Universal Serial Bus
Power Delivery Specification

Revision 3.0, V1.0a. 25 March 2016

Microchip Ex. 1020, p. 1
Microchip v. Aptiv
IPR2024-00496

Editors

Bob Dunstan	Intel Corporation
Richard Petrie	DisplayLink

Contributors

Charles Wang	ACON, Advanced-Connectek, Inc.
Conrad Choy	ACON, Advanced-Connectek, Inc.
Steve Sedio	ACON, Advanced-Connectek, Inc.
Vicky Chuang	ACON, Advanced-Connectek, Inc.
Joseph Scanlon	Advanced Micro Devices
Howard Chang	Allion Labs, Inc.
Greg Stewart	Analogix Semiconductor, Inc.
Mehran Badii	Analogix Semiconductor, Inc.
Bill Cornelius	Apple
Colin Whitby-Strevens	Apple
Corey Axelowitz	Apple
Corey Lange	Apple
Dave Conroy	Apple
David Sekowski	Apple
Girault Jones	Apple
James Orr	Apple
Jason Chung	Apple
Jennifer Tsai	Apple
Karl Bowers	Apple
Keith Porthouse	Apple
Matt Mora	Apple
Paul Baker	Apple
Reese Schreiber	Apple

Microchip Ex. 1020, p. 2
Microchip v. Aptiv
IPR2024-00496

Sameer Kelkar	Apple
Sasha Tietz	Apple
Sree Raman	Apple
William Ferry	Apple
Zaki Moussaoui	Apple
Bernard Shyu	Bizlink Technology, Inc.
Eric Wu	Bizlink Technology, Inc.
Morphy Hsieh	Bizlink Technology, Inc.
Shawn Meng	Bizlink Technology Inc.
Tiffany Hsiao	Bizlink Technology, Inc.
Weichung Ooi	Bizlink Technology, Inc.
Michal Staworko	Cadence Design Systems, Inc.
Alessandro Ingrassia	Canova Tech
Andrea Colognese	Canova Tech
Davide Ghedin	Canova Tech
Matteo Casalin	Canova Tech
Nicola Scantamburlo	Canova Tech
Yi-Feng Lin	Canyon Semiconductor
Anup Nayak	Cypress Semiconductor
Jagadeesan Raj	Cypress Semiconductor
Pradeep Bajpai	Cypress Semiconductor
Rushil Kadakia	Cypress Semiconductor
Steven Wong	Cypress Semiconductor
Subu Sankaran	Cypress Semiconductor
Sumeet Gupta	Cypress Semiconductor
Adolfo Montero	Dell Inc.

Microchip Ex. 1020, p. 3
Microchip v. Aptiv
IPR2024-00496

Bruce Montag	Dell Inc.	
Gary Verdun	Dell Inc.	
Merle Wood	Dell Inc.	
Mohammed Hijazi	Dell Inc.	
Siddhartha Reddy	Dell Inc.	
Dan Ellis	DisplayLink	
Jason Young	DisplayLink	
Peter Burgers	DisplayLink	
Richard Petrie	DisplayLink	PD Chair/Device Policy Lead
Abel Astley	Ellisys	
Chuck Trefts	Ellisys	
Emmanuel Durin	Ellisys	
Mario Pasquali	Ellisys	
Chien-Cheng Kuo	Etron Technology, Inc.	
Jack Yang	Etron Technology, Inc.	
Richard Crisp	Etron Technology, Inc.	
Shyanjia Chen	Etron Technology, Inc.	
TsungTa Lu	Etron Technology, Inc.	
Christian Klein	Fairchild Semiconductor	
Oscar Freitas	Fairchild Semiconductor	
Souhib Harb	Fairchild Semiconductor	
AJ Yang	Foxconn / Hon Hai	
Fred Fons	Foxconn / Hon Hai	
Steve Sedio	Foxconn / Hon Hai	
Terry Little	Foxconn / Hon Hai	
Bob McVay	Fresco Logic Inc.	

Microchip Ex. 1020, p. 4
Microchip v. Aptiv
IPR2024-00496

Christopher Meyers	Fresco Logic Inc.	
Tom Burton	Fresco Logic Inc.	
Dian Kurniawan	Fresco Logic Inc.	
Adam Rodriguez	Google Inc.	
Alec Berg	Google Inc.	
David Schneider	Google Inc.	
Jim Guerin	Google Inc.	
Juan Fantin	Google Inc.	
Ken Wu	Google Inc.	
Mark Hayter	Google Inc.	
Nithya Jagannathan	Google Inc.	
Todd Broch	Google Inc.	
Vincent Palatin	Google Inc.	
Mike Engbretson	Granite River Labs	
Rajaraman V	Granite River Labs	
Alan Berkema	Hewlett Packard	
Lee Atkinson	Hewlett Packard	
Rahul Lakdawala	Hewlett Packard	
Robin Castell	Hewlett Packard	
Roger Benson	Hewlett Packard	
Ron Schooley	Hewlett Packard	
Vaibhav Malik	Hewlett Packard	
Walter Fry	Hewlett Packard	
Bob Dunstan	Intel Corporation	PD Chair/Protocol WG Lead
Brad Saunders	Intel Corporation	
Chee Lim Nge	Intel Corporation	

Microchip Ex. 1020, p. 5
Microchip v. Aptiv
IPR2024-00496

Christine Krause	Intel Corporation	
Dan Froelich	Intel Corporation	
David Harriman	Intel Corporation	
David Hines	Intel Corporation	
David Thompson	Intel Corporation	
Guobin Liu	Intel Corporation	
Harry Skinner	Intel Corporation	
Henrik Leegaard	Intel Corporation	
Jervis Lin	Intel Corporation	
John Howard	Intel Corporation	
Karthi Vadivelu	Intel Corporation	
Leo Heiland	Intel Corporation	
Maarit Harkonen	Intel Corporation	
Nge Chee Lim	Intel Corporation	
Paul Durley	Intel Corporation	
Rahman Ismail	Intel Corporation	System Policy Lead
Ronald Swartz	Intel Corporation	
Sarah Sharp	Intel Corporation	
Scott Brenden	Intel Corporation	
Sridharan Ranganathan	Intel Corporation	
Steve McGowan	Intel Corporation	
Tim McKee	Intel Corporation	PD Chair/Compliance Lead
Toby Opferman	Intel Corporation	
Kenta Minejima	Japan Aviation Electronics Industry Ltd. (JAE)	
Mark Saubert	Japan Aviation Electronics Industry Ltd. (JAE)	
Toshio Shimoyama	Japan Aviation Electronics Industry Ltd. (JAE)	

Microchip Ex. 1020, p. 6
Microchip v. Aptiv
IPR2024-00496

Brian Fetz	Keysight Technologies Inc.
Babu Mailachalam	Lattice Semiconductor Corp
Gianluca Mariani	Lattice Semiconductor Corp
Joel Coplen	Lattice Semiconductor Corp
Thomas Watza	Lattice Semiconductor Corp
Vesa Lauri	Lattice Semiconductor Corp
Daniel H Jacobs	LeCroy Corporation
Jake Jacobs	LeCroy Corporation
Kimberley McKay	LeCroy Corporation
Mike Micheletti	LeCroy Corporation
Roy Chestnut	LeCroy Corporation
Phil Jakes	Lenovo
Dave Thompson	LSI Corporation
Alan Kinningham	Luxshare-ICT
Daniel Chen	Luxshare-ICT
Josue Castillo	Luxshare-ICT
Chris Yokum	MCCI Corporation
Geert Knapen	MCCI Corporation
Terry Moore	MCCI Corporation
Velmurugan Selvaraj	MCCI Corporation
Brian Marley	Microchip Technology Inc.
Dave Perchlik	Microchip Technology Inc.
Don Perkins	Microchip Technology Inc.
John Sisto	Microchip Technology Inc.
Josh Averyt	Microchip Technology Inc.
Kiet Tran	Microchip Technology Inc.

Microchip Ex. 1020, p. 7
Microchip v. Aptiv
IPR2024-00496

Mark Bohm	Microchip Technology Inc.
Matthew Kalibat	Microchip Technology Inc.
Mick Davis	Microchip Technology Inc.
Rich Wahler	Microchip Technology Inc.
Ronald Kunin	Microchip Technology Inc.
Shannon Cash	Microchip Technology Inc.
Anthony Chen	Microsoft Corporation
Dave Perchlik	Microsoft Corporation
David Voth	Microsoft Corporation
Geoff Shew	Microsoft Corporation
Jayson Kastens	Microsoft Corporation
Kai Inha	Microsoft Corporation
Marwan Kadado	Microsoft Corporation
Rahul Ramadas	Microsoft Corporation
Randy Aull	Microsoft Corporation
Shiu Ng	Microsoft Corporation
Timo Toivola	Microsoft Corporation
Toby Nixon	Microsoft Corporation
Vivek Gupta	Microsoft Corporation
Yang You	Microsoft Corporation
Dan Wagner	Motorola Mobility Inc.
Ben Crowe	MQP Electronics Ltd.
Pat Crowe	MQP Electronics Ltd.
Sten Carlsen	MQP Electronics Ltd.
Frank Borngräber	Nokia Corporation
Kai Inha	Nokia Corporation

Microchip Ex. 1020, p. 8
Microchip v. Aptiv
IPR2024-00496

Pekka Leinonen	Nokia Corporation	
Richard Petrie	Nokia Corporation	PD Vice-Chair/Device Policy Lead
Sten Carlsen	Nokia Corporation	Physical Layer WG Lead
Abhijeet Kulkarni	NXP Semiconductors	
Ahmad Yazdi	NXP Semiconductors	
Bart Vertenten	NXP Semiconductors	
Dong Nguyen	NXP Semiconductors	
Guru Prasad	NXP Semiconductors	
Ken Jaramillo	NXP Semiconductors	
Krishnan TN	NXP Semiconductors	
Michael Joehren	NXP Semiconductors	
Robert de Nie	NXP Semiconductors	
Rod Whitby	NXP Semiconductors	
Vijendra Kuroodi	NXP Semiconductors	
Robert Heaton	Obsidian Technology	
Bryan McCoy	ON Semiconductor	
Cor Voorwinden	ON Semiconductor	
Edward Berrios	ON Semiconductor	Power Supply WG Lead
Tom Duffy	ON Semiconductor	
Craig Wiley	Parade Technologies Inc.	
Ricardo Pregiteer	Power Integrations	
Chris Sporck	Qualcomm, Inc.	
Craig Aiken	Qualcomm, Inc.	
George Paparrizos	Qualcomm, Inc	
Giovanni Garcea	Qualcomm, Inc	
James Goel	Qualcomm, Inc	

Microchip Ex. 1020, p. 9
Microchip v. Aptiv
IPR2024-00496

Joshua Warner	Qualcomm, Inc
Narendra Mehta	Qualcomm, Inc.
Terry Remple	Qualcomm, Inc.
Yoram Rimoni	Qualcomm, Inc.
Atsushi Mitamura	Renesas Electronics Corp.
Dan Aoki	Renesas Electronics Corp.
Kiichi Muto	Renesas Electronics Corp.
Masami Katagiri	Renesas Electronics Corp.
Nobuo Furuya	Renesas Electronics Corp.
Patrick Yu	Renesas Electronics Corp.
Peter Teng	Renesas Electronics Corp.
Philip Leung	Renesas Electronics Corp.
Steve Roux	Renesas Electronics Corp.
Tetsu Sato	Renesas Electronics Corp.
Heinz Wei	Richtek Technology Corporation
Tatsuya Irisawa	Ricoh Company Ltd.
Akihiro Ono	Rohm Co. Ltd.
Chris Lin	Rohm Co. Ltd.
Hidenori Nishimoto	Rohm Co. Ltd.
Kris Bahar	Rohm Co. Ltd.
Manabu Miyata	Rohm Co. Ltd.
Ruben Balbuena	Rohm Co. Ltd.
Takashi Sato	Rohm Co. Ltd.
Vijendra Kuroodi	Rohm Co. Ltd.
Yusuke Kondo	Rohm Co. Ltd.
Matti Kulmala	Salcomp Plc

Microchip Ex. 1020, p. 10
Microchip v. Aptiv
IPR2024-00496

Toni Lehimo	Salcomp Plc	
Tong Kim	Samsung Electronics Co. Ltd.	
Alvin Cox	Seagate Technology LLC	Cab Con WG Lead
John Hein	Seagate Technology LLC	
Marc Noblitt	Seagate Technology LLC	
Ronald Rueckert	Seagate Technology LLC	
Tony Priborsky	Seagate Technology LLC	
John Sisto	SMSC	
Ken Gay	SMSC	
Mark Bohm	SMSC	
Richard Wahler	SMSC	
Shannon Cash	SMSC	
Tim Knowlton	SMSC	
William Chiechi	SMSC	
Fabien Friess	ST-Ericsson	
Giuseppe Platania	ST-Ericsson	
Jean-Francois Gatto	ST-Ericsson	
Milan Stamenkovic	ST-Ericsson	
Nicolas Florenchie	ST-Ericsson	
Patrizia Milazzo	ST-Ericsson	
Christophe Lorin	ST-Microelectronics	
John Bloomfield	ST-Microelectronics	
Massimo Panzica	ST-Microelectronics	
Meriem Mersel	ST-Microelectronics	
Nathalie Ballot	ST-Microelectronics	
Pascal Legrand	ST-Microelectronics	

Microchip Ex. 1020, p. 11
Microchip v. Aptiv
IPR2024-00496

Patrizia Milazzo	ST-Microelectronics	
Zongyao Wen	Synopsys, Inc.	
Joan Marrinan	Tektronix	
Kimberley McKay	Teledyne-LeCroy	
Matthew Dunn	Teledyne-LeCroy	
Tony Minchell	Teledyne-LeCroy	
Anand Dabak	Texas Instruments	
Bill Waters	Texas Instruments	
Deric Waters	Texas Instruments	Physical Layer WG Lead
Grant Ley	Texas Instruments	
Ingolf Frank	Texas Instruments	
Ivo Huber	Texas Instruments	
Javed Ahmad	Texas Instruments	
Jean Picard	Texas Instruments	
Martin Patoka	Texas Instruments	
Scott Jackson	Texas Instruments	
Srinath Hosur	Texas Instruments	
Steven Tom	Texas Instruments	
Dydron Lin	VIA Technologies, Inc.	
Fong-Jim Wang	VIA Technologies, Inc.	
Jay Tseng	VIA Technologies, Inc.	
Rex Chang	VIA Technologies, Inc.	
Terrance Shih	VIA Technologies, Inc.	
Charles Neumann	Western Digital Technologies, Inc.	
Curtis Stevens	Western Digital Technologies, Inc.	
John Maroney	Western Digital Technologies, Inc.	

Microchip Ex. 1020, p. 12
Microchip v. Aptiv
IPR2024-00496

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
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

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.

Please send comments via electronic mail to techsup@usb.org

For industry information, refer to the USB Implementers Forum web page at http://www.usb.org

All product names are trademarks, registered trademarks, or service marks of their respective owners.

Copyright © 2010-2016 Hewlett-Packard Company, Intel Corporation, Microsoft Corporation, Renesas, STMicroelectronics, and Texas Instruments

All rights reserved.

Microchip Ex. 1020, p. 13
Microchip v. Aptiv
IPR2024-00496

Table of Contents

Contributors 2

Revision History 13

Table of Contents 14

List of Tables 23

List of Figures 29

1. Introduction 36

1.1 Overview 36

1.2 Purpose 37

1.3 Scope 37

1.4 Conventions 37

1.4.1 Precedence 37

1.4.2 Keywords 37

1.4.3 Numbering 38

1.5 Related Documents 39

1.6 Terms and Abbreviations 39

1.7 Parameter Values 45

1.8 Changes From Revision 2.0 46

1.9 Compatibility with Revision 2.0 46

2. Overview 47

2.1 Introduction 47

2.2 Section Overview 48

2.3 Revision 2.0 Changes and Compatibility 49

2.3.1 Changes From Revision 2.0 49

2.3.2 Compatibility with Revision 2.0 49

2.4 USB Power Delivery Capable Devices 50

2.5 SOP* Communication 51

2.5.1 Introduction 51

2.5.2 SOP* Collision Avoidance 51

2.5.3 SOP Communication 51

Microchip Ex. 1020, p. 14
Microchip v. Aptiv
IPR2024-00496

2.5.4 SOP'/SOP'' Communication with Cable Plugs 51
2.6 Operational Overview 53
2.6.1 Source Operation 53
2.6.2 Sink Operation 56
2.6.3 Cable Plugs 58
2.7 Architectural Overview 59
2.7.1 Policy 61
2.7.2 Message Formation and Transmission 62
2.7.3 Collision Avoidance 62
2.7.4 Power supply 63
2.7.5 DFP/UFP 63
2.7.6 VCONN Source 63
2.7.7 Cable and Connectors 64
2.7.8 Interactions between Non-PD, BC and PD devices 64
2.7.9 Power Rules 64
3. USB Type-A and USB Type-B Cable Assemblies and Connectors 65
4. Electrical Requirements 66
4.1 Interoperability with other USB Specifications 66
4.2 Dead Battery Detection / Unpowered Port Detection 66
4.3 Cable IR Ground Drop (IR Drop) 66
4.4 Cable Type Detection 66
5. Physical Layer 67
5.1 Physical Layer Overview 67
5.2 Physical Layer Functions 67
5.3 Symbol Encoding 68
5.4 Ordered Sets 69
5.5 Transmitted Bit Ordering 70
5.6 Packet Format 71
5.6.1 Packet Framing 71
5.6.2 CRC 73
5.6.3 Packet Detection Errors 75
5.6.4 Hard Reset 75
5.6.5 Cable Reset 76

Microchip Ex. 1020, p. 15
Microchip v. Aptiv
IPR2024-00496

5.7 Collision Avoidance 76
5.8 Biphase Mark Coding (BMC) Signaling Scheme 77
5.8.1 Encoding and signaling 77
5.8.2 Transmit and Receive Masks 81
5.8.3 Transmitter Load Model 88
5.8.4 BMC Common specifications 89
5.8.5 BMC Transmitter Specifications 89
5.8.6 BMC Receiver Specifications 92
5.9 Built in Self-Test (BIST) 95
5.9.1 BIST Carrier Mode 95
5.9.2 BIST Test Data 95
6. Protocol Layer 96
6.1 Overview 96
6.2 Messages 96
6.2.1 Message Construction 96
6.3 Control Message 106
6.3.1 GoodCRC Message 107
6.3.2 GotoMin Message 107
6.3.3 Accept Message 107
6.3.4 Reject Message 108
6.3.5 Ping Message 108
6.3.6 PS_RDY Message 108
6.3.7 Get_Source_Cap Message 108
6.3.8 Get_Sink_Cap Message 108
6.3.9 DR_Swap Message 108
6.3.10 PR_Swap Message 109
6.3.11 VCONN_Swap Message 110
6.3.12 Wait Message 110
6.3.13 Soft Reset Message 111
6.3.14 Not_Supported Message 111
6.3.15 Get_Source_Cap_Extended Message 111
6.3.16 Get_Status Message 111
6.3.17 FR_Swap Message 112

Microchip Ex. 1020, p. 16
Microchip v. Aptiv
IPR2024-00496

6.4 Data Message 112
6.4.1 Capabilities Message 113
6.4.2 Request Message 120
6.4.3 BIST Message 124
6.4.4 Vendor Defined Message 126
6.4.5 Battery_Status Message 147
6.4.6 Alert Message 148
6.5 Extended Message 150
6.5.1 Source_Capabilities_Extended Message 150
6.5.2 Status Message 154
6.5.3 Get_Battery_Cap Message 156
6.5.4 Get_Battery_Status Message 156
6.5.5 Battery_Capabilities Message 156
6.5.6 Get_Manufacturer_Info Message 158
6.5.7 Manufacturer_Info Message 158
6.5.8 Security Messages 159
6.5.9 Firmware Update Messages 160
6.6 Timers 160
6.6.1 CRCReceiveTimer 160
6.6.2 SenderResponseTimer 161
6.6.3 Capability Timers 161
6.6.4 SinkRequestTimer 162
6.6.5 Power Supply Timers 162
6.6.6 NoResponseTimer 163
6.6.7 BIST Timers 164
6.6.8 Power Role Swap Timers 164
6.6.9 Soft Reset Timers 164
6.6.10 Hard Reset Timers 165
6.6.11 Structured VDM Timers 165
6.6.12 VCONN Timers 166
6.6.13 tCableMessage 167
6.6.14 DiscoverIdentityTimer 167
6.6.15 Collision Avoidance Timers 167
6.6.16 tFRSwapInit 167

Microchip Ex. 1020, p. 17
Microchip v. Aptiv
IPR2024-00496

6.6.17 Time Values and Timers 167
6.7 Counters 170
6.7.1 MessageID Counter 170
6.7.2 Retry Counter 170
6.7.3 Hard Reset Counter 171
6.7.4 Capabilities Counter 171
6.7.5 Discover Identity Counter 171
6.7.6 VDMBusyCounter 171
6.7.7 Counter Values and Counters 171
6.8 Reset 172
6.8.1 Soft Reset and Protocol Error 172
6.8.2 Hard Reset 172
6.8.3 Cable Reset 173
6.9 Collision Avoidance 173
6.10 Message Discarding 174
6.11 State behavior 175
6.11.1 Introduction to state diagrams used in Chapter 6 175
6.11.2 State Operation 175
6.11.3 List of Protocol Layer States 187
6.12 Message Applicability 188
6.12.1 Applicability of Control Messages 189
6.12.2 Applicability of Data Messages 190
6.12.3 Applicability of Extended Messages 190
6.12.4 Applicability of VDM Commands 191
6.12.5 Applicability of Reset Signaling 192
6.12.6 Applicability of Fast Role Swap signal 192
6.13 Value Parameters 194
7. Power Supply 195
7.1 Source Requirements 195
7.1.1 Behavioral Aspects 195
7.1.2 Source Bulk Capacitance 195
7.1.3 Types of Sources 195
7.1.4 Positive Voltage Transitions 196

Microchip Ex. 1020, p. 18
Microchip v. Aptiv
IPR2024-00496

7.1.5 Negative Voltage Transitions 196
7.1.6 Response to Hard Resets 197
7.1.7 Changing the Output Power Capability 198
7.1.8 Robust Source Operation 198
7.1.9 Output Voltage Tolerance and Range 199
7.1.10 Charging and Discharging the Bulk Capacitance on V_{BUS} 200
7.1.11 Swap Standby for Sources 200
7.1.12 Source Peak Current Operation 200
7.1.13 Source Capabilities Extended Parameters 202
7.1.14 Fast Role Swap 204
7.2 Sink Requirements 206
7.2.1 Behavioral Aspects 206
7.2.2 Sink Bulk Capacitance 206
7.2.3 Sink Standby 206
7.2.4 Suspend Power Consumption 207
7.2.5 Zero Negotiated Current 207
7.2.6 Transient Load Behavior 207
7.2.7 Swap Standby for Sinks 207
7.2.8 Sink Peak Current Operation 207
7.2.9 Robust Sink Operation 207
7.2.10 Fast Role Swap 209
7.3 Transitions 210
7.3.1 Increasing the Current 211
7.3.2 Increasing the Voltage 213
7.3.3 Increasing the Voltage and Current 215
7.3.4 Increasing the Voltage and Decreasing the Current 217
7.3.5 Decreasing the Voltage and Increasing the Current 219
7.3.6 Decreasing the Current 221
7.3.7 Decreasing the Voltage 223
7.3.8 Decreasing the Voltage and the Current 225
7.3.9 Sink Requested Power Role Swap 227
7.3.10 Source Requested Power Role Swap 230
7.3.11 GotoMin Current Decrease 233
7.3.12 Source Initiated Hard Reset 235

Microchip Ex. 1020, p. 19
Microchip v. Aptiv
IPR2024-00496

7.3.13 Sink Initiated Hard Reset 237
7.3.14 No change in Current or Voltage 239
7.3.15 Fast Role Swap 241
7.4 Electrical Parameters 243
7.4.1 Source Electrical Parameters 243
7.4.2 Sink Electrical Parameters 245
7.4.3 Common Electrical Parameters 246
8. Device Policy 247
8.1 Overview 247
8.2 Device Policy Manager 247
8.2.1 Capabilities 248
8.2.2 System Policy 248
8.2.3 Control of Source/Sink 248
8.2.4 Cable Detection 248
8.2.5 Managing Power Requirements 249
8.2.6 Use of "Externally Powered" bit with Batteries and AC supplies 250
8.2.7 Interface to the Policy Engine 252
8.3 Policy Engine 253
8.3.1 Introduction 253
8.3.2 Atomic Message Sequence Diagrams 253
8.3.3 State Diagrams 373
9. States and Status Reporting 453
9.1 Overview 453
9.1.1 PDUSB Device and Hub Requirements 455
9.1.2 Mapping to USB Device States 455
9.1.3 PD Software Stack 458
9.1.4 PDUSB Device Enumeration 458
9.2 PD Class Specific Descriptors 460
9.2.1 USB Power Delivery Capability Descriptor 460
9.2.2 Battery Info Capability Descriptor 461
9.2.3 PD Consumer Port Capability Descriptor 462
9.2.4 PD Provider Port Capability Descriptor 463
9.3 PD Class Specific Requests and Events 465

Microchip Ex. 1020, p. 20
Microchip v. Aptiv
IPR2024-00496

9.3.1 Class-specific Requests 465
9.4 PDUSB Hub and PDUSB Peripheral Device Requests 466
9.4.1 GetBatteryStatus 466
9.4.2 SetPDFeature 467
10. Power Rules 470
10.1 Introduction 470
10.2 Source Power Rules 470
10.2.1 Source Power Rule Considerations 470
10.2.2 Normative Voltages and Currents 471
10.2.3 Optional Voltages/Currents 474
10.2.4 Power sharing between ports 474
10.3 Sink Power Rules 474
10.3.1 Sink Power Rule Considerations 474
10.3.2 Normative Sink Rules 474
A. CRC calculation 475
A.1 C code example 475
A.2 Table showing the full calculation over one Message 477
B. PD Message Sequence Examples 478
B.1 External power is supplied downstream 478
B.2 External power is supplied upstream 482
B.3 Giving back power 489
C. VDM Command Examples 501
C.1 Discover Identity Example 501
C.1.1 Discover Identity Command request 501
C.1.2 Discover Identity Command response – Active Cable 501
C.1.3 Discover Identity Command response – Hub 502
C.2 Discover SVIDs Example 504
C.2.1 Discover SVIDs Command request 504
C.2.1 Discover SVIDs Command response 504
C.3 Discover Modes Example 506
C.3.1 Discover Modes Command request 506
C.3.2 Discover Modes Command response 506
C.4 Enter Mode Example 508

Microchip Ex. 1020, p. 21
Microchip v. Aptiv
IPR2024-00496

C.4.1 Enter Mode Command request 508
C.4.2 Enter Mode Command response 508
C.4.1 Enter Mode Command request with additional VDO 509
C.5 Exit Mode Example 510
C.5.1 Exit Mode Command request 510
C.5.2 Exit Mode Command response 510
C.6 Attention Example 512
C.6.1 Attention Command request 512
C.6.2 Attention Command request with additional VDO 512

Microchip Ex. 1020, p. 22
Microchip v. Aptiv
IPR2024-00496

List of Tables

Table 1-1 Terms and Abbreviations 39
Table 5-1 4b5b Symbol Encoding Table 68
Table 5-2 Ordered Sets 69
Table 5-3 Validation of Ordered Sets 69
Table 5-4 Data Size 70
Table 5-5 SOP ordered set 71
Table 5-6 SOP' ordered set 72
Table 5-7 SOP'' ordered set 72
Table 5-8 SOP'_Debug ordered set 73
Table 5-9 SOP''_Debug ordered set 73
Table 5-10 CRC-32 Mapping 74
Table 5-11 Hard Reset ordered set 75
Table 5-12 Cable Reset ordered set 76
Table 5-13 Rp values used for Collision Avoidance 77
Table 5-14 BMC Tx Mask Definition, X Values 82
Table 5-15 BMC Tx Mask Definition, Y Values 83
Table 5-16 BMC Rx Mask Definition 87
Table 5-17 BMC Common Normative Requirements 89
Table 5-18 BMC Transmitter Normative Requirements 89
Table 5-19 BMC Receiver Normative Requirements 92
Table 6-1 Message Header 97
Table 6-2 Revision Interoperability 99
Table 6-3 Extended Message Header 100
Table 6-4 Use of Unchunked Message Supported bit 102
Table 6-5 Control Message Types 106
Table 6-6 Data Message Types 113
Table 6-7 Power Data Object 114
Table 6-8 Fixed Supply PDO - Source 116
Table 6-9 Fixed Power Source Peak Current Capability 117
Table 6-10 Variable Supply (non-Battery) PDO - Source 118
Table 6-11 Battery Supply PDO - Source 118
Table 6-12 Fixed Supply PDO - Sink 119
Table 6-13 Variable Supply (non-Battery) PDO - Sink 120
Table 6-14 Battery Supply PDO - Sink 120
Table 6-15 Fixed and Variable Request Data Object 121

Microchip Ex. 1020, p. 23
Microchip v. Aptiv
IPR2024-00496

Table 6-16 Fixed and Variable Request Data Object with GiveBack Support 121
Table 6-17 Battery Request Data Object 121
Table 6-18 Battery Request Data Object with GiveBack Support 122
Table 6-19 BIST Data Object 125
Table 6-20 Unstructured VDM Header 127
Table 6-21 Structured VDM Header 127
Table 6-22 Structured VDM Commands 128
Table 6-23 SVID Values 128
Table 6-24 Commands and Responses 130
Table 6-25 ID Header VDO 131
Table 6-26 Product Types (UFP) 132
Table 6-27 Product Types (Cable Plug) 133
Table 6-28 Product Types (DFP) 133
Table 6-29 Cert Stat VDO 133
Table 6-30 Product VDO 134
Table 6-31 Passive Cable VDO 134
Table 6-32 Active Cable VDO 136
Table 6-33 AMA VDO 138
Table 6-34 Discover SVIDs Responder VDO 139
Table 6-35 Battery Status Data Object (BSDO) 147
Table 6-36 Alert Data Object 148
Table 6-37 Extended Message Types 150
Table 6-38 Source Capabilities Extended Data Block (SCEDB) 151
Table 6-39 Status Data Block (SSDB) 155
Table 6-40 Get Battery Cap Data Block (GBCDB) 156
Table 6-41 Get Battery Status Data Block (GBSDB) 156
Table 6-42 Battery Capability Data Block (BCDB) 157
Table 6-43 Get Manufacturer Info Data Block (GMIDB) 158
Table 6-44 Manufacturer Info Data Block (MIDB) 159
Table 6-45 Time Values 168
Table 6-46 Timers 169
Table 6-47 Counter parameters 171
Table 6-48 Counters 172
Table 6-49 Message discarding 174
Table 6-50 Protocol Layer States 187
Table 6-51 Applicability of Control Messages 189

