the Policy Engine **Shall** Request the Protocol Layer to send a get Source Capabilities Message in order to retrieve the Source Capabilities. The Policy Engine **Shall** send:

- A [Get_Source_Cap Message](#) when the Device Policy Manager requests SPR Capabilities or
- An [EPR_Get_Source_Cap Message](#) when the Device Policy Manager requests EPR Capabilities. The Policy Engine **Shall** then start the SenderResponseTimer.

On exit from the PE_DR_SRC_Get_Source_Cap State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout).

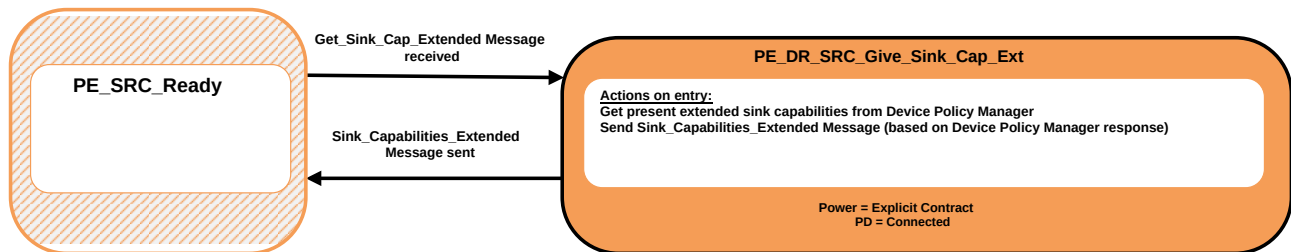
The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#)) when:

- In SPR Mode and SPR Source Capabilities were requested and a [Source_Capabilities Message](#) is received or
- In [EPR Mode](#) and EPR Source Capabilities were requested and an [EPR_Source_Capabilities Message](#) is received or
- The SenderResponseTimer times out.

9.2.20.8. Dual-Role (Source Port) Give Sink Capabilities State Diagram

[Figure 9.66](#) shows the State diagram for a Dual-Role Device, presently operating as a Source, on receiving a [Get_Sink_Cap Message](#). See also [Section 6.3.8](#).

Figure 9.66. Dual-Role (Source) Give Sink Capabilities diagram



9.2.20.8.1. PE_DR_SRC_Give_Sink_Cap State

The Policy Engine **Shall** transition to the PE_DR_SRC_Give_Sink_Cap State, from the PE_SRC_Ready State, when a [Get_Sink_Cap Message](#) or [EPR_Get_Sink_Cap Message](#) is received.

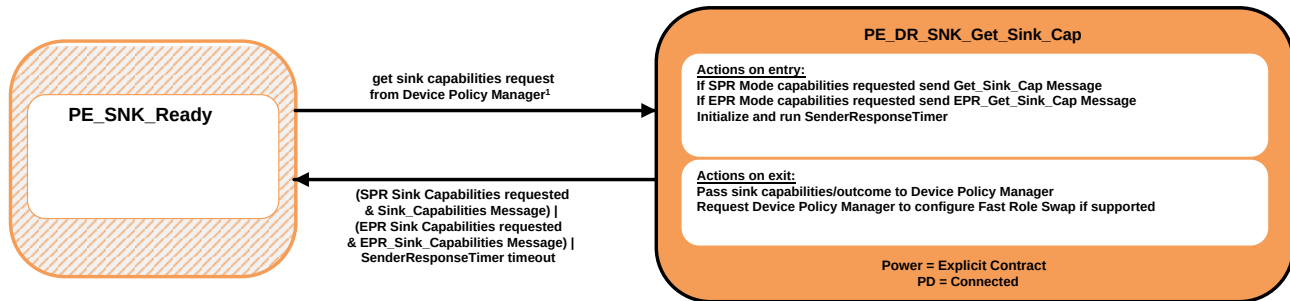
- On entry to the PE_DR_SRC_Give_Sink_Cap State the Policy Engine **Shall** Request the Device Policy Manager for the current system Capabilities. The Policy Engine **Shall** then Request the Protocol Layer to send a [Sink_Capabilities Message](#) containing these Capabilities. The Policy Engine **Shall** send:
- A [Sink_Capabilities Message](#) when a [Get_Sink_Cap Message](#) is received or
- An [EPR_Sink_Capabilities Message](#) when a [EPR_Get_Sink_Cap Message](#) is received.

The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#)) when:

- The [Sink_Capabilities Message](#) has been successfully sent.

9.2.20.9. Dual-Role (Sink Port) Get Sink Capabilities State Diagram

[Figure 9.67](#) shows the State diagram for a Dual-Role Device, presently operating as a Sink, on receiving a Request from the Device Policy Manager to get the Port Partner's Sink Capabilities. See also [Section 6.4.1.2](#).

Figure 9.67. Dual-Role (Sink) Get Sink Capabilities State Diagram

1. Either SPR or EPR Sink Capabilities **May** be requested, regardless of whether or not the Sink is currently operating in SPR or [EPR Mode](#)..

9.2.20.9.1. PE_DR_SNK_Get_Sink_Cap State

The Policy Engine **Shall** transition to the PE_DR_SNK_Get_Sink_Cap State, from the PE_SNK_Ready State, due to a Request to get the remote Source Capabilities from the Device Policy Manager.

- On entry to the PE_DR_SNK_Get_Sink_Cap State the Policy Engine **Shall** Request the Protocol Layer to send a [Get_Sink_Cap Message](#) in order to retrieve the Sink Capabilities. The Policy Engine **Shall** send:
- A [Get_Sink_Cap Message](#) when the Device Policy Manager requests SPR Capabilities or
- An [EPR_Get_Sink_Cap Message](#) when the Device Policy Manager requests EPR Capabilities. The Policy Engine **Shall** then start the SenderResponseTimer.

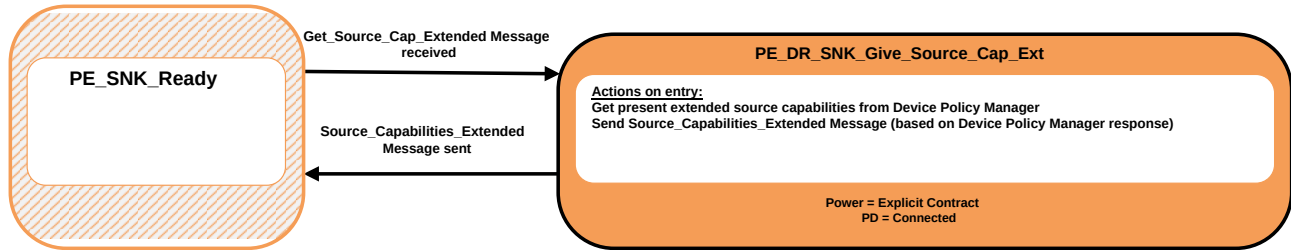
On exit from the PE_SRC_Get_Sink_Cap State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout). If [Fast Role Swap](#) is supported, Request Device Policy Manager prepare or disable 5V Source and configure the [Fast Role Swap](#) receiver based on the [Fast Role Swap](#) required USB Type- C Current bits in the received Sink Capabilities.

The Policy Engine **Shall** transition to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

- SPR Sink Capabilities were requested and a [Sink_Capabilities](#) Message is received or
- EPR Sink Capabilities were requested and an [EPR_Sink_Capabilities](#) Message is received or
- The SenderResponseTimer times out.

9.2.20.10. Dual-Role (Sink Port) Give Source Capabilities State Diagram

[Figure 9.68](#) shows the State diagram for a Dual-Role Device, presently operating as a Sink, on receiving a [Get_Source_Cap Message](#). See also [Section 6.3.7](#).

Figure 9.68. Dual-Role (Sink) Give Source Capabilities State Diagram**9.2.20.10.1. PE_DR_SNK_Give_Source_Cap State**

The Policy Engine **Shall** transition to the PE_DR_SNK_Give_Source_Cap State, from the PE_SNK_Ready State, when a [Get_Source_Cap Message](#) is received.

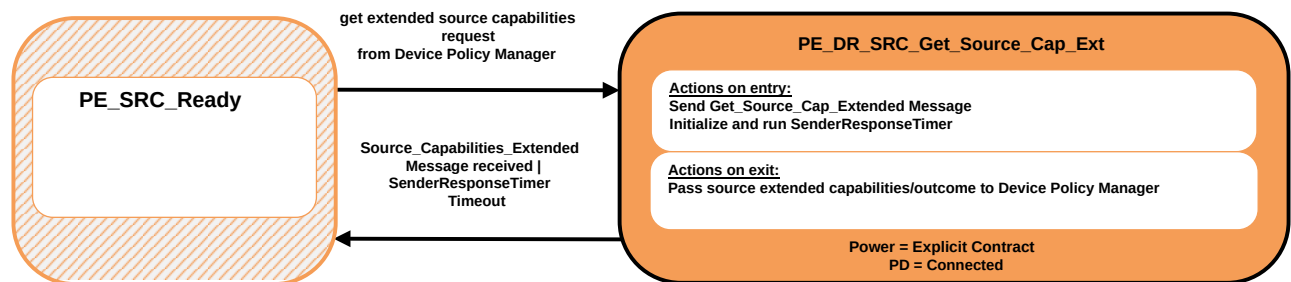
- On entry to the PE_DR_SNK_Give_Source_Cap State the Policy Engine **Shall** Request the Device Policy Manager for the current system Capabilities. The Policy Engine **Shall** then Request the Protocol Layer to send a Source Capabilities Message containing these Capabilities.
- The Policy Engine **Shall** send:
- A [Source_Capabilities Message](#) when a [Get_Source_Cap Message](#) is received or
- An [EPR_Source_Capabilities Message](#) when a [EPR_Get_Source_Cap Message](#) is received.

The Policy Engine **Shall** transition to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

- The Source Capabilities Message has been successfully sent.

9.2.20.11. Dual-Role (Source Port) Get Source Capabilities Extended State Diagram

[Figure 9.69](#) shows the State diagram for Dual-Role Device, presently operating as a Source, on receiving a Request from the Device Policy Manager to get the Port Partner's extended Source Capabilities. See also [Section 6.5.2](#).

Figure 9.69. Dual-Role (Source) Get Source Capabilities Extended State Diagram**9.2.20.11.1. PE_DR_SRC_Get_Source_Cap_Ext State**

The Policy Engine **Shall** transition to the PE_DR_SRC_Get_Source_Cap_Ext State, from the PE_SRC_Ready State, due to a Request to get the remote extended Source Capabilities from the Device Policy Manager.

On entry to the PE_DR_SRC_Get_Source_Cap_Ext State the Policy Engine **Shall** send a [Get_Source_Cap_Extended Message](#) and initialize and run the SenderResponseTimer.

On exit from the PE_DR_SRC_Get_Source_Cap_Ext State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout).

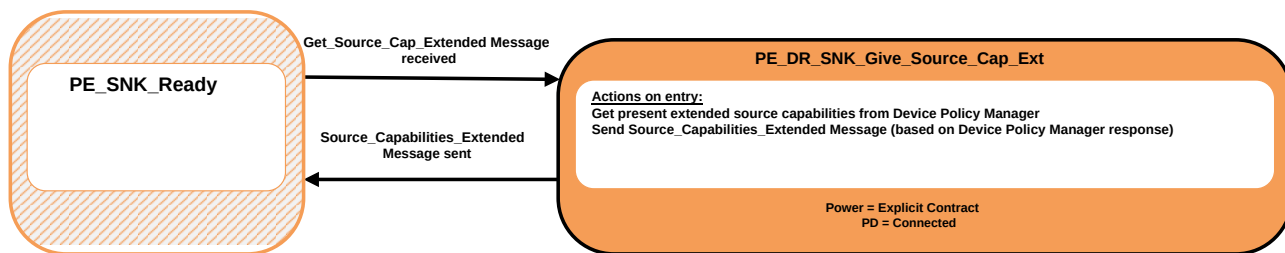
The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#)) when:

- A [Source_Capabilities_Extended Message](#) is received
- Or SenderResponseTimer times out.

9.2.20.12. Dual-Role (Sink Port) Give Source Capabilities Extended State Diagram

[Figure 9.68](#) shows the State diagram for a Dual-Role Device, presently operating as a Sink, on receiving a [Get_Source_Cap_Extended Message](#). See also [Section 7.19.2](#).

Figure 9.70. Dual-Role (Sink) Give Source Capabilities Extended diagram



9.2.20.12.1. PE_DR_SNK_Give_Source_Cap_Ext State

The Policy Engine **Shall** transition to the PE_DR_SNK_Give_Source_Cap_Ext State, from the PE_SNK_Ready State, when a [Get_Source_Cap_Extended Message](#) is received.

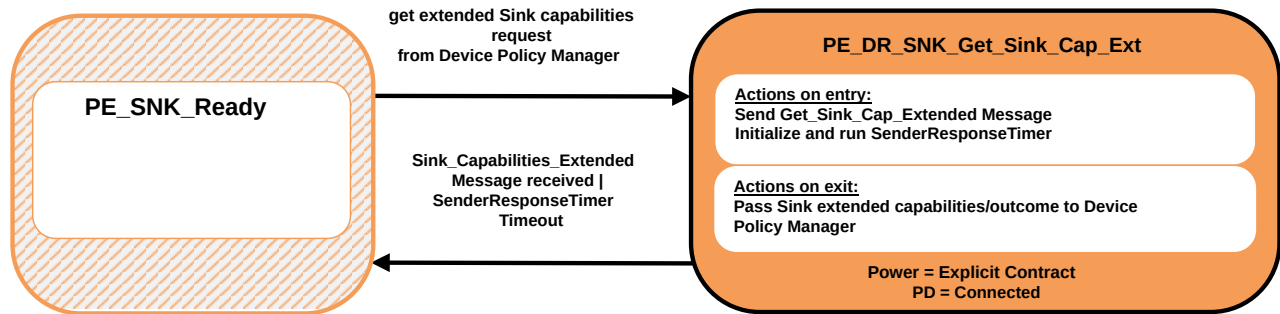
On entry to the PE_DR_SNK_Give_Source_Cap_Ext State the Policy Engine **Shall** Request the present extended Source Capabilities from the Device Policy Manager and then send a [Source_Capabilities_Extended Message](#) based on these Capabilities.

The Policy Engine **Shall** transition back to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

- The [Source_Capabilities_Extended Message](#) has been successfully sent.

9.2.20.13. Dual-Role (Sink Port) Get Sink Capabilities Extended State Diagram

[Figure 9.67](#) shows the State diagram for a Dual-Role Device, presently operating as a Sink, on receiving a Request from the Device Policy Manager to get the Port Partner's extended Sink Capabilities. See also [Section 6.5.16](#).

Figure 9.71. Dual-Role (Sink) Get Sink Capabilities Extended State Diagram**9.2.20.13.1. PE_DR_SNK_Get_Sink_Cap_Ext State**

The Policy Engine **Shall** transition to the PE_DR_SNK_Get_Sink_Cap_Ext State, from the PE_SNK_Ready State, due to a Request to get the remote extended Source Capabilities from the Device Policy Manager.

On entry to the PE_DR_SNK_Get_Sink_Cap_Ext State the Policy Engine **Shall** send a [Get_Sink_Cap_Extended Message](#) and initialize and run the SenderResponseTimer.

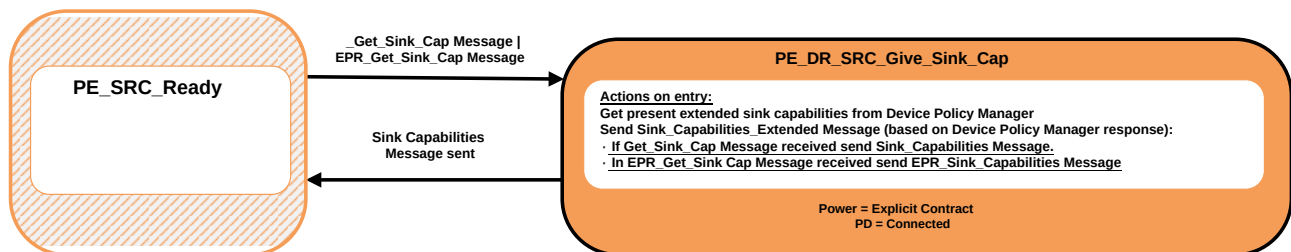
On exit from the PE_DR_SNK_Get_Sink_Cap_Ext State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (Capabilities or response timeout).

The Policy Engine **Shall** transition back to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

- A [Sink_Capabilities_Extended Message](#) is received.
- Or SenderResponseTimer times out.

9.2.20.14. Dual-Role (Source Port) Give Sink Capabilities Extended State Diagram

[Figure 9.72](#) shows the State diagram for a Dual-Role Device, presently operating as a Sink, on receiving a [Get_Sink_Cap_Extended Message](#). See also [Section 6.5.16](#).

Figure 9.72. Dual-Role (Source) Give Sink Capabilities Extended diagram**9.2.20.14.1. PE_DR_SRC_Give_Sink_Cap_Ext State**

The Policy Engine **Shall** transition to the PE_DR_SRC_Give_Sink_Cap_Ext State, from the PE_SRC_Ready State, when a [Get_Sink_Cap_Extended Message](#) is received.

On entry to the PE_DR_SRC_Give_Sink_Cap_Ext State the Policy Engine **Shall** Request the present extended Sink Capabilities from the Device Policy Manager and then send a [Sink_Capabilities_Extended Message](#) based on these Capabilities.

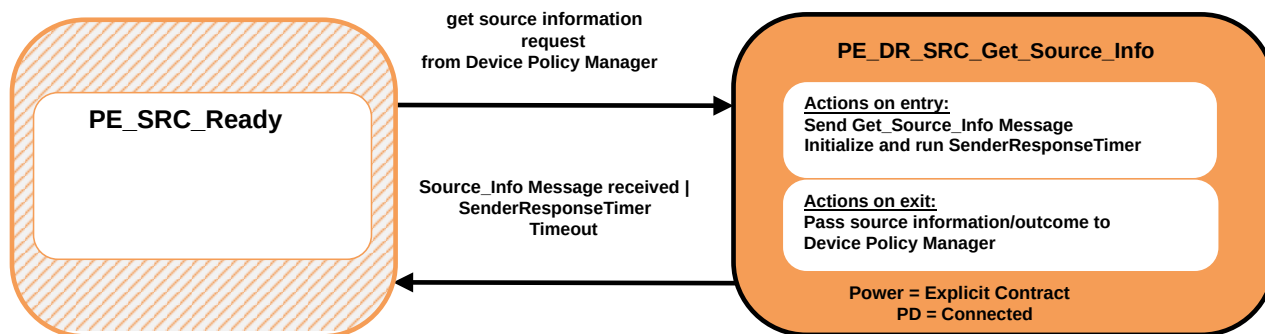
The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#))when:

- The [Sink_Capabilities_Extended Message](#) has been successfully sent.

9.2.20.15. Dual-Role (Source Port) Get Source Information State Diagram

[Figure 9.73](#) shows the State diagram for a Dual-Role Device, presently operating as a Source, on receiving a Request from the Device Policy Manager to get the Port Partner's Source information. See also [Section 6.3.23](#) and [Section 6.4.10](#).

Figure 9.73. Dual-Role (Source) Get Source Information State Diagram



9.2.20.15.1. PE_DR_SRC_Get_Source_Info State

The Policy Engine **Shall** transition to the PE_DR_SRC_Get_Source_Info State, from the PE_SRC_Ready State, due to a Request to get the remote Source information from the Device Policy Manager.

On entry to the PE_DR_SRC_Get_Source_Info State the Policy Engine **Shall** send a [Get_Source_Info Message](#) and initialize and run the SenderResponseTimer.

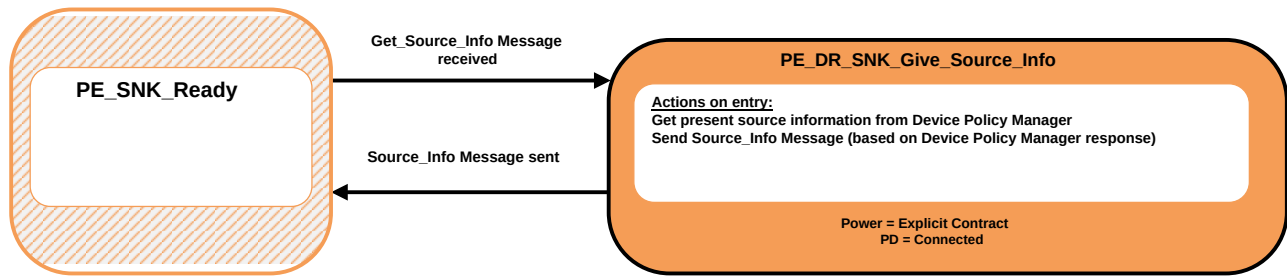
On exit from the PE_DR_SRC_Get_Source_Info State the Policy Engine **Shall** inform the Device Policy Manager of the outcome (information or response timeout).

The Policy Engine **Shall** transition back to the PE_SRC_Ready State (see [Figure 9.17](#)) when:

- A [Source_Info Message](#) is received.
- Or SenderResponseTimer times out.

9.2.20.16. Dual-Role (Sink Port) Give Source Information State Diagram

[Figure 9.68](#) shows the State diagram for a Dual-Role Device, presently operating as a Sink, on receiving a [Get_Source_Info Message](#). See also [Section 6.3.23](#) and [Section 6.4.10](#).

Figure 9.74. Dual-Role (Source) Give Source Information diagram

9.2.20.16.1. PE_DR_SNK_Give_Source_Info State

The Policy Engine **Shall** transition to the PE_DR_SNK_Give_Source_Info State, from the PE_SNK_Ready State, when a [Get_Source_Info Message](#) is received.

On entry to the PE_DR_SNK_Give_Source_Info State the Policy Engine **Shall** Request the present Source information from the Device Policy Manager and then send a [Source_Info Message](#) based on this information.

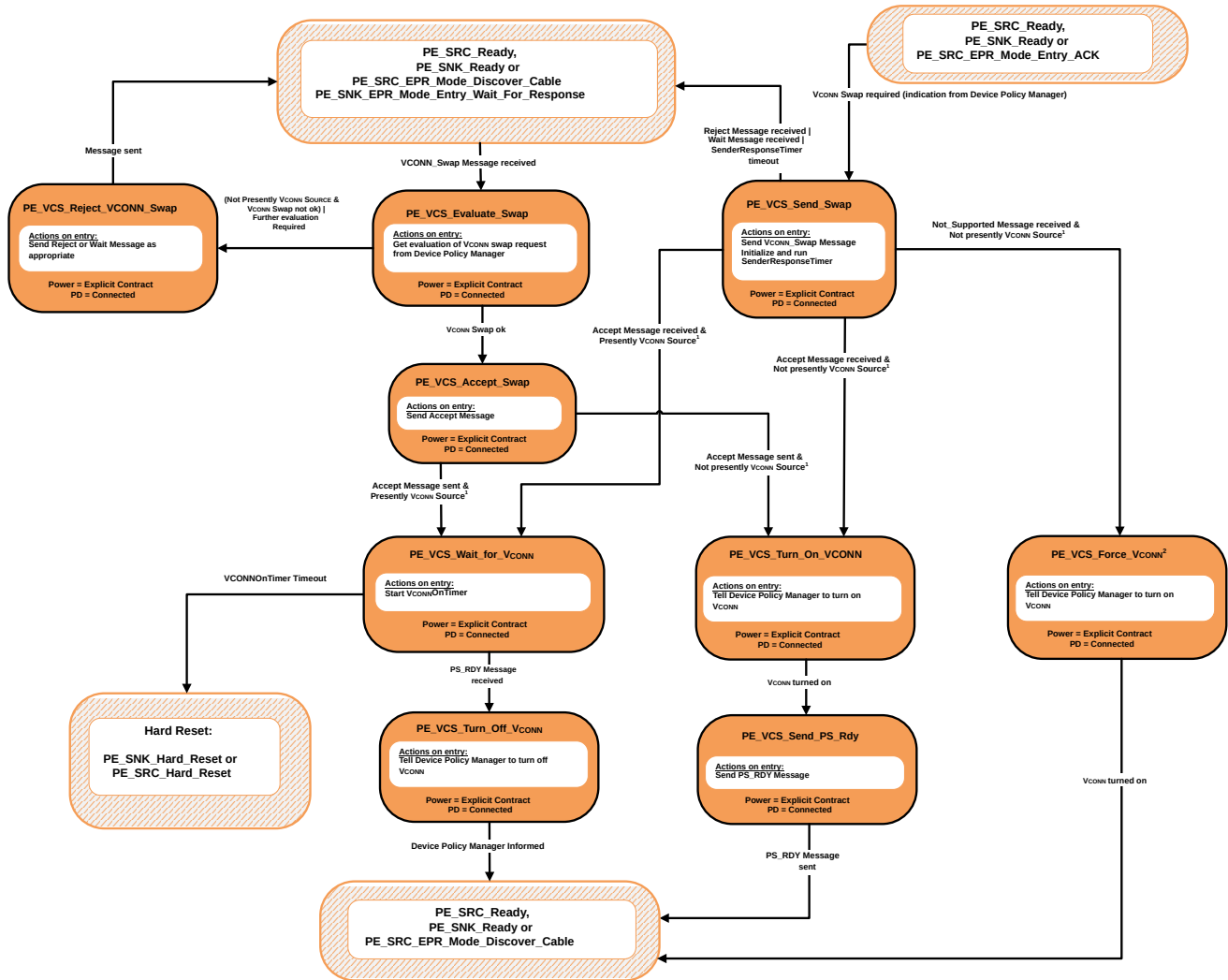
The Policy Engine **Shall** transition back to the PE_SNK_Ready State (see [Figure 9.18](#)) when:

- The [Source_Info Message](#) has been successfully sent.