Microchip Ex. 1020, p. 24
Microchip v. Aptiv
IPR2024-00496

Table 6-52 Applicability of Data Messages 190
Table 6-53 Applicability of Extended Messages 191
Table 6-54 Applicability of VDM Commands 192
Table 6-55 Applicability of Reset Signaling 192
Table 6-56 Applicability of Fast Role Swap signal 193
Table 6-57 Value Parameters 194
Table 7-1 Sequence Description for Increasing the Current 212
Table 7-2 Sequence Description for Increasing the Voltage 214
Table 7-3 Sequence Diagram for Increasing the Voltage and Current 216
Table 7-4 Sequence Description for Increasing the Voltage and Decreasing the Current 218
Table 7-5 Sequence Description for Decreasing the Voltage and Increasing the Current 220
Table 7-6 Sequence Description for Decreasing the Current 222
Table 7-7 Sequence Description for Decreasing the Voltage 224
Table 7-8 Sequence Description for Decreasing the Voltage and the Current 226
Table 7-9 Sequence Description for a Sink Requested Power Role Swap 228
Table 7-10 Sequence Description for a Source Requested Power Role Swap 231
Table 7-11 Sequence Description for a GotoMin Current Decrease 234
Table 7-12 Sequence Description for a Source Initiated Hard Reset 236
Table 7-13 Sequence Description for a Sink Initiated Hard Reset 238
Table 7-14 Sequence Description for no change in Current or Voltage 240
Table 7-15 Sequence Description for Fast Role Swap 241
Table 7-16 Source Electrical Parameters 243
Table 7-17 Sink Electrical Parameters 245
Table 7-18 Common Source/Sink Electrical Parameters 246
Table 8-1 Basic Message Flow 254
Table 8-2 Potential issues in Basic Message Flow 255
Table 8-3 Basic Message Flow with CRC failure 256
Table 8-4 Interruptible and Non-interruptible AMS 257
Table 8-5 Steps for a successful Power Negotiation 260
Table 8-6 Steps for a GotoMin Negotiation 263
Table 8-7 Steps for a Soft Reset 265
Table 8-8 Steps for Source initiated Hard Reset 269
Table 8-9 Steps for Sink initiated Hard Reset 272
Table 8-10 Steps for Source initiated Hard Reset – Sink long reset 275
Table 8-11 Steps for a Successful Source Initiated Power Role Swap Sequence 279
Table 8-12 Steps for a Successful Sink Initiated Power Role Swap Sequence 284

Microchip Ex. 1020, p. 25
Microchip v. Aptiv
IPR2024-00496

Table 8-13 Steps for a Successful Fast Role Swap Sequence 289
Table 8-14 Steps for Data Role Swap, UFP operating as Sink initiates 293
Table 8-15 Steps for Data Role Swap, UFP operating as Source initiates 295
Table 8-16 Steps for Data Role Swap, DFP operating as Source initiates 297
Table 8-17 Steps for Data Role Swap, DFP operating as Sink initiates 299
Table 8-18 Steps for Source to Sink VCONN Source Swap 302
Table 8-19 Steps for Sink to Source VCONN Source Swap 305
Table 8-20 Steps for Source Alert to Sink 307
Table 8-21 Steps for Sink Alert to Source 309
Table 8-22 Steps for a Sink getting Source status Sequence 311
Table 8-23 Steps for a Source getting Sink status Sequence 313
Table 8-24 Steps for a Sink getting Source capabilities Sequence 315
Table 8-25 Steps for a Dual-Role Source getting Dual-Role Sink's capabilities as a Source Sequence 317
Table 8-26 Steps for a Source getting Sink capabilities Sequence 319
Table 8-27 Steps for a Dual-Role Sink getting Dual-Role Source capabilities as a Sink Sequence 321
Table 8-28 Steps for a Sink getting Source extended capabilities Sequence 323
Table 8-29 Steps for a Dual-Role Source getting Dual-Role Sink extended capabilities Sequence 325
Table 8-30 Steps for a Sink getting Source Battery capabilities Sequence 327
Table 8-31 Steps for a Source getting Sink Battery capabilities Sequence 329
Table 8-32 Steps for a Source getting Sink's Port Manufacturer information Sequence 331
Table 8-33 Steps for a Source getting Sink's Port Manufacturer information Sequence 333
Table 8-34 Steps for a Source getting Sink's Battery Manufacturer information Sequence 335
Table 8-35 Steps for a Source getting Sink's Battery Manufacturer information Sequence 337
Table 8-36 Steps for a Source getting Sink's Port Manufacturer information Sequence 339
Table 8-37 Steps for a Source requesting a security exchange with a Sink Sequence 341
Table 8-38 Steps for a Sink requesting a security exchange with a Source Sequence 343
Table 8-39 Steps for a Vconn Source requesting a security exchange with a Cable Plug Sequence 345
Table 8-40 Steps for a Source requesting a firmware update exchange with a Sink Sequence 347
Table 8-41 Steps for a Sink requesting a firmware update exchange with a Source Sequence 349
Table 8-42 Steps for a Vconn Source requesting a firmware update exchange with a Cable Plug Sequence 351
Table 8-43 Steps for DFP to UFP Discover Identity 353
Table 8-44 Steps for Source Port to Cable Plug Discover Identity 355
Table 8-45 Steps for DFP to Cable Plug Discover Identity 357
Table 8-46 Steps for DFP to UFP Enter Mode 360
Table 8-47 Steps for DFP to UFP Exit Mode 362
Table 8-48 Steps for DFP to Cable Plug Enter Mode 365

Microchip Ex. 1020, p. 26
Microchip v. Aptiv
IPR2024-00496

Table 8-49 Steps for DFP to Cable Plug Exit Mode 367
Table 8-50 Steps for UFP to DFP Attention 369
Table 8-51 Steps for BIST Eye Pattern Test 372
Table 8-52 Policy Engine States 447
Table 9-1 USB Power Delivery Type Codes 460
Table 9-2 USB Power Delivery Capability Descriptor 460
Table 9-3 Battery Info Capability Descriptor 462
Table 9-4 PD Consumer Port Descriptor 463
Table 9-5 PD Provider Port Descriptor 464
Table 9-6 PD Class Requests 465
Table 9-7 PD Class Request Codes 465
Table 9-8 PD Class Feature Selectors 465
Table 9-9 Battery Status Structure 466
Table 9-10 Battery Wake Mask 468
Table 9-11 Charging Policy Encoding 468
Table 10-1 Considerations for Sources 470
Table 10-2 Normative Voltages and Currents 471
Table 10-3 Fixed Supply PDO – Source 5V 472
Table 10-4 Fixed Supply PDO – Source 9V 473
Table 10-5 Fixed Supply PDO – Source 15V 473
Table 10-6 Fixed Supply PDO – Source 20V 473
Table B-1 External power is supplied downstream 479
Table B-2 External power is supplied upstream 482
Table B-3 Giving back power 489
Table C-1 Discover Identity Command request from Initiator Example 501
Table C-2 Discover Identity Command response from Active Cable Responder Example 501
Table C-3 Discover Identity Command response from Hub Responder Example 503
Table C-4 Discover SVIDs Command request from Initiator Example 504
Table C-5 Discover SVIDs Command response from Responder Example 504
Table C-6 Discover Modes Command request from Initiator Example 506
Table C-7 Discover Modes Command response from Responder Example 506
Table C-8 Enter Mode Command request from Initiator Example 508
Table C-9 Enter Mode Command response from Responder Example 508
Table C-10 Enter Mode Command request from Initiator Example 509
Table C-11 Exit Mode Command request from Initiator Example 510
Table C-12 Exit Mode Command response from Responder Example 510

Microchip Ex. 1020, p. 27
Microchip v. Aptiv
IPR2024-00496

Table C-13 Attention Command request from Initiator Example 512
Table C-14 Attention Command request from Initiator with additional VDO Example 512

Microchip Ex. 1020, p. 28
Microchip v. Aptiv
IPR2024-00496

List of Figures

Figure 2-1 Logical Structure of USB Power Delivery Capable Devices 50
Figure 2-2 Example SOP' Communication between VCONN Source and Cable Plug(s) 52
Figure 2-3 USB Power Delivery Communications Stack 59
Figure 2-4 USB Power Delivery Communication Over USB 60
Figure 2-5 High Level Architecture View 61
Figure 5-1 Interpretation of ordered sets 69
Figure 5-2 Transmit Order for Various Sizes of Data 70
Figure 5-3 USB Power Delivery Packet Format 71
Figure 5-4 CRC 32 generation 74
Figure 5-5 Line format of Hard Reset 76
Figure 5-6 Line format of Cable Reset 76
Figure 5-7 BMC Example 77
Figure 5-8 BMC Transmitter Block Diagram 78
Figure 5-9 BMC Receiver Block Diagram 78
Figure 5-10 BMC Encoded Start of Preamble 79
Figure 5-11 Transmitting or Receiving BMC Encoded Frame Terminated by Zero with High-to-Low Last Transition 79
Figure 5-12 Transmitting or Receiving BMC Encoded Frame Terminated by One with High-to-Low Last Transition 80
Figure 5-13 Transmitting or Receiving BMC Encoded Frame Terminated by Zero with Low to High Last Transition 80
Figure 5-14 Transmitting or Receiving BMC Encoded Frame Terminated by One with Low to High Last Transition 81
Figure 5-15 BMC Tx 'ONE' Mask 81
Figure 5-16 BMC Tx 'ZERO' Mask 82
Figure 5-17 BMC Rx 'ONE' Mask when Sourcing Power 84
Figure 5-18 BMC Rx 'ZERO' Mask when Sourcing Power 85
Figure 5-19 BMC Rx 'ONE' Mask when Power neutral 85
Figure 5-20 BMC Rx 'ZERO' Mask when Power neutral 86
Figure 5-21 BMC Rx 'ONE' Mask when Sinking Power 86
Figure 5-22 BMC Rx 'ZERO' Mask when Sinking Power 87
Figure 5-23 Transmitter Load Model for BMC Tx from a Source 88
Figure 5-24 Transmitter Load Model for BMC Tx from a Sink 88
Figure 5-25 Transmitter diagram illustrating zDriver 90
Figure 5-26 Inter-Frame Gap Timings 91
Figure 5-27 Example Multi-Drop Configuration showing two DRPs 93
Figure 5-28 Example Multi-Drop Configuration showing a DFP and UFP 93
Figure 5-29 Test Data Frame 95
Figure 6-1 USB Power Delivery Packet Format including Control Message Payload 96

Microchip Ex. 1020, p. 29
Microchip v. Aptiv
IPR2024-00496

Figure 6-2 USB Power Delivery Packet Format including Data Message Payload 97
Figure 6-3 USB Power Delivery Packet Format including an Extended Message Header and Payload 97
Figure 6-4 Example Security_Request sequence Unchunked (Chunked bit = 0) 102
Figure 6-5 Example byte transmission for Security_Request Message of Data Size 7 (Chunked bit is set to 0) 103
Figure 6-6 Example byte transmission for Security_Response Message of Data Size 7 (Chunked bit is set to 0) 103
Figure 6-7 Example Security_Request sequence Chunked (Chunked bit = 1) 104
Figure 6-8 Example Security_Request Message of Data Size 7 (Chunked bit set to 1) 105
Figure 6-9 Example Chunk 0 of Security_Response Message of Data Size 30 (Chunked bit set to 1) 105
Figure 6-10 Example byte transmission for a Security_Request Message Chunk request (Chunked bit is set to 1) 105
Figure 6-11 Example Chunk 1 of Security_Response Message of Data Size 30 (Chunked bit set to 1) 106
Figure 6-12 Example Capabilities Message with 2 Power Data Objects 113
Figure 6-13 BIST Message 124
Figure 6-14 Vendor Defined Message 126
Figure 6-15 Discover Identity Command response 131
Figure 6-16 Example Discover SVIDs response with 3 SVIDs 139
Figure 6-17 Example Discover SVIDs response with 4 SVIDs 139
Figure 6-18 Example Discover SVIDs response with 12 SVIDs followed by an empty response 140
Figure 6-19 Example Discover Modes response for a given SVID with 3 Modes 140
Figure 6-20 Successful Enter Mode sequence 141
Figure 6-21 Enter Mode sequence Interrupted by Source Capabilities and then Re-run 142
Figure 6-22 Unsuccessful Enter Mode sequence due to NAK 142
Figure 6-23 Exit Mode sequence 143
Figure 6-24 Attention Command request/response sequence 144
Figure 6-25 Command request/response sequence 144
Figure 6-26 Enter/Exit Mode Process 146
Figure 6-27 Battery_Status Message 147
Figure 6-28 Alert Message 148
Figure 6-29 Source_Capabilities_Extended Message 150
Figure 6-30 Status Message 154
Figure 6-31 Get_Battery_Cap Message 156
Figure 6-32 Get_Battery_Status Message 156
Figure 6-33 Battery_Capabilities Message 157
Figure 6-34 Get_Manufacturer_Info Message 158
Figure 6-35 Manufacturer_Info Message 158
Figure 6-36 Security_Request Message 159
Figure 6-37 Security_Response Message 160

Microchip Ex. 1020, p. 30
Microchip v. Aptiv
IPR2024-00496

Figure 6-38 Firmware_Update_Request Message 160
Figure 6-39 Firmware_Update_Response Message 160
Figure 6-40 Outline of States 175
Figure 6-41 References to states 175
Figure 6-42 Common Protocol Layer Message transmission State Diagram 176
Figure 6-43 Source Protocol Layer Message transmission State Diagram 179
Figure 6-44 Sink Protocol Layer Message transmission State Diagram 181
Figure 6-45 Protocol layer Message reception 182
Figure 6-46 Hard/Cable Reset 184
Figure 7-1 Placement of Source Bulk Capacitance 195
Figure 7-2 Transition Envelope for Positive Voltage Transitions 196
Figure 7-3 Transition Envelope for Negative Voltage Transitions 197
Figure 7-4 Source V_{BUS} Response to Hard Reset 198
Figure 7-5 Application of vSrcNew and vSrcValid limits after tSrcReady 200
Figure 7-6 Source Peak Current Overload 201
Figure 7-7 Holdup Time Measurement 203
Figure 7-8 V_{BUS} Power during Fast Role Swap 204
Figure 7-9 V_{BUS} detection and timing during Fast Role Swap 205
Figure 7-10 Placement of Sink Bulk Capacitance 206
Figure 7-11 Transition Diagram for Increasing the Current 211
Figure 7-12 Transition Diagram for Increasing the Voltage 213
Figure 7-13 Transition Diagram for Increasing the Voltage and Current 215
Figure 7-14 Transition Diagram for Increasing the Voltage and Decreasing the Current 217
Figure 7-15 Transition Diagram for Decreasing the Voltage and Increasing the Current 219
Figure 7-16 Transition Diagram for Decreasing the Current 221
Figure 7-17 Transition Diagram for Decreasing the Voltage 223
Figure 7-18 Transition Diagram for Decreasing the Voltage and the Current 225
Figure 7-19 Transition Diagram for a Sink Requested Power Role Swap 227
Figure 7-20 Transition Diagram for a Source Requested Power Role Swap 230
Figure 7-21 Transition Diagram for a GotoMin Current Decrease 233
Figure 7-22 Transition Diagram for a Source Initiated Hard Reset 235
Figure 7-23 Transition Diagram for a Sink Initiated Hard Reset 237
Figure 7-24 Transition Diagram for no change in Current or Voltage 239
Figure 7-25 Transition Diagram for Fast Role Swap 241
Figure 8-1 Example of daisy chained displays 251
Figure 8-2 Basic Message Exchange (Successful) 253

Microchip Ex. 1020, p. 31
Microchip v. Aptiv
IPR2024-00496

Figure 8-3 Basic Message flow indicating possible errors 254
Figure 8-4 Basic Message Flow with Bad CRC followed by a Retry 256
Figure 8-5 Successful Power Negotiation 259
Figure 8-6 Successful GotoMin operation 263
Figure 8-7 Soft Reset 265
Figure 8-8 Source initiated Hard Reset 268
Figure 8-9 Sink Initiated Hard Reset 271
Figure 8-10 Source initiated reset - Sink long reset 274
Figure 8-11 Successful Power Role Swap Sequence Initiated by the Source 278
Figure 8-12 Successful Power Role Swap Sequence Initiated by the Sink 283
Figure 8-13 Successful Fast Role Swap Sequence 288
Figure 8-14 Data Role Swap, UFP operating as Sink initiates 292
Figure 8-15 Data Role Swap, UFP operating as Source initiates 295
Figure 8-16 Data Role Swap, DFP operating as Source initiates 297
Figure 8-17 Data Role Swap, DFP operating as Sink initiates 299
Figure 8-18 Source to Sink VCONN Source Swap 301
Figure 8-19 Sink to Source VCONN Source Swap 304
Figure 8-20 Source Alert to Sink 307
Figure 8-21 Sink Alert to Source 309
Figure 8-22 Sink Gets Source Status 311
Figure 8-23 Source Gets Sink Status 313
Figure 8-24 Sink Gets Source's Capabilities 315
Figure 8-25 Dual-Role Source Gets Dual-Role Sink's Capabilities as a Source 317
Figure 8-26 Source Gets Sink's Capabilities 319
Figure 8-27 Dual-Role Sink Gets Dual-Role Source's Capabilities as a Sink 321
Figure 8-28 Sink Gets Source's Extended Capabilities 323
Figure 8-29 Dual-Role Source Gets Dual-Role Sink's Extended Capabilities 325
Figure 8-30 Sink Gets Source's Battery Capabilities 327
Figure 8-31 Source Gets Sink's Battery Capabilities 329
Figure 8-32 Source Gets Sink's Port Manufacturer Information 331
Figure 8-33 Sink Gets Source's Port Manufacturer Information 333
Figure 8-34 Source Gets Sink's Battery Manufacturer Information 335
Figure 8-35 Sink Gets Source's Battery Manufacturer Information 337
Figure 8-36 VCONN Source Gets Cable Plug's Manufacturer Information 339
Figure 8-37 Source requests security exchange with Sink 341
Figure 8-38 Sink requests security exchange with Source 343

Microchip Ex. 1020, p. 32
Microchip v. Aptiv
IPR2024-00496

Figure 8-39 Vconn Source requests security exchange with Cable Plug 345
Figure 8-40 Source requests firmware update exchange with Sink 347
Figure 8-41 Sink requests firmware update exchange with Source 349
Figure 8-42 Vconn Source requests firmware update exchange with Cable Plug 351
Figure 8-43 DFP to UFP Discover Identity 353
Figure 8-44 Source Port to Cable Plug Discover Identity 355
Figure 8-45 DFP to Cable Plug Discover Identity 357
Figure 8-46 DFP to UFP Enter Mode 359
Figure 8-47 DFP to UFP Exit Mode 361
Figure 8-48 DFP to Cable Plug Enter Mode 364
Figure 8-49 DFP to Cable Plug Exit Mode 367
Figure 8-50 UFP to DFP Attention 369
Figure 8-51 BIST Carrier Mode Test 371
Figure 8-52 Outline of States 373
Figure 8-53 References to states 373
Figure 8-54 Example of state reference with conditions 373
Figure 8-55 Example of state reference with the same entry and exit 374
Figure 8-56 Source Port Policy Engine state diagram 375
Figure 8-57 Sink Port state diagram 381
Figure 8-58 Source Port Soft Reset and Protocol Error State Diagram 385
Figure 8-59 Sink Port Soft Reset and Protocol Error Diagram 387
Figure 8-60 Source Port Not Supported Message State Diagram 388
Figure 8-61 Sink Port Not Supported Message State Diagram 389
Figure 8-62 Source Port Ping State Diagram 389
Figure 8-63 Source Port Source Alert State Diagram 390
Figure 8-64 Sink Port Source Alert State Diagram 390
Figure 8-65 Sink Port Sink Alert State Diagram 391
Figure 8-66 Source Port Sink Alert State Diagram 391
Figure 8-67 Sink Port Get Source Capabilities Extended state diagram 392
Figure 8-68 Source Give Source Capabilities Extended state diagram 392
Figure 8-69 Sink Port Get Source Status state diagram 393
Figure 8-70 Source Give Source Status state diagram 393
Figure 8-71 Source Port Get Sink Status state diagram 394
Figure 8-72 Sink Give Sink Status state diagram 394
Figure 8-73 Get Battery Capabilities state diagram 395
Figure 8-74 Give Battery Capabilities state diagram 395

Microchip Ex. 1020, p. 33
Microchip v. Aptiv
IPR2024-00496

Figure 8-75 Get Battery Status state diagram 396
Figure 8-76 Give Battery Status state diagram 396
Figure 8-77 Get Manufacturer Information state diagram 397
Figure 8-78 Give Manufacturer Information state diagram 397
Figure 8-79 Send security request state diagram 398
Figure 8-80 Send security response state diagram 398
Figure 8-81 Security response received state diagram 399
Figure 8-82 Send firmware update request state diagram 400
Figure 8-83 Send firmware update response state diagram 400
Figure 8-84 Firmware update response received state diagram 401
Figure 8-85: DFP to UFP Data Role Swap State Diagram 402
Figure 8-86: UFP to DFP Data Role Swap State Diagram 404
Figure 8-87: Dual-Role Port in Source to Sink Power Role Swap State Diagram 407
Figure 8-88: Dual-role Port in Sink to Source Power Role Swap State Diagram 410
Figure 8-89: Dual-Role Port in Source to Sink Fast Role Swap State Diagram 413
Figure 8-90: Dual-role Port in Sink to Source Fast Role Swap State Diagram 416
Figure 8-91 Dual-Role (Source) Get Source Capabilities diagram 418
Figure 8-92 Dual-Role (Source) Give Sink Capabilities diagram 418
Figure 8-93 Dual-Role (Sink) Get Sink Capabilities State Diagram 419
Figure 8-94 Dual-Role (Sink) Give Source Capabilities State Diagram 419
Figure 8-95 Dual-Role (Source) Get Source Capabilities Extended state diagram 420
Figure 8-96 Dual-Role (Source) Give Sink Capabilities diagram 420
Figure 8-97 VCONN Swap State Diagram 421
Figure 8-98 Initiator to Port VDM Discover Identity State Diagram 424
Figure 8-99 Initiator VDM Discover SVIDs State Diagram 425
Figure 8-100 Initiator VDM Discover Modes State Diagram 426
Figure 8-101 Initiator VDM Attention State Diagram 427
Figure 8-102 Responder Structured VDM Discover Identity State Diagram 428
Figure 8-103 Responder Structured VDM Discover SVIDs State Diagram 429
Figure 8-104 Responder Structured VDM Discover Modes State Diagram 430
Figure 8-105 Receiving a Structured VDM Attention State Diagram 431
Figure 8-106 DFP VDM Mode Entry State Diagram 432
Figure 8-107 DFP VDM Mode Exit State Diagram 433
Figure 8-108 UFP Structured VDM Enter Mode State Diagram 434
Figure 8-109 UFP Structured VDM Exit Mode State Diagram 435
Figure 8-110 Cable Ready VDM State Diagram 436

Microchip Ex. 1020, p. 34
Microchip v. Aptiv
IPR2024-00496

Figure 8-111 Cable Plug Soft Reset State Diagram 437
Figure 8-112 Cable Plug Hard Reset State Diagram 437
Figure 8-113 DFP Soft Reset or Cable Reset of a Cable Plug State Diagram 438
Figure 8-114 UFP Source Soft Reset of a Cable Plug State Diagram 439
Figure 8-115 Source Startup Structured VDM Discover Identity State Diagram 440
Figure 8-116 Cable Plug Structured VDM Enter Mode State Diagram 442
Figure 8-117 Cable Plug Structured VDM Exit Mode State Diagram 443
Figure 8-118 BIST Carrier Mode State Diagram 445
Figure 9-1 Example PD Topology 454
Figure 9-2 Mapping of PD Topology to USB 455
Figure 9-3 USB Attached to USB Powered State Transition 456
Figure 9-4 Any USB State to USB Attached State Transition (When operating as a Consumer) 457
Figure 9-5 Any USB State to USB Attached State Transition (When operating as a Provider) 457
Figure 9-6 Any USB State to USB Attached State Transition (After a USB Type-C Data Role Swap) 458
Figure 9-7 Software stack on a PD aware OS 458
Figure 9-8 Enumeration of a PDUSB Device 459
Figure 10-1 Source Power Rule Illustration 471
Figure 10-2 Source Power Rule Example 472
Figure B-1 External Power supplied downstream 478
Figure B-2 External Power supplied upstream 482
Figure B-3 Giving Back Power 489

Microchip Ex. 1020, p. 35
Microchip v. Aptiv
IPR2024-00496

1. Introduction

USB has evolved from a data interface capable of supplying limited power to a primary provider of power with a data interface. Today many devices charge or get their power from USB ports contained in laptops, cars, aircraft or even wall sockets. USB has become a ubiquitous power socket for many small devices such as cell phones, MP3 players and other hand-held devices. Users need USB to fulfill their requirements not only in terms of data but also to provide power to, or charge, their devices simply, often without the need to load a driver, in order to carry out "traditional" USB functions.

There are however, still many devices which either require an additional power connection to the wall, or exceed the USB rated current in order to operate. Increasingly, international regulations require better energy management due to ecological and practical concerns relating to the availability of power. Regulations limit the amount of power available from the wall which has led to a pressing need to optimize power usage. The USB Power Delivery Specification has the potential to minimize waste as it becomes a standard for charging devices that are not satisfied by ***[USBBC 1.2]***.

Wider usage of wireless solutions is an attempt to remove data cabling but the need for "tethered" charging remains. In addition, industrial design requirements drive wired connectivity to do much more over the same connector.

USB Power Delivery is designed to enable the maximum functionality of USB by providing more flexible power delivery along with data over a single cable. Its aim is to operate with and build on the existing USB ecosystem; increasing power levels from existing USB standards, for example Battery Charging, enabling new higher power use cases such as USB powered Hard Disk Drives (HDDs) and printers.

With USB Power Delivery the power direction is no longer fixed. This enables the product with the power (Host or Peripheral) to provide the power. For example, a display with a supply from the wall can power, or charge, a laptop. Alternatively, USB power bricks or chargers are able to supply power to laptops and other battery powered devices through their, traditionally power providing, USB ports.

USB Power Delivery enables hubs to become the means to optimize power management across multiple peripherals by allowing each device to take only the power it requires, and to get more power when required for a given application. For example battery powered devices can get increased charging current and then give it back temporarily when the user's HDD requires spinning up. ***Optionally*** the hubs can communicate with the PC to enable even more intelligent and flexible management of power either automatically or with some level of user intervention.

USB Power Delivery allows Low Power cases such as headsets to negotiate for only the power they require. This provides a simple solution that enables USB devices to operate at their optimal power levels.

The Power Delivery Specification, in addition to providing mechanisms to negotiate power also can be used as a side-band channel for standard and vendor defined messaging. Power Delivery enables alternative modes of operation by providing the mechanisms to discover, enter and exit Alternate Modes. The specification also enables discovery of cable capabilities such as supported speeds and current levels.

1.1 Overview

This specification defines how USB Devices may negotiate for more current and/or higher or lower voltages over the USB cable (using the USB Type-C CC wire as the communications channel) than are defined in the ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** specifications. It allows Devices with greater power requirements than can be met with today's specification to get the power they require to operate from V_{BUS} and negotiate with external power sources (e.g. Wall Warts). In addition, it allows a Source and Sink to swap power roles such that a Device could supply power to the Host. For example, a display could supply power to a notebook to charge its battery.

The USB Power Delivery Specification is guided by the following principles:

1) Works seamlessly with legacy USB Devices
2) Compatible with existing spec-compliant USB cables
3) Minimizes potential damage from non-compliant cables (e.g. 'Y' cables etc.)
4) Optimized for low-cost implementations

Microchip Ex. 1020, p. 36
Microchip v. Aptiv
IPR2024-00496

This specification defines mechanisms to discover, enter and exit Modes defined either by a standard or by a particular vendor. These Modes can be supported either by the Port Partner or by a cable connecting the two Port Partners.

The specification defines mechanisms to discover the capabilities of cables which can communicate using Power Delivery.

This specification adds a mechanism to swap the data roles such that the upstream facing Port becomes the downstream facing Port and vice versa. It also enables a swap of the end supplying V_{CONN} to a powered cable.

1.2 Purpose

The USB Power Delivery specification defines a power delivery system covering all elements of a USB system including: Hosts, Devices, Hubs, Chargers and cable assemblies. This specification describes the architecture, protocols, power supply behavior, connectors and cabling necessary for managing power delivery over USB at up to 100W. This specification is intended to be fully compatible and extend the existing USB infrastructure. It is intended that this specification will allow system OEMs, power supply and peripheral developers adequate flexibility for product versatility and market differentiation without losing backwards compatibility.

USB Power Delivery is designed to operate independently of the existing USB bus defined mechanisms used to negotiate power which are:

- ***[USB 2.0]***, ***[USB 3.1]*** in band requests for high power interfaces.
- ***[USBBC 1.2]*** mechanisms for supplying higher power (not mandated by this specification).
- ***[USB Type-C 1.2]*** mechanisms for supplying higher power

Initial operating conditions remain the USB Default Operation as defined in ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]***.

- The DFP sources ***vSafe5V*** over V_{BUS}.
- The UFP consumes power from V_{BUS}.

1.3 Scope

This specification is intended as an extension to the existing ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** and ***[USBBC 1.2]*** specifications. It addresses only the elements required to implement USB Power Delivery. It is targeted at power supply vendors, manufacturers of ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** and ***[USBBC 1.2]*** Platforms, Devices and cable assemblies.

Normative information is provided to allow interoperability of components designed to this specification. Informative information, when provided, may illustrate possible design implementation.

1.4 Conventions

1.4.1 Precedence

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

1.4.2 Keywords

The following keywords differentiate between the levels of requirements and options.

1.4.2.1 Conditional Normative

Conditional Normative is a keyword used to indicate a feature that is mandatory when another related feature has been implemented. Designers are mandated to implement all such requirements, when the dependent features have been implemented, to ensure interoperability with other compliant Devices.