9.2.21. VCONN Swap State Diagram

The State Diagram in this section **Shall** apply to Ports that supply VCONN. [Figure 9.75](#) shows the State operation for a Port on sending or receiving a [VCONN Swap](#) Request.

Figure 9.75. VCONN Swap State Diagram



1. A Port is presently the VCONN Source if it has the responsibility for supplying VCONN even if VCONN has been turned off.
2. The **PE_VCS_Force_VCONN** State is **Optional**.

9.2.21.1. PE_VCS_Send_Swap State

The **PE_VCS_Send_Swap** State is entered from either the **PE_SRC_Ready** or **PE_SNK_Ready** State when the Policy Engine receives a Request from the Device Policy Manager to perform a [VCONN Swap](#).

On entry to the **PE_VCS_Send_Swap** State the Policy Engine **Shall** send a [VCONN_Swap Message](#) and start the **SenderResponseTimer**.

The Policy Engine **Shall** transition to the **PE_VCS_Wait_For_VCONN** State when:

- An [Accept Message](#) is received and
- The Port is presently the VCONN Source.

The Policy Engine **Shall** transition to the PE_VCS_Turn_On_VCONN State when:

- An [Accept Message](#) is received and
- The Port is not presently the VCONN Source.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_SRC_EPR_Mode_Discover_Cable State when:

- A [Reject Message](#) is received or
- A [Wait Message](#) is received or
- The SenderResponseTimer times out.

The Policy Engine **May** transition to the PE_VCS_Force_VCONN State when:

- A [Not_Supported Message](#) is received and
- The Port is not presently the VCONN Source.

9.2.21.2. PE_VCS_Evaluate_Swap State

The PE_VCS_Evaluate_Swap State is entered from either the PE_SRC_Ready or PE_SNK_Ready State when the Policy Engine receives a [VCONN_Swap Message](#).

On entry to the PE_VCS_Evaluate_Swap State the Policy Engine **Shall** Request the Device Policy Manager for an evaluation of the [VCONN Swap](#) Request.

The Policy Engine **Shall** transition to the PE_VCS_Accept_Swap State when:

- The Device Policy Manager indicates that a [VCONN Swap](#) is OK.

The Policy Engine **Shall** transition to the PE_VCS_Reject_Swap State when:

- The Port is not presently the VCONN Source and the Device Policy Manager indicates that a [VCONN Swap](#) is not OK or
- The Device Policy Manager indicates that a [VCONN Swap](#) cannot be done at this time.

9.2.21.3. PE_VCS_Accept_Swap State

On entry to the PE_VCS_Accept_Swap State the Policy Engine **Shall** send an [Accept Message](#). The Policy Engine **Shall** transition to the PE_VCS_Wait_For_VCONN State when:

- The [Accept Message](#) has been sent and
- The Port's VCONN is on.

The Policy Engine **Shall** transition to the PE_VCS_Turn_On_VCONN State when:

- The [Accept Message](#) has been sent and
- The Port's VCONN is off.

9.2.21.4. PE_VCS_Reject_Swap State

On entry to the PE_VCS_Reject_Swap State the Policy Engine **Shall** Request the Protocol Layer to send:

- A [Reject Message](#) if the Device is unable to perform a [VCONN Swap](#) at this time.
- A [Wait Message](#) if further evaluation of the [VCONN Swap](#) Request is required.

Note: In this case it is expected that the Port will send a [VCONN_Swap Message](#) at a later time. The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_SRC_EPR_Mode_Discover_Cable State when:

- The [Reject](#) or [Wait Message](#) has been sent.

9.2.21.5. PE_VCS_Wait_for_VCONN State

On entry to the PE_VCS_Wait_For_VCONN State the Policy Engine **Shall** start the [VconnOnTimer](#). The Policy Engine **Shall** transition to the PE_VCS_Turn_Off_VCONN State when:

- A [PS_RDY Message](#) is received.

The Policy Engine **Shall** transition to either the PE_SRC_Hard_Reset or PE_SNK_Hard_Reset State when:

- The [VconnOnTimer](#) times out.

9.2.21.6. PE_VCS_Turn_Off_VCONN State

On entry to the PE_VCS_Turn_Off_VCONN State the Policy Engine **Shall** tell the Device Policy Manager to turn off VCONN.

The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_SRC_EPR_Mode_Discover_Cable State when:

- The Device Policy Manager has been informed.

9.2.21.7. PE_VCS_Turn_On_VCONN State

On entry to the PE_VCS_Turn_On_VCONN State the Policy Engine **Shall** tell the Device Policy Manager to turn on VCONN.

The Policy Engine **Shall** transition to the PE_VCS_Send_Ps_Rdy State when:

- The Port's VCONN is on.

9.2.21.8. PE_VCS_Send_PS_Rdy State

On entry to the PE_VCS_Send_Ps_Rdy State the Policy Engine **Shall** send a [PS_RDY Message](#). The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_SRC_EPR_Mode_Discover_Cable State when:

- The [PS_RDY Message](#) has been sent.

9.2.21.9. PE_VCS_Force_VCONN State

On entry to the PE_VCS_Force_VCONN State the Policy Engine **Shall** tell the Device Policy Manager to turn on VCONN. The Policy Engine **Shall** transition back to either the PE_SRC_Ready, PE_SNK_Ready or PE_SRC_EPR_Mode_Discover_Cable State when:

- The Port's VCONN is on.

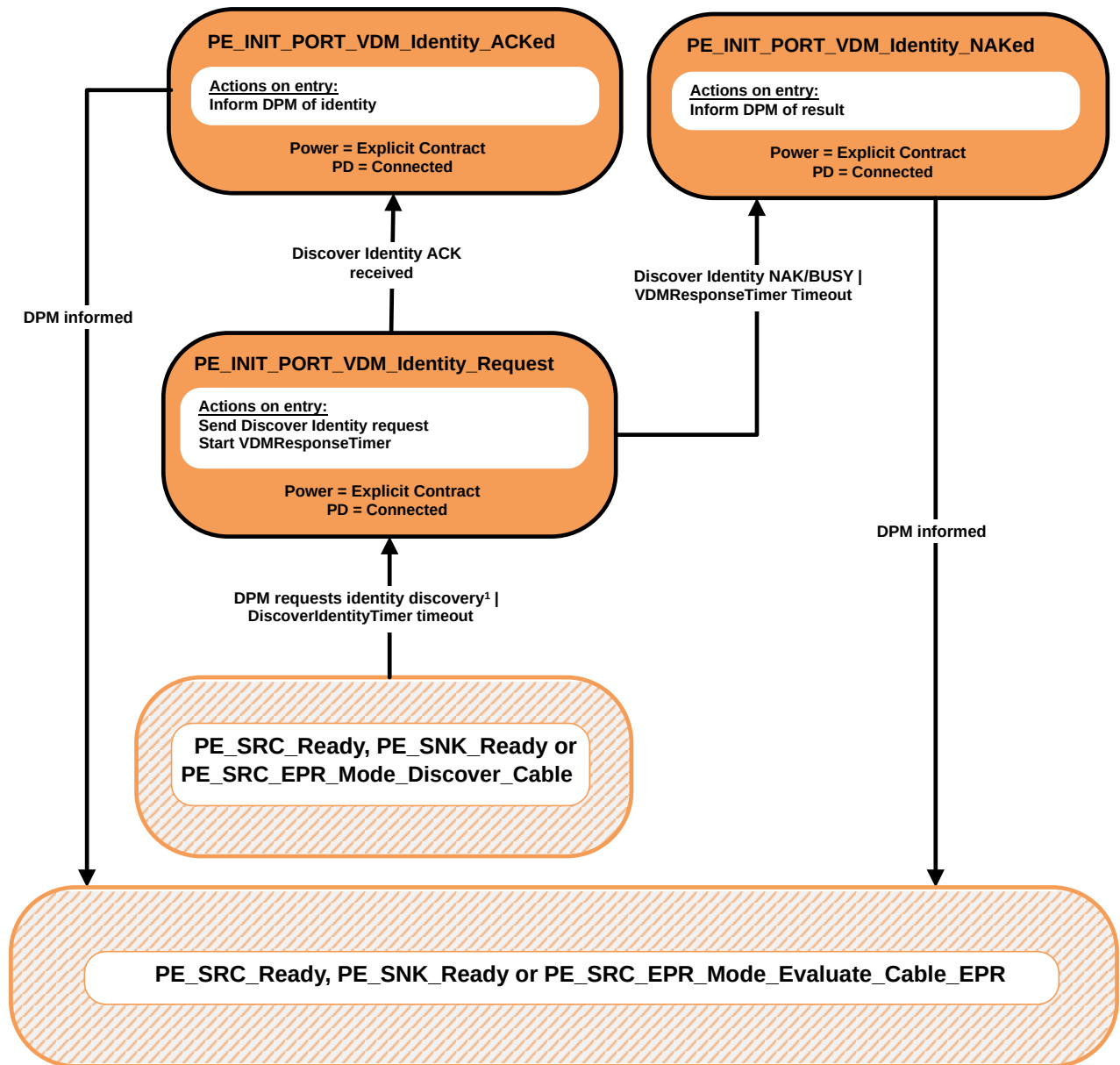
9.2.22. Initiator Structured VDM State Diagrams

The State Diagrams in this section **Shall** apply to all Initiators.

9.2.22.1. Initiator Structured VDM Discover Identity State Diagram

[Figure 9.76](#) shows the State diagram for an Initiator when discovering the identity of its Port Partner or Cable Plug.

Figure 9.76. Initiator to Port VDM Discover Identity State Diagram



1. The DPM in an EPR Source **Shall** Request the discovery of the identity of the Cable Plug at startup.

9.2.22.1.1. PE_INIT_Port_VDM_Identity_Request State

The Policy Engine transitions to the PE_INIT_Port_VDM_Identity_Request State from either the PE_SRC_Ready or PE_SNK_Ready State when:

- The Device Policy Manager requests the discovery of the identity of the Port Partner or Cable Plug or
- The DiscoverIdentityTimer times out.

The Policy Engine transitions to the PE_INIT_Port_VDM_Identity_Request State from the PE_SRC_EPR_Mode_Discover_Cable State when:

- The Cable Plug Discovery Process has been initiated.

On entry to the PE_INIT_Port_VDM_Identity_Request State the Policy Engine **Shall** send a [Structured VDM Discover Identity](#) Command Request and **Shall** start the VDMResponseTimer.

The Policy Engine **Shall** transition to the PE_INIT_Port_VDM_Identity_ACKed State when:

- A [Structured VDM Discover Identity](#) ACK Command response is received.

The Policy Engine **Shall** transition to the PE_INIT_Port_VDM_Identity_NAKed State when:

- A [Structured VDM Discover Identity](#) NAK or BUSY Command response is received or
- The VDMResponseTimer times out.

9.2.22.1.2. PE_INIT_Port_VDM_Identity_ACKed State

On entry to the PE_INIT_Port_VDM_Identity_ACKed State the Policy Engine **Shall** inform the Device Policy Manager of the Identity information.

The Policy Engine **Shall** transition to either the PE_SRC_Ready, PE_SNK_Ready or PE_SRC_EPR_Mode_Evaluate_Cable_EPR State when:

- The Device Policy Manager has been informed.

9.2.22.1.3. PE_INIT_Port_VDM_Identity_NAKed State

On entry to the PE_INIT_Port_VDM_Identity_NAKed State the Policy Engine **Shall** inform the Device Policy Manager of the result (NAK, BUSY or timeout).

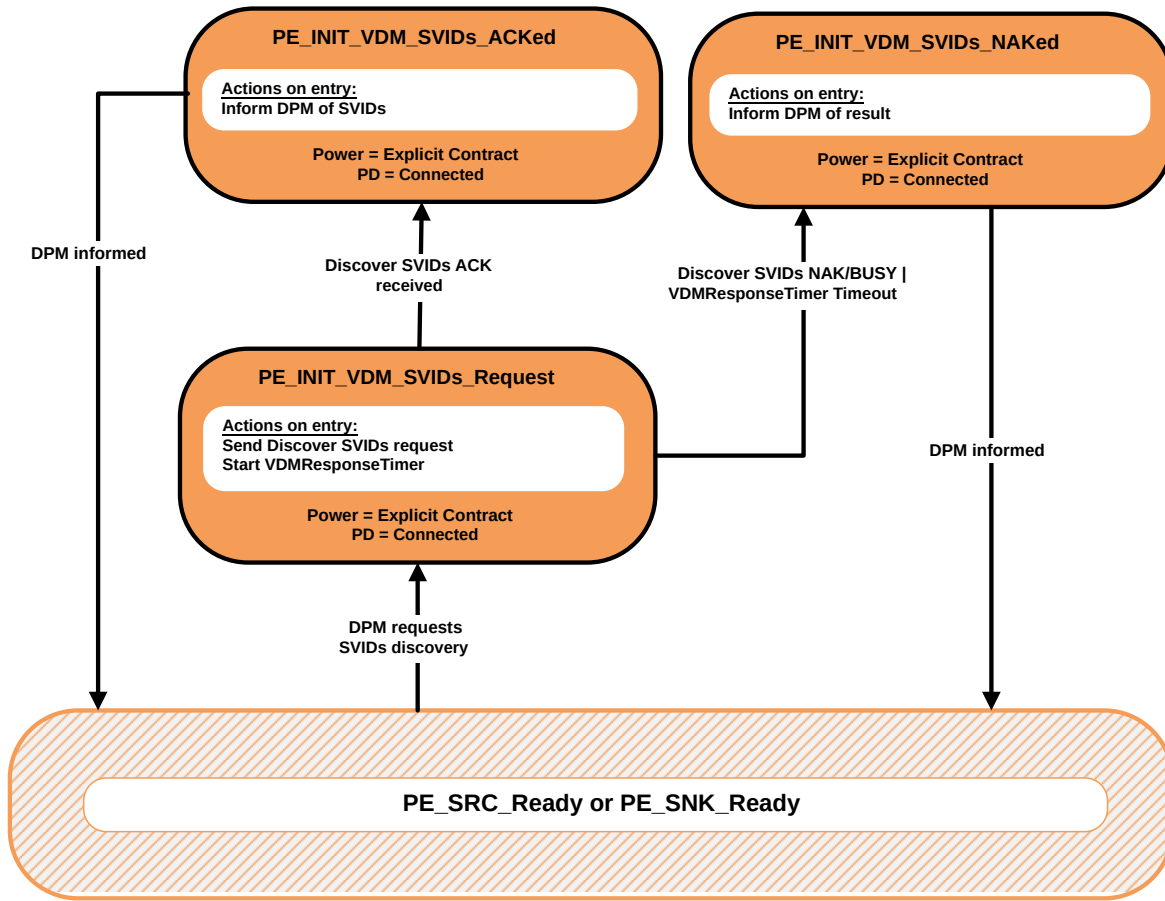
The Policy Engine **Shall** transition to either the PE_SRC_Ready, PE_SNK_Ready or PE_SRC_EPR_Mode_Evaluate_Cable_EPR State when:

- The Device Policy Manager has been informed.

9.2.22.2. Initiator Structured VDM Discover SVIDs State Diagram

[Figure 9.77](#) shows the State diagram for an Initiator when discovering SVIDs of its Port Partner or Cable Plug.

Figure 9.77. Initiator VDM Discover SVIDs State Diagram



9.2.22.2.1. PE_INIT_VDM_SVIDs_Request State

The Policy Engine transitions to the PE_INIT_VDM_SVIDs_Request State from either the PE_SRC_Ready or PE_SNK_Ready State when:

- The Device Policy Manager requests the discovery of the SVIDs of the Port Partner or a Cable Plug.

On entry to the PE_INIT_VDM_SVIDs_Request State the Policy Engine **Shall** send a [Structured VDM Discover SVIDs](#) Command Request and **Shall** start the VDMResponseTimer.

The Policy Engine **Shall** transition to the PE_INIT_VDM_SVIDs_ACKed State when:

- A [Structured VDM Discover SVIDs](#) ACK Command response is received. The Policy Engine **Shall** transition to the PE_INIT_VDM_SVIDs_NAKed State when:
 - A [Structured VDM Discover SVIDs](#) NAK or BUSY Command response is received or
 - The VDMResponseTimer times out.

9.2.22.2.2. PE_INIT_VDM_SVIDs_ACKed State

On entry to the PE_INIT_VDM_SVIDs_ACKed State the Policy Engine **Shall** inform the Device Policy Manager of the SVIDs information.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

- The Device Policy Manager has been informed.

9.2.22.2.3. PE_INIT_VDM_SVIDs_NAKed State

On entry to the PE_INIT_VDM_SVIDs_NAKed State the Policy Engine **Shall** inform the Device Policy Manager of the result (NAK, BUSY or timeout).

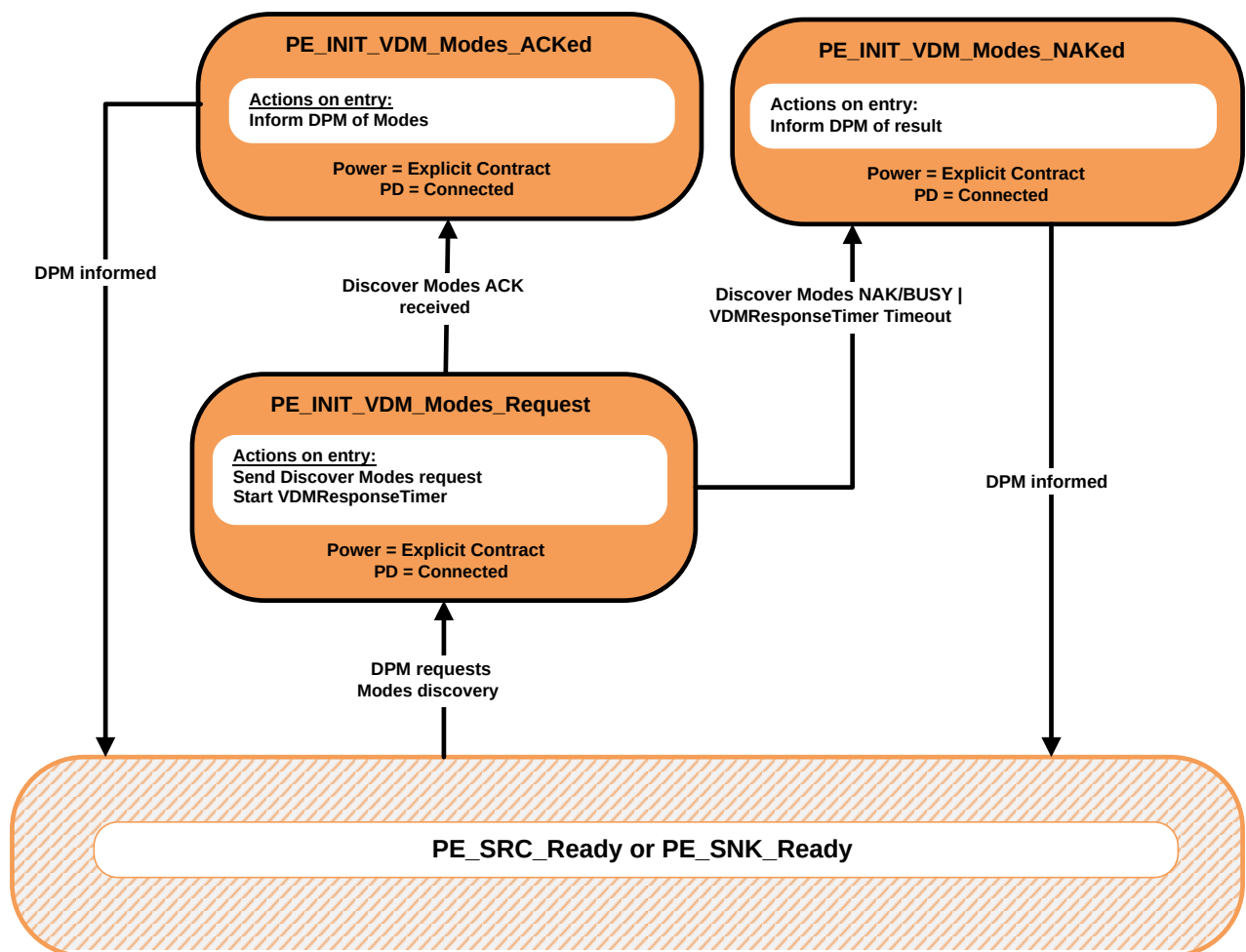
The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

- The Device Policy Manager has been informed.

9.2.22.3. Initiator Structured VDM Discover Modes State Diagram

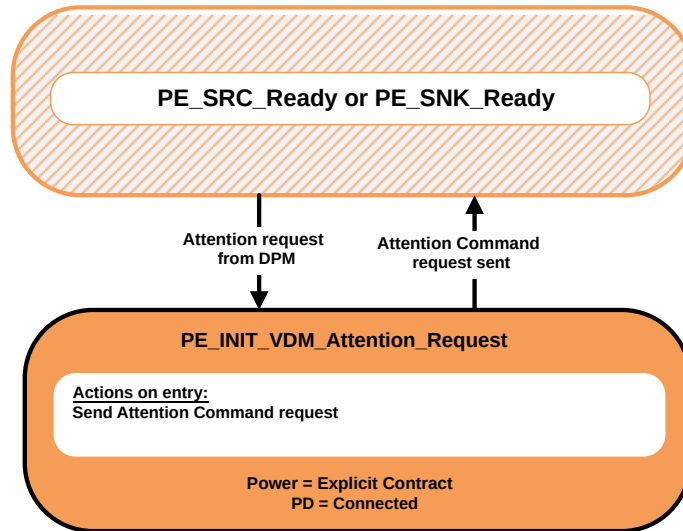
[Figure 9.78](#) shows the State diagram for an Initiator when discovering Modes of its Port Partner or Cable Plug.

Figure 9.78. Initiator VDM Discover Modes State Diagram



9.2.22.4. Initiator Structured VDM Attention State Diagram

[Figure 9.79](#) shows the State diagram for an Initiator when sending an [Attention](#) Command Request.

Figure 9.79. Initiator VDM Attention State Diagram**9.2.22.4.1. PE_INIT_VDM_Modes_Request State**

The Policy Engine transitions to the PE_INIT_VDM_Modes_Request State from either the PE_SRC_Ready or PE_SNK_Ready State when:

- The Device Policy Manager requests the discovery of the Modes of the Port Partner or a Cable Plug.

On entry to the PE_INIT_VDM_Modes_Request State the Policy Engine **Shall** send a [Structured VDM Discover Modes](#) Command Request and **Shall** start the VDMResponseTimer.

The Policy Engine **Shall** transition to the PE_INIT_VDM_Modes_ACKed State when:

- A [Structured VDM Discover Modes](#) ACK Command response is received. The Policy Engine **Shall** transition to the PE_INIT_VDM_Modes_NAKed State when:
 - A [Structured VDM Discover Modes](#) NAK or BUSY Command response is received or
 - The VDMResponseTimer times out.

9.2.22.4.2. PE_INIT_VDM_Modes_ACKed State

On entry to the PE_INIT_VDM_Modes_ACKed State the Policy Engine **Shall** inform the Device Policy Manager of the Modes information.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a DFP when:

- The Device Policy Manager has been informed.

9.2.22.4.3. PE_INIT_VDM_Modes_NAKed State

On entry to the PE_INIT_VDM_Modes_NAKed State the Policy Engine **Shall** inform the Device Policy Manager of the result (NAK, BUSY or timeout).

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a DFP when:

- The Device Policy Manager has been informed.

9.2.22.4.4. PE_INIT_VDM_Attention_Request State

The Policy Engine transitions to the PE_INIT_VDM_Attention_Request State from either the PE_SRC_Ready or PE_SNK_Ready State when:

- When the Device Policy Manager requests attention from its Port Partner.

On entry to the PE_INIT_VDM_Attention_Request State the Policy Engine **Shall** send an [Attention](#) Command Request.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

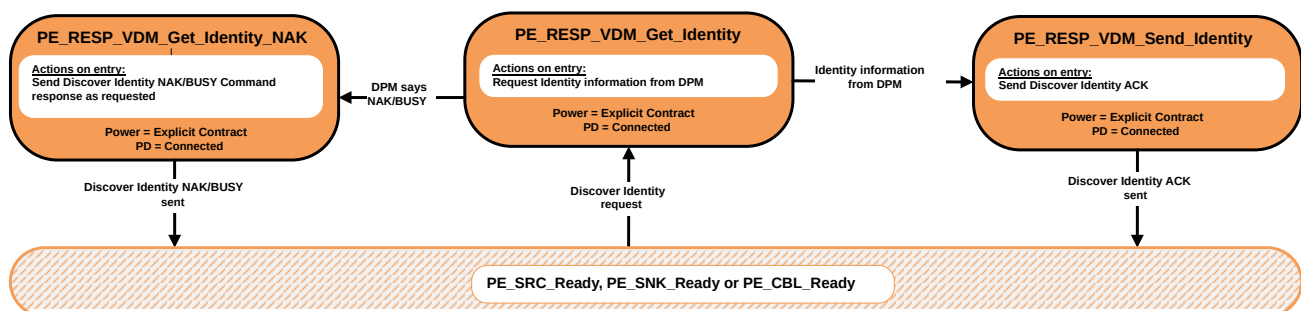
- The [Attention](#) Command Request has been sent.

9.2.23. Responder Structured VDM State Diagrams

9.2.23.1. Responder Structured VDM Discover Identity State Diagram

[Figure 9.80](#) shows the State diagram for a Responder receiving a [Discover Identity](#) Command Request.

Figure 9.80. Responder Structured VDM Discover Identity State Diagram



The Policy Engine transitions to the PE_RESP_VDM_Get_Identity State from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- A [Structured VDM Discover Identity](#) Command Request is received.

On entry to the PE_RESP_VDM_Get_Identity State the Responder **Shall** Request identity information from the Device Policy Manager.

The Policy Engine **Shall** transition to the PE_RESP_VDM_Send_Identity State when:

- Identity information is received from the Device Policy Manager.

The Policy Engine **Shall** transition to the PE_RESP_VDM_Get_Identity_NAK State when:

- The Device Policy Manager indicates that the response to the Discover Identity Command Request is NAK or BUSY.

9.2.23.1.1. PE_RESP_VDM_Send_Identity State

On entry to the PE_RESP_VDM_Send_Identity State the Responder **Shall** send the [Structured VDM Discover Identity](#) ACK Command response.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a UFP when:

- The [Structured VDM Discover Identity](#) ACK Command response has been sent.

9.2.23.1.2. PE_RESP_VDM_Get_Identity_NAK State

On entry to the PE_RESP_VDM_Get_Identity_NAK State the Policy Engine **Shall** send a [Structured VDM Discover Identity](#) NAK or BUSY Command response as indicated by the Device Policy Manager.

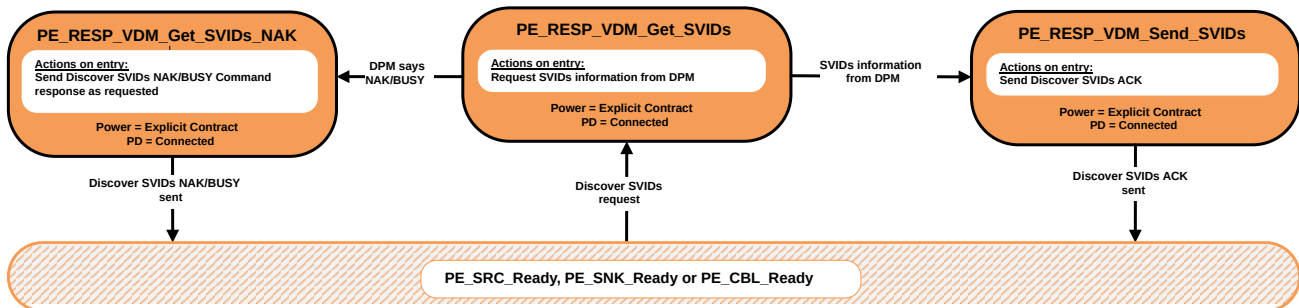
The Policy Engine **Shall** transition to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- The [Structured VDM Discover Identity](#) NAK or BUSY Command response has been sent.

9.2.23.2. Responder Structured VDM Discover SVIDs State Diagram

[Figure 9.81](#) shows the State diagram for a Responder when receiving a [Discover SVIDs](#) Command.

Figure 9.81. Responder Structured VDM Discover SVIDs State Diagram



The Policy Engine transitions to the PE_RESP_VDM_Get_SVIDs State from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- A [Structured VDM Discover SVIDs](#) Command Request is received.

On entry to the PE_RESP_VDM_Get_SVIDs State the Responder **Shall** Request SVIDs information from the Device Policy Manager.

The Policy Engine **Shall** transition to the PE_RESP_VDM_Send_SVIDs State when:

- SVIDs information is received from the Device Policy Manager.

The Policy Engine **Shall** transition to the PE_RESP_VDM_Get_SVIDs_NAK State when:

- The Device Policy Manager indicates that the response to the Discover SVIDs Command Request is NAK or BUSY.

9.2.23.2.1. PE_UFP_VDM_Send_SVIDs State

On entry to the PE_RESP_VDM_Send_SVIDs State the Responder **Shall** send the [Structured VDM Discover SVIDs](#) ACK Command response.

The Policy Engine **Shall** transition to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- The [Structured VDM Discover SVIDs](#) ACK Command response has been sent.

9.2.23.2.2. PE_UFP_VDM_Get_SVIDs_NAK State

On entry to the PE_RESP_VDM_Get_SVIDs_NAK State the Policy Engine **Shall** send a [Structured VDM Discover SVIDs](#) NAK or BUSY Command response as indicated by the Device Policy Manager.

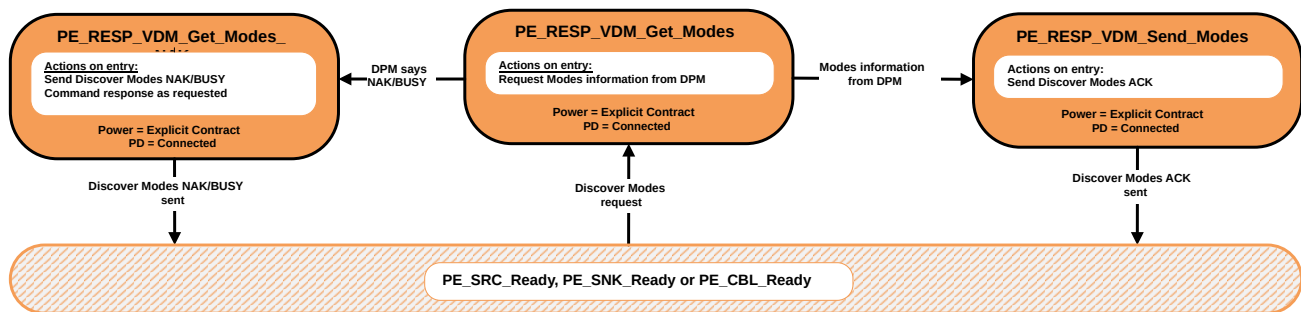
The Policy Engine **Shall** transition to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- The [Structured VDM Discover SVIDs](#) NAK or BUSY Command response has been sent.

9.2.23.3. Responder Structured VDM Discover Modes State Diagram

[Figure 9.82](#) shows the State diagram for a Responder on receiving a [Discover Modes](#) Command.

Figure 9.82. Responder Structured VDM Discover Modes State Diagram



The Policy Engine transitions to the PE_RESP_VDM_Get_Modes State from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- A [Structured VDM Discover Modes](#) Command Request is received.

On entry to the PE_RESP_VDM_Get_Modes State the Responder **Shall** Request Modes information from the Device Policy Manager.

The Policy Engine **Shall** transition to the PE_RESP_VDM_Send_Modes State when:

- Modes information is received from the Device Policy Manager.

The Policy Engine **Shall** transition to the PE_RESP_VDM_Get_Modes_NAK State when:

- The Device Policy Manager indicates that the response to the Discover Modes Command Request is NAK or BUSY.

9.2.23.3.1. PE_RESP_VDM_Send_Modes State

On entry to the PE_RESP_VDM_Send_Modes State the Responder **Shall** send the [Structured VDM Discover Modes](#) ACK Command response.

The Policy Engine **Shall** transition to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- The [Structured VDM Discover Modes](#) ACK Command response has been sent.

9.2.23.3.2. PE_RESP_VDM_Get_Modes_NAK State

On entry to the PE_RESP_VDM_Get_Modes_NAK State the Policy Engine **Shall** send a [Structured VDM Discover Modes](#) NAK or BUSY Command response as indicated by the Device Policy Manager.

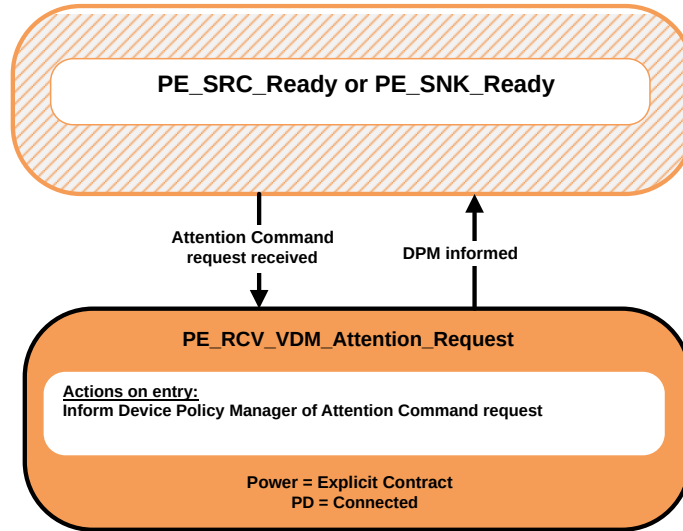
The Policy Engine **Shall** transition to either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- The [Structured VDM Discover Modes](#) NAK or BUSY Command response has been sent.

9.2.23.4. Receiving a Structured VDM Attention State Diagram

[Figure 9.83](#) shows the State diagram for a Responder when receiving an [Attention](#) Command Request.

Figure 9.83. Receiving a Structured VDM Attention State Diagram



9.2.23.4.1. PE_RCV_VDM_Attention_Request State

The Policy Engine transitions to the PE_RCV_VDM_Attention_Request State from either the PE_SRC_Ready or PE_SNK_Ready State when:

- An [Attention](#) Command Request is received.

On entry to the PE_RCV_VDM_Attention_Request State the Policy Engine **Shall** inform the Device Policy Manager of the [Attention](#) Command Request.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State when:

- The Device Policy Manager has been informed.

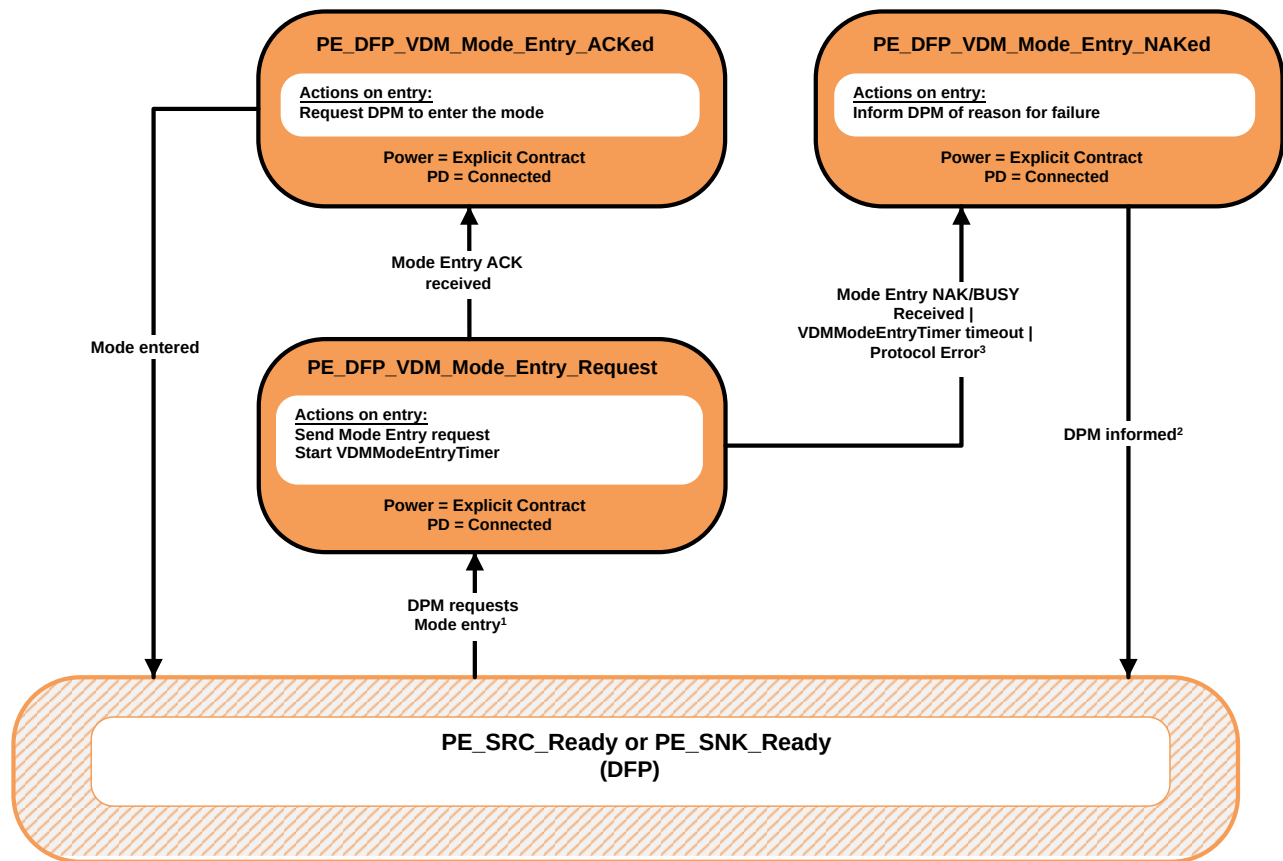
9.2.24. DFP Structured VDM State Diagrams

The State Diagrams in this section **Shall** apply to all DFPs that support Structured VDMs.

9.2.24.1. DFP Structured VDM Mode Entry State Diagram

[Figure 9.84](#) shows the State operation for a DFP when entering a Mode.

Figure 9.84. DFP VDM Mode Entry State Diagram



1. The Device Policy Manager **Shall** have placed the system into USB Safe State before issuing this Request when entering Modal Operation.
2. The Device Policy Manager **Shall** have returned the system to USB operation if not in Modal Operation at this point.
3. Protocol Errors are handled by informing the DPM, returning to USB Safe State and then processing the Message once the PE_SRC_Ready or PE_SNK_Ready State has been entered.

9.2.24.1.1. PE_DFP_VDM_Mode_Entry_Request State

The Policy Engine transitions to the PE_DFP_VDM_Mode_Entry_Request State from either the PE_SRC_Ready or PE_SNK_Ready State for a DFP when:

- The Device Policy Manager requests that the Port Partner or a Cable Plug enter a Mode.

On entry to the PE_DFP_VDM_Mode_Entry_Request State the Policy Engine **Shall** send a [Structured VDM Enter Mode](#) Command Request and **Shall** start the VDMModeEntryTimer.

The Policy Engine **Shall** transition to the PE_DFP_VDM_Mode_Entry_ACKed State when:

- A [Structured VDM Enter Mode](#) ACK Command response is received.

The Policy Engine **Shall** transition to the PE_DFP_VDM_Mode_Entry_NAKed State when:

- A [Structured VDM Enter Mode](#) NAK or BUSY Command response is received or
- The VDMModeEntryTimer times out.

9.2.24.1.2. PE_DFP_VDM_Mode_Entry_ACKed State

On entry to the PE_DFP_VDM_Mode_Entry_ACKed State the Policy Engine **Shall** Request the Device Policy Manager to enter the Mode.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a DFP when:

- The Mode has been entered.

9.2.24.1.3. PE_DFP_VDM_Mode_Entry_NAKed State

On entry to the PE_DFP_VDM_Mode_Entry_NAKed State the Policy Engine **Shall** inform the Device Policy Manager of the reason for failure (NAK, BUSY, timeout or Protocol Error).

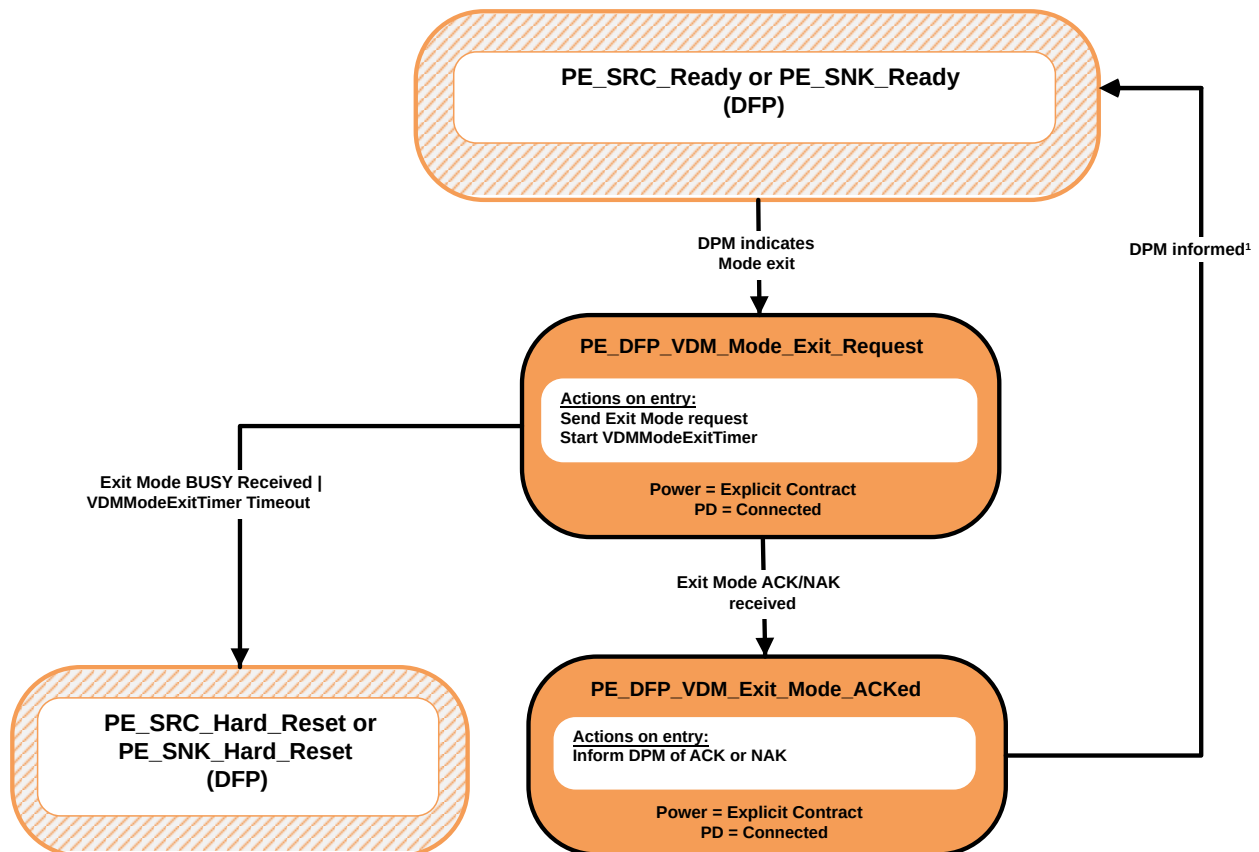
The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a DFP when:

- The Device Policy Manager has been informed.

9.2.24.2. DFP Structured VDM Mode Exit State Diagram

[Figure 9.85](#) shows the State operation for a DFP when exiting a Mode.

Figure 9.85. DFP VDM Mode Exit State Diagram



1. The Device Policy Manager is required to return the system to USB operation at this point when exiting Modal Operation.

9.2.24.2.1. PE_DFP_VDM_Mode_Exit_Request State

The Policy Engine transitions to the PE_DFP_VDM_Mode_Exit_Request State from either the PE_SRC_Ready or PE_SNK_Ready State for a DFP when:

- The Device Policy Manager requests that the Port Partner or a Cable Plug exit a Mode.

On entry to the PE_DFP_VDM_Mode_Exit_Request State the Policy Engine **Shall** send a [Structured VDM Exit Mode](#) Command Request and **Shall** start the VDMModeExitTimer.

The Policy Engine **Shall** transition to the PE_DFP_VDM_Mode_Entry_ACKed State when:

- A [Structured VDM Exit Mode](#) ACK or NAK Command response is received.

The Policy Engine **Shall** transition to either the PE_SRC_Hard_Reset or PE_SNK_Hard_Reset State depending on the present Power Role when:

- A [Structured VDM Exit Mode](#) BUSY Command response is received or
- The VDMModeExitTimer times out.

9.2.24.2.2. PE_DFP_VDM_DFP_Mode_Exit_ACKed State

On Exit to the PE_DFP_VDM_Mode_Entry_ACKed State the Policy Engine **Shall** inform the Device Policy Manager Of the result: ACK or NAK.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a DFP when:

- The Device Policy Manager has been informed.

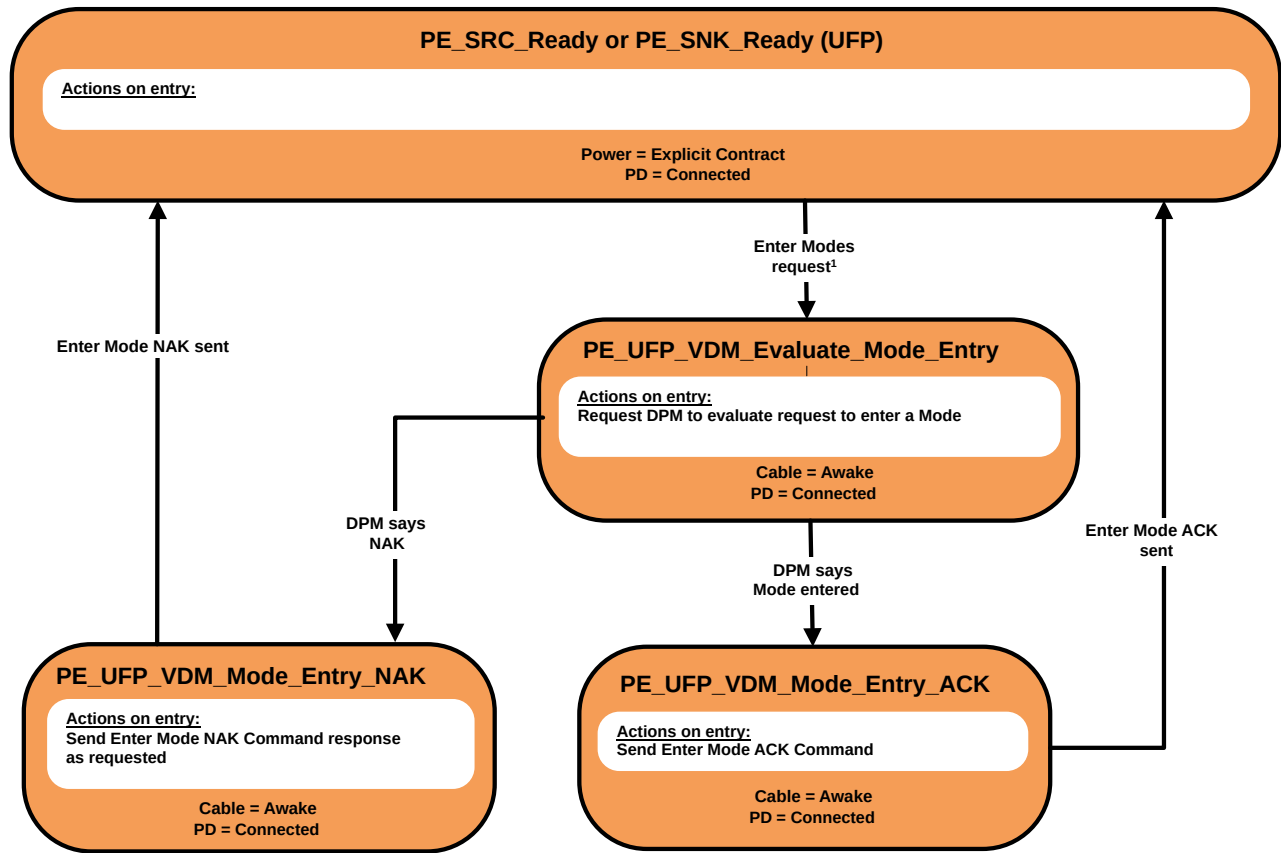
9.2.25. UFP Structured VDM State Diagrams

The State Diagrams in this section **Shall** apply to all UFPs that support Structured VDMs.