Microchip Ex. 1020, p. 37
Microchip v. Aptiv
IPR2024-00496

1.4.2.2 Deprecated

Deprecated is a keyword used to indicate a feature, supported in previous releases of the specification, which is no longer supported.

1.4.2.3 Discarded

Discarded is a keyword indicating that a Packet when received shall be thrown away by the PHY Layer and not passed to the Protocol Layer for processing. No ***GoodCRC*** Message shall be sent in response to the Packet.

1.4.2.4 Ignored

Ignored is a keyword indicating Messages or Message fields which, when received, shall result in no action by the receiver, aside from returning a ***GoodCRC*** Message to acknowledge Message receipt.

1.4.2.5 Invalid

Invalid is a keyword when used in relation to a Message indicates that the Message's usage or fields fall outside of the defined specification usage. When ***Invalid*** is used in relation to an Explicit Contract it indicates that a previously established Explicit Contract which can no longer be maintained by the Source.

1.4.2.6 May

May is a keyword that indicates a choice with no implied preference.

1.4.2.7 N/A

N/A is a keyword that indicates that a field or value is not applicable and has no defined value and shall not be checked or used by the recipient.

1.4.2.8 Optional/Optionally/Optional Normative

Optional, Optionally and ***Optional Normative*** are equivalent keywords that describe features not mandated by this specification. However, if an ***Optional*** feature is implemented, the feature shall be implemented as defined by this specification.

1.4.2.9 Reserved

Reserved is a keyword indicating reserved bits, bytes, words, fields, and code values that are set-aside for future standardization. Their use and interpretation may be specified by future extensions to this specification and shall not be utilized or adapted by vendor implementation. A ***Reserved*** bit, byte, word, or field shall be set to zero by the sender and shall be ***Ignored*** by the receiver. ***Reserved*** field values shall not be sent by the sender and shall be ***Ignored*** by the receiver.

1.4.2.10 Shall/Normative

Shall and ***Normative*** are equivalent keywords indicating a mandatory requirement. Designers are mandated to implement all such requirements to ensure interoperability with other compliant Devices.

1.4.2.11 Should

Should is a keyword indicating flexibility of choice with a preferred alternative. Equivalent to the phrase "it is recommended that".

1.4.3 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

Microchip Ex. 1020, p. 38
Microchip v. Aptiv
IPR2024-00496

"h" (e.g., 3Ah) or are preceded by “0x” (e.g. 0xFF00) are hexadecimal values. Numbers not immediately followed by either a "b", “B”, or "h" are decimal values.

1.5 Related Documents

- ***[USB 2.0]*** – Universal Serial Bus Specification, Revision 2.0, plus ECN and Errata http://www.usb.org/developers/docs/usb20_docs/.
- ***[USB 3.1]*** – Universal Serial Bus 3.1 Specification, Revision 1 plus ECN and Errata (this includes the entire document release package including the OTG&EH v3.0 specification). www.usb.org/developers/docs.
- ***[USBTypeCAuthentication 1.0]***, Universal Serial Bus Type-C Authentication Specification, Revision 1.0, March 25, 2016. www.usb.org/developers/docs.
- ***[USBPDFirmwareUpdate 1.0]***, Universal Serial Bus Power Delivery Firmware Update Specification, Revision 1.0. www.usb.org/developers/docs. Expected publication date H2 2016.
- ***[USBBC 1.2]*** – Universal Serial Bus Battery Charging Specification, Revision 1.2 plus Errata (referred to in this document as the Battery Charging specification). www.usb.org/developers/devclass_docs#approved.
- ***[USBBridge 1.0]*** – Universal Serial Bus Type-C Bridge Specification, Revision 1.0, March 25, 2016. www.usb.org/developers/docs.
- ***[USBTypeCBridge 1.0]*** – Universal Serial Bus Type-C Bridge Specification, Revision 1.0, March 25, 2016. www.usb.org/developers/docs.
- ***[USBPD 2.0]*** – Universal Serial Bus Power Delivery Specification, Revision 2, Version 1.2, March 25, 2016. www.usb.org/developers/docs.
- ***[USBPDCompliance]*** – USB Power Delivery Compliance Plan version 1.0 http://www.usb.org/developers/docs/devclass_docs/.
- ***[USB Type-C 1.2]*** – Universal Serial Bus Type-C Cable and Connector Specification, Revision 1.2, March 25, 2016. www.usb.org/developers/docs.
- ***[IEC 60958-1]*** IEC 60958-1 Digital Audio Interface Part:1 General Edition 3.0 2008-09 www.iec.ch
- ***[IEC 60950-1]*** IEC 60950-1:2005 Information technology equipment – Safety – Part 1: General requirements: Amendment 1:2009, Amendment 2:2013
- ***[IEC 62368-1]*** IEC 62368-1 Audio/Video, information and communication technology equipment – Part 1: Safety requirements
- ***[IEC 63002]*** Draft CD for IEC 63002 Identification and Communication Interoperability Method for External DC Power Supplies Used With Portable Computing Devices.

1.6 Terms and Abbreviations

This section defines terms used throughout this document. For additional terms that pertain to the Universal Serial Bus, see Chapter 2, “Terms and Abbreviations,” in ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** and ***[USBBC 1.2]***.

Table 1-1 Terms and Abbreviations

Term	Description
Active Cable	A cable with a USB Plug on each end at least one of which is a Cable Plug supporting SOP’, that also incorporates data bus signal conditioning circuits. The cable supports the Structured VDM ***Discover Identity*** Command to determine its characteristics in addition to other Structured VDM Commands (Electronically Marked Cable see ***[USB Type-C 1.2]***).
Active Mode	A Mode which has been entered and not exited.
Alternate Mode	As defined in ***[USB Type-C 1.2]***. Equivalent to Mode in the PD Specification.
Alternate Mode Adapter (AMA)	A PDUSB Device which supports Alternate Modes as defined in ***[USB Type-C 1.2]***. Note that since an AMA is a PDUSB Device it has a single UFP that is only addressable by SOP Packets.
Alternate Mode Controller (AMC)	A DFP that supports connection to AMAs as defined in ***[USB Type-C 1.2]***. A DFP that is an AMC can also be a PDUSB Host.
Atomic Message Sequence (AMS)	A fixed sequence of Messages as defined in Section 8.3.2 typically starting and ending in one of the following states: ***PE_SRC_Ready***, ***PE_SNK_Ready*** or ***PE_CBL_Ready***. An AMS can be Interruptible or Non-interruptible.

Microchip Ex. 1020, p. 39
Microchip v. Aptiv
IPR2024-00496

Term	Description
Attached	USB Power Delivery ports which are mechanically joined with USB cable.
Battery	A power storage device residing behind a Port that can either be a source or sink of power.
Battery Supply	A power supply that directly applies the output of a Battery to V_{BUS}. This is exposed by the Battery Supply PDO (see Section 6.4.1.2.4)
Binary Frequency Shift Keying (BFSK)	A Signaling Scheme now ***Deprecated*** in this specification. BFSK used a pair of discrete frequencies to transmit binary (0s and 1s) information over V_{BUS}. See ***[USBPD 2.0]*** for further details.
Biphase Mark Coding (BMC)	Modification of Manchester coding where each zero has one transition and a one has two transitions (see ***[IEC 60958-1]***).
BIST	Built In Self-Test – 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.
Cable Plug	Term used to describe a PD Capable element in a Multi-Drop system addressed by SOP'/SOP'' Packets. Logically the Cable Plug is associated with a USB plug at one end of the cable. In a practical implementation the electronics may 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 to its default state. It does not reset the Port Partners but does restore V_{CONN} to its Attachment state.
Chunk	A ***MaxExtendedMsgChunkLen*** (26 byte) or less portion of a Data Block. Data Blocks may be sent either as a single Message or as a series of Chunks.
Chunking	The process of breaking up a Data Block larger than ***MaxExtendedMsgLegacyLen*** (26-bytes) into two of more Chunks.
Cold Socket	A Port that does not apply ***vSafe5V*** on V_{BUS} until a Sink is Attached.
Command	Request and response pair defined as part of a Structured Vendor Defined Message (see Section 6.4.4.2)
Configuration Channel (CC)	Single wire used by the BMC PHY Layer Signaling Scheme (see ***[USB Type-C 1.2]***).
Connected	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.
Consumer	The capability of a PD Port (typically a Device's UFP) to sink power from the power conductor (e.g. V_{BUS}). This corresponds to a USB Type-C Port with Rd asserted on its CC Wire.
Consumer/Provider	A Consumer with the additional capability to act as a Provider. This corresponds to a Dual-Role Port with Rd asserted on its CC Wire.
Continuous BIST Mode	A 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 reached between a Port Pair. A Contract may be explicitly negotiated between the Port Pair or may be an Implicit power level defined by the current state. While operating in Power Delivery mode there will always be either an Explicit or Implicit Contract in place. The Contract may only be altered in the case of a (re-)negotiation, Power Role Swap, Data Role Swap, Hard Reset or failure of the Source.
Control Message	A Message is defined as a Control Message when the ***Number of Data Objects*** field in the Message Header is set to 0. The Control Message consists only of a Message Header and a CRC.
Data Block	An Extended Message payload data unit. The size of each type of Data Block is specified as a series of bytes up to ***MaxExtendedMsgLen*** bytes in length. 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 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. Power, Request, BIST and Vendor Data Objects are defined.
Data Role Swap	Process of exchanging the DFP (Host) and UFP (Device) roles between Port Partners using the ***[USB Type-C 1.2]*** connector.

Microchip Ex. 1020, p. 40
Microchip v. Aptiv
IPR2024-00496

Term	Description
Dead Battery	A device has a Dead Battery when the Battery in a device is unable to power its functions.
Detached	USB Power Delivery ports which are no longer mechanically joined with USB cable.
Device	When lower cased (device), it refers to any USB product, either USB Device or USB Host. When in upper case refers to a USB Device (Peripheral or Hub).
Device Policy Manager (DPM)	Module running in a Source or Sink that applies Local Policy to each Port in the Device via the Policy Engine.
Discovery Process	Command sequence using Structured Vendor Defined Messages resulting in identification of the Port Partner, its supported SVIDs and 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 Type-C 1.2]***. At connection the Port defaults to operation as a USB Host (when USB Communication is supported) and Source.
Dual-Role Data (DRD)	Capability of operating as either a DFP or UFP.
Dual-Role Data Port	A Port Capable of operating as either a DFP or UFP.
Dual-Role Power (DRP)	Capability of operating as either a Source or Sink.
Dual-Role Power Device	A product containing one or more Dual-Role Power Ports that are capable of operating as either a Source or a Sink.
Dual-Role Power Port	A Port capable of operating as either a Source or Sink.
End of Packet (EOP)	K-code marker used to delineate the end of a packet.
Enter Mode Process	Command sequence using Structured Vendor Defined Messages resulting in the Port Partners entering a Mode.
Error Recovery	Error recovery process as defined in ***[USB Type-C 1.2]***.
Exit Mode Process	Command sequence using Structured Vendor Defined Messages resulting in the Port Partners exiting a Mode.
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 indicating 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 Sink Policy Engine. The Explicit Contract may be altered through the re-negotiation process. All Port pairs are required to make an Explicit Contract.
Extended Message (EM)	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.
Fast Role Swap	Process of exchanging the Source and Sink roles between Port Partners rapidly due to the disconnection of an external power supply.
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. This is exposed by the Fixed Supply PDO (see Section 6.4.1.2.2)
Frame	Generic term referring to an atomic communication transmitted by PD such as a Packet, Test Frame or Signaling.
Hard Reset	This is initiated by ***Hard Reset*** Signaling from either Port Partner. It restores V_{BUS} 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 V_{CONN} Source to the Source Port.
HDD	A Hard Disk Drive.
Hot Swappable Battery	A Battery that is easily accessible for a user to remove or change for another Battery.
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.

Microchip Ex. 1020, p. 41
Microchip v. Aptiv
IPR2024-00496

Term	Description
Implicit Contract	An agreement on power levels between a Port Pair which occurs, not as a result of the Power Delivery negotiation process, but as a result 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 USB Type-C Current (see ***[USB Type-C 1.2]***).
Initiator	The initial sender of a Command request in the form of a query.
Interruptible	An AMS that, on receiving a Protocol Error, returns to the appropriate ready state in order to process the incoming Message is said to be Interruptible. Every AMS is Interruptible until the first Message in the AMS has been sent (a ***GoodCRC*** Message has been received). An AMS of Vendor Messages is Interruptible during the entire sequence.
IoC	The negotiated current value as defined in ***[IEC 63002]***.
IR Drop	The voltage drop across the cable and connectors between the Source and the Sink. 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 may be the default policy, hard coded or modified by changes in operating parameters or one provided by the system Host or some combination of these. The Local Policy ***Optionally*** may be changed by a System Policy Manager.
Message	The packet payload consisting of a Message Header for Control Messages and a Message Header and data for Data Messages as defined in Section 5.6.1.2.5.
Message Header	Every Message starts with a 16-bit Message Header containing basic information about the Message and the PD Port's Capabilities.
Messaging	Communication in the form of Messages as defined in Chapter 6.
Modal Operation	State where there are one or more Active Modes. Modal Operation ends when there are no longer any Active Modes.
Mode	Operation defined by a Vendor or Standard's organization, which is associated with a SVID, whose definition is outside the scope of USB-IF specifications. Entry to and exit from the Mode uses the Enter Mode and Exit Mode Processes. Modes are equivalent to "Alternate Modes" as described in ***[USB Type-C 1.2]***.
Multi-Drop	Refers to a Power Delivery system with one or more Cable Plugs where communication is to the Cable Plugs rather than the Port Partner. Multi-Drop systems share the Power Delivery communication channel with the Port Partners.
Negotiation	This is the PD process whereby: 1. The Source advertises its capabilities. 2. The Sink requests one of the advertised capabilities. 3. The Source acknowledges the request and alters its output to satisfy the request. The result of the negotiation is a Contract for power delivery/consumption between the Port Pair.
Non-interruptible	An AMS that, on receiving a Protocol Error, generates either a Soft Reset or Hard Reset. Any power related AMS is Non-interruptible once the first Message in the AMS has been sent (a ***GoodCRC*** Message has been received).
OCP	Over-Current Protection
OTP	Over-Temperature Protection
OVP	Over-Voltage Protection
Packet	One entire unit of PD communication including a Preamble, ***SOP****, payload, CRC and ***EOP*** as defined in Section 5.6.
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 Type-C 1.2]***). Note this specification does not discuss Passive Cables which are not Electronically Marked Cables.

Microchip Ex. 1020, p. 42
Microchip v. Aptiv
IPR2024-00496

Term	Description
PD	USB Power Delivery
PD Capable	A Port that supports USB Power Delivery.
PD Connection	See Connected.
PD Power (PDP)	The output power of a Source, as specified by the manufacturer and expressed in Fixed Supply PDOs as defined in Section 10.
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. A PDUSB Host is only addressed by SOP Packets.
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 PDUSB Hub is only addressed by SOP Packets.
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.
PHY Layer	The Physical Layer responsible for sending and receiving Messages across the USB Type-C CC wire between a Port Pair.
Policy	Policy defines 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 in order 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 delivering power from the Source to Sink. For example USB's V_{BUS}.
Power Consumer	See Consumer
Power Data Object (PDO)	Data Object used to expose a Source Port's power capabilities or a Sink's power requirements as part of a ***Source_Capabilities*** or ***Sink_Capabilities*** Message respectively. Fixed, Variable and Battery Power Data Objects are defined.
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 Role Swap. Power Delivery mode can only be exited by Detaching the ports, applying a Hard Reset or by the Source removing power (except when power is removed during the Power Role Swap procedure).
Power Provider	See Provider
Power Reserve	Power which is kept back by a Source in order to ensure that it can meet total power requirements of Attached Sinks on at least one Port.
Power Role Swap	Process of exchanging the Source and Sink roles between Port Partners.
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 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.
Protocol Error	An unexpected Message during an Atomic Message Sequence. A Protocol Error during a Non-interruptible AMS will result in either a Soft Reset or a Hard Reset. A Protocol Error during an Interruptible AMS will result in a return to the appropriate ready state where the Message will be handled.
Protocol Layer	The entity that forms the Messages used to communicate information between Port Partners.

Microchip Ex. 1020, p. 43
Microchip v. Aptiv
IPR2024-00496

Term	Description
Provider	A capability of a PD Port (typically a Host, Hub, or Wall Wart DFP) to source power over the power conductor (e.g. V_{BUS}). This corresponds to a USB Type-C Port with Rp asserted on its CC Wire.
Provider/Consumer	A Provider with the additional capability to act as a Consumer. This corresponds to a Dual-Role Power Port with Rp asserted on its CC Wire.
Rd	Pull-down resistor on the USB Type-C CC wire used to indicate that the Port is a Sink (see ***[USB Type-C 1.2]***).
Re-negotiation	A process wherein one of the Port Partners wants to alter the negotiated Contract.
Request Data Object (RDO)	Data Object used by a Sink Port to negotiate a Contact as a part of a ***Request*** Message.
Re-run	Start an Interruptible AMS again from the beginning after a Protocol Error.
Responder	The receiver of a Command request sent by an Initiator that replies with a Command response.
Rp	Pull-up resistor on the USB Type-C CC wire used to indicate that the Port is a Source (see ***[USB Type-C 1.2]***).
Safe Operation	Sources must have the ability to tolerate ***vSafe5V*** applied by both Port Partners.
Signaling	A Preamble followed by an ordered set of four K-codes used to indicate a particular line symbol e.g. ***Hard Reset*** as defined in Section 5.4.
Signaling Scheme	Physical mechanism used to transmit bits. Only the BMC Signaling Scheme is defined in this specification. Note: the BFSK Signaling Scheme supported in previous Revisions of this specification has been ***Deprecated***.
Single-Role Port	A Port that is a Port only capable of operating as a Source or Sink, but not both.
Sink	The Port consuming power from V_{BUS}; most commonly a Device.
Soft Reset	A process that resets the PD communications engine to its default state.
SOP Communication	Communication using SOP Packets also implies that a Message sequence is being followed.
SOP Packet	Any Power Delivery Packet which starts with an ***SOP***.
SOP* Communication	Communication with a Cable Plug using SOP* Packets, also implies a Message sequence is being followed.
SOP* Packet	A term referring to any Power Delivery Packet starting with either ***SOP***, ***SOP'*** or ***SOP''***.
SOP' Communication	Communication with a Cable Plug using SOP' Packets, also implies that a Message sequence 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 a Message sequence 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.
Source	A role a Port is currently taking to supply power over V_{BUS}; most commonly a Host or Hub downstream port.
Standard ID (SID)	16-bit unsigned value assigned by the USB-IF to a given industry standard.
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".
Start of Packet (SOP)	K-code marker used to delineate the start of a packet. Three start of packet sequences are defined: ***SOP, SOP'*** and ***SOP''***, with ***SOP**** used to refer to all three in place of ***SOP***/***SOP'***/***SOP''***.
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 Consumers and Providers that are also connected to the Host via USB.
Test Frame	Frame consisting of a Preamble, ***SOP****, followed by test data (See Section 5.9).
Test Pattern	Continuous stream of test data in a given sequence (See Section 5.9)
Tester	The Tester is assumed to be a piece of test equipment that manages the BIST testing process of a PD UUT.

Microchip Ex. 1020, p. 44
Microchip v. Aptiv
IPR2024-00496

Term	Description
Unit Interval (UI)	The time to transmit a single data bit on the wire.
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.
Upstream Facing Port (UFP)	Indicates the Port's position in the USB topology typically a Port on a Device as defined in ***[USB Type-C 1.2]***. At connection the Port defaults to operation as a USB Device (when USB Communication is supported) and Sink.
USB Attached State	Synonymous with the ***[USB 2.0]***] and ***[USB 3.1]*** definition of the Attached state
USB Default Operation	Operation of a Port at Attach or after a Hard Reset where the DFP Source applies ***vSafe0V*** or ***vSafe5V*** on V_{BUS} and the UFP Sink is operating at ***vSafe5V*** as defined in ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]***.
USB Device	Either a hub or a peripheral device as defined in ***[USB 2.0]*** and ***[USB 3.1]***.
USB Host	The host computer system where the USB host controller is installed as defined in ***[USB 2.0]*** and ***[USB 3.1]***.
USB Powered State	Synonymous with the ***[USB 2.0]*** and ***[USB 3.1]*** 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 Type-C 1.2]***) 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 Micro-A plugs and Standard-A plugs and receptacles. 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 Micro-B plugs and Standard-B plugs and receptacles, including the PD and non-PD versions. Micro-AB receptacles are assumed to be a combination of USB Type-A and USB Type-B.
USB Type-C	Term used to refer to the USB Type-C connector plug or receptacle as defined in ***[USB Type-C 1.2]***.
USB-IF PD SID (PD SID)	Standard ID allocated to this specification by the USB Implementer's Forum.
Variable Supply	A very poorly regulated power supply that is not a Battery. This is exposed by the Variable Supply PDO (see Section 6.4.1.2.3).
V_{CONN} Powered Accessory	An accessory that is powered from V_{CONN} to operate in a Mode (see ***[USB Type-C 1.2]***).
V_{CONN} Source	The USB Type-C Port responsible for sourcing V_{CONN}.
V_{CONN} Swap	Process of exchanging the V_{CONN} Source between Port Partners.
VDM Header	The first Data Object following the Message Header in a Vendor Defined Message. The VDM Header contains the SVID relating to the VDM being sent and provides information relating to the Command in the case of a Structured VDM (see Section 6.4.4).
Vendor Data Object (VDO)	Data Object used to send Vendor specific information as part of a ***Vendor_Defined*** Message.
Vendor Defined Message (VDM)	PD Data Message defined for vendor/standards usage. These are further partitioned into Structured VDM Messages, where Commands are defined in this specification, and Unstructured VDM Messages which are entirely Vendor Defined (see Section 6.4.4).
Vendor ID (VID)	16-bit unsigned value assigned by the USB-IF to a given Vendor.
VI	Same as power (i.e. voltage * current = power)
Wall Wart	A power supply or "power brick" that is plugged into an AC outlet. It supplies DC power to power a device or charge a Battery.

1.7 Parameter Values

The parameters in this specification are expressed in terms of absolute values. For details of how each parameter is measured in compliance please see ***[USBPDCompliance]***.

Microchip Ex. 1020, p. 45
Microchip v. Aptiv
IPR2024-00496

1.8 Changes From Revision 2.0

This specification includes the following updates:

- PD Power rules (also applied to *[USBPD 2.0]*)
- mechanisms to avoid collisions and simplify communication
- support for *[IEC 63002]* included extended power supply capabilities and status
- battery capabilities and status
- ability to perform a fast power role swap
- support for *[USBTypeCAuthentication 1.0]* and *[USBPDFirmwareUpdate 1.0]*

The following have been ***Deprecated*** from this specification:

- the BFSK signaling scheme
- definitions of Standard/Micro A/B cables and connectors
- dead battery operation for A/B Ports
- Profiles (replaced by PD Power Rules)

The following have been moved to other specifications:

- System Policy which is now defined in *[USBTypeCBridge 1.0]*

For more details see Section 2.3.1.

1.9 Compatibility with Revision 2.0

Revision 3.0 of the USB Power Delivery specification is designed to be fully interoperable with *[USBPD 2.0]* systems using BMC signaling over the *[USB Type-C 1.2]* connector and to be compatible with Revision 2.0 hardware.

Please see Section 2.3.2 for more details of the mechanisms defined to enable compatibility.

Microchip Ex. 1020, p. 46
Microchip v. Aptiv
IPR2024-00496

2. Overview

This section contains no ***Normative*** requirements.

2.1 Introduction

In USB Power Delivery, pairs of directly Attached ports negotiate voltage, current and/or direction of power flow over the USB cable, using the USB Type-C connector's CC wire as the communications channel. The mechanisms used, operate independently of other USB methods used to negotiate power.

USB Power Delivery also acts as a side-band channel enabling support for Standard or Vendor defined Modal Operation. Modes are associated with a Standard or Vendor ID (SVID). Power Delivery Structured VDM Messages can be used to discover supported SVIDs and Modes and then to enter and exit Modes as required. Multiple Active Modes can also be in operation at the same time.

Any Contract negotiated using this specification, supersedes any and all previous power contracts established whether from standard ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** mechanisms. While in Power Delivery Mode there will be a Contract in place (either Explicit or Implicit) determining the power level available and the direction of that power. The Port Pair remains in Power Delivery Mode until the Port Pair is Detached, there is a Hard Reset or the Source removes power (except during a Power Role Swap or Fast Role Swap when the initial Source removes power in order to for the new Source to apply power).

An Explicit Contract is negotiated by the process of the Source sending a set of Capabilities, from which the Sink is required to request a particular capability and then the Source accepting this request.

An Implicit Contract is the specified level of power allowed in particular states (i.e. during and after a Power Role Swap or Fast Role Swap). Implicit Contracts are temporary; Port Pairs are required to immediately negotiate an Explicit Contract.

Each Provider has a Local Policy, governing power allocation to its Ports. Sinks also have their own Local Policy governing how they draw power. A System Policy can be enacted over USB that allows modification to these local policies and hence management of overall power allocation in the system.

When PD Capable devices are Attached to each other, the DFPs and UFPs initially default to standard USB Default Operation. The DFP supplies ***vSafe5V*** and the UFP draws current in accordance with the rules defined by ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** specifications. After Power Delivery negotiation has taken place power can be supplied at higher, or lower, voltages and higher currents than defined in these specifications. It is also possible to perform a Power Role Swap or Fast Role Swap to exchange the power supply roles such that the DFP receives power and the UFP supplies power, to perform a Data Role Swap such that the DFP becomes the UFP and vice-versa and to perform a V_{CONN} Swap to change the end supplying V_{CONN} to the cable.

Prior to an Explicit Contract the Source can discover the capabilities of the Attached cable assembly. This is important for ***[USB Type-C 1.2]*** where 5A cabling is marked as well as other details of the cable assembly such as the supported speed. Cable discovery occurs on initial Attachment of a Port Pair, before an Explicit Contract has been established where the DFP is the Source. It is also possible to carry out cable discovery after a Power Role Swap or Fast Role Swap prior to establishing an Explicit Contract, where the UFP is the Source and an Implicit Contract is in place.

Once an Explicit Contract is in place only the DFP is permitted to communicate with the Attached cable assembly. This communication can consist of discovering capabilities but may also include discover of SVIDs, Modes and the entering/exiting of Modes supported by the cable assembly.

Microchip Ex. 1020, p. 47
Microchip v. Aptiv
IPR2024-00496

2.2 Section Overview

This specification contains the following sections:

Section 1	Introduction, conventions used in the document, list of terms and abbreviations, references and details of parameter usage.
Section 2	Overview of the document including a description of the operation of PD and the architecture.
Section 3	Mechanical and electrical characteristics of the cables and connectors used by PD. Section ***Deprecated***. See ***[USBPD 2.0]*** for legacy PD connector specification.
Section 4	Electrical requirements for Dead Battery operation and cable detection.
Section 5	Details of the PD PHY Layer requirements
Section 6	Protocol Layer requirements including the Messages, timers, counters and state operation.
Section 7	Power supply requirements for both Providers and Consumers.
Section 8	Device Policy Manager requirements. Policy Engine Message sequence diagrams and state diagrams
Section 9	USBPD Device requirements including mapping of V_{BUS} to USB states. System Policy Manager requirements including descriptors, events and requests.
Section 10	Rated Output Power definitions for PD.
Appendix A	Example CRC calculations.
Appendix B	Scenarios illustrating Device Policy Manager operation.
Appendix C	Examples of Structured VDM usage.

Microchip Ex. 1020, p. 48
Microchip v. Aptiv
IPR2024-00496

2.3 Revision 2.0 Changes and Compatibility

2.3.1 Changes From Revision 2.0

The following is a summary of the major changes between this specification (USB PD Revision 3.0) and ***[USBPD 2.0]***:

- Support for both Revision 2.0 and Revision 3.0 operation mandated for products to ensure backwards compatibility with existing products (see Section 6.2.1.1.5)
- Profiles ***Deprecated*** and replaced with PD Power Rules (see Section 2.7.9)
    - Change also applied to ***[USBPD 2.0]***
- BFSK support ***Deprecated*** including legacy cables, legacy connectors, legacy dead battery operation and related test modes
- Extended Messages with a data payload of up to 260 bytes defined (see Section 6.2.1.2):
    - Support for Chunking of Extended Messages to ***[USBPD 2.0]*** size mandated to enable compatibility with legacy PD hardware.
- Only the VCONN Source is allowed to communicate with the Cable Plugs (see Section 2.5.4)
- Source coordinated collision avoidance scheme to enable either the Source or Sink to initiate an Atomic Message Sequence (AMS):
    - ***[USB Type-C 1.2]*** Rp resistor values used by the Source to indicate when the Sink can/cannot initiate an AMS to either the Source or a Cable Plug (see Section 2.7.3)
    - Either the Source or Sink can initiate a Structured Vendor Defined AMS
    - Either the Source or the Sink can communicate with the Cable Plugs provided they are sourcing VCONN
    - Limitations on timing for ***Attention*** Commands removed
- Fast Role Swap defined to enable externally powered docks and hubs to rapidly switch to bus power when their external power supply is removed (see Section 6.3.17)
- Additional status and discovery of:
    - Power Supply extended capabilities and status
    - Battery capabilities and status
    - Manufacturer defined information
- Changes to fields in the Passive Cable, Active Cable and AMA VDOs indicated by a change in the Structured VDM Version to 2.0 (see Section 6.4.4.2).
- Support for USB Security related requests and responses (see ***[USBTypeCAuthentication 1.0]***)
- Support for USB PD firmware update requests and responses (see ***[USBPDFirmwareUpdate 1.0]***)
- System Policy now references ***[USBTypeCBridge 1.0]***

2.3.2 Compatibility with Revision 2.0

Revision 3.0 of the USB Power Delivery specification is designed to be fully interoperable with ***[USBPD 2.0]*** systems using BMC signaling over the ***[USB Type-C 1.2]*** connector and to be compatible with Revision 2.0 hardware.

This specification mandates that all Revision 3.0 systems fully support Revision 2.0 operation. They must discover the supported Revision used by their Port Partner and any connected Cable Plugs and revert to operation using the lowest common Revision number (see Section 6.2.1.1.5).