9.2.25.1. UFP Structured VDM Enter Mode State Diagram

[Figure 9.86](#) shows the State diagram for a UFP in response to an [Enter Mode](#) Command.

Figure 9.86. UFP Structured VDM Enter Mode State Diagram



1. The UFP is required to be in USB operation or USB Safe State at this point.

9.2.25.1.1. PE_UFP_VDM_Evaluate_Mode_Entry State

The Policy Engine transitions to the PE_UFP_VDM_Evaluate_Mode_Entry State from either the PE_SRC_Ready or PE_SNK_Ready State for a UFP when:

- A [Structured VDM Enter Mode](#) Command Request is received from the DFP.

On Entry to the PE_UFP_VDM_Evaluate_Mode_Entry State the Policy Engine **shall** Request the Device Policy Manager to evaluate the [Enter Mode](#) Command Request and enter the Mode indicated in the Command Request if the Request is acceptable.

The Policy Engine **shall** transition to the PE_UFP_VDM_Mode_Entry_ACK State when:

- The Device Policy Manager indicates that the Mode has been entered.

The Policy Engine **shall** transition to the PE_UFP_VDM_Mode_Entry_NAK State when:

- The Device Policy Manager indicates that the response to the Mode Request is NAK.

9.2.25.1.2. PE_UFP_VDM_Mode_Entry_ACK State

On entry to the PE_UFP_VDM_Mode_Entry_ACK State the Policy Engine **Shall** send a [Structured VDM Enter Mode](#) ACK Command response.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a UFP when:

- The [Structured VDM Enter Mode](#) ACK Command response has been sent.

9.2.25.1.3. PE_UFP_VDM_Mode_Entry_NAK State

On entry to the PE_UFP_VDM_Mode_Entry_NAK State the Policy Engine **Shall** send a [Structured VDM Enter Mode](#) NAK Command response as indicated by the Device Policy Manager.

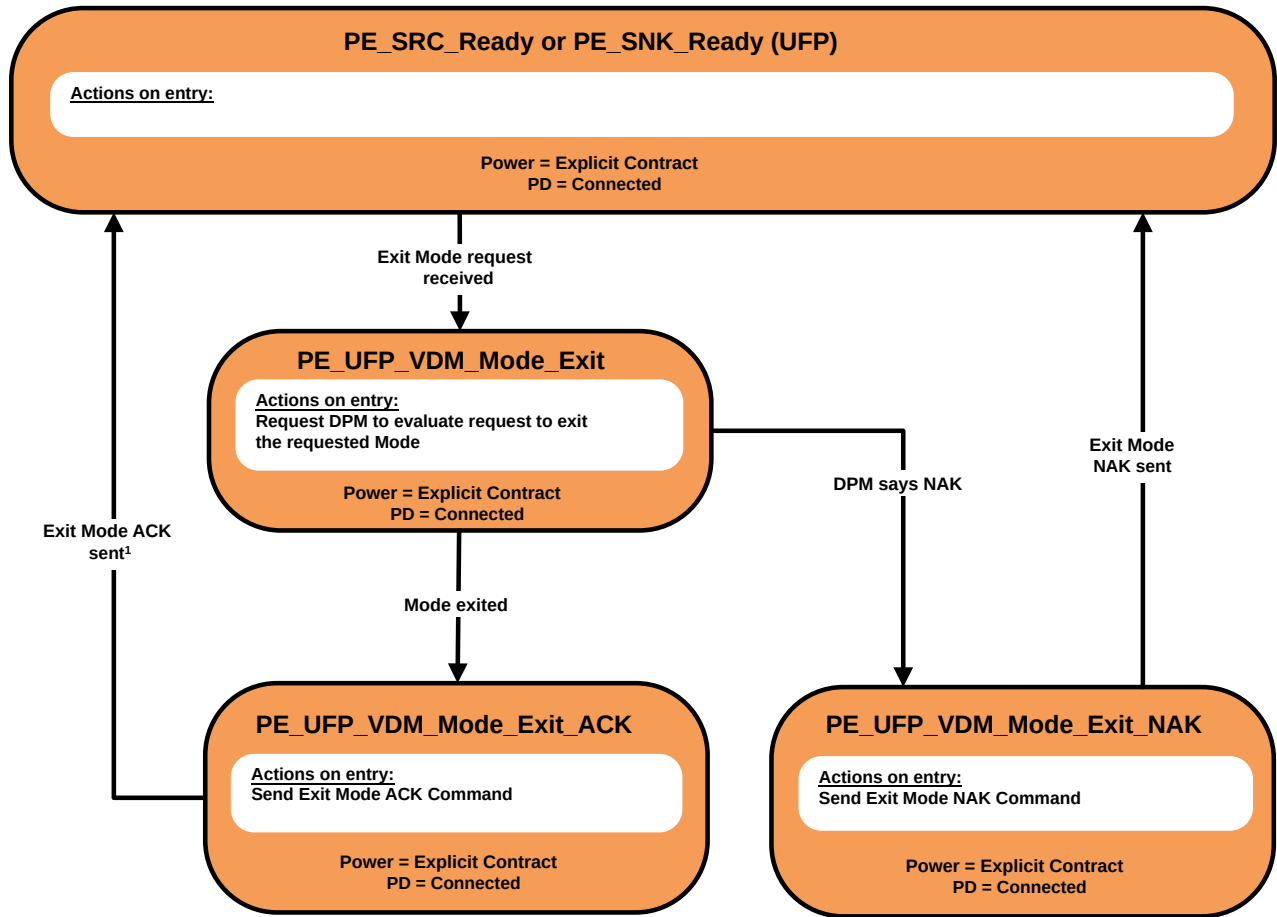
The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a UFP when:

- The [Structured VDM Enter Mode](#) NAK Command response has been sent.

9.2.25.2. UFP Structured VDM Exit Mode State Diagram

[Figure 9.87](#) shows the State diagram for a UFP in response to an [Exit Mode](#) Command.

Figure 9.87. UFP Structured VDM Exit Mode State Diagram



1. The UFP is required to be in USB operation or USB Safe State at this point.

9.2.25.2.1. PE_UFP_VDM_Mode_Exit State

The Policy Engine transitions to the PE_UFP_VDM_Mode_Exit State from either the PE_SRC_Ready or PE_SNK_Ready State for a UFP when:

- A [Structured VDM Exit Mode](#) Command Request is received from the DFP.

On entry to the PE_UFP_VDM_Mode_Exit State the Policy Engine **Shall** Request the Device Policy Manager to exit the Mode indicated in the Command.

The Policy Engine **Shall** transition to the PE_UFP_VDM_Mode_Exit_ACK State when:

- The Device Policy Manager indicates that the Mode has been exited.

The Policy Engine **Shall** transition to the PE_UFP_VDM_Mode_Exit_NAK State when:

- The Device Policy Manager indicates that the Command response to the [Exit Mode](#) Command Request is NAK.

9.2.25.2.2. PE_UFP_VDM_Mode_Exit_ACK State

On entry to the PE_UFP_VDM_Mode_Exit_ACK State the Policy Engine **Shall** send a [Structured VDM Exit Mode](#) ACK Command response.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State for a UFP when:

- The [Structured VDM Exit Mode](#) ACK Command response has been sent.

9.2.25.2.3. PE_UFP_VDM_Mode_Exit_NAK State

On entry to the PE_UFP_VDM_Mode_Exit_NAK State the Policy Engine **Shall** send a [Structured VDM Exit Mode](#) NAK Command response as indicated by the Device Policy Manager.

The Policy Engine **Shall** transition to either the either the PE_SRC_Ready or PE_SNK_Ready State for a UFP when:

- The [Structured VDM Exit Mode](#) NAK Command response has been sent.

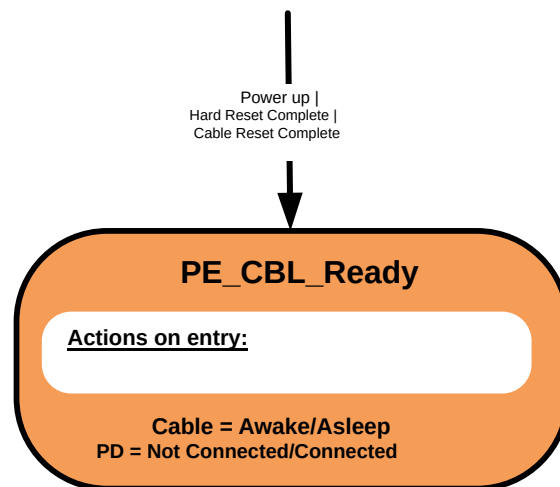
9.2.26. Cable Plug Specific State Diagrams

The State Diagrams in this section **Shall** apply to all Cable Plugs that support Structured VDMs.

9.2.26.1. Cable Plug Cable Ready State Diagram

[Figure 9.88](#) shows the Cable Ready State diagram for a Cable Plug.

Figure 9.88. Cable Ready State Diagram



9.2.26.1.1. PE_CBL_Ready State

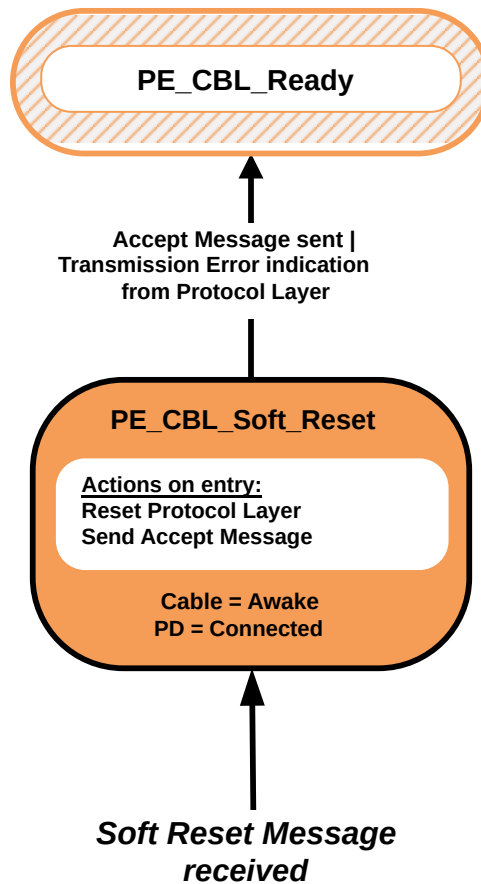
The PE_CBL_Ready State shown in the following sections is the normal operational State for a Cable Plug and where it starts after power up or a Hard/[Cable Reset](#).

9.2.26.2. Soft/Hard/Cable Reset

9.2.26.2.1. Cable Plug Soft Reset State Diagram

Figure 9.89 shows the Cable Plug State diagram for a [Soft Reset](#).

Figure 9.89. Cable Plug Soft Reset State Diagram



9.2.26.2.1.1. PE_CBL_Soft_Reset State

The PE_CBL_[Soft_Reset](#) State **Shall** be entered from any State when a [Soft_Reset Message](#) is received from the Protocol Layer.

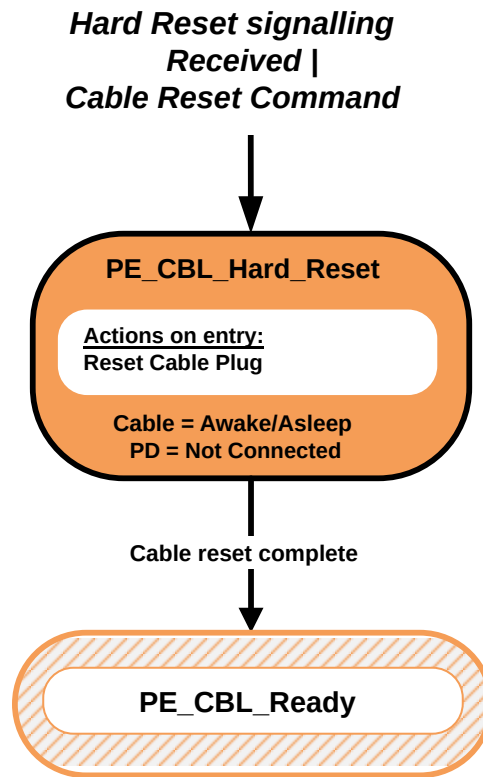
On entry to the PE_CBL_[Soft_Reset](#) State the Policy Engine **Shall** reset the Protocol Layer in the Cable Plug and **Shall** then Request the Protocol Layer to send an [Accept Message](#).

The Policy Engine **Shall** transition to the PE_CBL_Ready State when:

- The [Accept Message](#) has been sent or
- The Protocol Layer indicates that a transmission error has occurred.

9.2.26.2.2. Cable Plug Hard Reset State Diagram

Figure 9.90 shows the Cable Plug State diagram for a [Hard Reset](#).

Figure 9.90. Cable Plug Hard Reset State Diagram

9.2.26.2.2.1. PE_CBL_Hard_Reset State

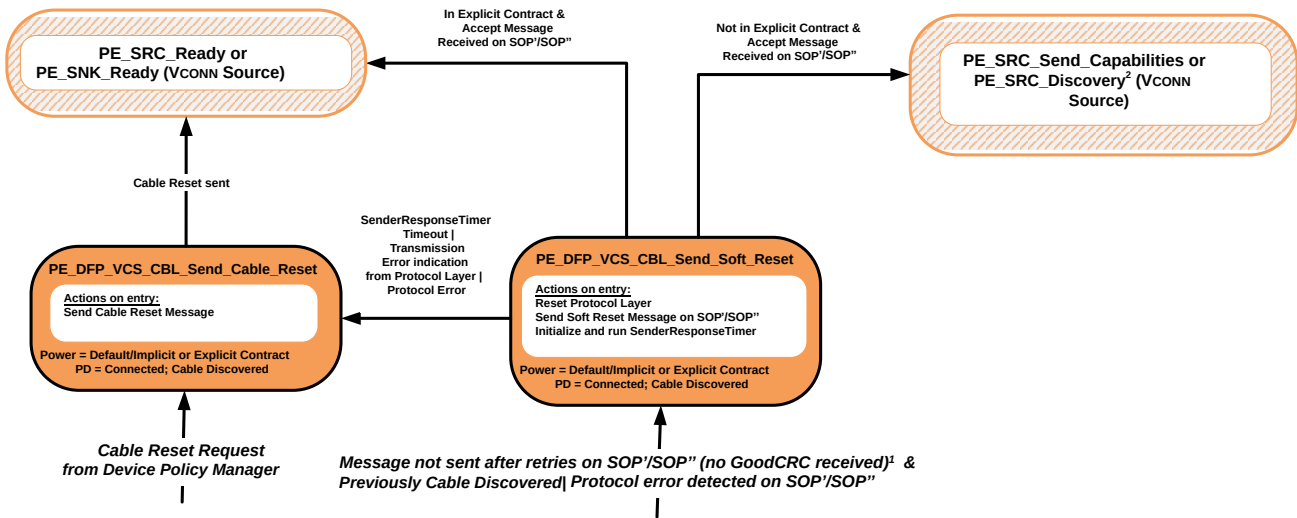
The PE_CBL_Hard_Reset State **Shall** be entered from any State when either [Hard Reset](#) Signaling or [Cable Reset](#) Signaling is detected.

On entry to the PE_CBL_Hard_Reset State the Policy Engine **Shall** reset the Cable Plug (equivalent to a power cycle). The Policy Engine **Shall** transition to the PE_CBL_Ready State when:

- The Cable Plug reset is complete.

9.2.26.2.3. DFP/VCONN Source SOP'/SOP" Soft Reset or Cable Reset of a Cable Plug or VPD State Diagram

This figure shows the State diagram for the Policy Engine in a VCONN Source when performing a [Soft Reset](#) or [Cable Reset](#) of a Cable Plug or VPD on SOP'/SOP". The following sections describe operation in each of the states.

Figure 9.91. DFP/VCONN Source Soft Reset or Cable Reset of a Cable Plug or VPD State Diagram

1. Excludes the [Soft_Reset Message](#) itself.
2. Sink only communicates with the Cable Plug when in an Explicit Contract. If the [Discover Identity](#) Command is being sent at startup, then the Policy Engine will subsequently transition to the PE_SRC_Send_Capabilities State as normal. Otherwise, the Policy Engine will transition to the PE_SRC_Discovery State.

9.2.26.2.3.1. PE_DFP_VCS_CBL_Send_Soft_Reset State

The PE_DFP_VCS_CBL_Send [Soft_Reset](#) State **shall** be entered from any State when a Protocol Error is detected on SOP'/SOP'' by the Protocol Layer (see [Section 7.1.1](#)) or when a Message has not been sent after retries on SOP'/SOP'' while communicating with a Cable Plug/VPD and when there was previous communication with the Cable Plug that did not result in a Transmission Error or whenever the Device Policy Manager directs a [Soft_Reset](#) on SOP'/SOP''.

On entry to the PE_DFP_VCS_CBL_Send [Soft_Reset](#) State the DFP Policy Engine **shall** Request the SOP'/SOP'' Protocol Layer to perform a [Soft_Reset](#), then **shall** send a [Soft_Reset Message](#) on SOP'/SOP'' to the Cable Plug/VPD, and initialize and run the SenderResponseTimer.

The Policy Engine **shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State, depending on the DFP VCONN Source's Power Role, when:

- There is no Explicit Contract in place and
- An [Accept Message](#) has been received on SOP'/SOP''.

The Policy Engine **shall** transition to either the PE_SRC_Send_Capabilities State or PE_SRC_Discovery State, depending on the DFP's VCONN Source's Power Role, when:

- There is an Explicit Contract in place and
- An [Accept Message](#) has been received on SOP'/SOP''.

The Policy Engine **shall** transition to the PE_DFP_VCS_CBL_Send_Cable_Reset State when:

- A SenderResponseTimer timeout occurs
- Or the Protocol Layer indicates that a transmission error has occurred
- Or when a Protocol Error is detected on SOP'/SOP'' by the Protocol Layer.

9.2.26.2.3.2. PE_DFP_VCS_CBL_Send_Cable_Reset State

The PE_DFP_VCS_CBL_Send_Cable_Reset State **Shall** be entered from any State when the Device Policy Manager requests a [Cable Reset](#).

On entry to the PE_DFP_VCS_CBL_Send_Cable_Reset State the DFP Policy Engine **Shall** Request the Protocol Layer to send [Cable Reset](#) Signaling.

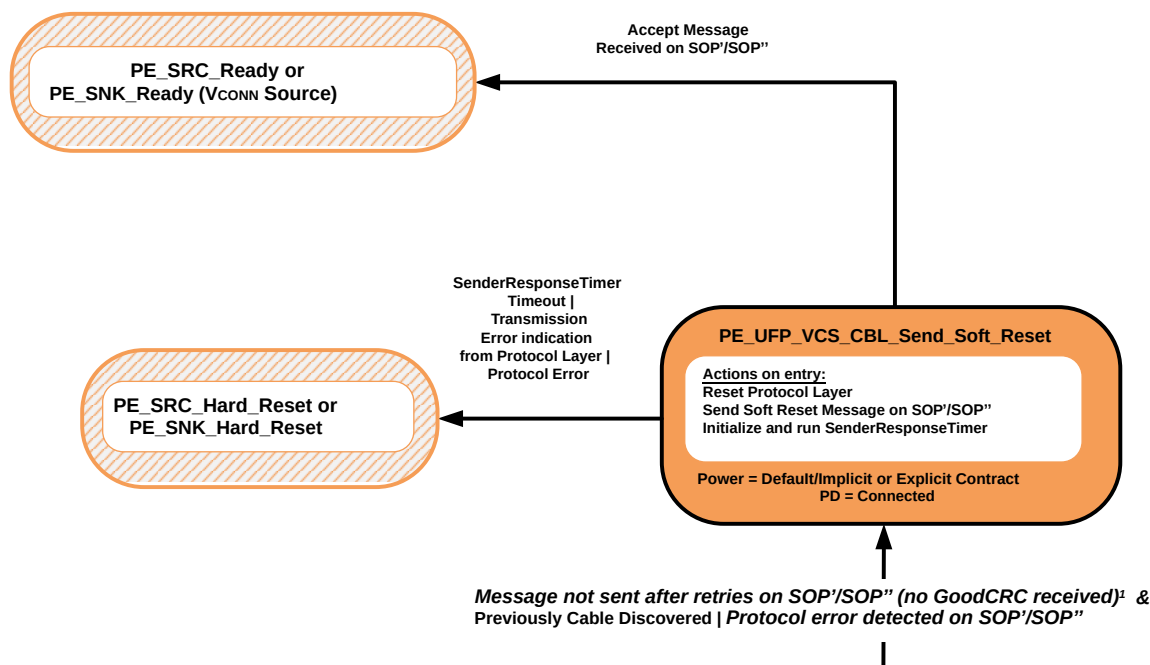
The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State, depending on the VCONN Source's Power Role, when:

- [Cable Reset](#) Signaling has been sent.

9.2.26.2.4. UFP/VCONN Source SOP'/SOP'' Soft Reset of a Cable Plug or VPD State Diagram

This figure shows the State diagram for the UFP Policy Engine in a VCONN Source when performing a [Soft Reset](#) of a Cable Plug or VPD on SOP'/SOP''. The following sections describe operation in each of the states.

Figure 9.92. UFP/VCONN Source Soft Reset of a Cable Plug or VPD State Diagram



1. Excludes the [Soft_Reset Message](#) itself.

9.2.26.2.4.1. PE_UFP_VCS_CBL_Send_Soft_Reset State

The PE_UFP_VCS_CBL_Send_Soft_Reset State **Shall** be entered from any State when a Protocol Error is detected on SOP'/SOP'' by the Protocol Layer (see [Section 7.1.1](#)) or when a Message has not been sent after retries on

SOP'/SOP'' while communicating with a Cable Plug/VPD and when there was previous communication with the Cable Plug that did not result in a Transmission Error or whenever the Device Policy Manager directs a [Soft Reset](#) on SOP'/SOP''.

On entry to the PE_UFP_VCS_CBL_Send_[Soft_Reset](#) State the Policy Engine **Shall** Request the SOP'/SOP'' Protocol Layer to perform a [Soft Reset](#), then **Shall** send a [Soft_Reset Message](#) on SOP'/SOP'' to the Cable Plug, and initialize and run the SenderResponseTimer.

The Policy Engine **Shall** transition to either the PE_SRC_Ready or PE_SNK_Ready State, depending on the UFP VCONN Source's Power Role, when:

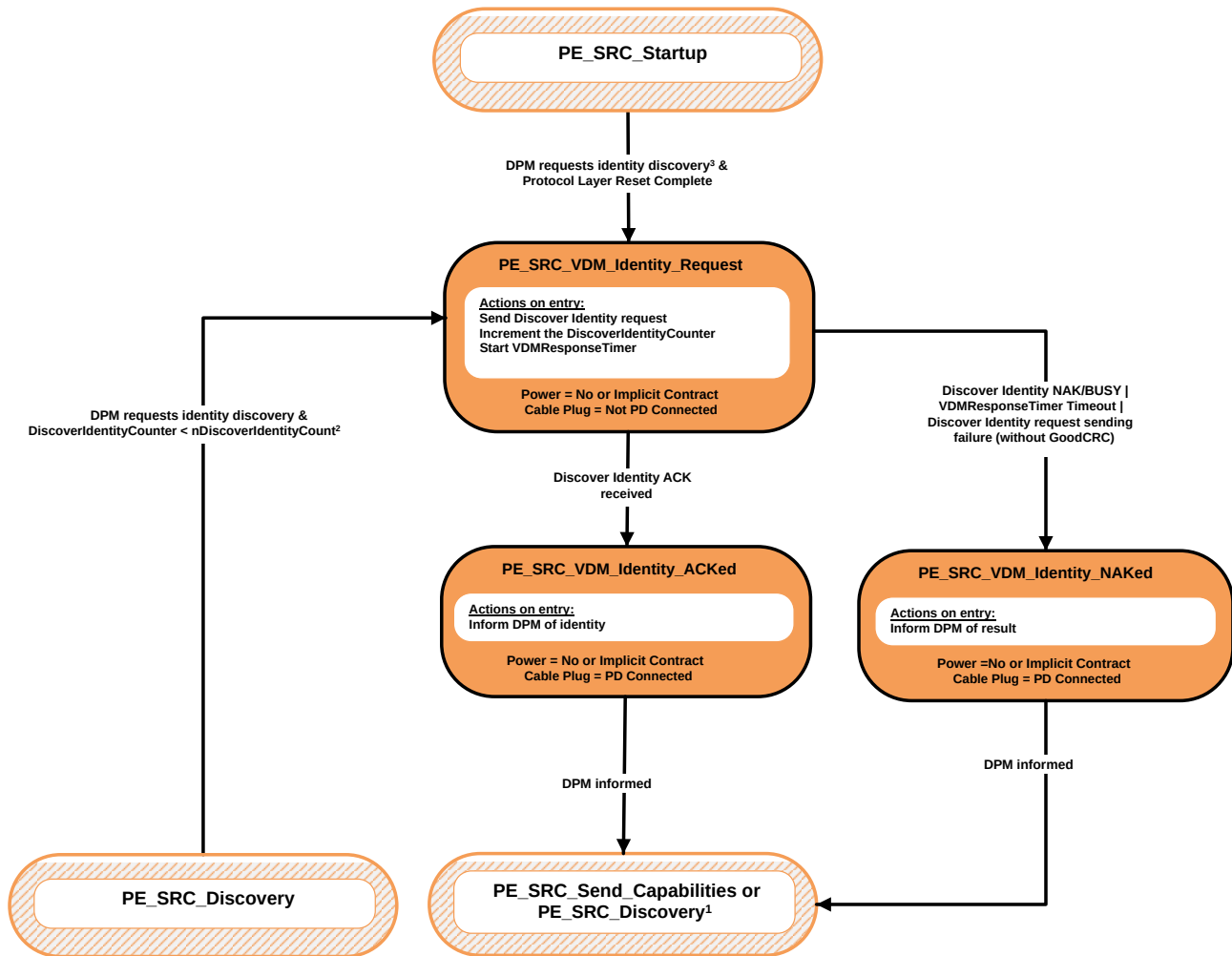
- An [Accept Message](#) has been received on SOP'/SOP''.

The Policy Engine **Shall** transition to either the PE_SRC_Hard_Reset or PE_SNK_Hard_Reset State, depending on the UFP VCONN Source's Power Role, when:

- A SenderResponseTimer timeout occurs
- Or the Protocol Layer indicates that a transmission error has occurred
- Or when a Protocol Error is detected on SOP'/SOP'' by the Protocol Layer.

9.2.26.2.5. Source Startup Structured VDM Discover Identity of a Cable Plug State Diagram

[Figure 9.93](#) shows the State diagram for Source discovery of identity information from a Cable Plug during the startup sequence.

Figure 9.93. Source Startup Structured VDM Discover Identity State Diagram

1. If the [Discover Identity](#) Command is being sent at startup, then the Policy Engine will subsequently transition to the PE_SRC_Send_Capabilities State as normal. Otherwise, the Policy Engine will transition to the PE_SRC_Discovery State.
2. The SourceCapabilityTimer continues to run during the states defined in this diagram even though there has been an exit from the PE_SRC_Discovery State. This ensures that [Source_Capabilities Messages](#) are sent out at a regular rate.
3. The DPM in an EPR Source **Shall** Request the discovery of the identity of the Cable Plug at startup.

9.2.26.2.5.1. PE_SRC_VDM_Identity_Request State

The Policy Engine **Shall** transition to the PE_SRC_VDM_Identity_Request State from the PE_SRC_Startup State when:

- The Device Policy Manager requests the discovery of the identity of the Cable Plug.

The Policy Engine **Shall** transition to the PE_SRC_VDM_Identity_Request State from the PE_SRC_Discovery State when:

- The Device Policy Manager requests the discovery of the identity of the Cable Plug and

- The DiscoverIdentityCounter < [nDiscoverIdentityCount](#).

Even though there has been a transition out of the PE_SRC_Discovery State the SourceCapabilityTimer **Shall** continue to run during the states shown in [Figure 9.93](#) and **Shall Not** be initialized on re-entry to PE_SRC_Discovery.

Note: An EPR Source is required to discover the identity of the Cable Plug prior to entering the First Explicit Contract (see [Section 7.30.1](#))

On entry to the PE_SRC_VDM_Identity_Request State the Policy Engine **Shall** send a [Structured VDM Discover Identity](#) Command Request, **Shall** increment the DiscoverIdentityCounter and **Shall** start the VDMResponseTimer.

The Policy Engine **Shall** transition to the PE_SRC_VDM_Identity_ACKed State when:

- A [Structured VDM Discover Identity](#) ACK Command response is received. The Policy Engine **Shall** transition to the PE_SRC_VDM_Identity_NAKed State when:
 - A [Structured VDM Discover Identity](#) NAK or BUSY Command response is received or
 - The VDMResponseTimer times out or
 - The [Structured VDM Discover Identity](#) Command [Request Message](#) sending fails (no [GoodCRC Message](#) received after retries).

9.2.26.2.5.2. PE_SRC_VDM_Identity_ACKed State

On entry to the PE_SRC_VDM_Identity_ACKed State the Policy Engine **Shall** inform the Device Policy Manager of the Identity information.

The Policy Engine **Shall** transition back to either the PE_SRC_Send_Capabilities or PE_SRC_Discovery State when:

- The Device Policy Manager has been informed.

9.2.26.2.5.3. PE_SRC_VDM_Identity_NAKed State

On entry to the PE_SRC_VDM_Identity_NAKed State the Policy Engine **Shall** inform the Device Policy Manager of the result (NAK, BUSY or timeout).

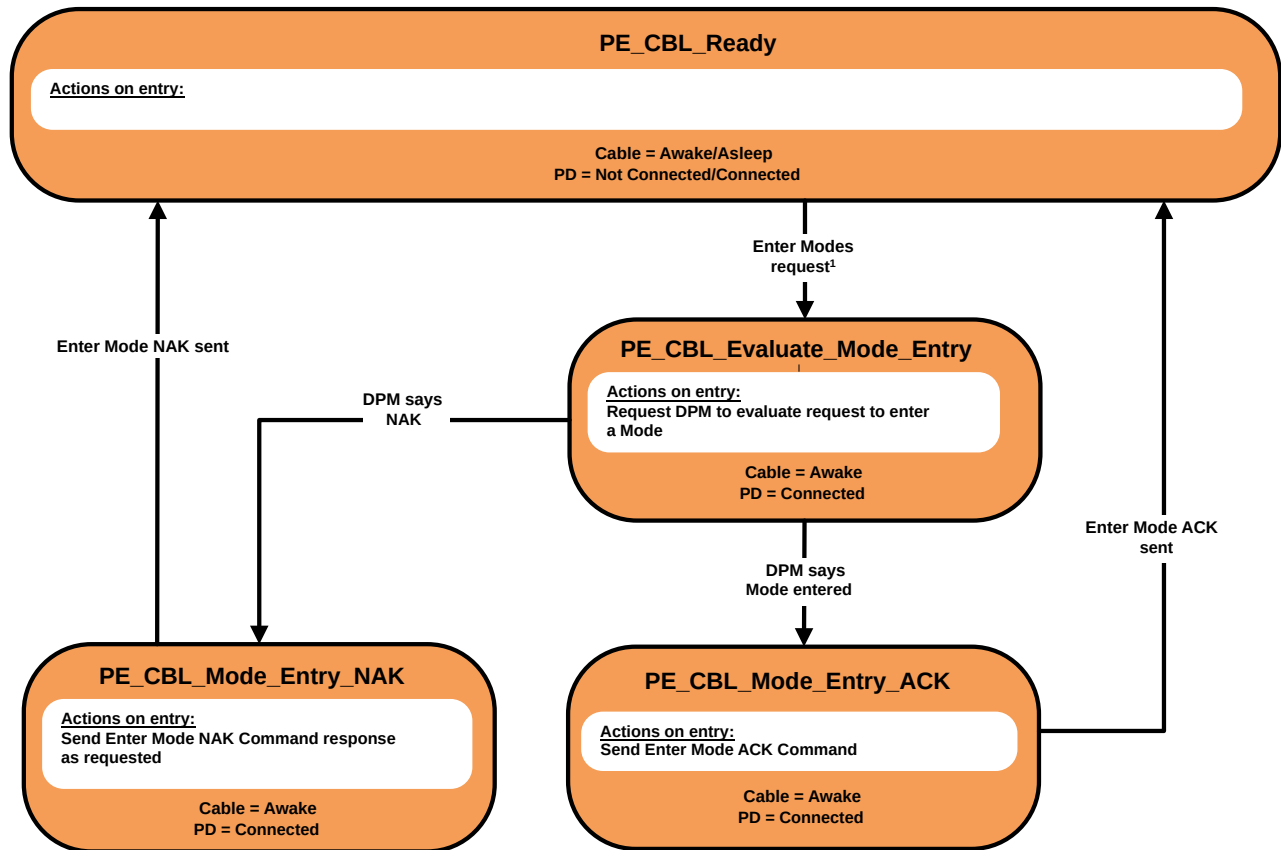
The Policy Engine **Shall** transition back to either the PE_SRC_Send_Capabilities or PE_SRC_Discovery State when:

- The Device Policy Manager has been informed.

9.2.26.2.6. Cable Plug Mode Entry/Exit

9.2.26.2.6.1. Cable Plug Structured VDM Enter Mode State Diagram

[Figure 9.94](#) shows the State diagram for a Cable Plug in response to an [Enter Mode](#) Command.

Figure 9.94. Cable Plug Structured VDM Enter Mode State Diagram

1. The Cable Plug is required to be in USB operation or USB Safe State at this point.

PE_CBL_Evaluate_Mode_Entry State

The Policy Engine transitions to the PE_CBL_Evaluate_Mode_Entry State from the PE_CBL_Ready State when:

- A [Structured VDM Enter Mode](#) Command Request is received from the DFP.

On Entry to the PE_CBL_Evaluate_Mode_Entry State the Policy Engine **Shall** Request the Device Policy Manager to evaluate the [Enter Mode](#) Command Request and enter the Mode indicated in the Command Request if the Request is acceptable.

The Policy Engine **Shall** transition to the PE_CBL_Mode_Entry_ACK State when:

- The Device Policy Manager indicates that the Mode has been entered. The Policy Engine **Shall** transition to the PE_CBL_Mode_Entry_NAK State when:
 - The Device Policy Manager indicates that the response to the Mode Request is NAK.

PE_CBL_Mode_Entry_ACK State

On entry to the PE_CBL_Mode_Entry_ACK State the Policy Engine **Shall** send a [Structured VDM Enter Mode](#) ACK Command response.

The Policy Engine **Shall** transition to the PE_CBL_Ready State when:

- The [Structured VDM Enter Mode](#) ACK Command response has been sent.

PE_CBL_Mode_Entry_NAK State

On entry to the PE_CBL_Mode_Entry_NAK State the Policy Engine **Shall** send a [Structured VDM Enter Mode](#) NAK Command response as indicated by the Device Policy Manager.

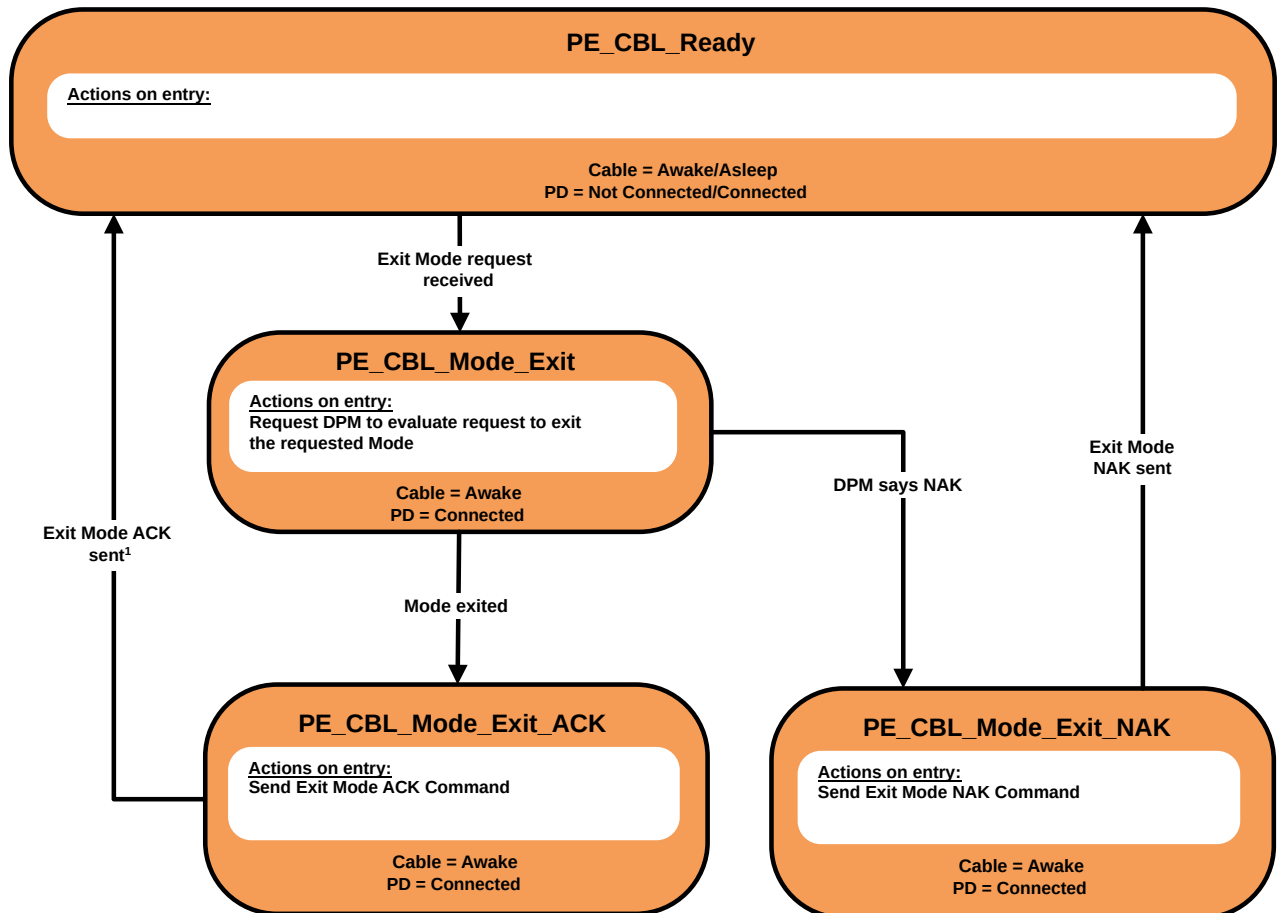
The Policy Engine **Shall** transition to the PE_CBL_Ready State when:

- The [Structured VDM Enter Mode](#) NAK Command response has been sent.

9.2.26.2.6.2. Cable Plug Structured VDM Exit Mode State Diagram

- [Figure 9.95](#) shows the State diagram for a Cable Plug in response to an [Exit Mode](#) Command.

Figure 9.95. Cable Plug Structured VDM Exit Mode State Diagram



1. The Cable Plug is required to be in USB operation or USB Safe State at this point.

PE_CBL_Mode_Exit State

The Policy Engine transitions to the PE_CBL_Mode_Exit State from the PE_CBL_Ready State when:

- A [Structured VDM Exit Mode](#) Command Request is received from the DFP.

On entry to the PE_CBL_Mode_Exit State the Policy Engine **Shall** Request the Device Policy Manager to exit the Mode indicated in the Command.

The Policy Engine **Shall** transition to the PE_CBL_Mode_Exit_ACK State when:

- The Device Policy Manager indicates that the Mode has been exited. The Policy Engine **Shall** transition to the PE_CBL_Mode_Exit_NAK State when:
 - The Device Policy Manager indicates that the Command response to the [Exit Mode](#) Command Request is NAK.

PE_CBL_Mode_Exit_ACK State

On entry to the PE_CBL_Mode_Exit_ACK State the Policy Engine **Shall** send a [Structured VDM Exit Mode](#) ACK Command response.

The Policy Engine **Shall** transition to the PE_CBL_Ready State when:

- The [Structured VDM Exit Mode](#) ACK Command response has been sent.

PE_CBL_Mode_Exit_NAK State

On entry to the PE_CBL_Mode_Exit_NAK State the Policy Engine **Shall** send a [Structured VDM Exit Mode](#) NAK Command response as indicated by the Device Policy Manager.

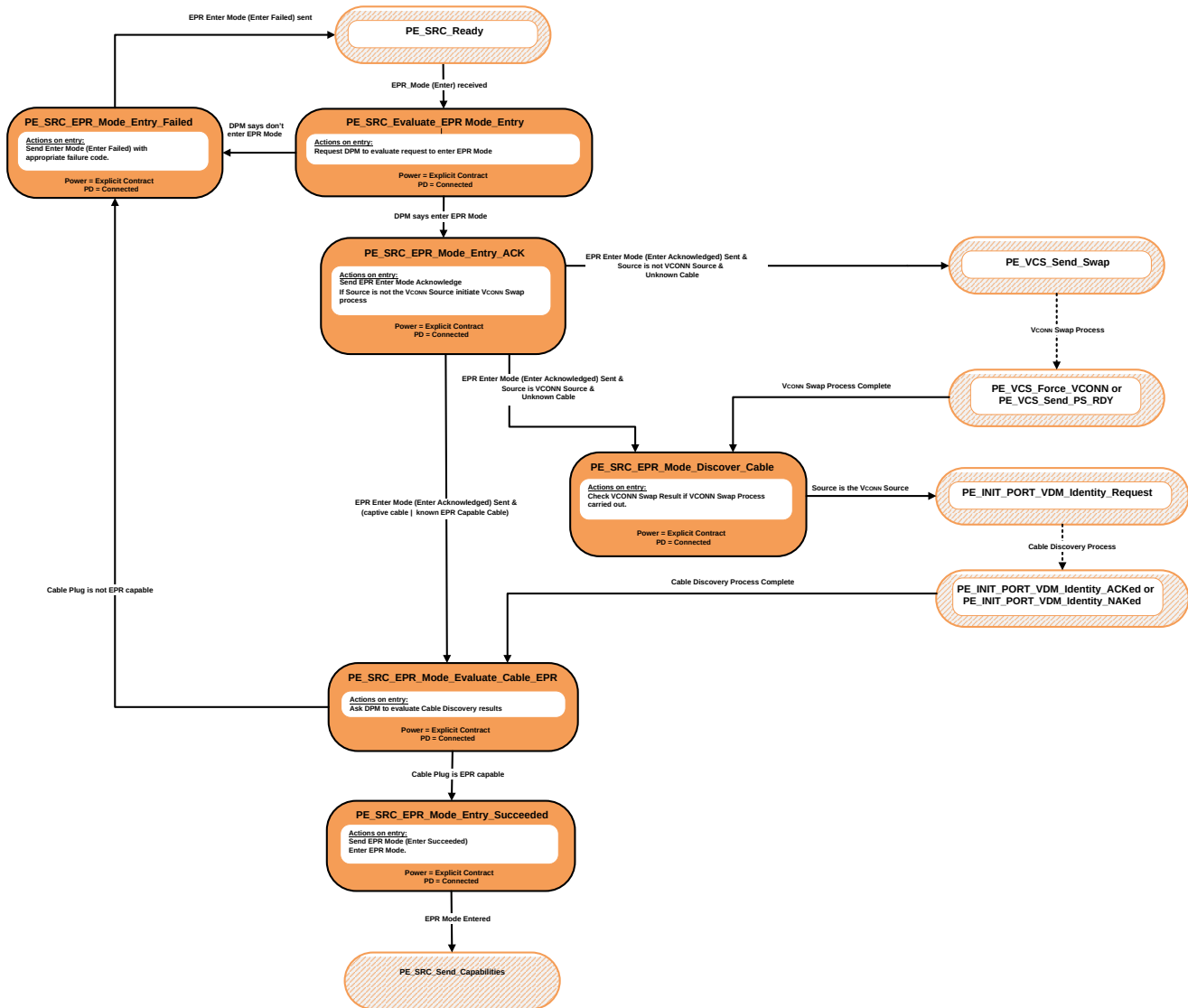
The Policy Engine **Shall** transition to the PE_CBL_Ready State when:

- The [Structured VDM Exit Mode](#) NAK Command response has been sent.

9.2.26.3. EPR Mode State Diagrams**9.2.26.3.1. Source EPR Mode Entry State Diagram**

[Figure 9.96](#) shows the State diagram for an EPR Source in response to an [EPR_Mode Message](#).

Figure 9.96. Source EPR Mode Entry State Diagram



9.2.26.3.1.1. PE_SRC_Evaluate_EPR_Mode_Entry State

The Policy Engine transitions to the PE_SRC_Evaluate_EPR_Mode_Entry State from the PE_SRC_Ready State when:

- An [EPR_Mode](#) (Enter) Message is received from the Sink.

On Entry to the PE_SRC_Evaluate_EPR_Mode_Entry State the Policy Engine **Shall** Request the Device Policy Manager to evaluate the [EPR_Mode](#) (Enter) Message.

The Policy Engine **Shall** transition to the PE_SRC_EPR_Mode_Entry_Ack State when:

- The Device Policy Manager indicates that [EPR_Mode](#) can be entered.

The Policy Engine **Shall** transition to the PE_SRC_EPR_Mode_Entry_Failed State when:

- The Device Policy Manager indicates that the [EPR_Mode](#) is not to be entered.

9.2.26.3.1.2. PE_SRC_EPR_Mode_Entry_Ack State

On entry to the PE_SRC_EPR_Mode_Entry_Ack State the Policy Engine **Shall** send a [EPR_Mode](#) (Enter Acknowledged) Message.

The Policy Engine **Shall** transition to the PE_SRC_EPR_Mode_Evaluate_Cable_EPR State when:

- The [EPR_Mode](#) (Enter Acknowledged) Message has been sent and
- The Source is not the VCONN Source and
- The cable is a captive cable or a known EPR Cable.

The Policy Engine **Shall** transition to the PE_VCS_Send_Swap State when:

- The [EPR_Mode](#) (Enter Acknowledged) Message has been sent and
- The Source is not the VCONN Source and
- The cable is unknown.

The Policy Engine **Shall** transition to the PE_SRC_EPR_Mode_Discover_Cable State when:

- The [EPR_Mode](#) (Enter Acknowledged) Message has been sent and
- The Source is the VCONN Source and
- The cable is unknown.

9.2.26.3.1.3. PE_SRC_EPR_Mode_Discover_Cable State

The Policy Engine transitions to the PE_SRC_EPR_Mode_Discover_Cable State from the PE_VCS_Force_VCONN State or PE_VCS_Send_Ps_Rdy State when:

- A Source initiated [VCONN Swap](#) process has completed.

The Policy Engine **Shall** transition to the PE_INIT_Port_VDM_Identity_Request State in order to perform Cable Plug discovery when:

- The Source is the VCONN Source.

The Policy Engine **Shall** transition to the PE_SRC_EPR_Mode_Entry_Failed State when:

- The [VCONN Swap](#) process failed (the Source is not the VCONN Source).

9.2.26.3.1.4. PE_SRC_EPR_Mode_Evaluate_Cable_EPR State

In the PE_SRC_EPR_Mode_Evaluate_Cable_EPR State the Policy Engine requests the DPM to evaluate the Cable Discovery results.

The Policy Engine **Shall** transition to the PE_SRC_EPR_Mode_Entry_Succeeded State when:

- The Cable Plug is capable of [EPR Mode](#).

The Policy Engine **Shall** transition to the PE_SRC_EPR_Mode_Entry_Failed State when:

- The Cable Plug is not capable of [EPR Mode](#).

9.2.26.3.1.5. PE_SRC_EPR_Mode_Entry_Succeeded State

On entry to the PE_SRC_EPR_Mode_Entry_Succeeded State the Policy Engine **Shall** send a [EPR_Mode](#) (Enter Succeeded) Message and enter [EPR Mode](#).

The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State when:

- [EPR Mode](#) has been entered.

9.2.26.3.1.6. PE_SRC_EPR_Mode_Entry_Failed State

On entry to the PE_SRC_EPR_Mode_Entry_Failed State the Policy Engine **Shall** send a [EPR_Mode](#) (Enter Failed) Message.