This specification defines Extended Messages containing data of up to 260 bytes (see Section 6.2.1.2). These Messages will be larger than expected by existing PHY HW. To accommodate Revision 2.0 based systems a Chunking mechanism is mandated such that Messages are limited to Revision 2.0 sizes unless it is discovered that both systems support the longer Message lengths.

This specification includes changes to the Vendor Defined Objects (VDO) used in the discovery of passive/active marked cables and Alternate Mode Adapters (AMA) (see Section 6.4.4.2). To enable systems to determine which VDO format is being used the Structured Vendor Defined Message (SVDM) version number has been incremented to 2.0.

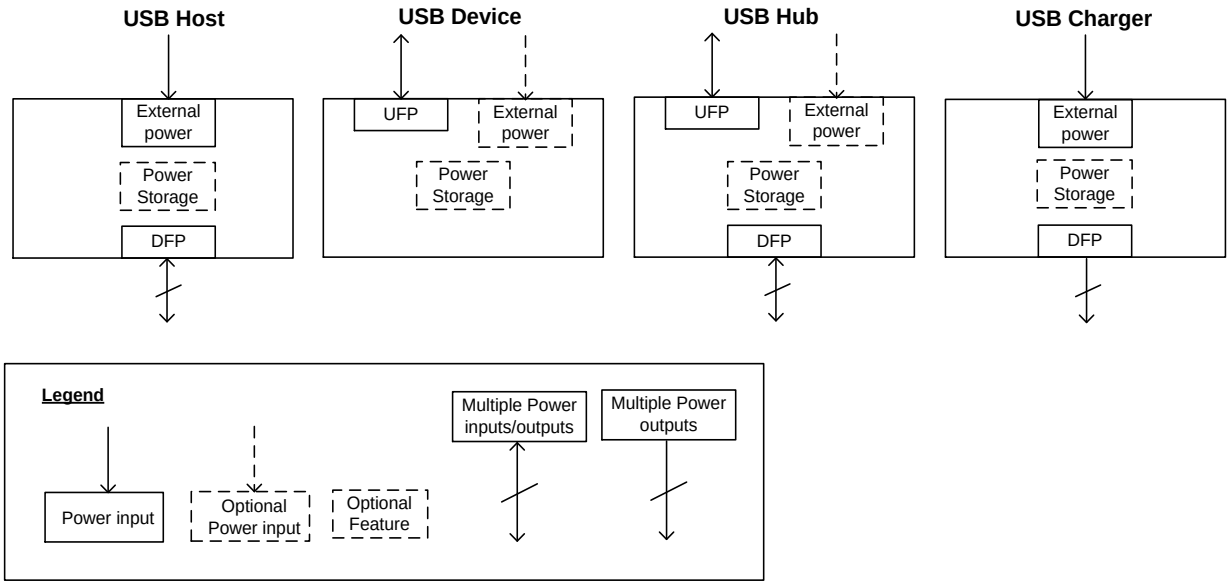
Microchip Ex. 1020, p. 49
Microchip v. Aptiv
IPR2024-00496

Version numbers have also been incorporated into the VDOs themselves to facilitate future changes should these become necessary.

2.4 USB Power Delivery Capable Devices

Some examples of USB Power Delivery capable devices can be seen in Figure 2-1 (a Host, a Device, a Hub, and a Charger). These are given for reference only and do not limit the possible configurations of products that can be built using this specification.

Figure 2-1 Logical Structure of USB Power Delivery Capable Devices



Each USB Power Delivery capable device is assumed to be made up of at least one Port. Providers are assumed to have a Source and Consumers a Sink. Each device contains one, or more, of the following components:

- UFPs that:
  - Sink power
  - ***Optionally*** source power (a Dual-Role Power Device).
  - ***Optionally*** communicate via USB.
  - Communicate using SOP Packets.
  - ***Optionally*** Communicate using SOP* Packets.
- DFPs that:
  - Source power
  - ***Optionally*** Sink power (a Dual-Role Power Device).
  - ***Optionally*** communicate via USB.
  - Communicate using SOP Packets
  - ***Optionally*** Communicate using SOP* Packets.
- A Source that may be:
  - An external power source e.g. AC.
  - Power Storage (e.g. Battery).
  - Derived from another Port (e.g. bus-powered Hub).

Microchip Ex. 1020, p. 50
Microchip v. Aptiv
IPR2024-00496

- A Sink that may be:
    - Power Storage (e.g. a Battery).
    - Used to power internal functions.
    - Used to power devices Attached to other devices (e.g. a bus-powered Hub).
- A Vconn Source that:
    - May be either Port Partner, either the DFP/UFP or Source/Sink
    - Powers the Cable Plug(s)
    - Is the only Port allowed to talk to the Cable Plug(s) at any given time

2.5 SOP* Communication

2.5.1 Introduction

The Start of Packet (or SOP) is used as an addressing scheme to identify whether the Communications were intended for one of the Port Partners (SOP Communication) or one of the Cable Plugs (SOP'/SOP'' Communication). SOP/SOP' and SOP'' are collectively referred to as SOP*. The term Cable Plug in the SOP'/SOP'' Communication case is used to represent a logical entity in the cable which is capable of PD Communication and which may or may not be physically located in the plug.

The following sections describe how this addressing scheme operates for Port to Port and Port to Cable Plug Communication.

2.5.2 SOP* Collision Avoidance

For all SOP* the Source co-ordinates communication in order to avoid bus collisions by allowing the Sink to initiate messaging when it does not need to communicate itself. Once an Explicit Contract is in place the Source indicates to the Sink that it can initiate a message sequence. This sequence can be communication with the Source or with one of the Cable Plugs. As soon as the Source itself needs to initiate a message sequence this will be indicated to the Sink. The Source then waits for any outstanding Sink SOP* Communication to complete before initiating a message sequence itself.

2.5.3 SOP Communication

SOP Communication is used for Port to Port communication between the Source and the Sink. SOP Communication is recognized by both Port Partners but not by any intervening Cable Plugs. SOP Communication takes priority over other SOP* Communications since it is critical to complete power related operations as soon as possible. Message sequences relating to power are also allowed to interrupt other sequences to ensure that negotiation and control of power is given priority on the bus.

2.5.4 SOP'/SOP'' Communication with Cable Plugs

SOP' Communication is recognized by electronics in one Cable Plug which is the Cable Plug that detected VCONN at Attach (see ***[USB Type-C 1.2]***). SOP'' Communication can also be supported when SOP' Communication is also supported. SOP'' Communication is recognized by the electronics in the Cable Plug that did not detect VCONN at Attach.

The Vconn Source is the DFP/Source at attach although all of these roles can later be swapped using PD messaging.

SOP Communication between the Port Partners is not recognized by the Cable Plug. Figure 2-2 outlines the usage of SOP* Communications between a VCONN Source (DFP/UFP) and the Cable Plugs.

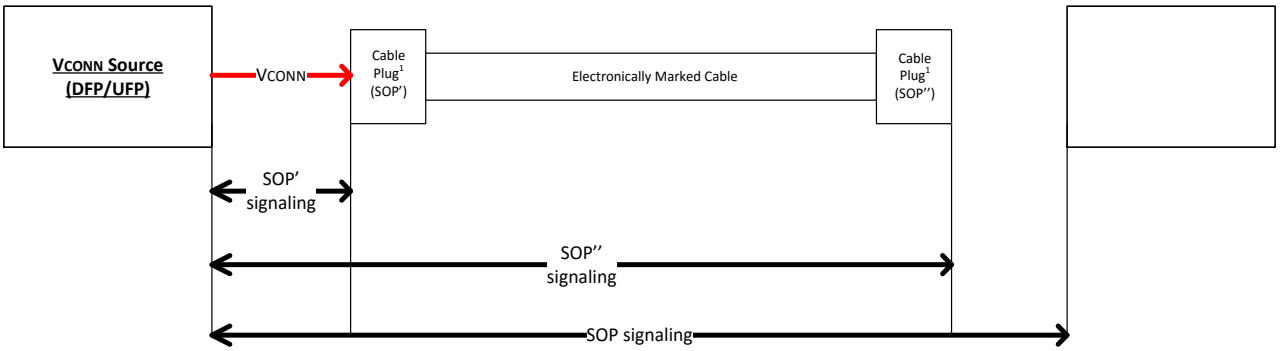
All SOP* Communications take place over a single wire (CC). This means that the SOP* Communication periods must be coordinated to prevent important communication from being blocked. For a product which does not recognize SOP/SOP' or SOP'' Packets, this will look like a non-idle channel, leading to missed packets and retries. Communications between the Port Partners take precedence meaning that communications with the Cable Plug can be interrupted, but will not lead to a Soft or Hard Reset.

Microchip Ex. 1020, p. 51
Microchip v. Aptiv
IPR2024-00496

When no Contract or an Implicit Contract is in place (e.g. after a Power Role Swap or Fast Role Swap) the Source (which can be either the DFP or UFP but must also be the VCONN Source) can communicate with a Cable Plug using SOP' Packets in order to discover its characteristics (see Figure 2-2). During this phase all communication with the Cable Plug is initiated and controlled by the Source which acts to prevent conflicts between SOP* Packets. The Sink does not communicate with the Cable Plug, even if it is the DFP, and Discards any SOP' Packets received.

When an Explicit Contract is in place the VCONN Source (either the DFP or the UFP) can communicate with the Cable Plug(s) using SOP'/SOP'' Packets (see Figure 2-2). During this phase all communication with the Cable Plug is initiated and controlled by the VCONN Source which acts to prevent conflicts between SOP* Packets. The Port that is not the VCONN Source does not communicate with the Cable Plug and does not recognize any SOP'/SOP'' Packets received. Only the DFP, when acting as a VCONN Source, is allowed to send SOP* in order to control the entry and exiting of Modes and to manage Modal Operation.

Figure 2-2 Example SOP' Communication between VCONN Source and Cable Plug(s)



[1] Cable Plug can be physically Attached to either the DFP or UFP.

Microchip Ex. 1020, p. 52
Microchip v. Aptiv
IPR2024-00496

2.6 Operational Overview

A USB Power Delivery Port supplying power is known as a Source and a Port consuming power is known as a Sink. There is only one Source Port and one Sink Port in each PD connection between Port Partners. At Attach the Source Port (the Port with Rp asserted see ***[USB Type-C 1.2]***) is also the DFP and VCONN Source. At Attach the Sink Port (the Port with Rd asserted) is also the UFP and is not the VCONN Source.

The Source/Sink roles, DFP/UFP roles and VCONN Source role can all subsequently be swapped orthogonally to each other. A Port that supports both Source and Sink roles is called a Dual-Role Power Port (DRP). A Port that supports both DFP and UFP roles is called a Dual-Role Data Port (DRD).

When USB Communications Capability is supported in the DFP role then the Port will also be able to act as a USB Host. Similarly when USB Communications Capability is supported in the UFP role then the Port will also be able to act as a USB Device.

The following sections describe the high level operation of ports taking on the roles of DFP, UFP, Source and Sink. These sections do not describe operation that is not allowed; however if a certain behavior is not described then it is probably not supported by this specification.

For details of how PD maps to USB states in a PDUSB Device see Section 9.1.2.

2.6.1 Source Operation

The Source operates differently depending on Attachment status:

- At Attach (no PD Connection or Contract):
    - For a Source-only Port the Source detects Sink Attachment.
    - For a DRP that toggles the Port becomes a Source Port on Attachment of a Sink
    - The Source then typically sets VBUS to ***vSafe5V***.
- Before PD Connection (no PD Connection or PD Contract):
    - Prior to sending ***Source_Capabilities*** Messages the Source can detect the type of cabling Attached and may alter its advertised capabilities depending on the type of cable detected:
        - The Source attempts to communicate with one of the Cable Plugs using SOP' Packets. If the Cable Plug responds then communication takes place.
        - The default capability of a USB Type-C cable is 3A, but SOP' Communication is used to discover other capabilities of the cable.
    - The Source periodically advertises its capabilities by sending ***Source_Capabilities*** Messages every ***tTypeCSendSourceCap.***
- Establishing PD Connection (no PD Connection or Contract):
    - Presence of a PD Capable Port Partner is detected either:
        - By receiving a ***GoodCRC*** Message in response to a ***Source_Capabilities*** Message.
        - By receiving ***Hard Reset*** Signaling.
- Establishing Explicit Contract (PD Connection but no Explicit Contract or Implicit Contract after a Power Role Swap or Fast Role Swap):
    - The Source receives a ***Request*** Message from the Sink and responds with an ***Accept*** Message, if this is a valid request, followed by a ***PS_RDY*** Message when its power supply is ready to source power at the agreed level. At this point an Explicit Contract has been agreed.
    - A DFP does not generate SOP' or SOP'' Packets, is not required to detect SOP' or SOP'' Packets and Discards them.
- During PD Connection (Explicit Contract - ***PE_SRC_Ready*** State):

Microchip Ex. 1020, p. 53
Microchip v. Aptiv
IPR2024-00496

- The Source processes and responds (if a response is required) to all Messages received and sends appropriate Messages whenever its Local Policy requires.
    - The Source informs the Sink whenever its capabilities change, by sending a ***Source_Capabilities*** Message.
    - The Source will always have Rp asserted on its CC wire.
    - When this Port is a DRP the Source can initiate or receive a request for the exchange of power roles. After the Power Role Swap this Port will be a Sink and an Implicit Contract will be in place until an Explicit Contract is negotiated immediately afterwards.
    - When this Port is a DRD the Source can initiate or receive a request for an exchange of data roles. After a Data Role Swap the DFP (Host) becomes a UFP (Device). The Port remains a Source and the VCONN Source role (or not) remains unchanged.
    - The Source can initiate or receive a request for an exchange of VCONN Source. During a VCONN Swap VCONN is applied by both ends (make before break). The Port remains a Source and DFP/UFP roles remain unchanged.
- The Source when it is the VCONN Source can communicate with a Cable Plug using SOP' or SOP'' Communication at any time it is not engaged in any other SOP Communications.
    - If SOP Packets are received by the Source, during SOP' or SOP'' Communication, the SOP' or SOP'' Communication is immediately terminated (the Cable Plug times out and does not retry)
    - If the Source needs to initiate an SOP Communication during an ongoing SOP' or SOP'' Communication (e.g. for a Capabilities change) then the SOP' or SOP'' Communications will be interrupted.
    - When the Source Port is also a DFP the Source can control the entry and exiting of modes in the Cable Plug(s) and control Modal Operation.
- When the Source Port is also a DFP
    - The Source can initiate Unstructured or Structured VDMs.
    - The Source can control the entry and exiting of modes in the Sink and control Modal Operation using Structured VDMs.
- When the Source Port is part of a multi-port system:
    - will issue GotoMin requests when the power reserve is needed.

- Detach or Communications Failure
    - A Source detects plug Detach and takes V_{BUS} down to ***vSafe5V*** within ***tSafe5V*** and ***vSafe0V*** within ***tSafe0V*** (i.e. using USB Type-C Detach detection via CC).
    - When the Source detects the failure to receive a ***GoodCRC*** Message in response to a Message within ***tReceive:***
        - Leads to a Soft Reset, within ***tSoftReset*** of the ***CRCReceiveTimer*** expiring.
        - If the soft reset process cannot be completed a Hard Reset will be issued within ***tHardReset*** of the ***CRCReceiveTimer*** to restore V_{BUS} to USB Default Operation within ~1-1.5s.
            - When the Source is also the VCONN Source, VCONN will also be power cycled during the Hard Reset.
        - Receiving no response to further attempts at communication is interpreted by the Source as an error (see Error handling).
    - Errors during power transitions will automatically lead to a Hard Reset in order to restore power to default levels.
- Error handling

Microchip Ex. 1020, p. 54
Microchip v. Aptiv
IPR2024-00496

- Protocol Errors are handled by a ***Soft_Reset*** Message issued by either Port Partner, that resets counters, timers and states, but does not change the negotiated voltage and current or the Port's role (e.g. Source, DFP/UFP, VCONN Source) and does not cause an exit from Modal Operation.
- Serious errors are handled by ***Hard Reset*** Signaling issued by either Port Partner. A Hard Reset:
    - Resets protocol as for a Soft Reset but also returns the power supply to USB Default Operation (***vSafe0V*** or ***vSafe5V*** output) in order to protect the Sink.
    - Restores the Port's data role to DFP.
    - When the Sink is the VCONN Source it removes VCONN then the Source Port is restored as the VCONN Source.
    - Causes all Active Modes to be exited such that the Source is no longer in Modal Operation.
- After a Hard Reset it is expected that the Port Partner will respond within ***tNoResponse***. If this does not occur then ***nHardResetCount*** further Hard Resets are carried out before the Source performs additional Error Recovery steps, as defined in ***[USB Type-C 1.2]*** , by entering the ***ErrorRecovery*** state.

Microchip Ex. 1020, p. 55
Microchip v. Aptiv
IPR2024-00496

2.6.2 Sink Operation

- At Attach (no PD Connection or Contract):
    - Sink detects Source Attachment through the presence of ***vSafe5V***.
    - For a DRP that toggles the Port becomes a Sink Port on Attachment of a Source.
    - Once the Sink detects the presence of ***vSafe5V*** on V_{BUS} it waits for a ***Source_Capabilities*** Message indicating the presence of a PD capable Source.
    - If the Sink does not receive a ***Source_Capabilities*** Message within ***tTypeCSinkWaitCap*** then it issues ***Hard Reset*** Signaling in order to cause the Source Port to send a ***Source_Capabilities*** Message if the Source Port is PD capable.
    - The Sink does not generate SOP' or SOP'' Packets, is not required to detect SOP' or SOP'' Packets and does not recognize them.
- Establishing PD Connection (no PD Connection or Contract):
    - The Sink receives a ***Source_Capabilities*** Message and responds with a ***GoodCRC*** Message.
    - The Sink does not generate SOP' or SOP'' Packets, is not required to detect SOP' or SOP'' Packets and Discards them.
- Establishing Explicit Contract (PD Connection but no Explicit Contract or Implicit Contract after a Power Role Swap or Fast Role Swap):
    - The Sink receives a ***Source_Capabilities*** Message from the Source and responds with a ***Request*** Message. If this is a valid request the Sink receives an ***Accept*** Message followed by a ***PS_RDY*** Message when the Source's power supply is ready to source power at the agreed level. At this point the Source and Sink have entered into an Explicit Contract.
        - The Sink Port may request one of the capabilities offered by the Source, even if this is the ***vSafe5V*** output offered by ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]***, in order to enable future power negotiation.
            - A Sink not requesting any capability with a ***Request*** Message results in an error.
        - A Sink unable to fully operate at the offered capabilities requests the default capability but indicates that it would prefer another power level and provide a physical indication of the failure to the end user (e.g. using an LED).
        - A Sink does not generate SOP' or SOP'' Packets, is not required to detect SOP' or SOP'' Packets and Discards them.
- During PD Connection (Explicit Contract – ***PE_SNK_Ready*** state)
    - The Sink processes and responds (if a response is required) to all Messages received and sends appropriate Messages whenever its Local Policy requires.
    - A Sink whose power needs have changed indicates this to the Source with a new ***Request*** Message. The Sink Port may request one of the capabilities previously offered by the Source, even if this is the ***vSafe5V*** output offered by ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]***, in order to enable future power negotiation.
        - Not requesting any capability with a ***Request*** Message results in an error.
        - A Sink unable to fully operate at the offered capabilities requests an offered capability but indicates a capability mismatch i.e. that it would prefer another power level also providing a physical indication of the failure to the End User (e.g. using an LED).
    - The Sink will always have Rd asserted on its CC wire.
    - When this Port is a DRP the Sink can initiate or receive a request for the exchange of power roles. After the Power Role Swap this Port will be a Source and an Implicit Contract will be in place until an Explicit Contract is negotiated immediately afterwards..

Microchip Ex. 1020, p. 56
Microchip v. Aptiv
IPR2024-00496

- When this Port is a DRD the Sink can initiate or receive a request for an exchange of data roles. After a Data Role Swap the DFP (Host) becomes a UFP (Device). The Port remains a Sink and VCONN Source role (or not) remains unchanged.
- The Sink can initiate or receive a request for an exchange of VCONN Source. During a VCONN Swap VCONN is applied by both ends (make before break). The Port remains a Sink and DFP/UFP roles remain unchanged.
- The Sink when it is the VCONN Source can communicate with a Cable Plug using SOP' or SOP'' Communication at any time it is not engaged in any other SOP Communications.
    - If SOP Packets are received by the Sink, during SOP' or SOP'' Communication, the SOP' or SOP'' Communication is immediately terminated (the Cable Plug times out and does not retry)
    - If the Sink needs to initiate an SOP Communication during an ongoing SOP' or SOP'' Communication (e.g. for a Capabilities change) then the SOP' or SOP'' Communications will be interrupted.
    - When the Sink Port is also a DFP the Source can control the entry and exiting of modes in the Cable Plug(s) and control Modal Operation.
- When the Sink Port is also a DFP
    - The Sink can initiate Unstructured or Structured VDMs.
    - The Sink can control the entry and exiting of modes in the Sink and control Modal Operation using Structured VDMs.

- Detach or Communications Failure
    - A Sink detects the removal of V_{BUS} and interprets this as the end of the PD Connection.
        - This is unless the ***vSafe0V*** is due to either a Hard Rest, Power Role Swap or Fast Role Swap.
    - A Sink detects plug removal and discharges V_{BUS}.
    - When the Sink detects the failure to receive a ***GoodCRC*** Message in response to a Message within ***tReceive:***
        - leads to a Soft Reset, within ***tSoftReset*** of the ***CRCReceiveTimer*** expiring.
        - If the soft reset process cannot be completed a Hard Reset will be issued within ***tHardReset*** of the ***CRCReceiveTimer*** to restore V_{BUS} to USB Default Operation within ~1-1.5s.
        - Receiving no response to further attempts at communication is interpreted by the Sink as an error (see Error handling).
    - Errors during power transitions will automatically lead to a Hard Reset in order to restore power to default levels.
    -

- Error handling
    - Protocol Errors are handled by a ***Soft_Reset*** Message issued by either Port Partner, that resets counters, timers and states, but does not change the negotiated voltage and current or the Port's role (e.g. Sink, DFP/UFP, VCONN Source) and does not cause an exit from Modal Operation.
    - Serious errors are handled by ***Hard Reset*** Signaling issued by either Port Partner. A Hard Reset:
        - resets protocol as for a Soft Reset but also returns the power supply to USB Default Operation (***vSafe0V*** or ***vSafe5V*** output) in order to protect the Sink.
        - restores the Port's data role to UFP.
        - when the Sink is the VCONN Source it removes VCONN then the Source Port is restored as the VCONN Source.
        - causes all Active Modes to be exited such that the Source is no longer in Modal Operation.
    - After a Hard Reset it is expected that the Port Partner will respond within ***tTypeCSinkWaitCap***. If this does not occur then 2 further Hard Resets are carried out before the UFP stays in the ***PE_SNK_Wait_for_Capabilities*** state.

Microchip Ex. 1020, p. 57
Microchip v. Aptiv
IPR2024-00496

2.6.3 Cable Plugs

- Cable Plugs are powered when VCONN is present but are not aware of the status of the Contract.
- Cable Plugs do not initiate message sequences and only respond to messages sent to them.
- Detach or Communications Failure
    - Communications can be interrupted at any time
    - There is no communication timeout scheme between the DFP/UFP and Cable Plug
    - The Cable Plug is ready to respond to potentially repeated requests.
- Error handling:
    - The Cable Plug detects ***Hard Reset*** Signaling to determine that the Source and Sink have been reset and will need to reset itself (equivalent to a power cycle).
        - The Cable Plug cannot generate ***Hard Reset*** Signaling itself.
        - The Hard Reset process power cycles both V_{BUS} and VCONN so this is expected to reset the Cable Plugs by itself.
    - A Cable Plug detects ***Cable Reset*** Signaling to determine that it will need to reset itself (equivalent to a power cycle).

Microchip Ex. 1020, p. 58
Microchip v. Aptiv
IPR2024-00496

2.7 Architectural Overview

This logical architecture is not intended to be taken as an implementation architecture. An implementation architecture is, by definition, a part of product definition and is therefore outside of the scope of this specification.

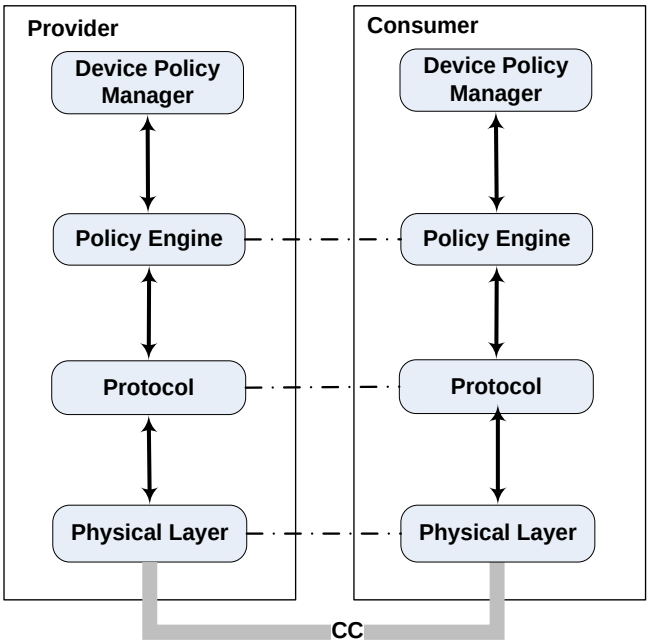
This section outlines the high level logical architecture of USB Power Delivery referenced throughout this specification. In practice various implementation options are possible based on many different possible types of PD device. PD devices may have many different configurations e.g. USB or non-USB communication, single versus multiple ports, dedicated power supplies versus supplies shared on multiple ports, hardware versus software based implementations etc. The architecture outlined in this section is therefore provided only for reference in order to indicate the high level logical model used by the PD specification. This architecture is used to identify the key concepts and also to indicate logical blocks and possible links between them.

The USB Power Delivery architecture in each USB Power Delivery capable Device is made up of a number of major components.

The communications stack seen in Figure 2-3 consists of:

- A **Device Policy Manager** (see Section 8.2) that exists in all devices and manages USB Power Delivery resources within the device across one or more ports based on the Device's Local Policy.
- A **Policy Engine** (see Section 8.3) that exists in each USB Power Delivery Port implements the Local Policy for that Port.
- A **Protocol Layer** (see Chapter 6) that enables Messages to be exchanged between a Source Port and a Sink Port.
- A **Physical Layer** (see Chapter 5) that handles transmission and reception of bits on the wire and handles data transmission.

Figure 2-3 USB Power Delivery Communications Stack



Additionally USB Power Delivery devices which can operate as USB devices may communicate over USB (see Figure 2-4). An ***Optional* System Policy Manager** (see Chapter 9) that resides in the USB Host communicates with the PD Device over USB, via the root Port and potentially over a tree of USB Hubs. The **Device Policy Manager** interacts with the USB interface in each device in order to provide and update PD related information in the USB domain. Note that a PD device is not required to have a USB device interface.

Microchip Ex. 1020, p. 59
Microchip v. Aptiv
IPR2024-00496

Figure 2-4 USB Power Delivery Communication Over USB

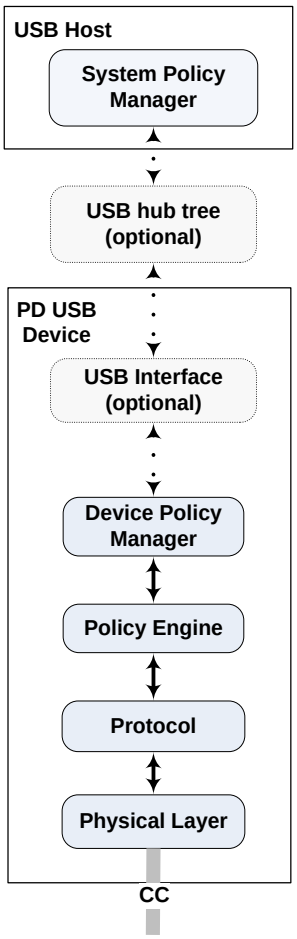


Figure 2-5 shows the logical blocks between two Attached PD ports. In addition to the communication stack described above there are also:

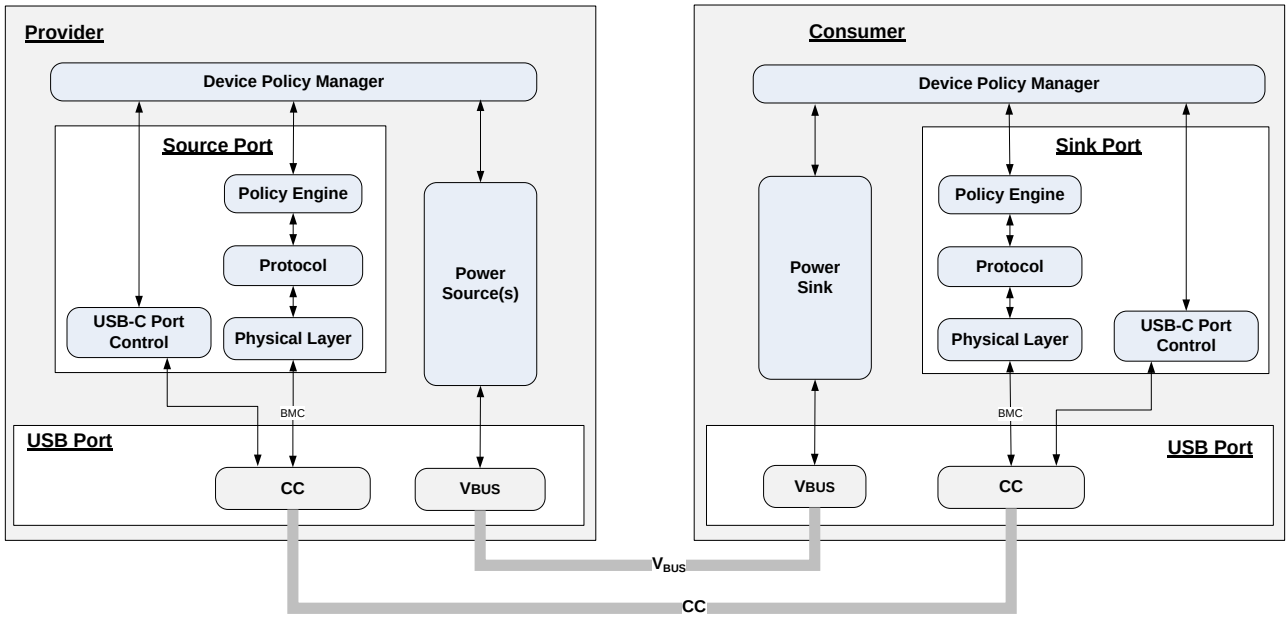
- For a Provider or Dual-Role Power Device: one or more **Sources** providing power to one or more ports.
- For a Consumer or Dual-Role Power Device: a **Sink** consuming power.
- A **USB-C Port Control** module (see Section4.4) that detects cable Attach/Detach as defined in ***[USB Type-C 1.2]***.
- USB Power Delivery uses standard cabling as defined in ***[USB Type-C 1.2]***.