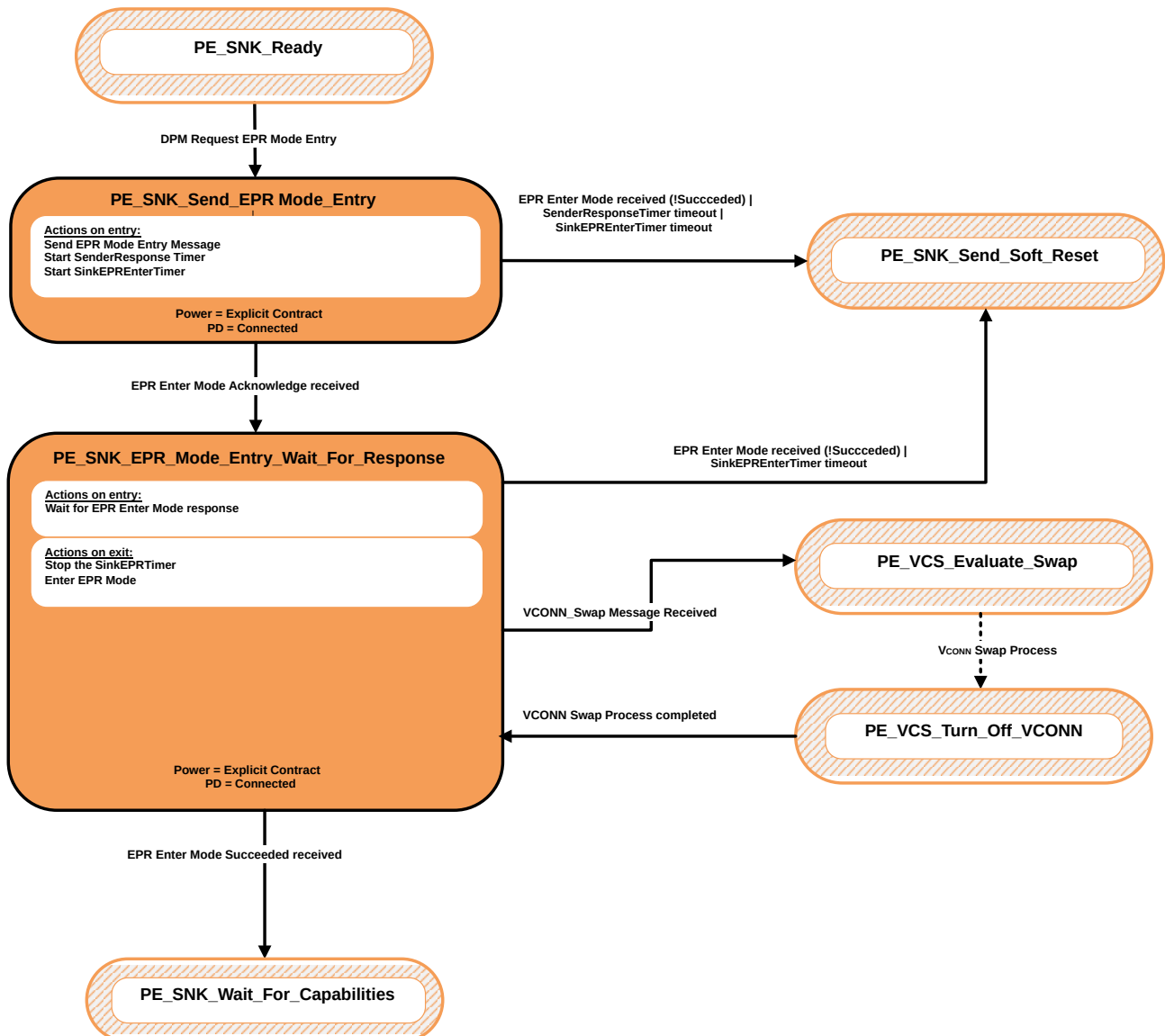
The Policy Engine **Shall** transition to the PE_SRC_Ready State when:

- The [EPR_Mode](#) (Enter Failed) Message has been sent.

9.2.26.3.2. Sink EPR Mode Entry State Diagram

[Figure 9.97](#) shows the State diagram for an EPR Sink initiating the [EPR Mode](#) Entry process.

Figure 9.97. Sink EPR Mode Entry State Diagram



9.2.26.3.2.1. PE_SNK_Send_EPR_Mode_Entry State

The Policy Engine transitions to the PE_SNK_Send_EPR_Mode_Entry State from the PE_SNK_Ready State when:

- The DPM requests entry into [EPR Mode](#).

On Entry to the PE_SNK_Send_EPR_Mode_Entry State the Policy Engine **Shall** send an [EPR_Mode](#) (Enter) Message and starts the SenderResponseTimer and the SinkEPREnterTimer.

Note: The SinkEPREnterTimer **Shall** continue to run in every State until it is stopped or times out. The Policy Engine **Shall** transition to the PE_SNK_EPR_Mode_Wait_For_Response State when:

- An [EPR_Mode](#) (Enter Acknowledge) Message is received.

The Policy Engine **Shall** transition to the PE_SNK_Send_Soft_Reset State when:

- An [EPR_Mode Message](#) is received which is not Enter Succeeded or
- The SenderResponseTimer times out or
- The SinkEPREnterTimer times out.

9.2.26.3.2.2. PE_SNK_EPR_Mode_Wait_For_Response State

In the State the Policy Engine waits for a confirmation that the [EPR_Mode](#) entry Request has succeeded. On exit from the PE_SNK_EPR_Mode_Wait_For_Response State the Policy Engine **Shall** stop the SinkEPREnterTimer and enter [EPR_Mode](#).

The Policy Engine **Shall** transition to the PE_SNK_Send_Soft_Reset State when:

- An [EPR_Mode Message](#) is received which is not Enter Succeeded or
- The SinkEPREnterTimer times out.

The Policy Engine **Shall** transition to the PE_VCS_Evaluate_Swap State when:

- A [VCONN_Swap Message](#) is received.

The Policy Engine **Shall** transition back from the PE_VCS_Turn_Off_VCONN State to the PE_SNK_EPR_Mode_Wait_For_Response State when:

- The [VCONN Swap](#) process has completed.

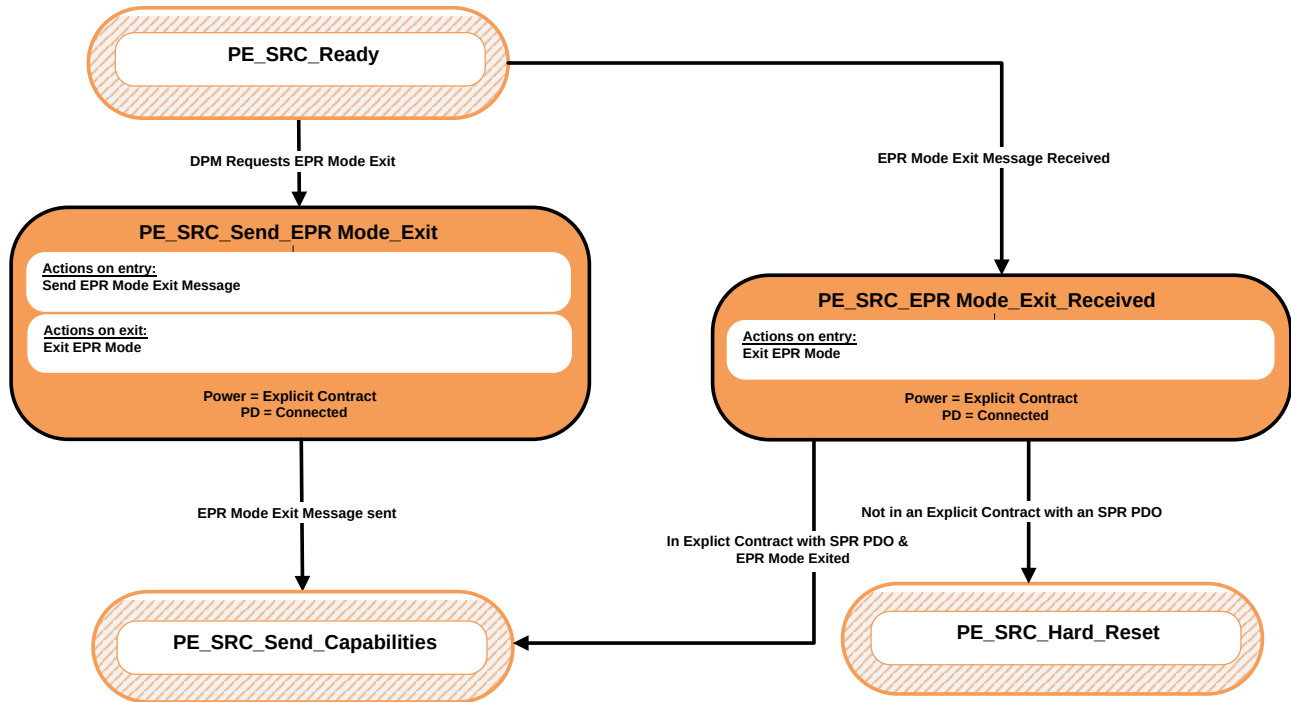
The Policy Engine **Shall** transition to the PE_SNK_Wait_for_Capabilities State when:

- An [EPR_Mode](#) (Enter Succeeded) Message has been received.

9.2.26.3.3. Source EPR Mode Exit State Diagram

[Figure 9.98](#) shows the State diagram for an EPR Source initiating the [EPR_Mode](#) Exit process.

Figure 9.98. Source EPR Mode Exit State Diagram



9.2.26.3.3.1. PE_SRC_Send_EPR_Mode_Exit State

The Policy Engine transitions to the PE_SRC_Send_EPR_Mode_Exit State from the PE_SRC_Ready State when:

- The DPM requests exit from [EPR Mode](#).

On Entry to the PE_SRC_Send_EPR_Mode_Exit State the Policy Engine **Shall** send an [EPR_Mode](#) (Exit) Message. On Exit from the PE_SRC_Send_EPR_Mode_Exit State the Policy Engine **Shall** exit [EPR Mode](#).

The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State when:

- The [EPR_Mode](#) (Exit) Message has been sent.

9.2.26.3.3.2. PE_SRC_EPR_Mode_Exit_Received State

The Policy Engine transitions to the PE_SRC_EPR_Mode_Exit_Received State from the PE_SRC_Ready State when:

- An [EPR_Mode](#) (Exit) Message is received.

On Entry to the PE_SRC_EPR_Mode_Exit_Received State the Policy Engine **Shall** exit [EPR Mode](#). The Policy Engine **Shall** transition to the PE_SRC_Send_Capabilities State when:

- In an Explicit Contract with an SPR PDO or APDO and
- [EPR Mode](#) has been exited.

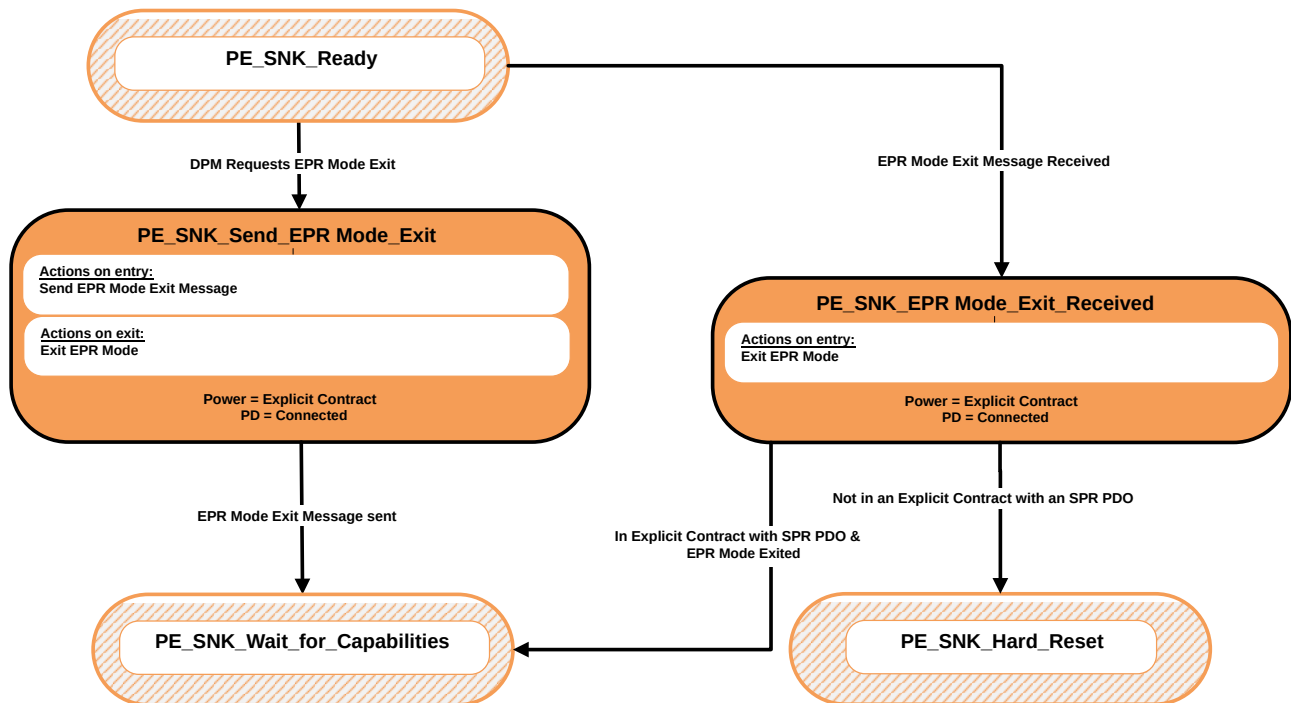
The Policy Engine **Shall** transition to the PE_SRC_Hard_Reset State when:

- Not in an Explicit Contract with an SPR PDO or APDO.

9.2.26.3.4. Sink EPR Mode Exit State Diagram

Figure 9.99 shows the State diagram for an EPR Sink initiating the [EPR Mode](#) Exit process.

Figure 9.99. Sink EPR Mode Exit State Diagram



9.2.26.3.4.1. PE_SNK_Send_EPR_Mode_Exit State

The Policy Engine transitions to the PE_SNK_Send_EPR_Mode_Exit State from the PE_SNK_Ready State when:

- The DPM requests exit from [EPR Mode](#).

On Entry to the PE_SNK_Send_EPR_Mode_Exit State the Policy Engine **Shall** send an [EPR_Mode](#) (Exit) Message. On Exit from the PE_SNK_Send_EPR_Mode_Exit State the Policy Engine **Shall** exit [EPR Mode](#).

The Policy Engine **Shall** transition to the PE_SNK_Wait_for_Capabilities State when:

- The [EPR_Mode](#) (Exit) Message has been sent.

9.2.26.3.4.2. PE_SNK_EPR_Mode_Exit_Received State

The Policy Engine transitions to the PE_SNK_EPR_Mode_Exit_Received State from the PE_SNK_Ready State when:

- An [EPR_Mode](#) (Exit) Message is received.

On Entry to the PE_SNK_EPR_Mode_Exit_Received State the Policy Engine **Shall** exit [EPR Mode](#). The Policy Engine **Shall** transition to the PE_SNK_Wait_for_Capabilities State when:

- In an Explicit Contract with an SPR PDO or APDO and

- [EPR Mode](#) has been exited.

The Policy Engine **Shall** transition to the PE_SNK_Hard_Reset State when:

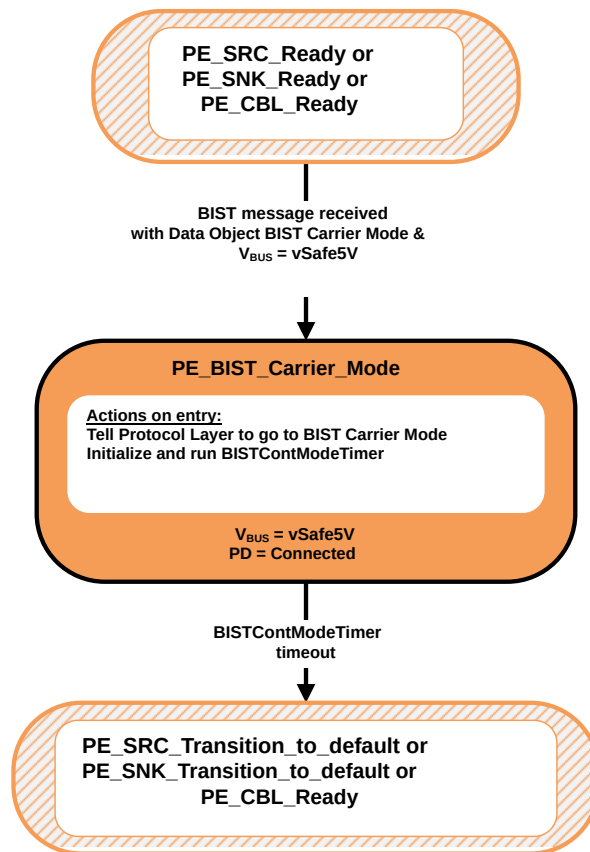
- Not in an Explicit Contract with an SPR PDO or APDO.

9.2.26.4. BIST State diagrams

9.2.26.4.1. BIST Carrier Mode State Diagram

[Figure 9.100](#) shows the State diagram required by a UUT, which can be either a Source, Sink or Cable Plug, when operating in BIST Carrier Mode. Transitions **Shall** be from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready states.

Figure 9.100. BIST Carrier Mode State Diagram



9.2.26.4.1.1. PE_BIST_Carrier_Mode State

The Source, Sink or Cable Plug **Shall** enter the PE_BIST_Carrier_Mode State from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- A [BIST Message](#) is received with a BIST Carrier Mode BIST Data Object and
- V_{BUS} is at [vSafe5V](#).

On entry to the PE_BIST_Carrier_Mode State the Policy Engine **Shall** tell the Protocol Layer to go to BIST Carrier Mode (see [Section 5.4.1](#)) and **Shall** initialize and run the BISTContModeTimer.

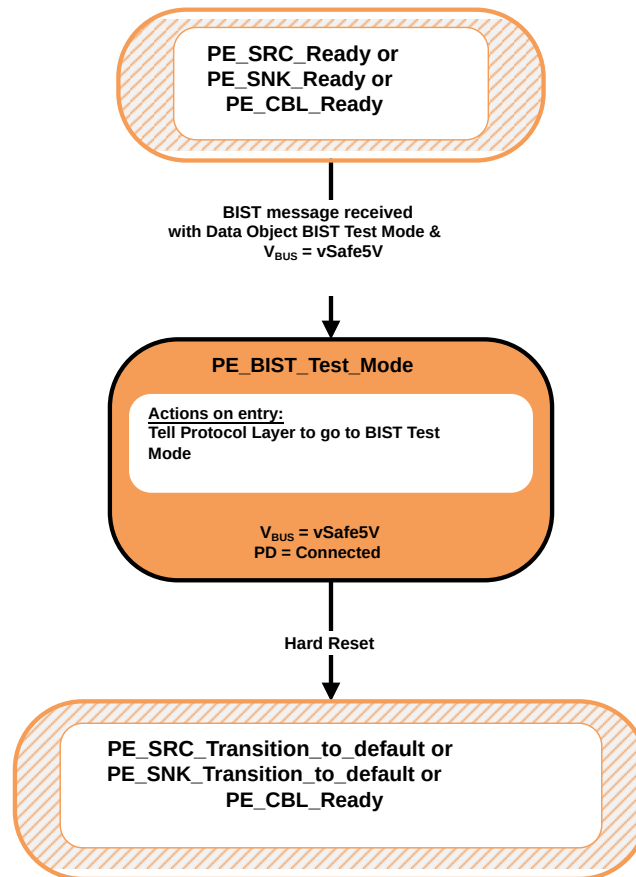
The Policy Engine **Shall** transition to either the PE_SRC_Transition_to_default State, PE_SNK_Transition_to_default State or PE_CBL_Ready State (as appropriate) when:

- The BISTContModeTimer times out.

9.2.26.4.2. BIST Test Data Mode State Diagram

[Figure 9.101](#) shows the State diagram required by a UUT, which can be either a Source, Sink, or Cable Plug, when operating in BIST Test Data Mode. Transitions **Shall** be from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready states.

Figure 9.101. BIST Test Data Mode State Diagram



9.2.26.4.2.1. PE_BIST_Test_Mode State

The Source, Sink or Cable Plug **Shall** enter the PE_BIST_Test_Mode State from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready State when:

- A [BIST Message](#) is received with a BIST Test Data BIST Data Object and
- VBUS is at [vSafe5V](#).

On entry to the PE_BIST_Test_Mode State the Policy Engine **Shall** tell the Protocol Layer to go into BIST Test Data Mode where it sends no further Messages except for [GoodCRC Messages](#) in response to received Messages (see [Section 6.4.3](#)).

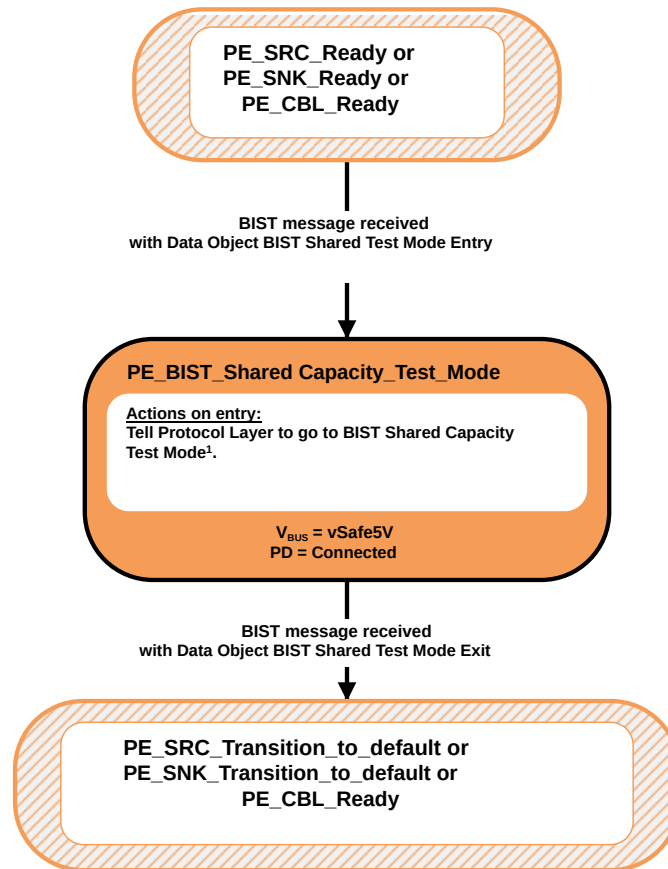
The Policy Engine **Shall** transition to either the PE_SRC_Transition_to_default State, PE_SNK_Transition_to_default State or PE_CBL_Ready State (as appropriate) when:

- A [Hard Reset](#) occurs.

9.2.26.4.3. BIST Shared Capacity Test Mode State Diagram

[Figure 9.102](#) shows the State diagram required by a UUT, which can be either a Source, Sink, or Cable Plug, when operating in BIST Shared Capacity Test Mode. Transitions **Shall** be from either the PE_SRC_Ready, PE_SNK_Ready or PE_CBL_Ready states.

Figure 9.102. BIST Shared Capacity Test Mode State Diagram



1. The UUT **Shall** exit BIST Shared Capacity Test Mode when It is powered off. The UUT **Shall** remain in BIST Shared Capacity Test Mode for any PD event (except when a BIST Shared Test Mode Exit BIST Data Object, is received); specifically the UUT **Shall** remain in BIST Shared Capacity Test Mode when any of the following PD events occurs: [Hard Reset](#), [Cable Reset](#), [Soft Reset](#), [Data Role Swap](#), [Power Role Swap](#), [Fast Role Swap](#), [VCONN Swap](#). The UUT **May** leave test Mode if the Tester makes a Request that exceeds the Capabilities of the UUT.

9.2.26.4.3.1. PE_BIST_Shared_Capacity_Test_Mode State

The Source, Sink or Cable Plug **Shall** enter the PE_BIST_Shared_Capacity_Test_Mode state when:

- A [BIST Message](#) is received with a BIST Shared Test Mode Entry BIST Data Object and

- VBUS is at [vSafe5V](#).

On entry to the PE_BIST_Shared_Capacity_Test_Mode State the Policy Engine **Shall** tell the Protocol Layer to go to BIST Shared Capacity Test Mode (see [Section 6.4.3](#)).

The Policy Engine **Shall** transition to either the PE_SRC_Transition_to_default State, PE_SNK_Transition_to_default State or PE_CBL_Ready State (as appropriate) when:

- A [BIST Message](#) is received with a BIST Shared Test Mode Exit BIST Data Object.

9.2.26.5. USB Type-C Referenced States

This section contains states cross-referenced from the [USB-C] specification.

9.2.26.5.1. ErrorRecovery State

The ErrorRecovery State is used to electronically disconnect Port Partners using the USB Type-C connector. The ErrorRecovery State **Shall** be entered when there are errors on USB Type-C Ports which cannot be recovered by [Hard Reset](#). The ErrorRecovery State **Shall** map to USB Type-C ErrorRecovery State operation as defined in the [USB-C] specification. Bus powered Sinks **Shall Not** be required to meet this requirement as removal of their power will serve the same purpose.