The **Device Policy Manager** talks to the communication stack, Source/Sink and the USB-C Port Control block in order to manage the resources in the Provider or Consumer.

Figure 2-5 illustrates a Provider and a Consumer. Dual-Role Power Devices can be constructed by combining the elements of both Provider and Consumer into a single device. Providers can also contain multiple Source Ports each with their own communications stack and USB-C Port Control.

Microchip Ex. 1020, p. 60
Microchip v. Aptiv
IPR2024-00496

Figure 2-5 High Level Architecture View



2.7.1 Policy

There are two possible levels of Policy:

1) System Policy applied system wide by the System Policy Manager across multiple Providers or Consumers.
2) Local Policy enforced on a Provider or Consumer by the Device Policy Manager.

Policy comprises several logical blocks:

- System Policy Manager (system wide).
- Device Policy Manager (one per Provider or Consumer).
- Policy Engine (one per Source or Sink Port).

2.7.1.1 System Policy Manager

Since the USB Power Delivery protocol is essentially point to point, implementation of a System Policy requires communication by an additional data communication mechanism i.e. USB. The System Policy Manager monitors and controls System Policy between various Providers and Consumers connected via USB. The System Policy Manager resides in the USB Host and communicates via USB with the Device Policy Manager in each connected Device. Devices without USB data communication capability or are not data connected, will not be able to participate in System Policy.

The System Policy Manager is ***Optional*** in any given system so USB Power Delivery Providers and Consumers can operate without it being present. This includes systems where the USB Host does not provide a System Policy Manager and may also include "headless" systems without any USB Host. In those cases where a Host is not present, USB Power Delivery is useful for charging purposes, or the powering of devices since useful USB functionality is not possible. Where there is a USB Host but no System Policy Manager, Providers and Consumers can negotiate power between themselves, independently of USB power rules, but are more limited in terms of the options available for managing power.

Microchip Ex. 1020, p. 61
Microchip v. Aptiv
IPR2024-00496

2.7.1.2 Device Policy Manager

The Device Policy Manager provides mechanisms to monitor and control the USB Power Delivery system within a particular Consumer or Provider. The Device Policy Manager enables Local Policies to be enforced across the system by communication with the System Policy Manager. Local Policies are enacted on a per Port basis by the Device Policy Manager's control of the Source/Sink Ports and by communication with the Policy Engine and USB-C Port Control for that Port.

2.7.1.3 Policy Engine

Providers and Consumers are free to implement their own Local Policies on their directly connected Source or Sink Ports. These will be supported by negotiation and status mechanisms implemented by the Policy Engine for that Port. The Policy Engine interacts directly with the Device Policy Manager in order to determine the present Local Policy to be enforced. The Policy Engine will also be informed by the Device Policy Manager whenever there is a change in Local Policy (e.g. a capabilities change).

2.7.2 Message Formation and Transmission

2.7.2.1 Protocol Layer

The Protocol Layer forms the Messages used to communicate information between a pair of ports. It is responsible for forming Capabilities Messages, requests and acknowledgements. Additionally it forms Messages used to swap roles and maintain presence. It receives inputs from the Policy Engine indicating which Messages to send and indicates the responses back to the Policy Engine.

The basic protocol uses a push model where the Provider pushes it capabilities to the Consumer that in turn responds with a request based on the offering. However, the Consumer may asynchronously request the Provider's present capabilities and may select another voltage/current.

Extended Messages of up to a Data Size of ***MaxExtendedMsgLen*** can be sent and received provided the Protocol Layer determines that both Port Partners support this capability. When one of both Port Partners do not support Extended Messages of Data Size greater than ***MaxExtendedMsgLegacyLen*** then the Protocol Layer supports a Chunking mechanism to break larger Messages into smaller Chunks of size ***MaxExtendedMsgChunkLen***.

2.7.2.2 PHY Layer

The PHY Layer is responsible for sending and receiving Messages across the USB Type-C CC wire and for managing data. It tries to avoid collisions on the wire, recovering from them when they occur. It also detects errors in the Messages using a CRC.

2.7.3 Collision Avoidance

2.7.3.1 Policy Engine

The Policy Engine in a Source will indicate to the Protocol Layer the start and end of each Atomic Message Sequence (AMS) that the Source initiates. The Policy Engine in a Sink will indicate to the Protocol Layer the start of each AMS the Sink initiates. This enables co-ordination of AMS initiation between the Port Partners.

2.7.3.2 Protocol Layer

The Protocol Layer in the Source will request the PHY to set the Rp value to ***SinkTxOk*** to indicate that the Sink can initiate an AMS by sending the first Message in the sequence. The Protocol Layer in the Source will request the PHY to set the Rp value to ***SinkTxNG*** to indicate that the Sink cannot initiate an AMS since the Source is about to initiate an AMS.

The Protocol Layer in the Sink, when the Policy Engine indicates that an AMS is being initiated, will wait for the Rp value to be set to ***SinkTxOk*** before initiating the AMS by sending the first Message in the sequence.

Microchip Ex. 1020, p. 62
Microchip v. Aptiv
IPR2024-00496

2.7.3.3 PHY Layer

The PHY Layer in the Source will set the Rp value to either ***SinkTxOk*** or ***SinkTxNG*** as directed by the Protocol Layer. The PHY Layer in the Sink will detect the present Rp value and inform the Protocol Layer.

2.7.4 Power supply

2.7.4.1 Source

Each Provider will contain one or more Sources that are shared between one or more ports. These Sources are controlled by the Local Policy. Sources start up in USB Default Operation where the Port applies ***vSafe0V*** or ***vSafe5V*** on V_{BUS} and return to this state on Detach or after a Hard Reset. If the Source applies ***vSafe0V*** as their default, it detects Attach events and transitions its output to ***vSafe5V*** upon detecting an Attach.

2.7.4.2 Sink

Consumers are assumed to have one Sink connected to a Port. This Sink is controlled by Local Policy. Sinks start up in USB Default Operation where the Port can operate at ***vSafe5V*** with USB default specified current levels and return to this state on Detach or after a Hard Reset.

2.7.4.3 Dual-Role Power Ports

Dual-Role Power Ports have the ability to operate as either a Source or a Sink and to swap between the two roles using Power Role Swap or Fast Role Swap.

2.7.4.4 Dead Battery or Lost Power Detection

[USB Type-C 1.2] defines mechanisms intended to communicate with and charge a Sink or DRP with a Dead Battery.

2.7.5 DFP/UFP

2.7.5.1 Downstream Facing Port (DFP)

The Downstream Facing Port or DFP is equivalent in the USB topology to the USB A-Port. The DFP will also correspond to the USB Host but only if USB Communication is supported while acting as a DFP. Products such as Wall Warts can be a DFP while not having USB Communication capability. The DFP also acts as the bus master when controlling alternate mode operation.

2.7.5.2 Upstream Facing Port (UFP)

The Upstream Facing Port or UFP is equivalent in the USB topology to the USB B-Port. The UFP will also correspond to the USB Device but only if USB Communication is supported while acting as a UFP. Products which charge can be a UFP while not having USB Communication capability.

2.7.5.3 Dual-Role Data Ports

Dual-Role Data Ports have the ability to operate as either a DFP or a UFP and to swap between the two roles using Data Role Swap. Note that products can be Dual-Role Data Ports without being Dual-Role Power ports i.e. they can switch logically between DFP and UFP roles even if they are Source-only or Sink-only Ports.

2.7.6 VCONN Source

One Port, initially the Source Port, is the VCONN Source. The Cable Plugs use this supply to determine which Cable Plug is SOP'. The responsibility for sourcing VCONN can be swapped between the Source and Sink Ports in a make before break fashion to ensure that the Cable Plugs continue to be powered. To ensure reliable communication with the Cable Plugs only the VCONN Source is permitted to communicate with the Cable Plugs. Prior to a Power Role Swap, Data Role Swap or Fast Role Swap each Port needs to ensure that it is the VCONN Source if it needs to communicate with the Cable Plugs after the swap.

Microchip Ex. 1020, p. 63
Microchip v. Aptiv
IPR2024-00496

2.7.7 Cable and Connectors

2.7.7.1 USB-C Port Control

The USB-C Port Control block provides mechanisms to inform the Device Policy Manager of cable Attach/Detach events.

The USB Power Delivery specification assumes certified USB cables and associated detection mechanisms as defined in the ***[USB Type-C 1.2]*** specification.

2.7.8 Interactions between Non-PD, BC and PD devices

USB Power Delivery only operates when two USB Power Delivery devices are directly connected. When a Device finds itself a mixed environment, where the other device does not support the USB Power Delivery Specification, the existing rules on supplying ***vSafe5V*** as defined in the ***[USB 2.0]***, ***[USB 3.1]***, ***[USBBC 1.2]*** or ***[USB Type-C 1.2]*** specifications are applied.

There are two primary cases to consider:

- The Host (DFP/Source) is non-PD and as such will not send any advertisements. An Attached PD capable Device will not see any advertisements and operates using the rules defined in the ***[USB 2.0]***, ***[USB 3.1]***, ***[USBBC 1.2]*** or ***[USB Type-C 1.2]*** specifications.
- The Device (UFP/Sink) is non-PD and as such will not see any advertisements and therefore will not respond. The Host (DFP/Source) will continue to supply ***vSafe5V*** to V_{BUS} as specified in the ***[USB 2.0]***, ***[USB 3.1]***, ***[USBBC 1.2]*** or ***[USB Type-C 1.2]*** specifications.

2.7.9 Power Rules

Power Rules define voltages and current ranges that are offered by USB Power Delivery Sources and used by a USB Power Delivery Sink for a given value of PD Power. See Section 10 for further details.

Microchip Ex. 1020, p. 64
Microchip v. Aptiv
IPR2024-00496

3. USB Type-A and USB Type-B Cable Assemblies and Connectors

This section has been ***Deprecated***. Please refer to ***[USBPD 2.0]*** for details of cables and connectors used in scenarios utilizing the BFSK Signaling scheme in conjunction with USB Type-A or USB Type-B connectors.

Microchip Ex. 1020, p. 65
Microchip v. Aptiv
IPR2024-00496

4. Electrical Requirements

This chapter covers the platform's electrical requirements for implementing USB Power Delivery.

4.1 Interoperability with other USB Specifications

USB Power Delivery may be implemented alongside the ***[USB 2.0]***, ***[USB 3.1]***, ***[USBBC 1.2]*** and ***[USB Type-C 1.2]*** specifications. In the case where a Device requests power via the Battery Charging Specification and then the USB Power Delivery Specification, it shall follow the USB Power Delivery Specification until the Port Pair is Detached or there is a Hard Reset. If the USB Power Delivery connection is lost, the Port shall return to its default state, see Section 6.8.2.

4.2 Dead Battery Detection / Unpowered Port Detection

Dead Battery/Unpowered operation is when a USB Device needs to provide power to a USB Host under the circumstances where the USB Host:

- Has a Dead Battery that requires charging or
- Has lost its power source or
- Does not have a power source or
- Does not want to provide power.

Dead Battery charging operation for connections between USB Type-C connectors is defined in ***[USB Type-C 1.2]***.

4.3 Cable IR Ground Drop (IR Drop)

Every PD Sink Port capable of USB communications may be susceptible to unreliable USB communication if the voltage drop across ground falls outside of the acceptable common mode range for the USB Hi-Speed transceivers data lines due to excessive current draw. Certified USB cabling is specified such that such errors should not occur (See ***[USB Type-C 1.2]***).

4.4 Cable Type Detection

Standard USB Type-C cable assemblies are rated for PD voltages higher than ***vSafe5V*** and current levels of at least 3A (See ***[USB Type-C 1.2]***). The Source shall limit maximum capabilities it offers so as not to exceed the capabilities of the type of cabling detected.

Sources shall detect the type of Attached cable and limit the Capabilities they offer based on the current carrying capability of the cable determined by the Cable capabilities determined using the ***Discover Identity*** Command (see Section 6.4.4.2) sent using SOP' Communication (see Section 2.5) to the Cable Plug. The Cable VDO returned as part of the ***Discover Identity*** Command details the maximum current and voltage values that shall be negotiated for a given cable as part of an Explicit Contract.

The cable detection process is usually run when the Source is powered up, after a Power Role Swap or Fast Role Swap or when power is applied to a Sink. The exact method used to detect these events is up to the manufacturer and shall meet the following requirements:

- Sources shall run the cable detection process prior to the Source sending ***Source_Capabilities*** Messages offering currents in excess of 3A and/or voltages in excess of 20V.
- Sinks with USB Type-C connectors shall select Capabilities from the offered Source Capabilities assuming that the Source has already determined the Capabilities of the cable.

Microchip Ex. 1020, p. 66
Microchip v. Aptiv
IPR2024-00496

5. Physical Layer

5.1 Physical Layer Overview

The Physical Layer (PHY Layer) defines the signaling technology for USB Power Delivery. This chapter defines the electrical requirements and parameters of the PD Physical Layer required for interoperability between USB PD devices.

5.2 Physical Layer Functions

The USB PD Physical Layer consists of a pair of transmitters and receivers that communicate across a single signal wire (CC). All communication is half duplex. The PHY Layer practices collision avoidance to minimize communication errors on the channel.

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 Biphase Mark Coding (BMC) over CC

The receiver performs the following functions:

- Recover the clock and lock onto the Packet from the Preamble
- 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 not valid, flush the received data.

Microchip Ex. 1020, p. 67
Microchip v. Aptiv
IPR2024-00496

5.3 Symbol Encoding

Except for the Preamble, all communications on the line shall be encoded with a line code to ensure a reasonable level of DC-balance and a suitable number of transitions. This encoding makes receiver design less complicated and allows for more variations in the receiver design.

4b5b line code shall be used. 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 4b5b code provides data encoding along with special symbols. Special symbols are used to signal ***Hard Reset***, and delineate packet boundaries.

Table 5-1 4b5b Symbol Encoding Table

Name	4b	5b Symbol	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	Startsynch #1
Sync-2	K-code	10001	Startsynch #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
Reserved	Error	00000	Shall not be used
Reserved	Error	00001	Shall not be used
Reserved	Error	00010	Shall not be used
Reserved	Error	00011	Shall not be used
Reserved	Error	00100	Shall not be used
Reserved	Error	00101	Shall not be used
Sync-3	K-code	00110	Startsynch #3
Reserved	Error	01000	Shall not be used
Reserved	Error	01100	Shall not be used
Reserved	Error	10000	Shall not be used
Reserved	Error	11111	Shall not be used

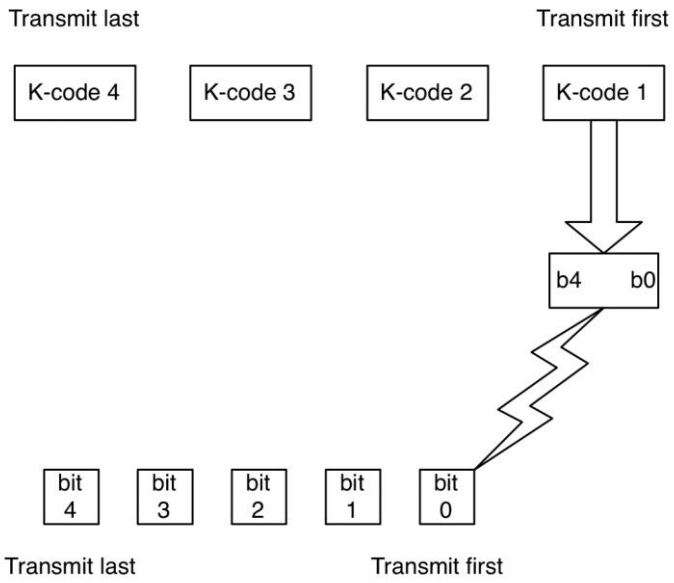
Microchip Ex. 1020, p. 68
Microchip v. Aptiv
IPR2024-00496

5.4 Ordered Sets

Ordered sets shall be interpreted according to Figure 5-1.

An ordered set consists of 4 K-codes sent as shown in Figure 5-1.

Figure 5-1 Interpretation of ordered sets



A list of the ordered sets used by USB Power Delivery can be seen in Table 5-2. ***SOP**** is a generic term used in place of ***SOP***/***SOP'***/***SOP''***.

Table 5-2 Ordered Sets

Ordered Set	Reference
Cable Reset	Section 5.6.5
Hard Reset	Section 5.6.4
SOP	Section 5.6.1.2.1
SOP'	Section 5.6.1.2.2
SOP'_Debug	Section 5.6.1.2.4
SOP''	Section 5.6.1.2.3
SOP''_Debug	Section 5.6.1.2.5

The receiver shall search for all four K-codes and when it finds all four in the correct place, it may interpret it as a valid ordered set (see Table 5-3).

Table 5-3 Validation of Ordered Sets

	1st code	2nd code	3rd code	4th code
Valid	Corrupt	K-code	K-code	K-code
Valid	K-code	Corrupt	K-code	K-code
Valid	K-code	K-code	Corrupt	K-code
Valid	K-code	K-code	K-code	Corrupt
Valid (perfect)	K-code	K-code	K-code	K-code
Not Valid (example)	K-code	Corrupt	K-code	Corrupt

Microchip Ex. 1020, p. 69
Microchip v. Aptiv
IPR2024-00496

5.5 Transmitted Bit Ordering

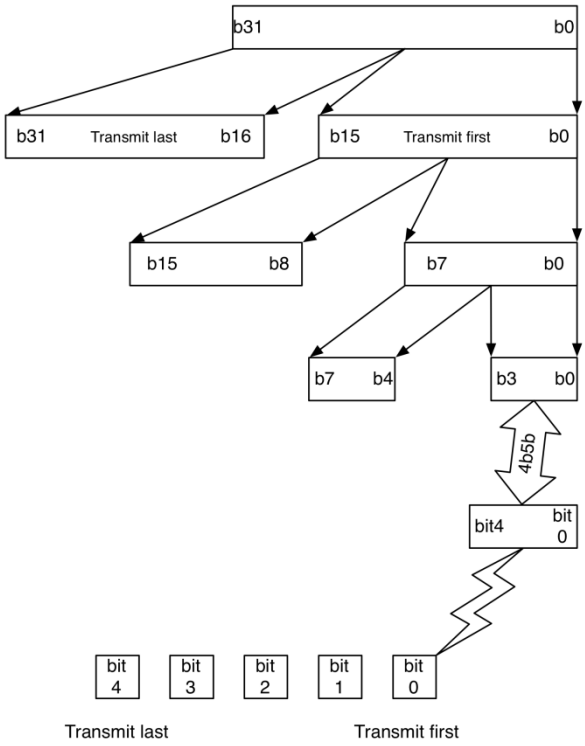
This section describes the order of bits on the wire that shall be used when transmitting data of varying sizes. Table 5-4 shows the different data sizes that are possible.

Figure 5-2 shows the transmission order that shall be followed.

Table 5-4 Data Size

	Unencoded	Encoded
Byte	8-bits	10-bits
Word	16-bits	20- bits
DWord	32-bits	40-bits

Figure 5-2 Transmit Order for Various Sizes of Data

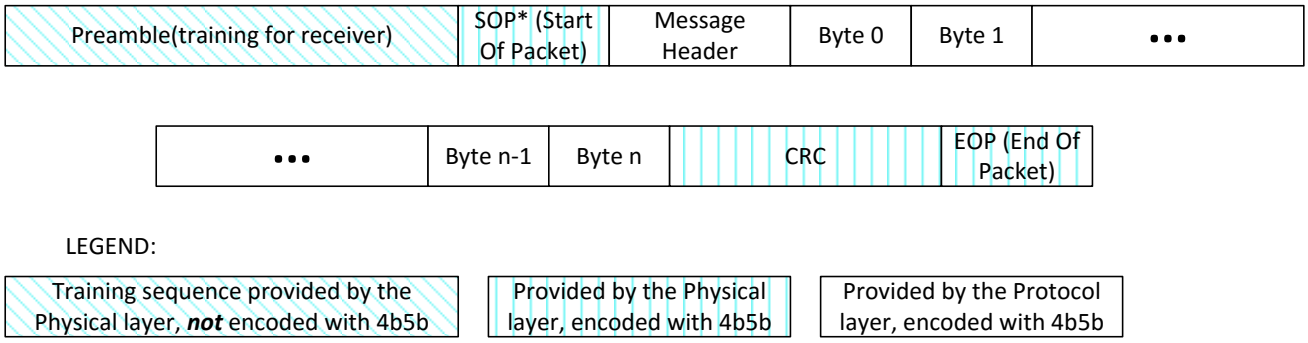


Microchip Ex. 1020, p. 70
Microchip v. Aptiv
IPR2024-00496

5.6 Packet Format

The packet format shall consist of a Preamble, an ***SOP****, (see Section 5.6.1.2), packet data including the Message Header, a CRC and an ***EOP*** (see Section 5.6.1.5). The packet format is shown in Figure 5-3 and indicates which parts of the packet shall be 4b/5b encoded. Once 4b/5b encoded, the entire Packet shall be transmitted using BMC over CC. Note that all the bits in the Packet, including the Preamble, are BMC encoded. See Section 6.2.1 for more details of the Packet construction for Control, Data and Extended Messages.

Figure 5-3 USB Power Delivery Packet Format



5.6.1 Packet Framing

The transmission starts with a Preamble that is used to allow the receiver to lock onto the carrier. It is followed by a ***SOP**** (Start of Packet). The packet is terminated with an ***EOP*** (End of Packet) K-code.

5.6.1.1 Preamble

The Preamble is used to achieve lock in the receiver by presenting an alternating series of "0s" and "1s", so the average frequency is the carrier frequency. Unlike the rest of the packet, the Preamble shall not be 4b/5b encoded.

The Preamble shall consist of a 64-bit sequence of alternating 0s and 1s. The Preamble shall start with a "0" and shall end with a "1".

5.6.1.2 Start of Packet Sequences

5.6.1.2.1 Start of Packet Sequence (SOP)

SOP is an ordered set. The ***SOP*** ordered set is defined as: three ***Sync-1*** K-codes followed by one ***Sync-2*** K-code (see Table 5-5).

Table 5-5 SOP ordered set

K-code number	K-code in code table
1	***Sync-1***
2	***Sync-1***
3	***Sync-1***
4	***Sync-2***

A Power Delivery Capable Source or Sink shall be able to detect and communicate with packets using ***SOP***. If a valid ***SOP*** is not detected (see Table 5-3) then the whole transmission shall be ***Discarded***.

Sending and receiving of SOP Packets shall be limited to PD Capable Ports on PDUSB Hosts and PDUSB Devices. Cable Plugs shall neither send nor receive SOP Packets. Note that PDUSB Devices, even if they have the physical form of a cable (e.g. AMAs), are still required to respond to SOP Packets.

Microchip Ex. 1020, p. 71
Microchip v. Aptiv
IPR2024-00496

5.6.1.2.2 Start of Packet Sequence Prime (SOP')

The ***SOP'*** ordered set is defined as: two ***Sync-1*** K-codes followed by two ***Sync-3*** K-codes (see Table 5-6).

Table 5-6 SOP' ordered set

K-code number	K-code in code table
1	***Sync-1***
2	***Sync-1***
3	***Sync-3***
4	***Sync-3***

A Cable Plug capable of SOP' Communications shall only detect and communicate with packets starting with ***SOP'***.

A Port needing to communicate with a Cable Plug capable of SOP' Communications, Attached between a Port Pair will be able to communicate using both packets starting with ***SOP'*** to communicate with the Cable Plug and starting with ***SOP*** to communicate with its Port Partner.

For a Cable Plug supporting SOP' Communications, if a valid ***SOP'*** is not detected (see Table 5-3) then the whole transmission shall be ***Discarded***. For a Port supporting SOP' Communications if a valid ***SOP*** or ***SOP'*** is not detected (see Table 5-3) then the whole transmission shall be ***Discarded***. When there is no Explicit Contract or an Implicit Contract in place a Sink shall not send SOP' Packets and shall Discard all packets starting with ***SOP'***.

5.6.1.2.3 Start of Packet Sequence Double Prime (SOP'')

The ***SOP''*** ordered set is defined as the following sequence of K-codes: ***Sync-1***, ***Sync-3***, ***Sync-1***, ***Sync-3*** (see Table 5-7).

Table 5-7 SOP'' ordered set

K-code number	K-code in code table
1	***Sync-1***
2	***Sync-3***
3	***Sync-1***
4	***Sync-3***

A Cable Plug capable of SOP'' Communication, shall have a SOP' Communication capability in the other Cable Plug. No cable shall only support SOP'' Communication. A Cable Plug to which SOP'' Communication is assigned shall only detect and communicate with packets starting with ***SOP''*** and shall Discard any other packets.

A Port needing to communicate with such a Cable Plug, Attached between a Port Pair will be able to communicate using packets starting with ***SOP'*** and ***SOP''*** to communicate with the Cable Plugs and packets starting with ***SOP*** to communicate with its Port Partner. A Port which supports SOP'' Communication shall also support SOP' Communication and shall co-ordinate SOP* Communication so as to avoid collisions.

For the Cable Plug supporting SOP'' Communication, if a valid ***SOP''*** is not detected (see Table 5-3) then the whole transmission shall be ***Discarded***. For the Port if a valid ***SOP**** is not detected (see Table 5-3) then the whole transmission shall be ***Discarded***.

5.6.1.2.4 Start of Packet Sequence Prime Debug (SOP'_Debug)

The ***SOP'_Debug*** ordered set is defined as the following sequence of K-codes: ***Sync-1***, ***RST-2***, ***RST-2***, ***Sync-3*** (see Table 5-8). The usage of this Ordered Set is presently undefined.

Microchip Ex. 1020, p. 72
Microchip v. Aptiv
IPR2024-00496

Table 5-8 SOP'_Debug ordered set

K-code number	K-code in code table
1	*Sync-1*
2	*RST-2*
3	*RST-2*
4	*Sync-3*

5.6.1.2.5 Start of Packet Sequence Double Prime Debug (SOP''_Debug)

The ***SOP''_Debug*** ordered set is defined as the following sequence of K-codes: ***Sync-1***, ***RST-2***, ***Sync-3***, ***Sync-2*** (see Table 5-9). The usage of this Ordered Set is presently undefined.

Table 5-9 SOP''_Debug ordered set

K-code number	K-code in code table
1	*Sync-1*
2	*RST-2*
3	*Sync-3*
4	*Sync-2*

5.6.1.3 Packet Payload

The packet payload is delivered from the protocol layer (Section 6.2) and shall be encoded with the hex data codes from Table 5-1.

5.6.1.4 CRC

The CRC shall be inserted just after the payload. It is described in Section 5.6.2.

5.6.1.5 End of Packet (EOP)

The end of packet marker shall be a single ***EOP*** K-code as defined in Table 5-1. This shall mark the end of the CRC. After the ***EOP***, the CRC-residual shall be checked. If the CRC is not good, the whole transmission shall be ***Discarded***, if it is good, the packet shall be delivered to the Protocol Layer. Note an ***EOP*** may be used to prematurely terminate a Packet e.g. before sending ***Hard Reset*** Signaling.

5.6.2 CRC

The Message Header and data shall be protected by a 32-bit CRC.

CRC-32 protects the data integrity of the data payload. CRC-32 is defined as follows:

- The CRC-32 polynomial shall be = 04C1 1DB7h.
- The CRC-32 Initial value shall be = FFFF FFFFh.
- CRC-32 shall be calculated for all bytes of the payload not inclusive of any packet framing symbols (i.e. excludes the Preamble, ***SOP****, ***EOP***).
- CRC-32 calculation shall begin at byte 0 bit 0 and continue to bit 7 of each of the bytes of the packet.
- The remainder of CRC-32 shall be complemented.
- The residual of CRC-32 shall be C704 DD7Bh.

Note: This inversion of the CRC-32 remainder adds an offset of FFFF FFFFh that will create a constant CRC-32 residual of C704 DD7Bh at the receiver side.

Note: The CRC implementation is identical to the one used in ***[USB 3.1]***.

Figure 5-4 is an illustration of CRC-32 generation. The output bit ordering shall be as detailed in Table 5-10.

Microchip Ex. 1020, p. 73
Microchip v. Aptiv
IPR2024-00496

Figure 5-4 CRC 32 generation

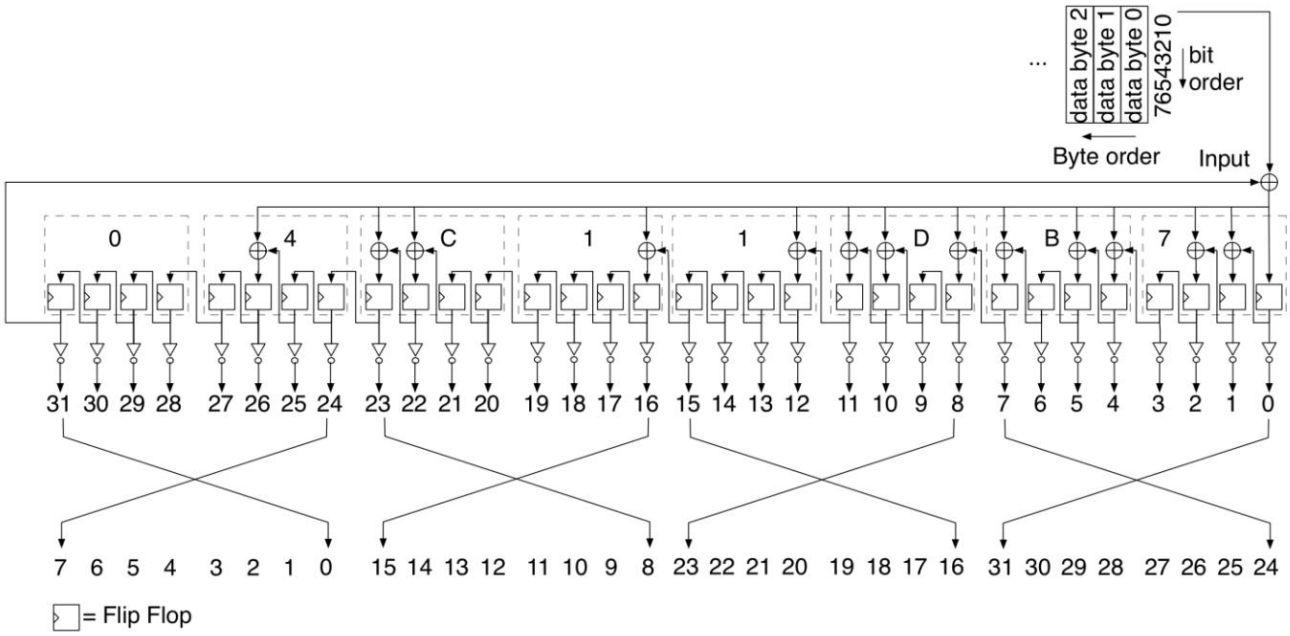


Table 5-10 CRC-32 Mapping

CRC-32	Result bit Position in CRC-32 Field
0	31
1	30
2	29
3	28
4	27
5	26
6	25
7	24
8	23
9	22
10	21
11	20
12	19
13	18
14	17
15	16
16	15
17	14
18	13
19	12
20	11
21	10
22	9
23	8
24	7

Microchip Ex. 1020, p. 74
Microchip v. Aptiv
IPR2024-00496

CRC-32	Result bit Position in CRC-32 Field
25	6
26	5
27	4
28	3
29	2
30	1
31	0

The CRC-32 shall be encoded before transmission.