On entry to the ErrorRecovery State the Explicit Contract and PD Connection **Shall** be ended.

On exit from the ErrorRecovery State a new Explicit Contract **Should** be established once the Port Partners have re-Connected over the CC wire.

Chapter 10. Fast Role Swap (FRS)

10.1. Introduction

A [Fast Role Swap](#) (FRS) is like a [Power Role Swap](#) but is done much faster and without initial Message exchanges to start the Power Role process. FRS is designed for use by charge-through accessories. A Charge-Through Accessory is a USB PD-capable DRP Hub or Device that meets all the following:

- Capable of receiving power from an External Power Supply.
- Capable of receiving power from an upstream Host (e.g., notebook, tablet, phone).
- Capable of supplying power to the upstream Host when powered by an external supply.
- Powers its internal circuitry from whichever power Source is active.
- Capable of powering any downstream ports from whichever power Source is active.

For a Charge-Through Accessory to be an FRS-capable Charge-Through Accessory, it must also meet the following:

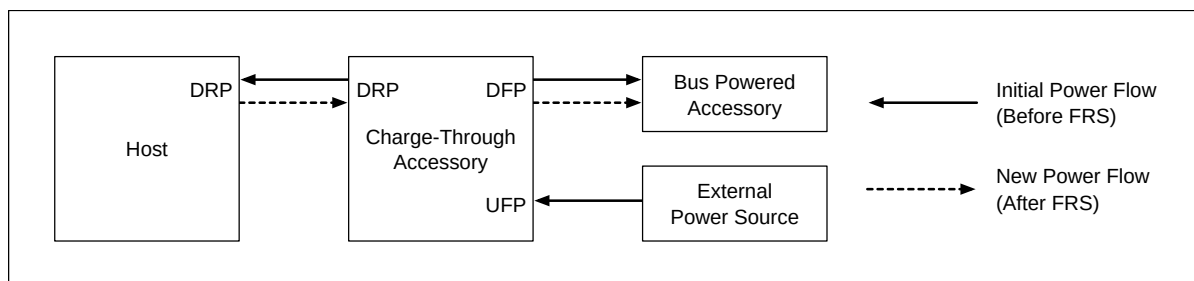
- Capable of powering its internal circuitry and downstream ports when receiving 5V from an upstream Host at 15W or less. If higher voltage or more current is required to complete the swap and function, then FRS will not work.

When an FRS-capable Charge-Through Accessory is supplying power to the upstream Host, the loss of the External Supply triggers the FRS mechanism. If the upstream Host also supports and is initialized for FRS, an FRS-capable Charge-Through Accessory will quickly swap power roles with the Host, causing the Host to become the power Source, without disrupting ongoing data connections or Device functionality. This transition relies on temporary internal power storage within the accessory to maintain power within the accessory and continue providing power to any downstream ports during the swap.

FRS ensures power continuity by enabling the Host to apply [vSafe5V](#) to VBUS when it detects VBUS drooping to or below [vSafe5V](#) after receiving a valid [Fast Role Swap](#) signal. The [Fast Role Swap AMS](#) (Atomic Messaging Sequence) is used in parallel to the power swap on VBUS to reestablish correct Source/Sink power roles and to properly configure the USB-C Rp/Rd terminations.

[Figure 10.1](#) shows an example power flow direction before and after an FRS. In this example, an external power Source is powering a Host and a bus-powered accessory through the Charge-Through Accessory. When external power is lost, the FRS event occurs, and the Charge-Through Accessory and the bus-powered accessory are then powered by the Host.

Figure 10.1. Power Flow Before and After an FRS



10.2. FRS Initialization

Prior to an FRS event, the accessory providing power to an upstream Host is the Initial Source and the upstream Host receiving power is the Initial Sink.

The Initial Sink **Shall** only be initialized for FRS and respond to a [Fast Role Swap](#) signal when the following conditions are true:

1. An Explicit Contract is established between the Initial Source and the Initial Sink.
2. Initial Sink has sent a [Get Sink Cap Message](#) and received the Sink Capabilities from the Initial Source with at least one of the [Fast Role Swap](#) bits set in the 5V Fixed Supply PDO.
3. The Initial Sink is capable of sourcing the current (iNewFrsSnk) indicated by the Initial Source in the [Fast Role Swap](#) bits of the [Sink Capabilities Message](#).

Rules for the Initial Sink FRS initialization are:

- The Initial Sink **Should** check for condition 2 as soon as practical.
- When either condition 1 or 2 is not met, the Initial Sink **Shall** not initialize for FRS and **Shall** not respond to an FRS signal.
- When condition 1 or 2 is met but condition 3 is not met, the Initial Sink **May** initialize for FRS.
- When all three conditions are met, the Initial Sink **Should** initialize for FRS.

When initialized for FRS, the Initial Sink is ready to respond to an FRS signal. At initialization:

- the Initial Sink **Shall** disable Type-C disconnect detection and **Shall** become the VCONN Source if it is required to support VCONN otherwise (i.e., USB 3, DisplayPort Alt Mode, etc.). This avoids requiring a [VCONN Swap](#) during the FRS event.
- If the Initial Sink would not be required to Source VCONN as a Source normally (e.g., USB2 only), then it is not required to perform a [VCONN Swap](#).

Note: The Initial Source is unaware whether the Initial Sink is initialized for FRS or not and will attempt to signal FRS on external power loss regardless of the Initial Sink's initialization.

10.3. FRS Sequence

[Figure 10.2](#), [Figure 10.3](#), and [Figure 10.4](#) illustrate the FRS sequence. [Figure 10.2](#) shows the Signaling and response activity diagrams of an FRS. [Figure 10.3](#) and [Figure 10.4](#) show the relative timing diagrams for a slow/long FRS event and a fast/short FRS event. Each figure shows a corresponding numbered event that matches between the activity diagrams, the timing diagrams, and the numbered sequence in the text following the figures.

The Initial Source follows the 'Signal FRS' activity diagram shown in [Figure 10.2](#) when external power loss is detected and the Initial Sink follows the 'FRS Signal Received' activity diagram when it detects the FRS Signal. To swap power quickly, without waiting for Message exchanges, the power activity happens in parallel to, and generally faster than, the messaging on CC. To show this, each activity diagram is broken into swim lanes for Power/Control and CC Signaling and shows both sequential activity paths and parallel paths. Due to the parallel nature of activity, the numbered steps aren't always sequential through the diagram and text.

Figure 10.2. FRS Signaling and Receiving Activity Diagrams

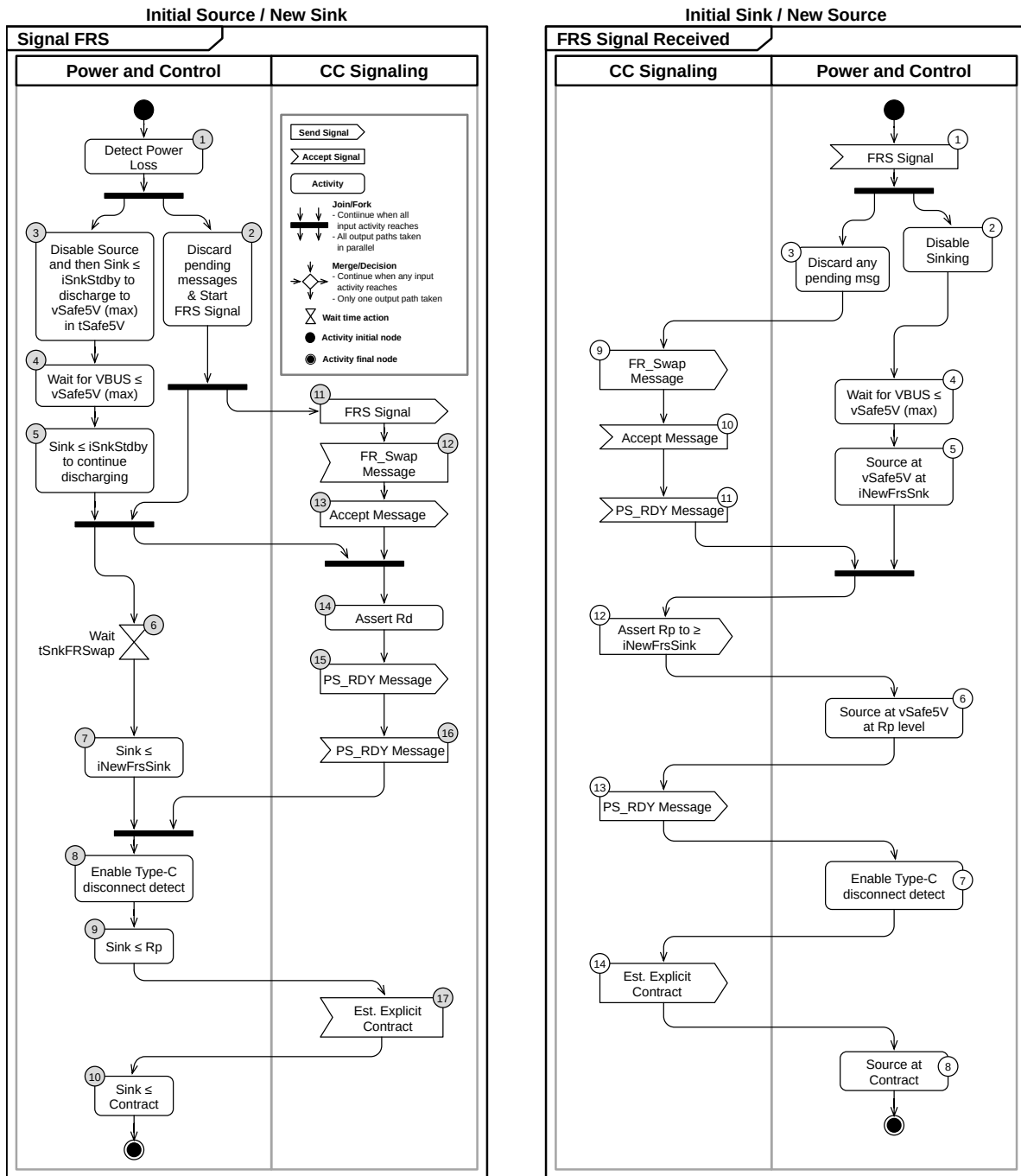


Figure 10.3. FRS Sequence with VBUS Starting at Voltage > vSafe5V (long discharge)

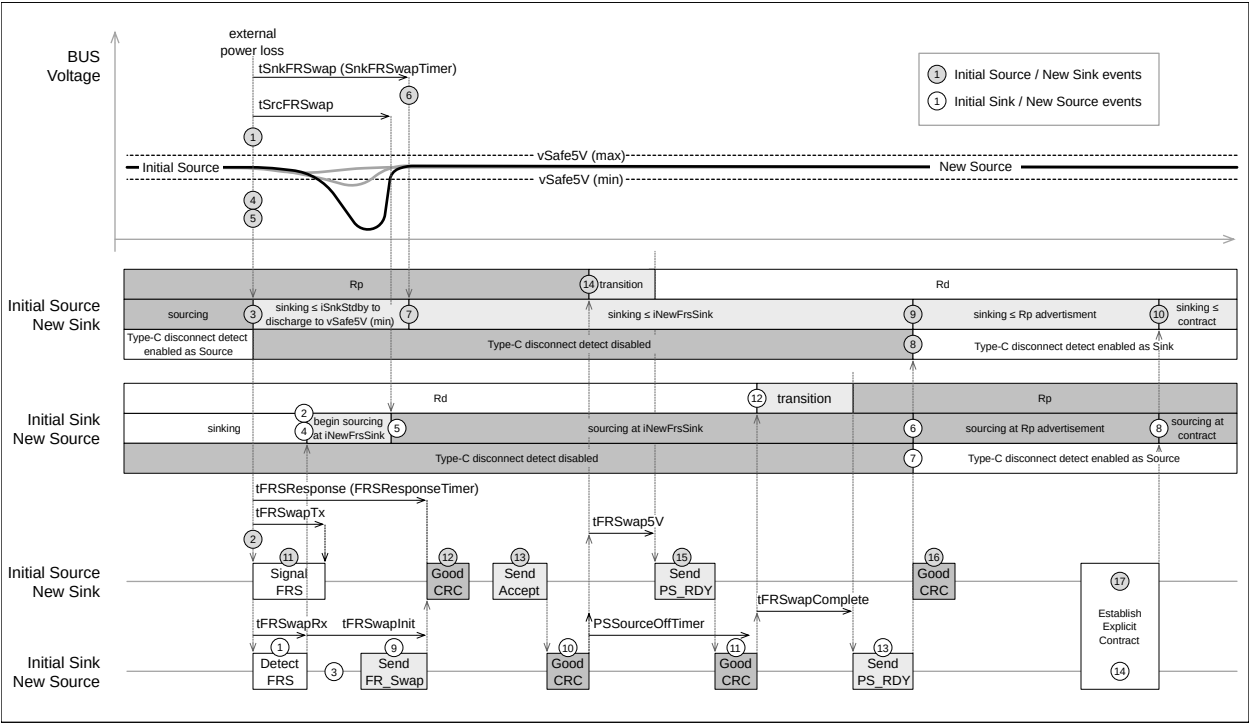
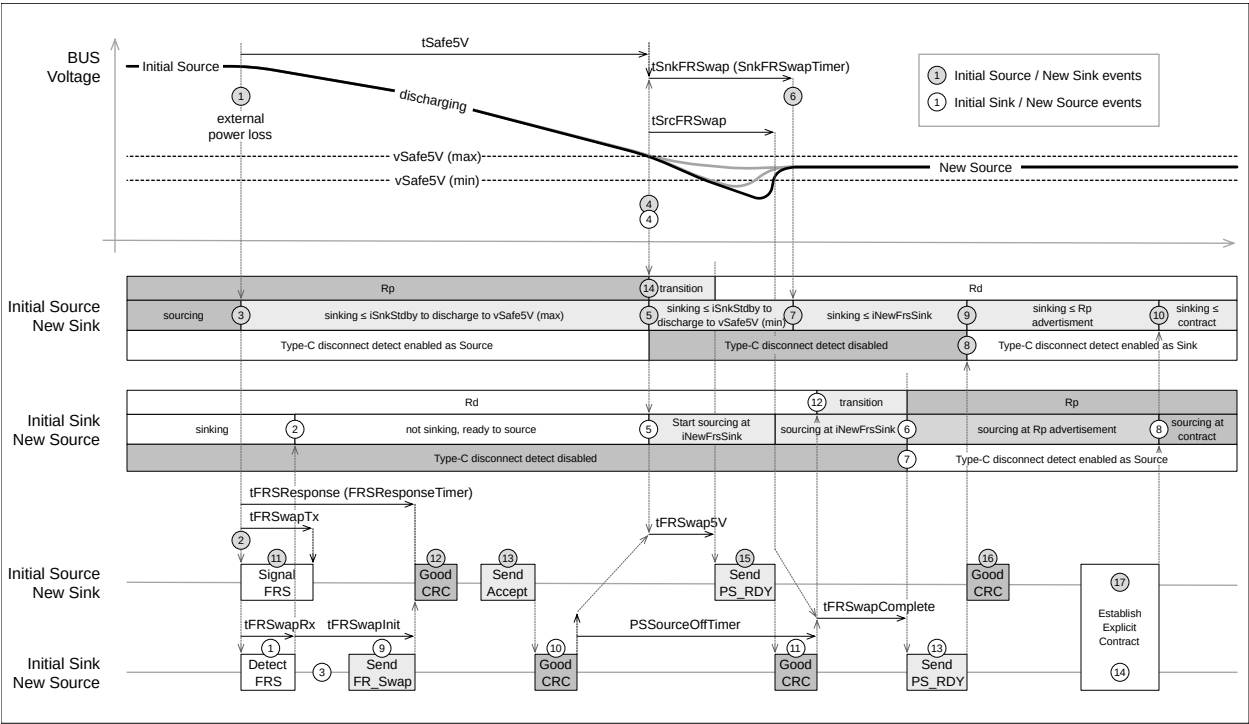


Figure 10.4. FRS Sequence with VBUS starting at vSafe5V (short or no discharge)



10.4. Initial Source/New Sink

An FRS Request is signaled by the Initial Source when it has lost external power and needs the Initial Sink to provide power to VBUS. The Initial Source **Should** signal the FRS as soon as it is aware that it has lost power. The following steps describe the behavior shown in the 'Signal FRS' activity diagram in [Figure 10.2](#) and the timing diagram shown in [Figure 10.3](#) and [Figure 10.4](#) match the numbers indicated in the figures.

Power and Control

1. The Initial Source detects external power loss.
2. The Initial Source **Shall** discard any pending messages and start sending the FRS Signal. The FRS Signal is transmitted by pulling the CC line below [vFRSwapCableTx](#) for at least [tFRSwapTx](#).
3. At the same time as starting the FRS Signal, the Initial Source **Shall** stop Sourcing VBUS and **Shall** start sinking current \leq [iSnkStdby](#) from VBUS. While $VBUS \geq$ [vSafe5V](#) (max), the New Sink **Shall** Sink enough current to discharge to [vSafe5V](#) (max) within [tSafe5V](#).
4. [Wait](#) for $VBUS \leq$ [vSafe5V](#) (max) and initialize the [SnkFRSwapTimer](#) with [tSnkFRSwap](#) and start the timer. VBUS may already be \leq [vSafe5V](#) (max) when this step is reached.
5. The New Sink **Shall** continue to pull current to discharge VBUS to [vSafe5V](#) (min) during [SnkFRSwapTimer](#). This prevents the process from stalling where the New Source's threshold for [vSafe5V](#) (max) might be lower than the New Sink's threshold. The New Source might begin to Source VBUS before reaching [vSafe5V](#) (min) resulting in VBUS staying above [vSafe5V](#) (min). If VBUS falls below [vSafe5V](#) (min), the New Sink is no longer required to discharge VBUS.
6. [iSnkStdby](#) is the (max) current that can be drawn until [tSnkFRSwap](#) after both the FRS Signal was started and $VBUS <$ [vSafe5V](#) (max).
7. After [tSnkFRSwap](#), the New Sink **Shall** Sink current \leq [iNewFrsSnk](#). [iNewFrsSnk](#) is the current set by the FRS bits in the New Sink's Sink Capabilities initially provided to the Initial Sink. The New Sink **Shall** Sink current \leq [iNewFrsSnk](#) until it has detected the New Source's Rp termination.
8. When the [PS_RDY Message](#) from the New Source is received, the New Sink is now in an Implicit Contract and **Shall** enable Type-C disconnect detection.
9. The New Sink **Shall** limit the Sink current based on the Implicit Contract until an Explicit Contract is established.
10. Once an Explicit Contract is established, normal operation resumes.

CC Signaling

11. The FRS **Shall** be signaled by pulling low on CC for a length [tFRSwapTx](#) and the [FRSResponseTimer](#) is initialized with [tFRSResponse](#) and started.
12. After sending the FRS Signal, the Initial Source waits to receive an [FR_Swap Message](#). If the [FR_Swap Message](#) is not received and responded to within expiration of [FRSResponseTimer](#), the New Sink **Shall** go to errorRecovery.
13. The New Sink **Shall** send an [Accept Message](#) in response to the [FR_Swap Message](#) within [tReceiverResponse](#).
14. When both the [GoodCRC Message](#) response following the [Accept Message](#) has been received and $VBUS \leq$ [vSafe5V](#) (max), the New Sink **Shall** assert Rd and send a [PS_RDY Message](#) within [tFRSwap5V](#). The Port Power Role field of the [PS_RDY Message](#) **Shall** be set to Sink.
15. After sending the [PS_RDY](#), the New Sink **Shall** wait for a [PS_RDY Message](#) from the New Source.

16. After the [PS_RDY Message](#) is received from the New Source, the New Sink waits to enter an Explicit Contract with the New Source.

10.4.1. Initial Sink/New Source

An Initial Sink, when initialized for FRS, monitors for the FRS Signal from the Initial Source and disables Type-C disconnect detection via VBUS. The Initial Sink **shall** also be ready to Source VBUS as soon as the FRS Signal is received. When the FRS Signal is received, the Initial Sink follows the FRS Signal Received activity diagram shown in [Figure 10.2](#).

Power and Control

1. The FRS signal is received by the Initial Sink for at least [tFRSwapRx](#).
2. The Initial Sink becomes the New Source and **shall** immediately stop sinking current from VBUS.
3. At the same time, the New Source **shall** discard any pending messages.
4. [Wait](#) for $VBUS \leq vSafe5V$ (max) and then proceed.
5. The New Source **should** immediately start sourcing VBUS at [vSafe5V](#) and [iNewFrsSnk](#) and **shall** start sourcing within [tSrcFRSwap](#) of $VBUS \leq vSafe5V$ (min).
6. When the [PS_RDY Message](#) from the New Sink is received, the New Source **shall** Source VBUS based on the Implicit Contract at a Type-C current $\geq iNewFrsSnk$.
7. After sending a [PS_RDY Message](#), the New Source **shall** enable Type-C disconnect detection.
8. Once an Explicit Contract is established, normal operation resumes.

CC Signaling

9. The New Source responds to the FRS signal by sending the [FR_Swap Message](#) within [tFRSwapInit](#) of completing the detection.
10. After sending the [FR_Swap Message](#), the New Source waits for an [Accept Message](#). If the [Accept Message](#) is not received and responded to with a [GoodCRC Message](#) within [tSenderResponse](#) time, the New Source **shall** go to ErrorRecovery.
11. After receiving the [Accept Message](#), the New Source initializes the [PSSourceOffTimer](#) with [tPSSourceOff](#) and waits to receive a [PS_RDY Message](#). If the [PS_RDY Message](#) is not received and responded to with a [GoodCRC Message](#) when the [PSSourceOffTimer](#) expires, the New Source **shall** go to ErrorRecovery. If the FRS Signal was received while in SPR Mode, the SPR [tPSSourceOff](#) time is used. If the FRS Signal was received while in [EPR Mode](#), the EPR [tPSSourceOff](#) time is used.
12. After receiving the [PS_RDY Message](#) and responding with a [GoodCRC Message](#) and $VBUS \leq vSafe5V$ (max), the New Source **shall** assert Rp termination for a Type-C current $\geq iNewFrsSnk$, and send a [PS_RDY Message](#) within [tFRSwapComplete](#) of sending the [GoodCRC Message](#). The Port Power Role field of the [PS_RDY Message](#) **shall** be set to Source.
13. After sending the [PS_RDY Message](#), the New Source **shall** establish an Explicit Contract. Note: the New Source **may** query the Cable Plug with a Discovery_Identity Message prior to sending Source Capabilities.

10.4.2. Constraints During FRS

- The FRS process is considered an AMS. No other messages may be exchanged until all messages in the [Fast Role Swap](#) sequences have been completed and an Explicit Contract has been established. If any Unexpected Message is received by either the Source or the Sink, the recipient **shall** go to ErrorRecovery.
- If any step in the sequence is missed or delayed, the Initial Source (now New Sink) **may** run out of internal power and disconnect.

10.5. FRS Parameters

Table 10.1. FRS Timers

Timer Name	Timer Parameter	Description
FRSResponseTimer	tFRSResponse	Timer to enforce tFRSResponse time.
SnkFRSwapTimer	tSnkFRSwap	Timer to enforce tSnkFRSwap time.

Table 10.2. FRS Parameters

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
iNewFrsSink			Default USB	A	Maximum current the New Sink can draw during a Fast Role Swap until the New Source applies Rp.
			1.5	A	Matches the required Fast Role Swap required USB Type-C Current field of the Fixed Supply PDO of the Initial Source's Sink_Capabilities Message .
			3.0	A	
tFRSResponse			50	ms	Interval for the Initial Source to receive an FR_Swap Message from the Initial Sink after sending an FRS Signal. Start: the leading edge of the FRS Signal. End: the last bit of the FR_Swap Message EOP has been transmitted
tFRSwap5V			15	ms	Interval for the New Sink to assert Rd and send a PS_RDY Message indicating it has completed the swap from the Initial Source. Start: The later of (both have occurred): <ul style="list-style-type: none"> The last bit of the GoodCRC Message response EOP following the Accept Message has been transmitted. VBUS \leq vSafe5V. End: the first bit of the PS_RDY Message Preamble has been transmitted.
tFRSwapComplete			15	ms	Interval for the New Source to assert Rp and send a PS_RDY Message indicating it has completed the swap from the Initial Sink. Start: The later of (both have occurred): <ul style="list-style-type: none"> The last bit of the GoodCRC response EOP following the New Sink's PS_RDY Message has been transmitted. New Source is sourcing VBUS at vSafe5V. End: first bit of the response PS_RDY Message Preamble has been transmitted.
tFRSwapInit			15	ms	Interval for the Initial Sink to send an FR_Swap Message after detecting an FRS Signal. Start: the detection of an FRS signal. End: the last bit of the FR_Swap Message EOP has been transmitted.
tFRSwapRx	30		50	μ s	FRS signal receive duration. Duration of the FRS signal to be detected as valid by the Initial Sink. Start: falling edge of CC low
tFRSwapTx	60		120	μ s	FRS signal transmit duration. Duration of the Initial Source driving CC low to signal an FRS. Start: falling edge of CC low by driving with a resistance to ground of rFRSwap-Tx . End: rising edge of CC when removing the rFRSwapTx resistance.