5.6.3 Packet Detection Errors

CRC errors, or errors detected while decoding encoded symbols using the code table, shall be treated the same way; the Message shall be ***Discarded*** and a ***GoodCRC*** Message shall not be returned.

While the receiver is processing a packet, if at any time V_{BUS} becomes idle the receiver shall stop processing the packet and discard it (no ***GoodCRC*** Message is returned). See Section 5.8.6.1 for the definition of BMC idle.

5.6.4 Hard Reset

Hard Reset Signaling is an ordered set of bytes sent with the purpose to be recognized by the PHY Layer. The ***Hard Reset*** Signaling ordered set is defined as: three ***RST-1*** K-codes followed by one ***RST-2*** K-code (see Table 5-11).

Table 5-11 Hard Reset ordered set

K-code number	K-code in code table
1	***RST-1***
2	***RST-1***
3	***RST-1***
4	***RST-2***

A device shall perform a Hard Reset when it receives ***Hard Reset*** Signaling. After receiving the ***Hard Reset*** Signaling, the device shall reset as described in Section 6.8.2. If a valid ***Hard Reset*** is not detected (see Table 5-3) then the whole transmission shall be ***Discarded***.

A Cable Plug shall perform a Hard Reset when it detects ***Hard Reset*** Signaling being sent between the Port Partners. After receiving the ***Hard Reset*** Signaling, the device shall reset as described in Section 6.8.2.

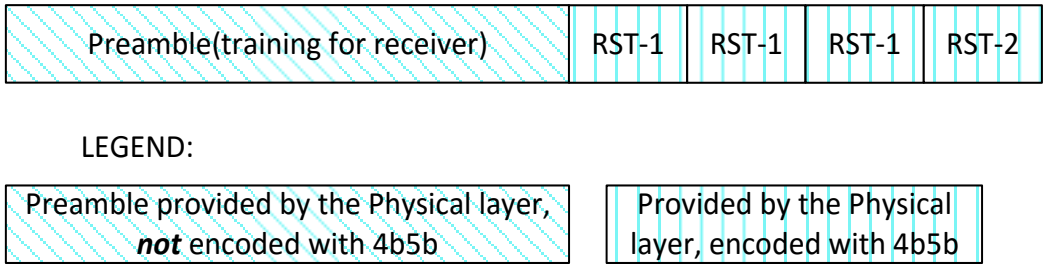
The procedure for sending ***Hard Reset*** Signaling shall be as follows:

1. If the PHY Layer is currently sending a Message, the Message shall be interrupted by sending an ***EOP*** K-code and the rest of the Message ***Discarded***.
2. If CC is not idle, wait for it to become idle (see Section 5.8.6.1)
3. Wait ***tInterFrameGap.***
4. If CC is still idle send the Preamble followed by the 4 K-codes for ***Hard Reset*** Signaling.
5. 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.
6. Re-enable the channel when requested by the Protocol Layer.

Figure 5-5 shows the line format of ***Hard Reset*** Signaling which is a Preamble followed by the ***Hard Reset*** Ordered Set.

Microchip Ex. 1020, p. 75
Microchip v. Aptiv
IPR2024-00496

Figure 5-5 Line format of Hard Reset



5.6.5 Cable Reset

Cable Reset Signaling is an ordered set of bytes sent with the purpose to be recognized by the PHY Layer. The ***Cable Reset*** Signaling ordered set is defined as the following sequence of K-codes: ***RST-1***, ***Sync-1***, ***RST-1***, ***Sync-3*** (see Table 5-12).

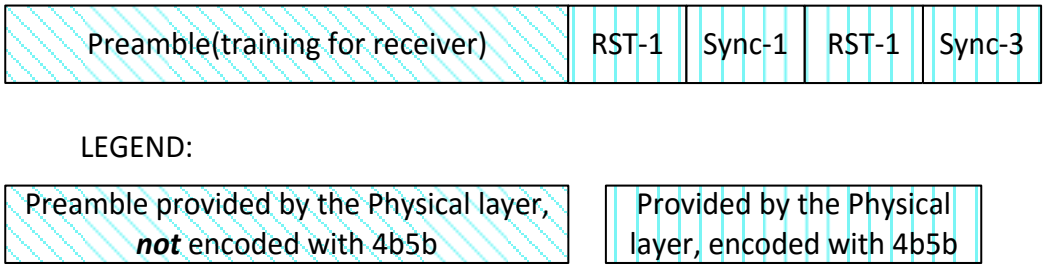
Table 5-12 Cable Reset ordered set

K-code number	K-code in code table
1	***RST-1***
2	***Sync-1***
3	***RST-1***
4	***Sync-3***

Cable Reset Signaling shall only be sent by the DFP. The ***Cable Reset*** Ordered Set is used to reset the Cable Plugs without the need to Hard Reset the Port Partners. The state of the Cable Plug after the ***Cable Reset*** Signaling shall be equivalent to power cycling the Cable Plug.

Figure 5-6 shows the line format of ***Cable Reset*** Signaling which is a Preamble followed by the ***Cable Reset*** Ordered Set.

Figure 5-6 Line format of Cable Reset



5.7 Collision Avoidance

The PHY Layer shall monitor the channel for data transmission and only initiate transmissions when CC is idle. If the bus idle condition is present, it shall be considered safe to start a transmission provided the conditions detailed in Section 5.8.5.4 are met. The bus idle condition shall be checked immediately prior to transmission. If transmission cannot be initiated then the packet shall be ***Discarded***. If the packet is ***Discarded*** because CC is not idle, the PHY Layer shall signal to the protocol layer that it has ***Discarded*** the Message as soon as CC becomes idle. See Section 5.8.6.1 for the definition of idle CC.

In addition the PHY Layer shall control the Rp resistor value to avoid collisions between Source and Sink transmissions. The Source shall set an Rp value corresponding to a current of 3A to indicate to the Sink that it may initiate an AMS. The Source shall set an Rp value corresponding to a current of 1.5A this shall indicate to the Sink that

Microchip Ex. 1020, p. 76
Microchip v. Aptiv
IPR2024-00496

it shall not initiate an AMS and shall only respond to Messages as part of an AMS. See ***[USB Type-C 1.2]*** for details of the corresponding Rp values.

Table 5-13 details the Rp values that shall be used by the Source to control Sink initiation of an AMS.

Table 5-13 Rp values used for Collision Avoidance

Source Rp	Parameter	Description	Sink operation	Source operation
1.5A@5V	***SinkTxNG***	Sink Transmit "No Go",	Sink cannot initiate an AMS. Sink can only respond to Messages as part of an AMS	Source can initiate an AMS ***tSinkTx*** after setting Rp to this value.
3A@5V	***SinkTxOk***	Sink Transmit "Ok"	Sink can initiate an AMS.	Source cannot initiate an AMS while it has this value set.

See also Section 6.6.15 and Section 6.11.2.1.

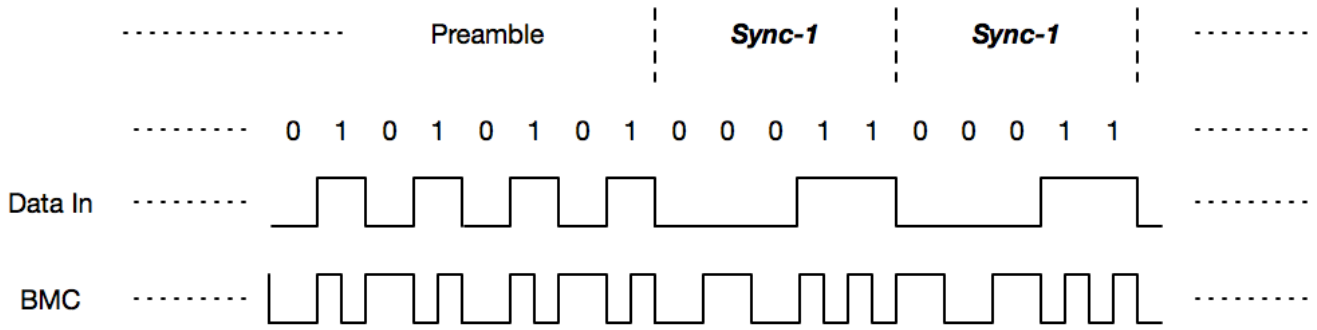
5.8 Biphase Mark Coding (BMC) Signaling Scheme

Biphase Mark Coding (BMC) is the physical layer Signaling Scheme for carrying USB Power Delivery Messages. This encoding assumes a dedicated DC connection, identified as the CC wire, which is used for sending PD Messages.

Biphase 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 zeroes 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 5-7 illustrates Biphase 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 that other K-codes can occur after the Preamble for Signaling such as ***Hard Reset*** and ***Cable Reset***.

Figure 5-7 BMC Example



5.8.1 Encoding and signaling

BMC uses DC coupled baseband signaling on CC. Figure 5-8 shows a block diagram for a Transmitter and Figure 5-9 shows a block diagram for the corresponding Receiver.

Microchip Ex. 1020, p. 77
Microchip v. Aptiv
IPR2024-00496

Figure 5-8 BMC Transmitter Block Diagram

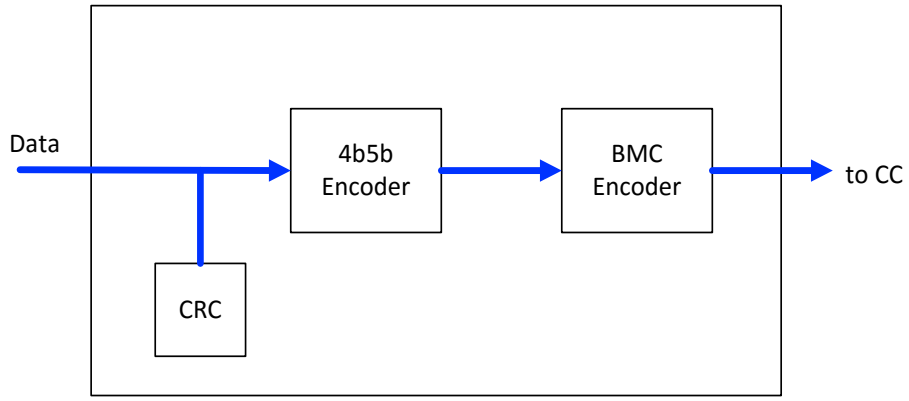
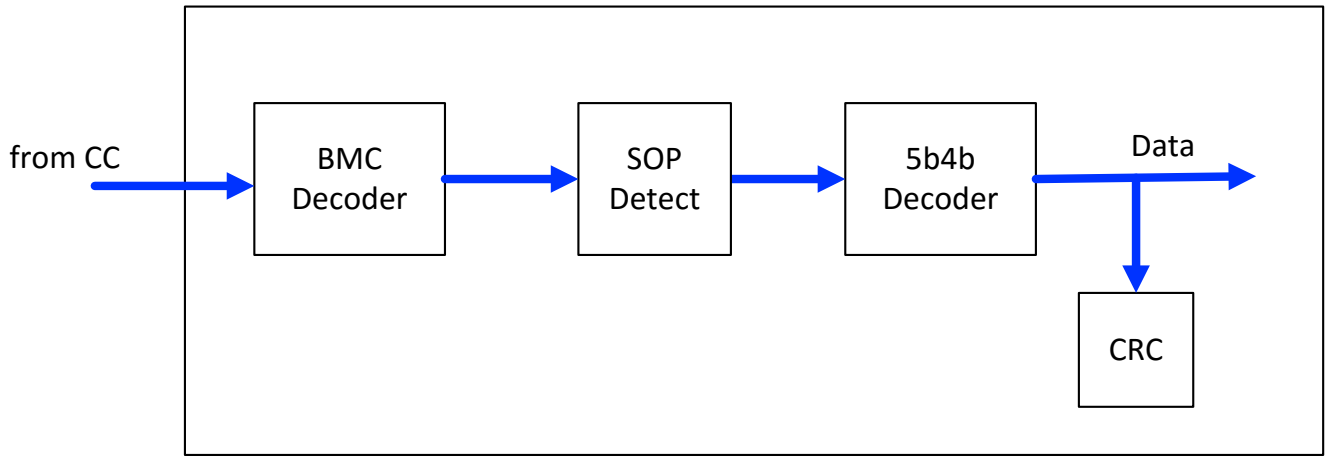


Figure 5-9 BMC Receiver Block Diagram

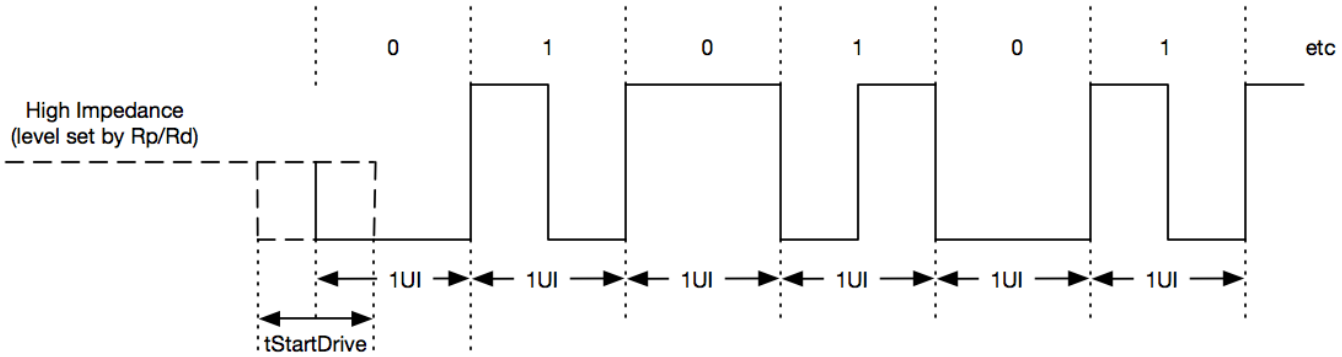


The USB PD baseband signal shall be driven on the CC wire with a tristate driver that shall cause a ***vSwing*** swing on CC. The tristate driver is slew rate limited (see min rise/fall time in Section 5.8.5) to limit coupling to D+/D- and to other signal lines in the USB Type-C fully featured cables (see ***[USB Type-C 1.2]***). This slew rate limiting can be performed either with driver design or an RC filter on the driver output.

When sending the Preamble, the transmitter shall start by transmitting a low level. The receiver shall tolerate the loss of the first edge. The transmitter may vary the start of the Preamble by ***tStartDrive*** min (see Figure 5-10).

Microchip Ex. 1020, p. 78
Microchip v. Aptiv
IPR2024-00496

Figure 5-10 BMC Encoded Start of Preamble

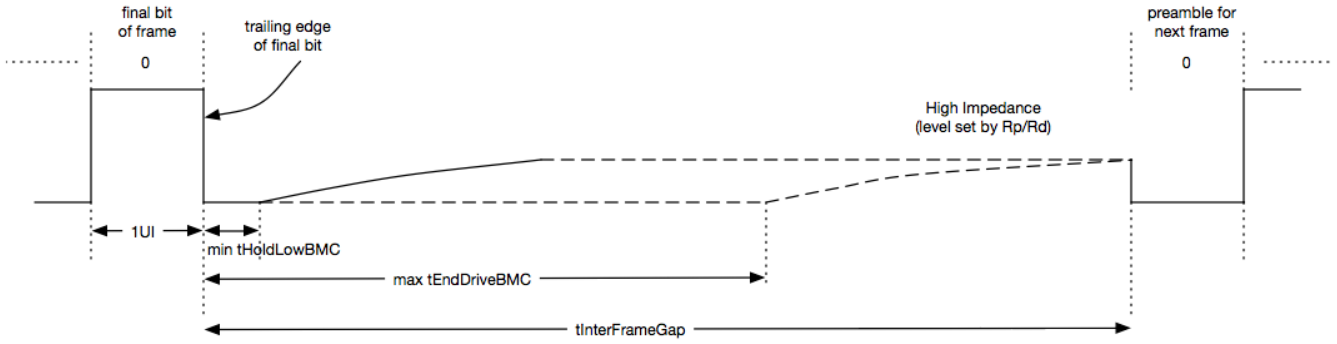


The transmitter shall terminate the final bit of the Frame by an edge (the "trailing edge") to help ensure that the receiver clocks the final bit. If the trailing edge results in the transmitter driving CC low (i.e. the final half-UI of the frame is high), then the transmitter:

1. Shall continue to drive CC low for ***tHoldLowBMC.***
2. Then shall continue to drive CC low for ***tEndDriveBMC*** measured from the trailing edge of the final bit of the Frame.
3. Then shall release CC to high impedance.

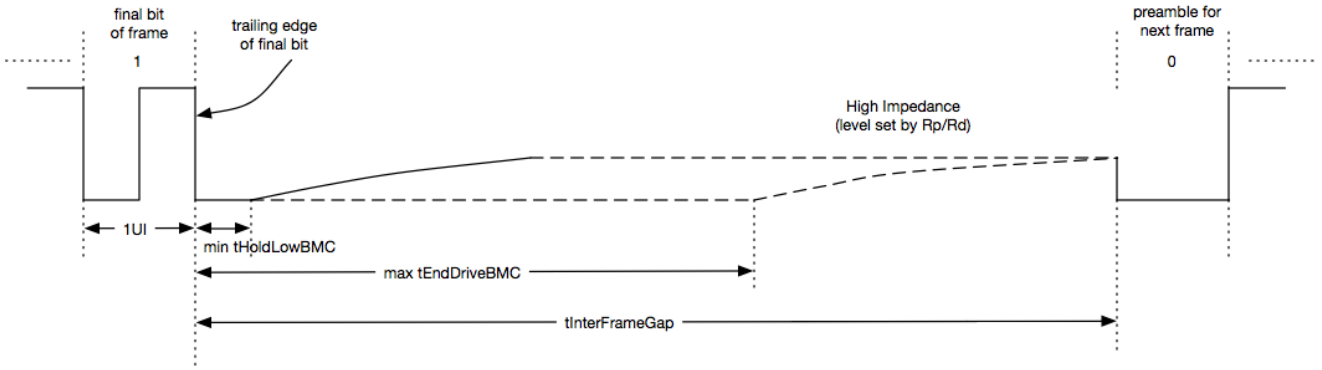
Figure 5-11 illustrates the end of a BMC encoded Frame with an encoded zero for which the final bit of the Frame is terminated by a high to low transition. Figure 5-12 illustrates the end of a BMC Encoded frame with an encoded one for which the final bit of the Frame is terminated by a high to low transition. Both figures also illustrate the ***tInterFrameGap*** timing requirement before the start of the next Frame when the Port has either been transmitting or receiving the previous Frame (see Section 5.8.5.4).

Figure 5-11 Transmitting or Receiving BMC Encoded Frame Terminated by Zero with High-to-Low Last Transition



Microchip Ex. 1020, p. 79
Microchip v. Aptiv
IPR2024-00496

Figure 5-12 Transmitting or Receiving BMC Encoded Frame Terminated by One with High-to-Low Last Transition

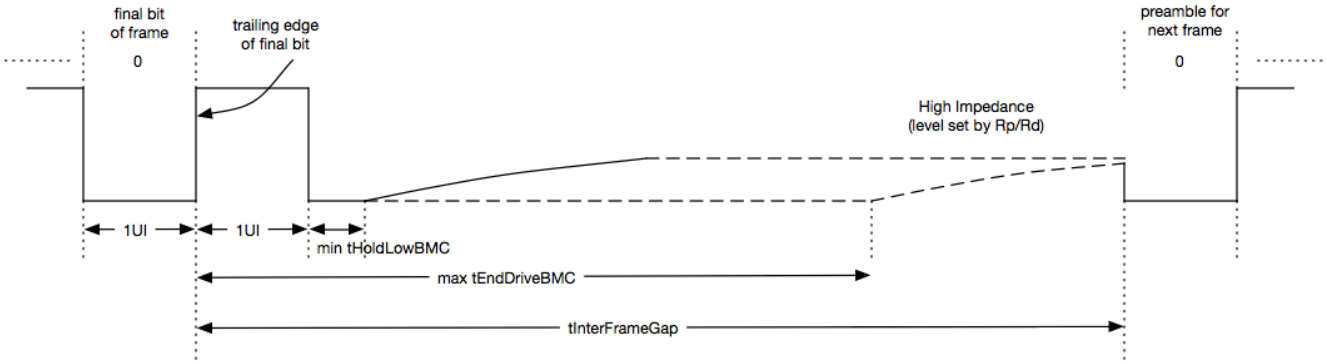


If the trailing edge results in the transmitter driving CC high (i.e. the final half-UI of the frame is low), then the transmitter:

1. Shall continue to drive CC high for 1 UI.
2. Then shall drive CC low for ***tHoldLowBMC***.
3. Then shall continue to drive CC low for ***tEndDriveBMC*** measured from the final edge of the final bit of the Frame.
4. Then shall release CC to high impedance.

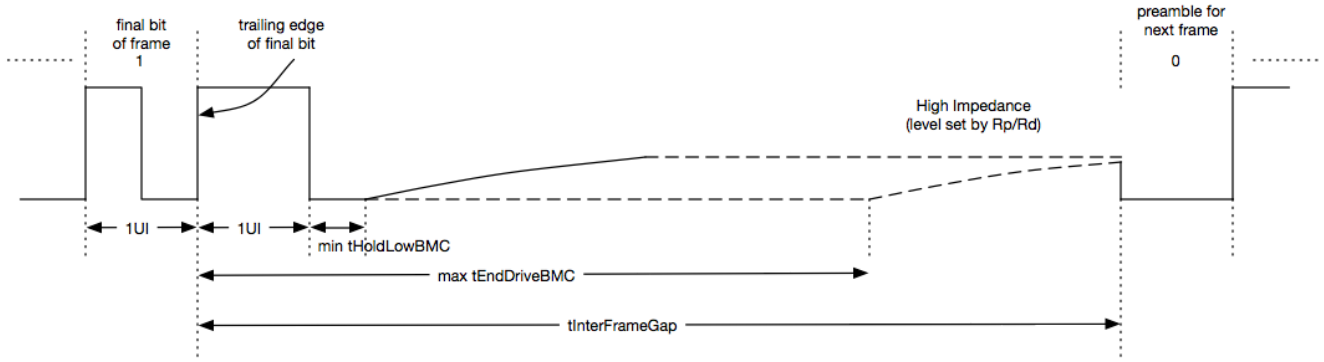
Figure 5-13 illustrates the ending of a BMC encoded Frame that ends with an encoded zero for which the final bit of the Frame is terminated by a low to high transition. Figure 5-14 illustrates the ending of a BMC encoded Frame that ends with an encoded one for which the final bit of the Frame is terminated by a low to high transition. Both figures also illustrate the ***tInterFrameGap*** timing requirement before the start of the next Frame when the Port has either been transmitting or receiving the previous Frame (see Section 5.8.5.4).

Figure 5-13 Transmitting or Receiving BMC Encoded Frame Terminated by Zero with Low to High Last Transition



Microchip Ex. 1020, p. 80
Microchip v. Aptiv
IPR2024-00496

Figure 5-14 Transmitting or Receiving BMC Encoded Frame Terminated by One with Low to High Last Transition



Note: There is no requirement to maintain a timing phase relationship between back-to-back packets.

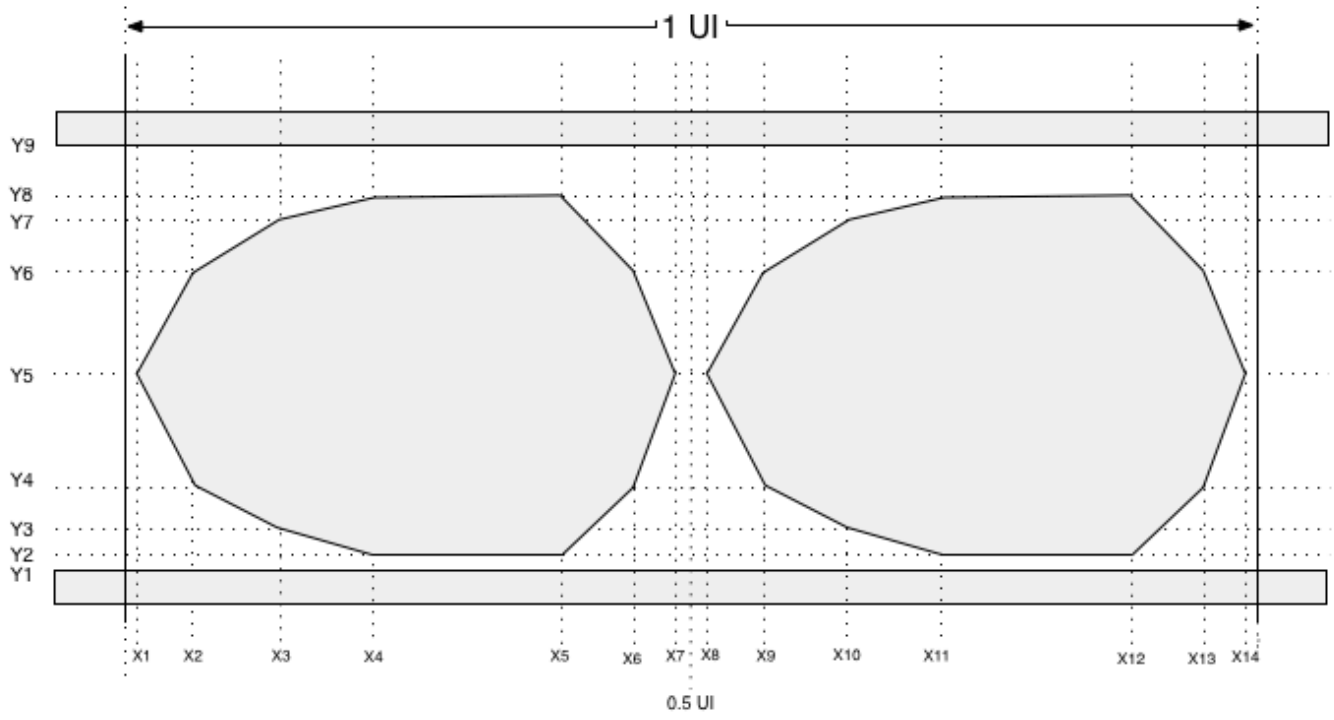
5.8.2 Transmit and Receive Masks

5.8.2.1 Transmit Masks

The transmitted signal shall not violate the masks defined in Figure 5-15, Figure 5-16, Table 5-14 and Table 5-15 at the output of a load equivalent to the cable model and receiver load model described in Section 5.8.3. The masks apply to the full range of Rp/Rd values as defined in ***[USB Type-C 1.2]***. Note: the measurement of the transmitter does not need to accommodate a change in signal offset due to the ground offset when current is flowing in the cable.

The transmitted signal shall have a rise time no faster than ***tRise***. The transmitted signal shall have a fall time no faster than ***tFall***. The maximum limits on the rise and fall times are enforced by the Tx inner masks.

Figure 5-15 BMC Tx 'ONE' Mask



Microchip Ex. 1020, p. 81
Microchip v. Aptiv
IPR2024-00496

Figure 5-16 BMC Tx 'ZERO' Mask

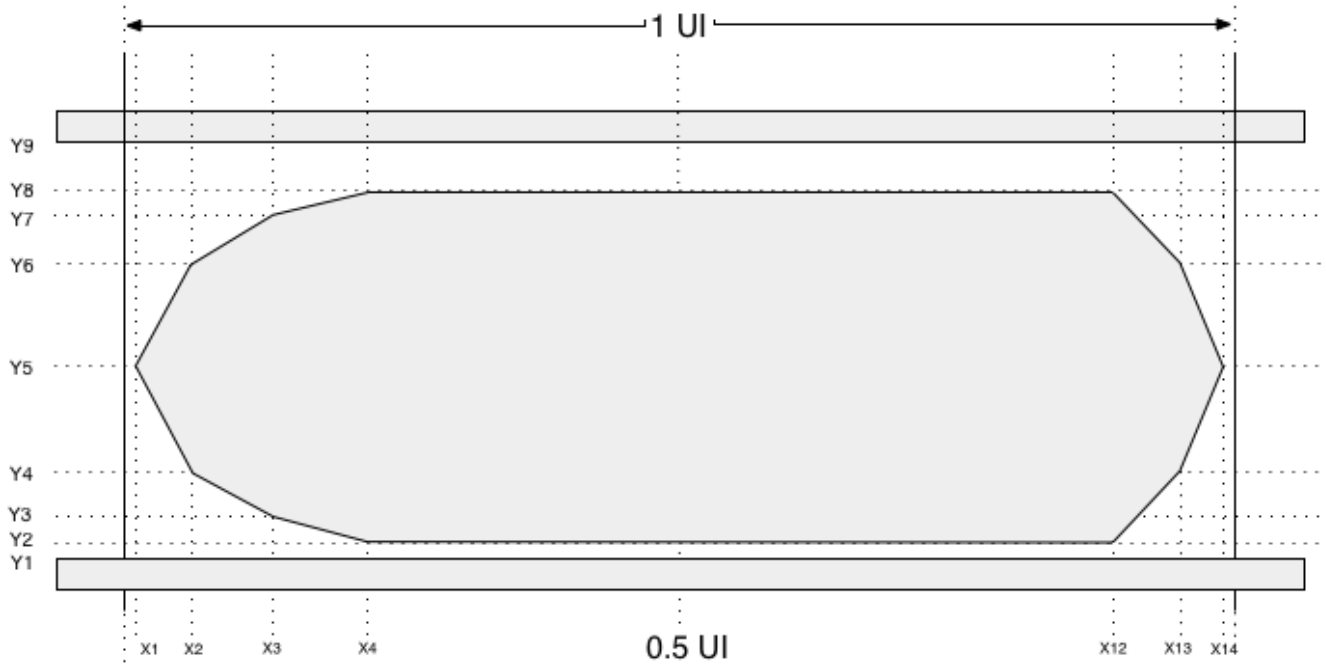


Table 5-14 BMC Tx Mask Definition, X Values

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

Microchip Ex. 1020, p. 82
Microchip v. Aptiv
IPR2024-00496

Table 5-15 BMC Tx Mask Definition, Y Values

Name	Description	Value	Units
Y1Tx	Lower bound of Outer mask	-0.075	V
Y2Tx	Lower bound of inner mask	0.075	V
Y3Tx	see figure	0.15	V
Y4Tx	see figure	0.325	V
Y5Tx	Inner mask vertical midpoint	0.5625	V
Y6Tx	see figure	0.8	V
Y7Tx	see figure	0.975	V
Y8Tx	see figure	1.04	V
Y9Tx	Upper Bound of Outer mask	1.2	V

5.8.2.2 Receive Masks

A Source using the BMC Signaling Scheme shall be capable of receiving a signal that complies with the mask when sourcing power as defined in Figure 5-17, Figure 5-18 and Table 5-16. The Source Rx mask is bounded by sweeping a Tx mask compliant signal, with added ***vNoiseActive*** between power neutral and Source offsets.

A Consumer using the BMC Signaling Scheme shall be capable of receiving a signal that complies with the mask when sinking power as defined in Figure 5-21, Figure 5-22 and Table 5-16. The Consumer Rx mask is bounded by sweeping a Tx mask compliant signal, with added ***vNoiseActive*** between power neutral and Consumer offsets.

Every product using the BMC Signaling Scheme shall be capable of receiving a signal that complies with the mask when power neutral as defined in Figure 5-19, Figure 5-20 and Table 5-16.

Dual-Role Power Devices shall meet the receiver requirements for a Source when providing power during any transmission using the BMC Signaling Scheme or a Sink when consuming power during any transmission using the BMC Signaling Scheme.

Cable Plugs shall meet the receiver requirements for both a Source and a Sink during any transmission using the BMC Signaling Scheme.

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.

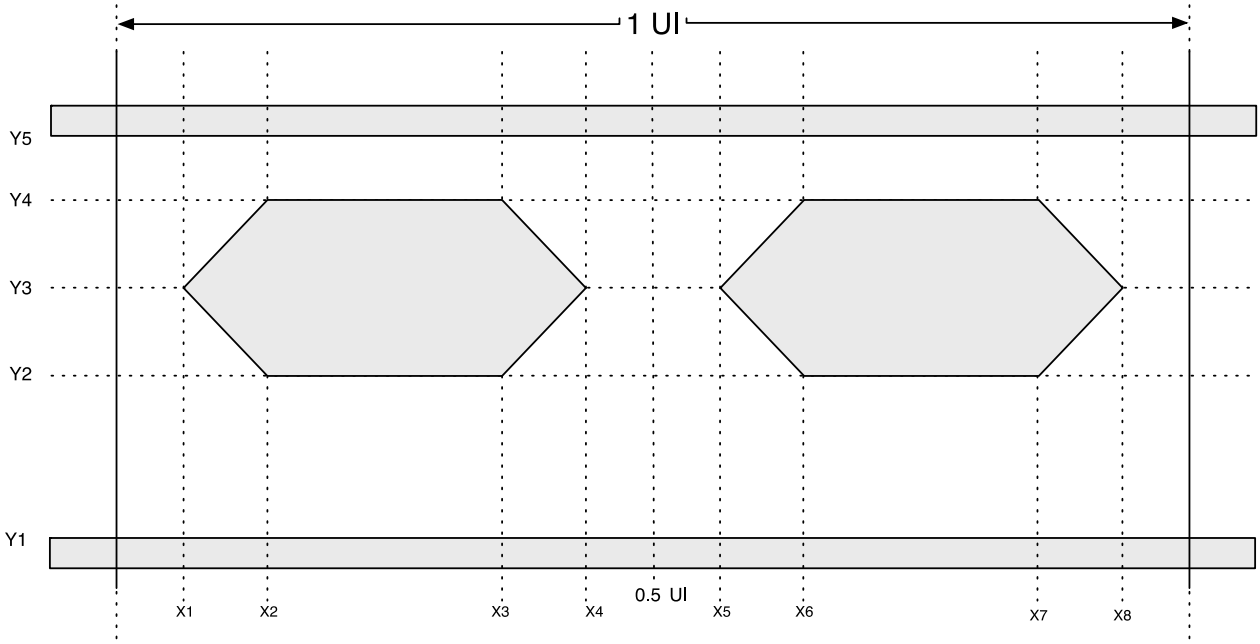
Microchip Ex. 1020, p. 83
Microchip v. Aptiv
IPR2024-00496

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.

The receiver sensitivity shall be set such that the receiver does not treat noise on an undriven signal path as an incoming signal. Signal amplitudes below ***vNoiseIdle*** max shall be treated as noise when BMC is idle.

Figure 5-17 BMC Rx 'ONE' Mask when Sourcing Power



Microchip Ex. 1020, p. 84
Microchip v. Aptiv
IPR2024-00496

Figure 5-18 BMC Rx 'ZERO' Mask when Sourcing Power

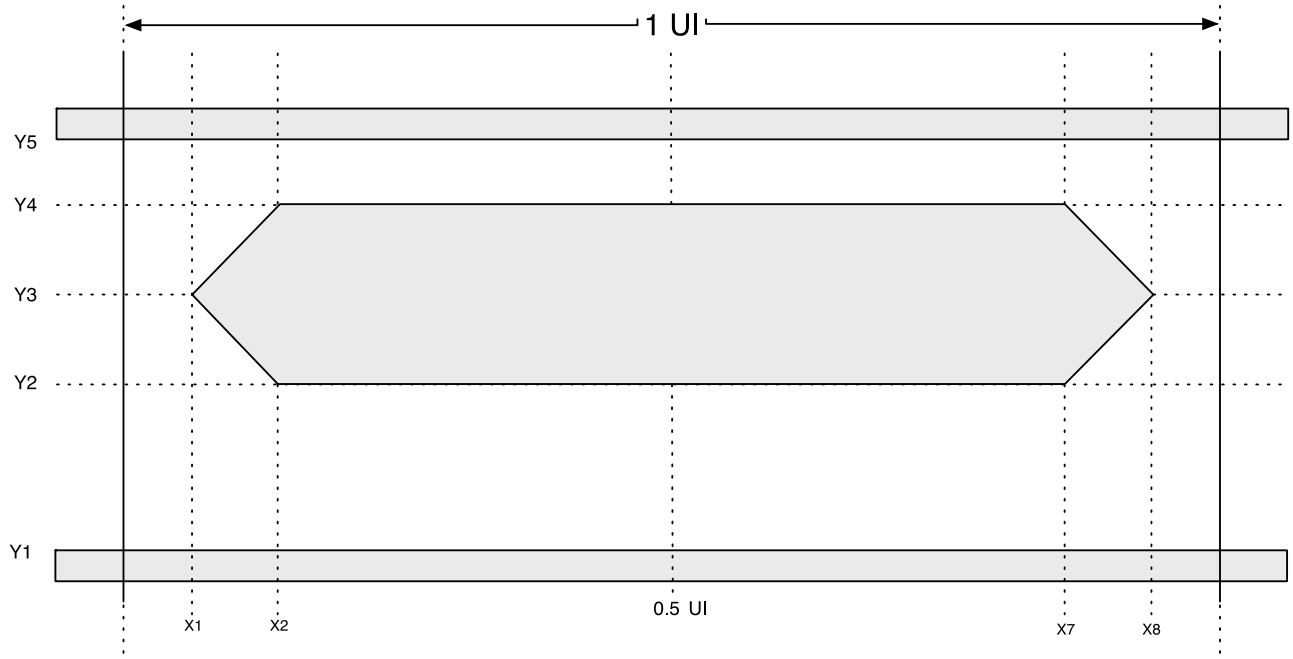
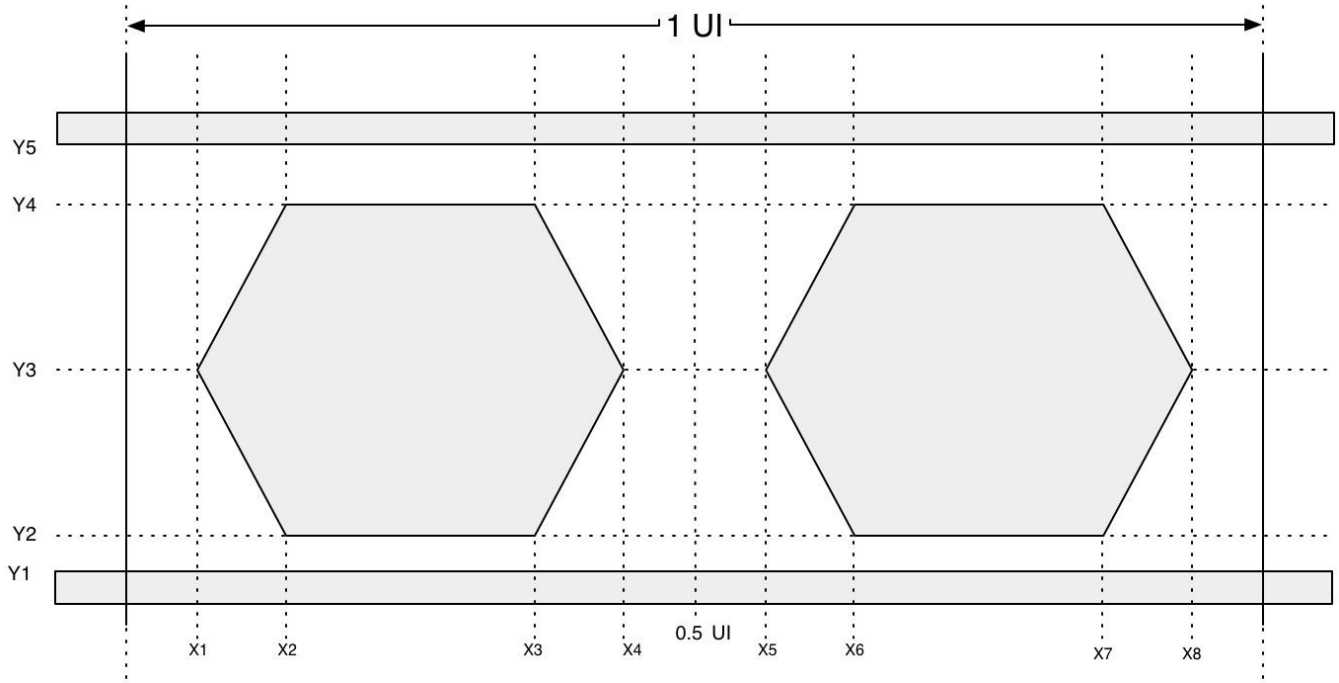


Figure 5-19 BMC Rx 'ONE' Mask when Power neutral



Microchip Ex. 1020, p. 85
Microchip v. Aptiv
IPR2024-00496

Figure 5-20 BMC Rx 'ZERO' Mask when Power neutral

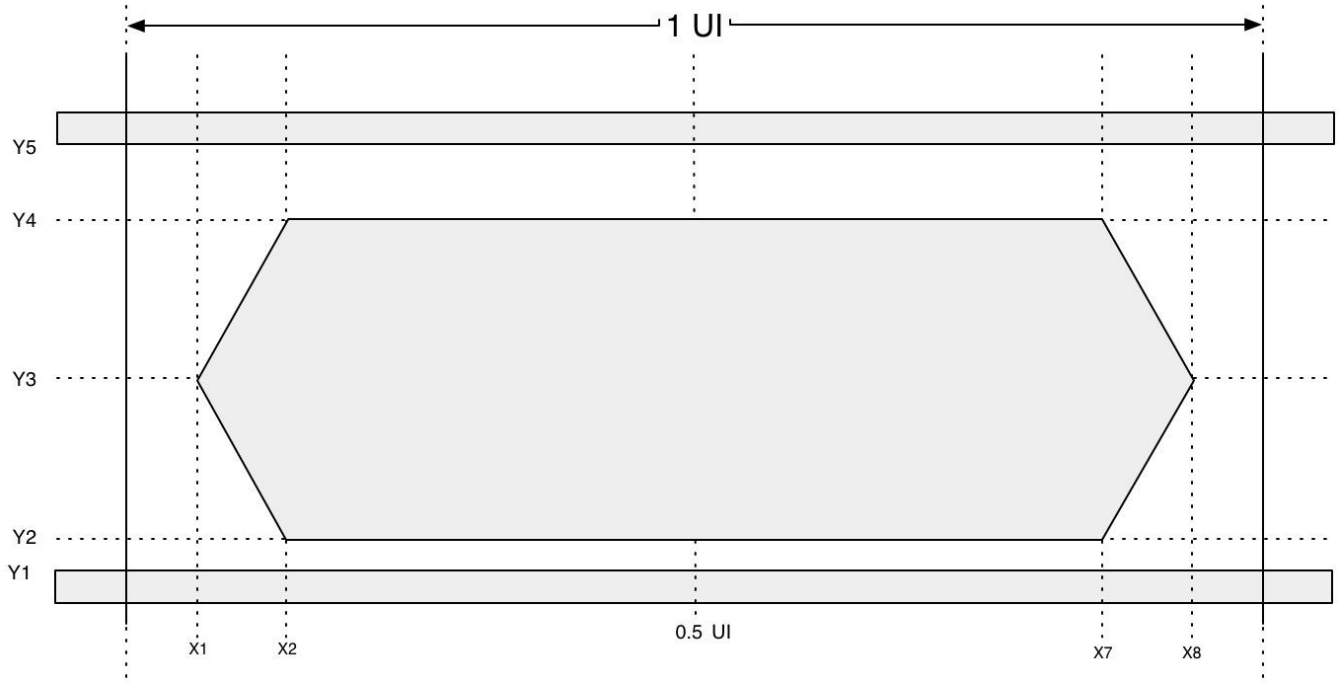
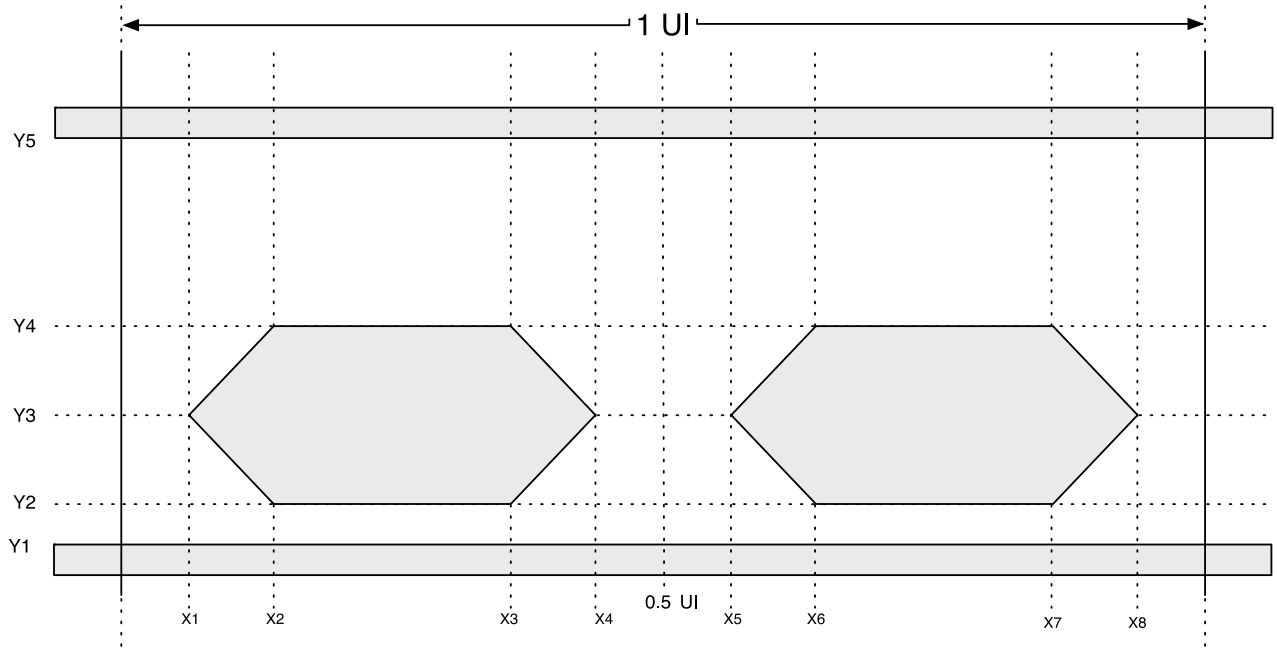


Figure 5-21 BMC Rx 'ONE' Mask when Sinking Power



Microchip Ex. 1020, p. 86
Microchip v. Aptiv
IPR2024-00496

Figure 5-22 BMC Rx 'ZERO' Mask when Sinking Power

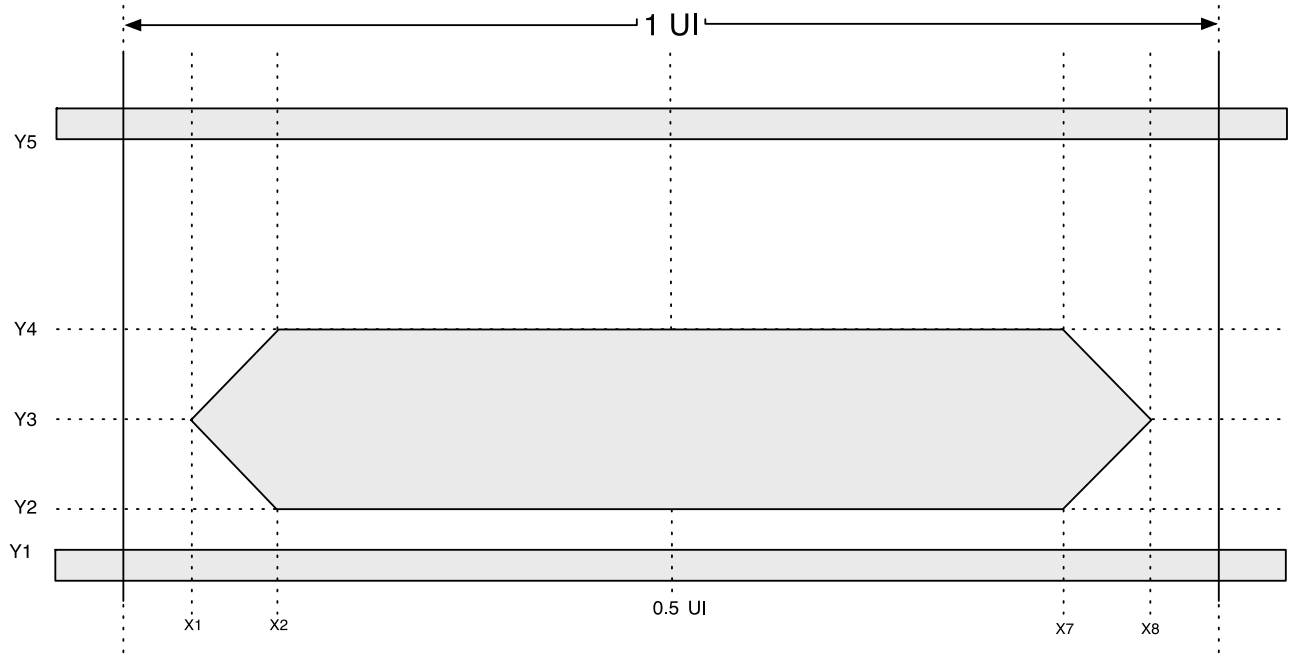


Table 5-16 BMC Rx Mask Definition

Name	Description	Value	Units
X1Rx	Left Edge of Mask	0.07	UI
X2Rx	Top Edge of Mask	0.15	UI
X3Rx	See figure	0.35	UI
X4Rx	See figure	0.43	UI
X5Rx	See figure	0.57	UI
X6Rx	See figure	0.65	UI
X7Rx	See figure	0.85	UI
X8Rx	See figure	0.93	UI
Y1Rx	Lower bound of Outer Mask	-0.3325	V
Y2Rx	Lower Bound of Inner Mask	*Y3Rx* – 0.205 when sourcing power[1] or sinking power[1] *Y3Rx* – 0.33 when power neutral[1]	V
Y3Rx	Center line of Inner Mask	0.6875 Sourcing Power[1] 0.5625 Power Neutral[1] 0.4375 Sinking Power[1]	V
Y4Rx	Upper bound of Inner mask	*Y3Rx* + 0.205 when sourcing power[1] or sinking power[1] *Y3Rx* + 0.33 when power neutral[1]	V
Y5Rx	Upper bound of the Outer mask	1.5325	V
Note 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).			

Microchip Ex. 1020, p. 87
Microchip v. Aptiv
IPR2024-00496

5.8.3 Transmitter Load Model

The transmitter load model shall be equivalent to the circuit outlined in Figure 5-23 for a Source and Figure 5-24 for a Sink. It is formed by the concatenation of a cable load model and a receiver load model. See ***[USB Type-C 1.2]*** for details of the Rp and Rd resistors. Note the parameters *zCable_CC*, *tCableDelay_CC* and *cCablePlug_CC* are defined in ***[USB Type-C 1.2]***.

Figure 5-23 Transmitter Load Model for BMC Tx from a Source

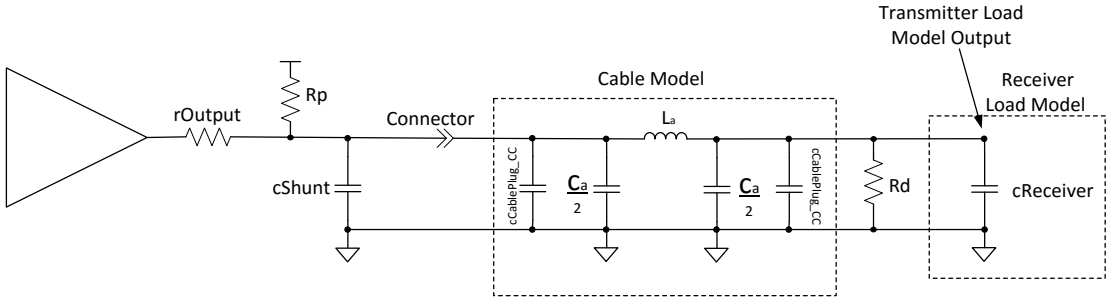
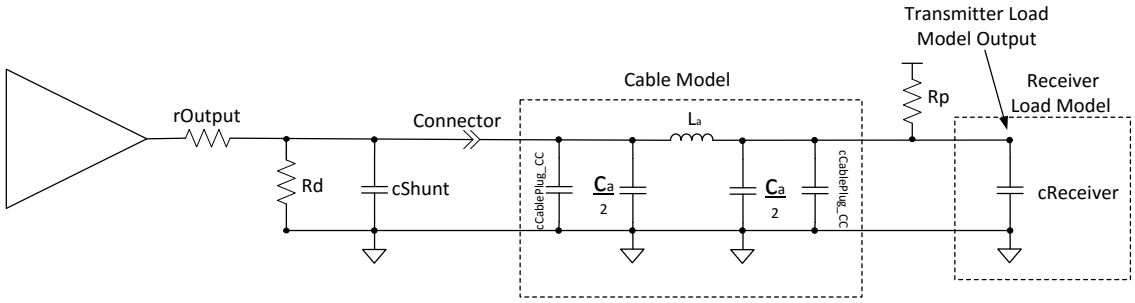


Figure 5-24 Transmitter Load Model for BMC Tx from a Sink



The transmitter system components *rOutput* and *cShunt* are illustrated for informative purposes, and do not form part of the transmitter load model. See Section 5.8.5 for a description of the transmitter system design.

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

$$La = tCableDelay\_CC_{max} * zCable\_CC_{min}$$

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

The modeled cable inductance is 640 nH for a cable with $zCable\_CC_{min}$ = 32 Ω and $tCableDelay\_CC_{max}$ = 20 nS.

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

$$Ca = \frac{tCableDelay\_CC_{max}}{zCable\_CC_{min}}$$

The modeled cable capacitance is Ca = 625 pF for a cable with $zCable\_CC_{min}$ = 32 Ω and $tCableDelay\_CC_{max}$ = 20 nS. Therefore, Ca/2 = 312.5 pF.

cCablePlug_CC models the capacitance of the plug at each end of the cable. ***cReceiver*** models the capacitance of the receiver. The maximum values shall be used in each case.

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

Microchip Ex. 1020, p. 88
Microchip v. Aptiv
IPR2024-00496

5.8.4 BMC Common specifications

This section defines the common receiver and transmitter requirements.

5.8.4.1 BMC Common Parameters

The electrical requirements specified in Table 5-17 shall apply to both the transmitter and receiver.

Table 5-17 BMC Common Normative Requirements

Name	Description	Min	Nom	Max	Units	Comment
fBitRate	Bit rate	270	300	330	Kbps	
tUnitInterval[1]	Unit Interval	3.03		3.70	µs	1/***fBitRate***
Note 1: ***tUnitInterval*** 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.						

5.8.5 BMC Transmitter Specifications

The transmitter shall meet the specifications defined in Table 5-18.

Table 5-18 BMC Transmitter Normative Requirements

Name	Description	Min	Nom	Max	Units	Comment
pBitRate	Maximum difference between the bit-rate during the part of the packet following the Preamble and the reference bit-rate.			0.25	%	The reference bit rate is the average bit rate of the last 32 bits of the Preamble.
rFRSwapTx	Fast Role Swap request transmit driver resistance (excluding cable resistance)			5	Ω	Maximum driver resistance of a Fast Role Swap request transmitter. Assumes a worst case cable resistance of 15Ω as defined in ***[USB Type-C 1.2]***. Note: based on this value the maximum combined driver and cable resistance of a Fast Role Swap request transmitter is 20Ω.
tEndDriveBMC	Time to cease driving the line after the end of the last bit of the Frame.			23	µs	Min value is limited by ***tHoldLowBMC***.
tFall	Fall Time	300			ns	10 % and 90 % amplitude points, minimum is under an unloaded condition.
tHoldLowBMC	Time to cease driving the line after the final high-to-low transition.	1			µs	Max value is limited by ***tEndDriveBMC***.
tInterFrameGap	Time from the end of last bit of a Frame until the start of the first bit of the next Preamble.	25			µs	
tFRSwapTx	Fast Role Swap request transmit duration	60		120	µs	Fast Role Swap request is indicated from the initial Source to the initial Sink by driving CC low for this time.

Microchip Ex. 1020, p. 89
Microchip v. Aptiv
IPR2024-00496

Name	Description	Min	Nom	Max	Units	Comment
tRise	Rise time	300			ns	10 % and 90 % amplitude points, minimum is under an unloaded condition.
tStartDrive	Time before the start of the first bit of the Preamble when the transmitter shall start driving the line.	-1		1	µs	
vSwing	Voltage Swing	1.05	1.125	1.2	V	Applies to both no load condition and under the load condition specified in Section 5.8.3.
zDriver	Transmitter output impedance	33		75	Ω	Source output impedance at the Nyquist frequency of *[USB 2.0]* low speed (750 kHz) while the source is driving the CC line.

5.8.5.1 Capacitance when not transmitting

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. The transmitter may have more capacitance than *cReceiver* while driving the CC line, but must meet the waveform mask requirements. Once transmission is complete, the transmitter shall disengage capacitance in excess of *cReceiver* from the CC wire within *tInterFrameGap*.

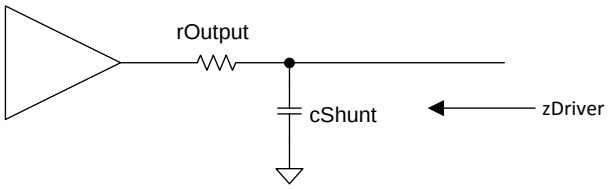
5.8.5.2 Source Output Impedance

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 ingression in the cable. It is specified such that the noise at the Receiver is bounded.

zDriver is defined by the following equation:

$$zDriver = \frac{rOutput}{1 + s * rOutput * cShunt}$$

Figure 5-25 Transmitter diagram illustrating zDriver



cShunt shall not cause a violation of *cReceiver* when not transmitting.

5.8.5.3 Bit Rate Drift

Limits on the drift in *fBitRate* are set in order 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. The change in *fBitRate* during a packet shall be less than *pBitRate*. 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:

pBitRate = | *fBitRate* – refBitRate | / refBitRate x 100%

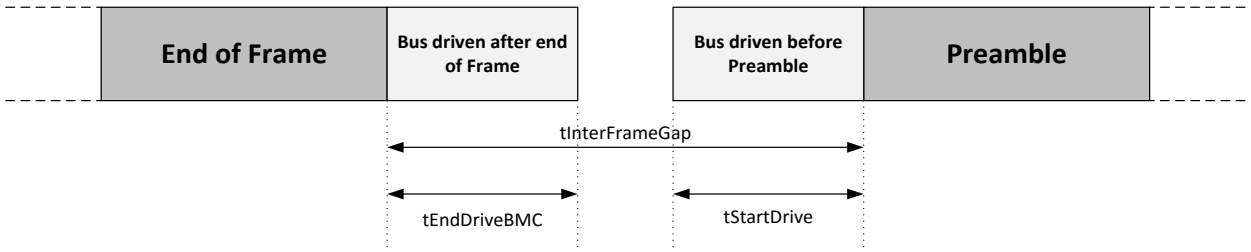
Microchip Ex. 1020, p. 90
Microchip v. Aptiv
IPR2024-00496

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. 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 .

5.8.5.4 Inter-Frame Gap

Figure 5-26 illustrates the inter-Frame gap timings.