Parameter Name	Min Value	Nom Value	Max Value	Unit	Description
tSnkFRSwap			200	μs	<p>Wait time during an FRS before the New Sink can draw up to iNewFrsSink.</p> <p>Start: The later of (both have occurred):</p> <ul style="list-style-type: none"> Start of the FRS signal. VBUS falling below vSafe5V (min).
tSrcFRSwap			150	μs	<p>Interval for the New Source to be actively sourcing VBUS at vSafe5V and iNewFrsSink.</p> <p>Start: VBUS ≤ vSafe5V (max).</p>
rFRSwapTx			5	Ω	FRS Signal transmit driver pull-down resistance. Resistance is between the CC pin and ground.
vFRSwapCableTx	490	520	550	mV	FRS Request voltage detection threshold.

Appendix A. Revision History

Table A.1. Revision History

Revision	Version	Comments	Issue Date
1.0	1.0	Initial release Revision 1.0	5 July 2012
1.0	1.1	Including errata through 31-October-2012	31 October 2012
1.0	1.2	Including errata through 26-June-2013	26 June 2013
1.0	1.3	Including errata through 11-March-2014	11 March 2014
2.0	1.0	Initial release Revision 2.0	11 August 2014
2.0	1.1	Including errata through 7-May 2015	7 May 2015
2.0	1.2	Including errata through 25-March-2016	25 March 2016
2.0	1.3	Including errata through 11-January-2017	11 January 2017
3.0	1.0	Initial release Revision 3.0	11 December 2015
3.0	1.0a	Including errata through 25-March-2016	25 March 2016
3.0	1.1	Including errata through 12-January-2017	12 January 2017
3.0	1.2	Including errata through 21-June-2018	21 June 2018
3.0	2.0	Including errata through 29-August-2019	29 August 2019
3.1	1.0	Including errata through May 2021	May 2021
3.1	1.1	Including errata through July 2021 This Version incorporates the following ECNs: <ul style="list-style-type: none"> • EPR Clarifications • Define AMS starting point 	July 2021
3.1	1.2	Including errata through October 2021 This Version incorporates the following ECNs: <ul style="list-style-type: none"> • Clarify use of Retries • Battery Capabilities • FRS timing problem • PPS power rule clarifications • Peak current support for EPR AVS APDO 	October 2021
3.1	1.3	This Version incorporates the following ECNs: <ul style="list-style-type: none"> • Robust EPR Source Operation • EPR Source Caps Editorial • SRC PPS behavior in low current Request • Enter USB 	January 2022
3.1	1.4	Editorial changes This Version incorporates the following ECNs: <ul style="list-style-type: none"> • Capabilities Mismatch Update • Chunking Timing Issue • OT Mitigation 	April 2022

Revision	Version	Comments	Issue Date
3.1	1.5	Editorial changes This Version incorporates the following ECNs: <ul style="list-style-type: none"> • Timer Description Corrections • Change Source_Info Requirements • AMS Update 	July 2022
3.1	1.6	Editorial changes This Version incorporates the following ECNs: <ul style="list-style-type: none"> • USB4 V2 Updates • Data Reset Issues • Increase tSenderResponse • PPS Power Limit Bit Update • Support for Asymmetric Mode • Timer Description Corrections Revisited 	October 2022
3.1	1.7	Editorial Changes This Version incorporates the following ECNs: <ul style="list-style-type: none"> • Data Reset Invalid Reject Handling • Source Request • Source Transition • EPR Entry 	January 2023
3.1	1.8	Editorial Changes This Version incorporates the following ECNs: <ul style="list-style-type: none"> • Slew rate exemption for Power Role Swap. • EUDO cable speed clarification. • Update to PPS Requirements. • Deprecate Interruptibility. • Section 7.3 restructure and update. 	April 2023
3.1	1.9	Editorial Changes	July 2023
3.2	1.0	This Version incorporates the following ECNs: <ul style="list-style-type: none"> • VDM-use Conditions. • tTypeCSinkWaitCap. • tFirstSourceCap Clarification • Hard Reset Clarification. • Unrecognized Country Code • EPR Entry Process-1 • SPR AVS Definition • EPR Power Rules Clarifications 	October 2023

Revision	Version	Comments	Issue Date
3.2	1.1	<p>This Version incorporates the following ECNs:</p> <ul style="list-style-type: none">• Power Transition time from EPR to PR_Swap• Capabilities Mismatch Update• Deprecate GotoMin and GiveBack Features and Update Power Reserve• EPR Entry requirements Clarification• EPRMDO and Entry Clarification.• Remove 10.2.4 power sharing between ports• Source PDP Rating field clarifications• Source Power Rules update.• Source_Info Message Clarifications.• Correction to BMC description.• EPR Source cap clarification.• Delaying of VCONN Swap.• EPR_Request in SPR Mode.• Generic transition diagram.• Removing the usage of Ping Message• Sink Standby• Source Info Support	October 2024

Revision	Version	Comments	Issue Date
3.2	1.2	<p>Reformatted and restructured the specification; all previous text has been rewritten and reformatted.</p> <p>This version incorporates the following ECNs; note that they were authored against the prior structure/wording, so their content may appear in different locations in this document.</p> <ul style="list-style-type: none"> • Dynamic Power Sources (DPS). • Clarify Assured and Shared Capacity Definitions. • Clarify EPR_Get_Source_Cap negative Response. • Clarify meaning of full Capabilities • Errata PDO type. • Remove vEPRmax. • Wait in EPR_Request • Clarify requirement of PSSourceOffTimer in FRS • HardReset Processing Time Clarification • FRS update to discharge and detect disconnect • Fix Regression in Active Cable VDO Version • iOvershoot removal • Clarify SVDM Major Version 1.0 • cSnkBulkPd Clarifications • EPR_Get_Source_Cap – EPR_Get_Sink_Cap applicability • Source power rule of first source caps 	January 2026
		<p>Corrected Printing (March 2026):</p> <p>This corrected printing fixes the following editorial errata:</p> <ul style="list-style-type: none"> • Errata #1: Corrected broken cross-reference in Section 9.2.26.4.3.1 (PE_BIST_Shared_Capacity_Test_Mode) where state name incorrectly appeared as "Error! Reference Source not found" in two locations. • Errata #2: Corrected DNL error value in Section 4.1.3.2.6 (Source DNL Tolerance) where the text incorrectly stated "The ideal scenario for a DNL error of 1" when it should state "The ideal scenario for a DNL error of 0". 	March 2026

Revision	Version	Comments	Issue Date
		Corrected Printing (May 2026): This corrected printing fixes the following editorial errata: <ul style="list-style-type: none">• Errata #3: Corrected parameter name iSnkStby to iSnkStdby throughout the specification with proper cross-reference links in FRS and electrical requirement sections.• Errata #4: Corrected bit range errors in SIDO1 and SIDO2 status data objects.• Errata #5: Corrected EPR AVS Sink PDO field name and description, changing PDP to Maximum Power for consistency.• Errata #6: Corrected Manufacturer Info Target/Ref byte ordering in the data object definition.• Errata #7: Added missing Fast Role Swap field in Fixed 5V Sink PDO specification.• Errata #8: Added EPR Source Capabilities sequence to EPR Mode Enter AMS definition.• Errata #9: Corrected PDO descriptions where "current values" should read "power values".• Errata #10: Corrected DNL sentence in Section 4.1.3.2.6 for additional clarity.	May 2026

Appendix B. Contributors

ACON, Advanced-Connectek, Inc.

Charles Wang
Conrad Choy

Dennis Chuang
Steve Sedio

Sunney Yang
Vicky Chuang

ASMedia Technology Inc.

Jeff Liu
Kuo Lung Li
Ming-Wei Hsu

PS Tseng
Sam Tzeng
Thomas Hsu

Weikao Chang
Yang Cheng

Advanced Micro Devices

Joseph Scanlon

Sujan Thomas

Allion Labs, Inc.

Caspar Lin
Casper Lee

Danny Shih
Howard Chang

Analogix Semiconductor, Inc.

Greg Stewart

Mehran Badii

Apple Inc.

Alexei Kosut
Bill Cornelius
Brett Saunders
Carlos Calderon
Chris Uiterwijk
Colin Whitby-Strevens
Corey Axelowitz
Corey Lange
Dave Conroy
David Sekowski
Girault Jones

James Orr
Jason Chung
Jay Kim
Jeff Wilcox
Jennifer Tsai
Karl Bowers
Keith Porthouse
Kevin Hsiue
Matt Mora
Michael Iannamico
Paul Baker

Reese Schreiber
Ricardo Janezic Pregitzer
Ruchi Chaturvedi
Sameer Kelkar
Sasha Tietz
Scott Jackson
Sree Raman
William Ferry
Zaki Moussaoui

Bizlink Technology Inc.

Aaron Hou

Shawn Meng

Bizlink Technology, Inc.

Bernard Shyu
Eric Wu

Morphy Hsieh
Sean O'Neal

Tiffany Hsiao
Weichung Ooi

Broadcom Corp.

Rahul Bhushan

Cadence Design Systems, Inc.

Asila nahas
Claire Ying

Jie min
Mark Summers

Michal Staworko
Sathish Kumar Ganesan

Canova Tech

Alessandro Ingrassia
Andrea Colognese
Antonio Orzelli

Davide Ghedin
Matteo Casalin
Michael Marioli

Nicola Scantamburlo
Paolo Pilla

Canyon Semiconductor

Ray Huang

Yi-Feng Lin

YuHung Lin

Chrontel, Inc.

David Tsai

Cypress Semiconductor

Anshul Gulati
Anup Nayak
Benjamin Kropf
Dhanraj Rajput
Ganesh Subramaniam
Jagadeesan Raj
Junjie cui
Manu Kumar

Muthu M
Nicholas Bodnaruk
Pradeep Bajpai
Rajaram R
Rama Vakkantula
Rushil Kadakia
Simon Nguyen
Steven Wong

Subu Sankaran
Sumeet Gupta
Tejender Sheoran
Venkat Mandagulathar
Xiaofeng Shen
Zeng Wei

Dell Inc.

Adie Tan
Adolfo Montero
Bruce Montag
Gary Verdun

Ken Nicholas
Marcin Nowak
Merle Wood
Mohammed Hijazi

Siddhartha Reddy
Terry Matula

Derun Semiconductor

Jay Hu

Shelly Liu

Dialog Semiconductor (UK) Ltd

Bindhu Vasu
Chanchal Gupta

Jianming Yao
John Shi

Mengfei Liu
Scott Brown

Dipti Baheti
Duc Doan
Holger Petersen

KE Hong
Kevin Mori
Larry Ping

Yimin Chen
Yong Li

Diodes Incorporated

Justin Lee

DisplayLink (UK) Ltd.

Dan Ellis
Jason Young

Kevin Jacobs
Paulo Alcobia

Peter Burgers
Richard Petrie

Ellisys

Abel Astley
Chuck Trefts

Emmanuel Durin
Mario Pasquali

Tim Wei

Etron Technology, Inc.

Chien-Cheng Kuo
Jack Yang

Richard Crisp
Shyanjia Chen

TsungTa Lu

Fairchild Semiconductor

Christian Klein

Oscar Freitas

Souhib Harb

Feature Integration Technology Inc.

Amanda Ying
Jacky Chan
Kenny Hsieh

KungAn Lin
Paul Yang
Su Jaden

Yu-Lin Chu
Yulin Lan

Foxconn / Hon Hai

AJ Yang
Bob Hall
Chihyin Kan

Fred Fons
Jie Zheng
Patrick Casher

Shruti Deore
Steve Sedio
Terry Little

Fresco Logic Inc.

Bob McVay
Christopher Meyers

Dian Kurniawan
Tom Burton

Google Inc.

Abraham Levkoy
Adam Rodriguez
Alec Berg

Eric Herrmann
Jameson Thies
Jim Guerin

Nithya Jagannathan
Srikanth Lakshmikanthan
Todd Broch

Benson Leung
Chao Fei
Dave Bernard
David Schneider
Diana Zigterman

Juan Fantin
Ken Wu
Kyle Tso
Mark Hayter
Nathan Kolluru

Toshak Singhal
Vincent Palatin
Xuelin Wu
Zhenxue Xu

Granite River Labs

Alan Kinningham
Anand Murugan
Balamurugan Manialagan
Medipalli Sowmya
Mike Engbretson
Mike Wu

Mukesh Tatiya
Naresh Botsa
PoornaKumar M.
Prajwal Rathod
Rajaraman V
Sivan Perumal

Sivaram Murugesan
Tim Lin
Vijay S.
Vijayakumar P
Vishal Kakade
Yogeshwaran Venkatesan

GuangDong OPPO Mobile Telecommunications Corp., Ltd.

Jerry Qin

HP Inc.

Alan Berkema
Kenneth Chan
Lee Atkinson
Lee Leppo

Rahul Lakdawala
Robin Castell
Roger Benson
Ron Schooley

Steve Chen
Suketa Partiwala
Vaibhav Malik
Walter Fry

Hosiden Corporation

Hideyuki Hayafuji
Keiji Mine

Masaki Yamaoka
Takashi Muto

Yasunori Nishikawa

Huawei Technologies Co., Ltd.

Bai Sean
Chunjiang Zhao
JianQuan Wu
Li Zongjian

Liansheng Zheng
Lihua Duan
Min Chen
Wang Feng

Wei Haihong
Zhenning Shi

Hynetek Semiconductor Co., Ltd

James Xie

Yingyang Ou

ITE Tech. Inc.

Al Hsiao
Greg Song

Richard Guo
Victor Lin

Y.C. Chou

Indie Semiconductor

Robert Heaton

Vincent Wang

Infineon Technologies

Benjamin Kropf
Sie Boo Chiang

Tue Fatt David Wee
Wee Tar Richard Ng

Wolfgang Furtner

Intel Corporation

Aruni Nelson
Bob Dunstan
Brad Saunders
Chee Lim Nge
Christine Krause
Chuen Ming Tan
Dan Froelich
David Harriman
David Hines
David Thompson
Guobin Liu

Harry Skinner
Henrik Leegaard
Jenn Chuan Cheng
Jervis Lin
John Howard
Karthi Vadivelu
Leo Heiland
Maarit Harkonen
Nge Chee Lim
Paul Durley
Rahman Ismail

Rajaram Regupathy
Ronald Swartz
Sarah Sharp
Scott Brenden
Sridharan Ranganathan
Steve McGowan
Tim McKee
Toby Opferman
Uma Medepalli
Ziv Kabiry

Intersil Corporation

Jia Wei

Japan Aviation Electronics Industry Ltd. (JAE)

Kenta Minejima

Toshio Shimoyama

Keysight Technologies Inc.

Brian Fetz

Jit Lim

Kinetic Technologies Inc.

Koji Asakawa

LG Electronics Ltd.

Won-Jong Choi

LG electronics

Do Kyun Kim

Won-Jong Choi

LSI Corporation

Dave Thompson

Lattice Semiconductor Corp

Babu Mailachalam

Joel Coplen

Vesa Lauri

Gianluca Mariani

Thomas Watza

LeCroy Corporation

Chetan Kopalle
Daniel H Jacobs
Jake Jacobs

Kimberley McKay
Mike Engbretson
Mike Micheletti

Roy Chestnut
Tyler Joe

Leadtrend

Bruce Chuang

Eilian Liu

Lenovo

Phil Jakes

Lion Semiconductor

Aaron Melgar
Chris Zhou

Sehyung Jeon
Wonyoung Kim

Yongho Kim

Luxshare-ICT

Alan Kinningham
Alan Liu
Daniel Chen
Eric Wen

James Kirk
James Stevens
Josue Castillo
Pat Young

Scott Shuey
Stone Lin

MCCI Corporation

Chris Yokum
Geert Knapen

Terry Moore
Velmurugan Selvaraj

MQP Electronics Ltd.

Ben Crowe

Pat Crowe

Sten Carlsen

Maxim Integrated Products

Chikara Kakizawa
Jacob Scott

Ken Helfrich
Michael Miskho

MediaTek Inc.

Tung-Sheng Lin

MegaChips Corporation

Satoru Kumashiro

Microchip Technology Inc.

Brian Marley
Dave Perchlik
Don Perkins
Fernando Gonzalez
John Sisto
Josh Averyt

Kiet Tran
Mark Bohm
Matthew Kalibat
Mick Davis
Prasanna Vengateshan
Rich Wahler

Richard Petrie
Ronald Kunin
Shannon Cash
Thomas Farkas
Venkataraman Krishnamoorthy

Microsoft Corporation

Andrew Yang
Anthony Chen
Arvind Murching
Dave Perchlik
David Voth
Geoff Shew
Jayson Kastens

Kai Inha
Marwan Kadado
Michelle Bergeron
Nathan Sherman
Rahul Ramadas
Randy Aull
Shiu Ng

Tieyong Yin
Timo Toivola
Toby Nixon
Vahid Vassey
Vivek Gupta
Yang You

Molex LLC

Adib Al Abaji

Monolithic Power Systems Inc.

Aaron Xu
Bo Zhou

Christian Sporck
Di Han

Zhihong Yu

Motorola Mobility Inc.

Dan Wagner

NEC Corporation

Kenji Oguma

NXP Semiconductors

Abhijeet Kulkarni
Ahmad Yazdi
Bart Vertenten
Dennis Ha
Dong Nguyen

Guru Prasad
Ken Jaramillo
Krishnan TN
Michael Joehren
Robert de Nie

Rod Whitby
Vijendra Kuroodi
Winston Langeslag

Nexperia B.V.

Stefan Seider

Max Guan

ChinJui Lin

Nokia Corporation

Frank Borngraber
Kai Inha

Pekka Leinonen
Richard Petrie

Sten Carlsen

ON Semiconductor

Andrew Yoo
Brady Maasen
Bryan McCoy

Christian Klein
Cor Voorwinden
Edward Berrios

Michael Smith
Oscar Freitas
Tom Duffy

Obsidian Technology

Robert Heaton

Parade Technologies Inc.

Brian Collins

Craig Wiley

Power Forest Technology Corporation

Hung-Chih Chiu

Jay Tu

Power Integrations

Adel Lahham
Aditya Kulkarni
Akshay Nayaknur

Amruta Patra
K R Rahul Raj
Kaushik Raam

Rahul Joshi
Shruti Anand

Qualcomm, Inc

Amit gupta
Chris Sporck
Craig Aiken
George Paparrizos
Giovanni Garcea
Jack Pham

James Goel
Joshua Warner
Karyn Vuong
Lalan Mishra
Narendra Mehta
Nicholas Cadieux

Terry Remple
Vamsi Samavedam
Vatsal Patel
Will Kun
Yoram Rimoni

ROHM Co., Ltd.

Kazuomi Nagai

Realtek Semiconductor Corp.

Fan-Hau Hsu

Tsung-Peng Chuang

Renesas Electronics Corp.

Atsushi Mitamura

John Carpenter

Philip Leung

Bob Dunstan
Brian Allen
Dan Aoki
Fengshuan Zhou
Hajime Nozaki

Kiichi Muto
Masami Katagiri
Nobuo Furuya
Patrick Yu
Peter Teng

Steve Roux
Tetsu Sato
Toshifumi Yamaoka
Yimin Chen

Richtek Technology Corporation

Chunan Kuo
Heinz Wei

Max Huang
TZUHSIEN CHUANG

Ricoh Company Ltd.

Tatsuya Irisawa

Rohm Co. Ltd.

Akihiro Ono
Chris Lin
Hidenori Nishimoto

Kris Bahar
Manabu Miyata
Ruben Balbuena

Takashi Sato
Vijendra Kuroodi
Yusuke Kondo

SMSC

John Sisto
Ken Gay
Mark Bohm

Richard Wahler
Shannon Cash
Tim Knowlton

William Chiechi

ST-Ericsson

Fabien Friess
Giuseppe Platania

Jean-Francois Gatto
Milan Stamenkovic

Nicolas Florenchie
Patrizia Milazzo

STMicroelectronics

Christophe Cochard
Christophe Lorin
Filippo Bonaccorso
Jessy Guilbot

Joel Huloux
John Bloomfield
Massimo Panzica
Meriem Mersel

Nathalie Ballot
Pascal Legrand
Patrizia Milazzo
Richard O'Connor

Salcomp Plc

Matti Kulmala

Toni Lehimo

Samsung Electronics Co. Ltd.

Edward Lee

Tong Kim

Scosche Industries

Amit Bouzaglo

Seagate Technology LLC

Alvin Cox
Emmanuel Lemay
John Hein

Marc Noblitt
Michael Morgan
Ronald Rueckert

Tony Priborsky

Semtech Corporation

Chin Chang

Tom Farkas

Siemens Industry Software Inc.

Ankit Garg

Silergy Corp.

Ning Dai

Wanfeng Zhang

SiliConch Systems Private Limited

Abhishek Sardeshpande
Aniket Mathad
Chandana N
Jaswanth Ammineni

Jinisha Patel
Kaustubh Kumar
Nitish
Pavitra Balasubramanian

Rakesh Polasa
Satish Anand Verkila
Shubham Paliwal
Vishnu Pusuluri

Silicon Laboratories, Inc.

Kafai Leung
Kok Hong Soh

Sorin Badiu
Steven Ghang

Sony Group Corporation

Shigenori Tagami

Shinichi Hirata

Specwerkz

Amanda Hosler
Bob Dunstan

Brad Saunders
Diane Lenox

StarTech.com Ltd.

Michael Munn

Synopsys, Inc.

Morten Christiansen
Nivin George

Prishkit Abrol
Zongyao Wen

Tektronix

Joan Marrinan

Teledyne-LeCroy

Kimberley McKay

Matthew Dunn

Tony Minchell

Texas Instruments

Anand Dabak
Annamalai Kasthuri
Bill Waters
Bing Lu
Deric Waters
Grant Ley
Gregory Watkins

Ingolf Frank
Ivo Huber
Javed Ahmad
Jean Picard
John Perry
Kasthuri Annamalai
Martin Patoka

Mike Campbell
Scott Jackson
Shafiuddin Mohammed
Srinath Hosur
Steven Tom
Yoon Lee

The Silanna Group Pty. Ltd.

Tim Wilhelm

Tod Wolf

Total Phase

Chris Yokum

UL LLC

Dylan Su
Eric Wall

Jason Smith
Terry Kao

Unigraf OY

Steven Chen

Topi Lampiranta

VIA Technologies, Inc.

Dydron Lin
Fong-Jim Wang

Jay Tseng
Rex Chang

Terrance Shih

Ventev Mobile

Brad Cox

Colin Vose

Weltrend Semiconductor

Ho Wen Tsai
Hung Chiang

Jeng Cheng Liu
Priscilla Lee

Wayne Lo

Western Digital Technologies, Inc.

Charles Neumann

Curtis Stevens

John Maroney

Wilder Technologies

Joe O'Brien

Will Miller

Xiaomi Communications Co., Ltd.

Juejia Zhou

Xiaoxing Yang

Zhuhai Ismartware Technology Co., Ltd.

Liu Qiong

Long Zhang

Yuanchao Liang

eEver Technology, Inc.

Chien-Cheng Kuo

Shyanjia Chen

iST - Integrated Service Technology Inc.

Weijie Huang

Key Contributors to the Revision 3.2 Version 1.2 Specification Rewrite

The following individuals provided significant contributions to the rewrite and restructuring of this specification:

Scott Jackson (Apple Inc.) - Editor
Nathalie Ballot (STMicroelectronics) - Editor
Michael Iannamico (Apple, Inc.) - Technical Writer
Neha Bagga (Apple Inc.) - Technical Writer
Roger Benson (HP Inc.)
Carlos Calderon (Apple Inc.)
William Ferry (Apple Inc.)
Ricardo Janezic Pregitzer (Apple Inc.)

Alexei Kosut (Apple Inc.)
Corey Lange (Apple Inc.)
Abraham Levkoy (Google Inc.)
Benson Leung (Google Inc.)
Brett Saunders (Apple Inc.)
Jameson Thies (Google Inc.)
Deric Waters (Texas Instruments)