Figure 5-26 Inter-Frame Gap Timings



The transmitter shall drive the bus for no longer than ***tEndDriveBMC*** after transmitting the final bit of the Frame.

Before starting to transmit the next Frame's Preamble the transmitter of the next Frame shall ensure that it waits for ***tInterFrameGap*** after either:

1. Transmitting the previous frame, for example sending the next Message in an AMS immediately after having sent a ***GoodCRC*** Message, or
2. Receiving the previous frame, for example when responding to a received Message with a ***GoodCRC*** Message, or
3. Observing an idle condition on CC (see Section 5.7). In this case the Port is waiting to initiate an AMS observes idle (see Section 5.8.6.1) and then waits ***tInterFrameGap*** before transmitting the Frame. See also Section 5.7 for details on when an AMS can be initiated.

Note: the transmitter is also required to verify a bus idle condition immediately prior to starting transmission of the next Frame (see Section 5.8.6.1).

The transmitter of the next Frame may vary the start of the Preamble by ***tStartDrive*** (see Section 5.8.1).

See also Section 5.8.1 for figures detailing the timings relating to transmitting, receiving and observing idle in relating to Frames.

5.8.5.5 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 while the Transmitter is in the process of transmitting (see Section 5.8.5.6).

5.8.5.6 Fast Role Swap Transmission

The Fast Role Swap process is intended for use by a PDUSB HUB that presently has an external wall supply, and is providing power both through its downstream Ports to USB Devices and upstream to a USB Host such as a notebook. On removal of the external wall supply Fast Role Swap enables a V_{BUS} supply to be maintained by allowing the USB Host to apply ***vSafe5V*** when it sees V_{BUS} droop below ***vSafe5V*** after having detected Fast Role Swap signaling. The Fast Role Swap AMS is then used to correctly assign Source/Sink roles and configure the Rp/Rd resistors (see Section 8.3.2.7).

The initial Source shall signal a Fast Role Swap request by driving CC to ground with a resistance of less than ***rFRSwapTx*** for ***tFRSwapTx***. The initial Source shall only signal a Fast Role Swap when it has an Explicit Contract. On transmission of the Fast Role Swap signal any pending Messages shall be ***Discarded*** (see Section 6.11.2.1.1).

Microchip Ex. 1020, p. 91
Microchip v. Aptiv
IPR2024-00496

The Fast Role Swap signal may override any active transmissions.

Since the initial Sink's response to the Fast Role Swap signal is to send an ***FR_Swap*** Message, the initial Source shall ensure Rp is set to ***SinkTxOk*** once the Fast Role Swap signal is complete.

5.8.6 BMC Receiver Specifications

The receiver shall meet the specifications defined in Table 5-19.

Table 5-19 BMC Receiver Normative Requirements

Name	Description	Min	Nom	Max	Units	Comment
cReceiver	CC receiver capacitance	200		600	pF	The DFP or UFP system shall have capacitance within this range when not transmitting on the line.
nBER	Bit error rate, S/N = 25 dB			10^{-6}		
nTransitionCount	Transitions for signal detect	3				Number of transitions to be detected to declare bus non-idle.
tFRSwapRx	Fast Role Swap request detection time	30		50	µs	A Fast Role Swap request results in the receiver detecting a signal low for at least this amount of time.
tRxFilter	Rx bandwidth limiting filter (digital or analog)	100			ns	Time constant of a single pole filter to limit broad-band noise ingression[1].
tTransitionWindow	Time window for detecting non-idle	12		20	µs	
vFRSwapCableTx	Fast Role Swap request voltage detection threshold	490	520	550	mV	The Fast Role Swap request has to be below this voltage threshold to be detected.
vIRDropGNDC	Cable Ground IR Drop			250	mV	As specified in ***[USB Type-C 1.2]***
vNoiseActive	Noise amplitude when BMC is active.			165	mV	Peak-to-peak noise from V_{BUS}, USB 2.0 and SBU lines after the Rx bandwidth limiting filter with the time constant ***tRxFilter*** has been applied.
vNoiseIdle	Noise amplitude when BMC is idle.			300	mV	Peak-to-peak noise from V_{BUS}, USB 2.0 and SBU lines after the Rx bandwidth limiting filter with the time constant ***tRxFilter*** has been applied.
zBmcRx	Receiver Input Impedance	1			MΩ	
Note 1: Broad-band noise ingression is due to coupling in the cable interconnect.						

5.8.6.1 Definition of Idle

BMC packet collision is avoided by the detection of signal transitions at the receiver. This is the equivalent of squelch for FSK modulation. Detection is active when ***nTransitionCount*** transitions occur at the receiver within a time window of ***tTransitionWindow***. After waiting ***tTransitionWindow*** without detecting ***nTransitionCount*** transitions the bus shall be declared idle.

Refer to Section 5.8.5.4 for details of when transmissions may start.

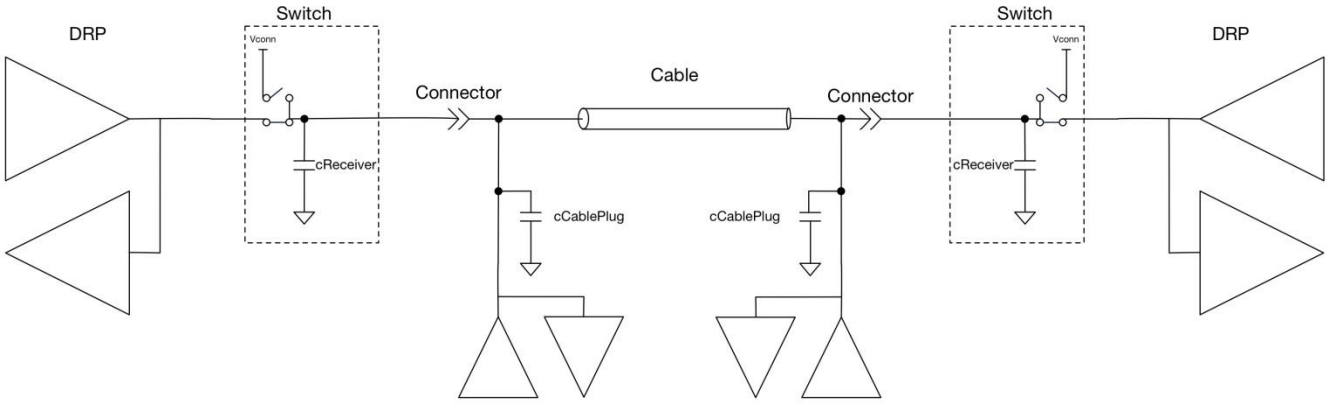
Microchip Ex. 1020, p. 92
Microchip v. Aptiv
IPR2024-00496

5.8.6.2 Multi-Drop

The BMC Signaling Scheme is suitable for use in Multi-Drop configurations containing one or two BMC Multi-Drop transceivers connected to the CC wire, for example where one or both ends of a cable contains a Multi-Drop transceiver. In this specification the location of the Multi-Drop transceiver is referred to as the Cable Plug.

Figure 5-27 below illustrates a typical Multi Drop configuration with two DRPs.

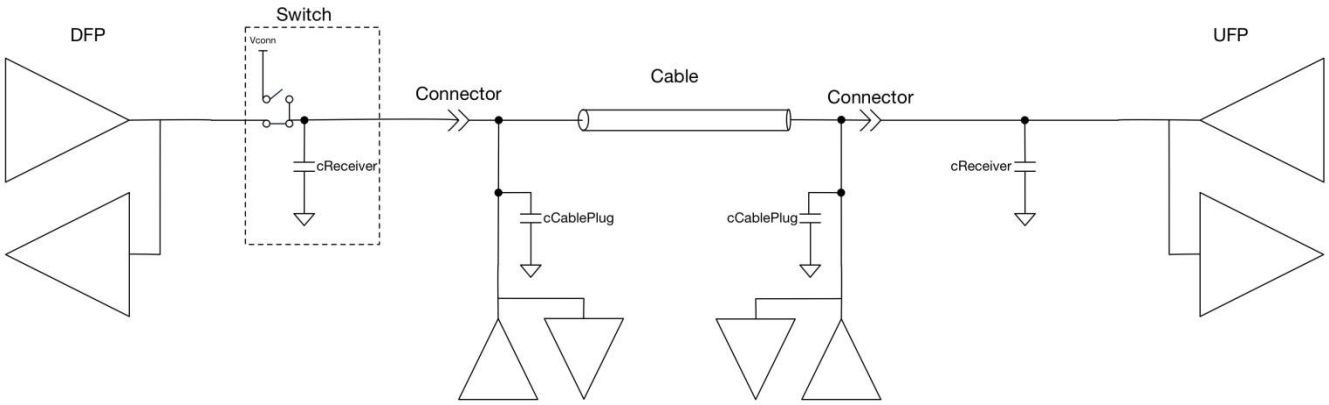
Figure 5-27 Example Multi-Drop Configuration showing two DRPs



The Multi-Drop transceiver shall obey all the electrical characteristics specified in this section except for those relating to capacitance. The maximum capacitance allowed for the Multi-Drop node when not driving the line is *cCablePlug_CC* defined in ***[USB Type-C 1.2]*** . There are no constraints as to the distance of the Multi-Drop transceiver from the end of the plug. The Multi-Drop transceiver(s) may be located anywhere along the cable including the plugs. The Multi-Drop transceiver suffers less from ground offset compared to the transceivers in the host or device and contributes no significant reflections.

It is possible to have a configuration at Attach where one Port is able to be a Vconn Source and the other Port is not able to be a Vconn Source, such that there is no switch in the second Port. An example of a DFP with a switch Attached to a UFP without a switch is outlined in Figure 5-28. The capacitance on the CC line for a Port not able to be a VCONN Source shall still be within ***cReceiver*** except when transmitting.

Figure 5-28 Example Multi-Drop Configuration showing a DFP and UFP



5.8.6.3 Fast Role Swap Detection

An initial Sink prepares for a Fast Role Swap by ensuring that once it has detected the Fast Role Swap signal its power supply is ready to respond by applying ***vSafe5V*** according to the timing detailed in Section 7.1.14. The initial Sink

Microchip Ex. 1020, p. 93
Microchip v. Aptiv
IPR2024-00496

shall only respond to the Fast Role Swap signal when it has an Explicit Contract. On detection of the Fast Role Swap signal any pending Messages shall be ***Discarded*** (see Section 6.11.2.1.1).

When the initial Sink is prepared for a Fast Role Swap and the bus is idle the CC voltage averaged over ***tFRSwapRx*** min remains above 0.7V (see ***[USB Type-C 1.2]***) since the Source Rp is either 1.5A or 3.0A. However ***vNoiseIdle*** noise may cause the CC line voltage to reach 0.7V-***vNoiseIdle***/2 for short durations. When the initial Sink is prepared for a Fast Role swap while it is transmitting and the initial Source is signaling a Fast Role swap request, the transmission will be attenuated such that the peak CC voltage will not exceed ***vFRSwapCableTx*** min. Therefore, when the initial Sink is prepared for a Fast Role Swap, it shall not detect a Fast Swap signal when the CC voltage, averaged over ***tFRSwapRx*** min, is above 0.7V. When the initial Sink is prepared for a Fast Role Swap, it shall detect a CC voltage lower than ***vFRSwapCableTx*** min for ***tFRSwapRx*** as a Fast Role Swap request. Note: the initial Sink is not required to average the CC voltage to meet these requirements.

The initial Sink shall initiate the Fast Role Swap AMS within ***tFRSwapInit*** of detecting the Fast Role Swap request in order to assign the Rp/Rd resistors to the correct Ports and to re-synchronize the state machines (see Section 6.3.17).

The initial Sink shall become the new Source and shall start supplying ***vSafe5V*** at USB Type-C Current (see ***[USB Type-C 1.2]***) no later than ***tSrcFRSwap*** after V_{BUS} has dropped below ***vSafe5V***. An initial Sink shall disable its V_{BUS} Disconnect Threshold detection circuitry while Fast Role Swap detection is active.

Note: while power is transitioning the V_{CONN} Source to the Cable Plug(s) cannot be guaranteed.

Microchip Ex. 1020, p. 94
Microchip v. Aptiv
IPR2024-00496

5.9 Built in Self-Test (BIST)

The following sections define BIST functionality which shall be supported.

5.9.1 BIST Carrier Mode

In ***BIST Carrier Mode***, the Physical 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.

Note that this transmission is a purely a sequence of alternating bits and shall not be formatted as a Packet.

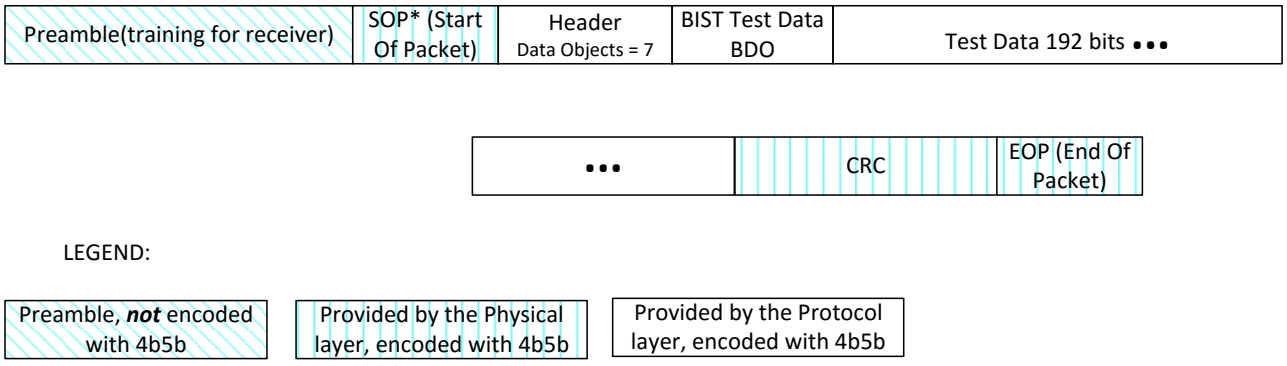
See also Section 6.4.3.

5.9.2 BIST Test Data

A ***BIST Test Data*** Message is used by the Tester to send various Tester generated test patterns to the UUT in order to test the UUT’s receiver. See also Section 6.4.3.

Figure 5-29 shows the Test Data Frame which shall be sent by the Tester to the UUT. The ***BIST*** Message, with a ***BIST Test Data*** BIST Data Object consists of a Preamble, followed by ***SOP****, followed by the Message Header with a data length of 7 Data Objects, followed a ***BIST Test Data*** BIST Data Object, followed by 6 Data Objects containing Test data, followed by the CRC and then an ***EOP***.

Figure 5-29 Test Data Frame



Microchip Ex. 1020, p. 95
Microchip v. Aptiv
IPR2024-00496

6. Protocol Layer

6.1 Overview

This chapter describes the requirements of the USB Power Delivery Specification's protocol layer including:

- Details of how Messages are constructed and used
- Use of timers and timeout values
- Use of Message and retry counters
- Reset operation
- Error handling
- State behavior

Refer to Section 2.6 for an overview of the theory of operation of USB Power Delivery.

6.2 Messages

This specification defines three types of Messages:

- Control Messages that are short and used to manage the Message flow between Port Partners or to exchange Messages that require no additional data. Control Messages are 16 bits in length.
- Data Messages that are used to exchange information between a pair of Port Partners. Data Messages range from 48 to 240 bits in length. There are three types of Data Messages:
    - Those used to expose capabilities and negotiate power
    - Those used for the BIST
    - Those that are Vendor Defined
- Extended Messages that are used to exchange information between a pair of Port Partners. Extended Messages are up to MaxExtendedMsgLen bytes.
- There are several types of Extended Messages:
    - Those used for Source and Battery information
    - Those used for Security
    - Those used for Firmware Update
    - Those that are vendor defined

6.2.1 Message Construction

All Messages shall be composed of a Message Header and a variable length (including zero) data portion. A Message either originates in the Protocol Layer and is passed to the Physical Layer, or it is received by the Physical Layer and is passed to the Protocol Layer.

Figure 6-1 illustrates a Control Message as part of a Packet showing the parts are provided by the Protocol and PHY Layers.

Figure 6-1 USB Power Delivery Packet Format including Control Message Payload

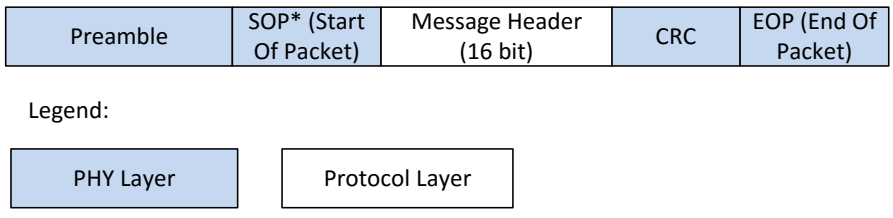


Figure 6-2 illustrates a Data Message as part of a Packet showing the parts are provided by the Protocol and PHY Layers.

Microchip Ex. 1020, p. 96
Microchip v. Aptiv
IPR2024-00496

Figure 6-2 USB Power Delivery Packet Format including Data Message Payload

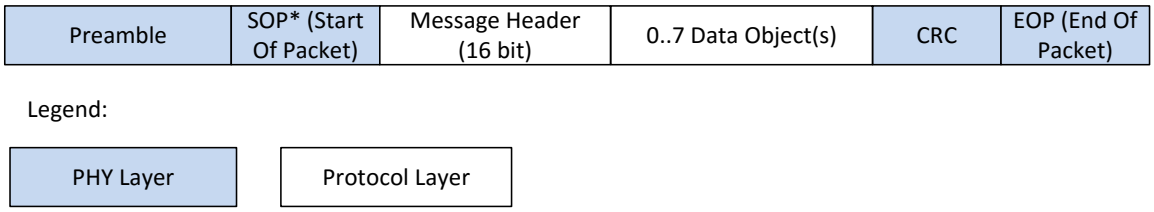
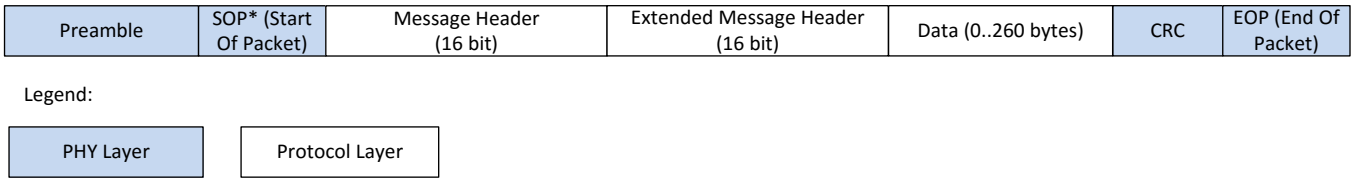


Figure 6-3 illustrates an Extended Message as part of a Packet showing the parts are provided by the Protocol and PHY Layers.

Figure 6-3 USB Power Delivery Packet Format including an Extended Message Header and Payload



6.2.1.1 Message Header

Every Message shall start with a Message Header as shown in Figure 6-1, Figure 6-2 and Figure 6-3 and as defined in Table 6 1. The Message Header contains basic information about the Message and the PD Port Capabilities.

The Message Header may be used standalone as a Control Message when the Number of Data Objects field is zero or as the first part of a Data Message when the ***Number of Data Objects*** field is non-zero.

Table 6-1 Message Header

Bit(s)	Start of Packet	Field Name	Reference
15	SOP*	***Extended***	Section 6.2.1.1.1
14..12	SOP*	***Number of Data Objects***	Section 6.2.1.1.2
11..9	SOP*	***MessageID***	Section 6.2.1.1.3
8	SOP only	***Port Power Role***	Section 6.2.1.1.4
	SOP'/SOP''	***Cable Plug***	Section 6.2.1.1.7
7..6	SOP*	***Specification Revision***	Section 6.2.1.1.5
5	SOP only	***Port Data Role***	Section 6.2.1.1.6
	SOP'/SOP''	***Reserved***	Section 1.4.2.9
4..0	SOP*	***Message Type***	Section 6.2.1.1.8

6.2.1.1.1 Extended

The 1-bit ***Extended*** field shall be set to zero to indicate a Control Message or Data Message and set to one to indicate an Extended Message.

The ***Extended*** field shall apply to all SOP* Packet types.

6.2.1.1.2 Number of Data Objects

When the ***Extended*** field is set to zero the 3-bit ***Number of Data Objects*** field shall indicate the number of 32-bit Data Objects that follow the Message Header. When this field is zero the Message is a Control Message and when it is non-zero, the Message is a Data Message.

Microchip Ex. 1020, p. 97
Microchip v. Aptiv
IPR2024-00496

The ***Number of Data Objects*** field shall apply to all SOP* Packet types.

When both the ***Extended*** bit and ***Chunked*** bit are set to one, the ***Number of Data Objects*** field shall indicate the number of Data Objects in the Message padded to the 4-byte boundary including the Extended Header as part of the first Data Object.

When the ***Extended*** bit is set to one and ***Chunked*** bit is set to zero, the ***Number of Data Objects*** field shall be ***Reserved***. Note that in this case, the message length is determined solely by the ***Data Size*** field in the Extended Message Header.

6.2.1.1.3 MessageID

The 3-bit ***MessageID*** field is the value generated by a rolling counter maintained by the originator of the Message. The ***MessageIDCounter*** shall be initialized to zero at power-on as a result of a Soft Reset, or a Hard Reset. The ***MessageIDCounter*** shall be incremented when a Message is successfully received as indicated by receipt of a ***GoodCRC*** Message. Note: the usage of ***MessageID*** during testing with BIST Messages is defined in ***[USBPDCompliance]***.

The ***MessageID*** field shall apply to all SOP* Packet types.

6.2.1.1.4 Port Power Role

The 1-bit ***Port Power Role*** field shall indicate the Port's present power role:

- 0b Sink
- 1b Source

Messages, such as ***Ping***, and ***GotoMin***, that are only ever sent by a Source, shall always have the ***Port Power Role*** field set to Source. Similarly Messages such as the ***Request*** Message that are only ever sent by a Sink shall always have the ***Port Power Role*** field set to Sink.

During the Power Role Swap Sequence, 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 (see Figure 8-6 and Figure 8-7).

During the Power Role Swap Sequence, for the initial Sink Port, 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 (see Figure 8-6 and Figure 8-7).

During the Fast Role Swap Sequence, for the initial Source Port, the ***Port Power Role*** field shall be set to Sink in the ***PS_RDY*** Message indicating that V_{BUS} is not being driven by the initial Source and is within ***vSafe5V*** (see Figure 8-13).

During the Fast Role Swap Sequence, for the initial Sink Port, 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 (see Figure 8-13).

Note that the ***GoodCRC*** Message sent by the initial Sink in response to the ***PS_RDY*** Message from the initial Source will have its ***Port Power Role*** field set to Sink since this is initiated by the Protocol Layer. Subsequent Messages initiated by the Policy Engine, such as the ***PS_RDY*** Message sent to indicate that V_{BUS} is ready, will have the ***Port Power Role*** field set to Source.

The ***Port Power Role*** field of a received Message shall not be verified by the receiver and shall not lead to Soft Reset, Hard Reset or Error Recovery if it is incorrect.

The ***Port Power Role*** field shall only be defined for SOP Packets.

6.2.1.1.5 Specification Revision

The 2-bit ***Specification Revision*** field for all Messages shall contain highest the Revision of the Power Delivery Specification supported by the device. For reasons of implementation the ***Specification Revision*** field may contain a different Revision in ***GoodCRC*** Message to other the Messages; all other Messages shall contain the same Revision value.

The ***Specification Revision*** field shall be one of the following values (except 11b):

Microchip Ex. 1020, p. 98
Microchip v. Aptiv
IPR2024-00496

- 00b –Revision 1.0
- 01b –Revision 2.0
- 10b – Revision 3.0
- 11b – ***Reserved***, shall not be used

To ensure interoperability with existing USBPD Products, USBPD Products shall support every PD Specification Revision starting from ***[USBPD 2.0]***.

The ***Specification Revision*** field shall apply to all SOP* Packet types.

The ***Specification Revision*** field of the ***Source_Capabilities*** Message sent by the Source during the negotiation of an Explicit Contract shall be used by the Sink to determine the Specification Revision supported by the Source. Similarly ***Specification Revision*** field of the ***Request*** Message sent by the Sink during the negotiation of an Explicit Contract shall be used by the Source to determine the Specification Revision supported by the Sink. Port Partners shall establish an Explicit Contract using their highest common Specification Revision and continue to operate using this Specification Revision until they are Detached. After the Explicit Contract has been made the Revision in the ***Specification Revision*** field shall match the agreed Revision.

Before there is an Explicit Contract the VCONN Source shall communicate with a Cable Plug using ***[USBPD 2.0]***. After there is an Explicit Contract the VCONN Source shall communicate with a Cable Plug using the highest common Specification Revision based on the Revision agreed between the Port Partners and the highest Revision supported by the Cable Plug it is communicating with. A Cable Plug shall use the Revision indicated by the Vconn Source in its ***Specification Revision*** field to determine its operational Revision. A Cable Plug shall always return its highest supported Revision in its ***Specification Revision*** field.

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 6-2 for details of how various Revisions shall interoperate.

Table 6-2 Revision Interoperability

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
3	2	3	3	2
2	3	2	2	2
2	3	3	2	2
3	3	2	2	2
3	3	3	3	3

6.2.1.1.6 Port Data Role

The 1-bit ***Port Data Role*** field shall indicate the Port's present data role:

- 0b UFP
- 1b DFP

The ***Port Data Role*** field shall only be defined for SOP Packets. For all other SOP* Packets the ***Port Data Role*** field is ***Reserved*** and shall be set to zero.

Should a USB Type-C Port receive 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 Type-C 1.2]*** shall be performed.

For a USB Type-C Port the ***Port Data Role*** field shall be set to the default value at Attachment after a Hard Reset: 0b for a Port with Rd asserted and 1b for a Port with Rp asserted.

Microchip Ex. 1020, p. 99
Microchip v. Aptiv
IPR2024-00496

In the case that a Port is not USB Communications Capable, at Attachment a Source Port shall default to DFP and a Sink Port shall default to UFP.

6.2.1.1.7 Cable Plug

The 1-bit ***Cable Plug*** field shall indicate whether this Message originated from a Cable Plug:

- 0b Message originated from a DFP or UFP
- 1b Message originated from a Cable Plug

The ***Cable Plug*** field shall only apply to SOP' and SOP'' Packet types.

6.2.1.1.8 Message Type

The 5-bit ***Message Type*** field shall indicate the type of Message being sent. To fully decode the ***Message Type***, the ***Number of Data Objects*** field is first examined to determine whether the Message is a Control Message or a Data Message. Then the specific ***Message Type*** can be found in Table 6-5 (Control Message) or Table 6-6 (Data Message).

The ***Message Type*** field shall apply to all SOP* Packet types.

6.2.1.2 Extended Message Header

Every Extended Message (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 6-3 and defined in Table 6-3.

The Extended Message Header is used to support ECMs and EDMs containing Data Blocks of ***Data Size*** either sent in a single Message or as a series of Chunks. When the Data Block is sent as a series of 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 may be less than ***MaxExtendedMsgChunkLen*** bytes and shall be padded to the next 4-byte Data Object boundary.

Table 6-3 Extended Message Header

Bit(s)	Start of Packet	Field Name	Reference
15	SOP*	***Chunked***	Section 6.2.1.2.1
14..11	SOP*	***Chunk Number***	Section 6.2.1.2.2
10	SOP*	***Request Chunk***	Section 6.2.1.2.3
9	SOP*	***Reserved***	Section 1.4.2.9
8..0	SOP*	***Data Size***	Section 6.2.1.2.4

6.2.1.2.1 Chunked

The Port Partners shall use the Unchunked Extended Messages Supported fields in the ***Source_Capabilities*** Message and the ***Request*** Message to determine whether to send Messages of Data Size > ***MaxExtendedMsgLegacyLen*** bytes in a single Unchunked Extended Message (see Section 6.4.1.2.2.6 and Section 6.4.2.6).

When either Port Partner only supports Chunked Extended Messages:

1. The ***Chunked*** bit in every Extended Message shall be set to one
2. Every Extended Message of Data Size > ***MaxExtendedMsgLegacyLen*** shall be transmitted between the Port Partners in Chunks
3. Point 1 and Point 2 above shall apply until the Port Pair is Detached, there is a Hard Reset or the Source removes power (except during a Power Role Swap or Fast Role Swap when the initial Source removes power in order to for the new Source to apply power).

When both Port Partners support Unchunked Extended Messages:

1. The ***Chunked*** bit in every Extended Message shall be set to zero.
2. Every Extended Message shall be transmitted between the Port Partners Unchunked

Microchip Ex. 1020, p. 100
Microchip v. Aptiv
IPR2024-00496

3. Point 1 and Point 2 above shall apply until the Port Pair is Detached, there is a Hard Reset or the Source removes power (except during a Power Role Swap or Fast Role Swap when the initial Source removes power in order to for the new Source to apply power).

The VCONN Source shall only send Chunked Extended Messages to a Cable Plug. Cable Plugs shall always send Extended Messages of Data Size > ***MaxExtendedMsgLegacyLen*** Chunked and shall set the ***Chunked*** bit in every Extended Message to one.

When Extended Messages are supported Chunking shall be supported.

6.2.1.2.2 Chunk Number

The ***Chunk Number*** field shall only be valid in a Message if the ***Chunked*** flag is set to one. if the ***Chunked*** flag is set to zero the ***Chunk Number*** field shall also be set to zero.

The ***Chunk Number*** field is used differently depending on whether the Message is a request for Data or a requested Data Block being returned:

In a request for data the ***Chunk Number*** field indicates the number of the Chunk being requested. The requestor shall only set this field to one of the following values:

- Zero to request the first Chunk in the series
- The number of the last received Chunk in the series
- The number of the next Chunk in the series (the next Chunk after the last received Chunk)

In the requested Data Block the ***Chunk Number*** field indicates the number of the Chunk being returned. The Chunk number for each Chunk in the series shall start at zero and shall increment for each Chunk by one up to a maximum of 9 corresponding to 10 Chunks in total.

6.2.1.2.3 Request Chunk

The ***Request Chunk*** bit shall only be used for the Chunked transfer of an Extended Message when the ***Chunked*** bit is set to 1 (see Figure 6-7). For Unchunked Extended Message transfers Messages shall be sent and received without the request/response mechanism (see Figure 6-4).

The ***Request Chunk*** bit shall be set to one to indicate that this is a request for a Chunk of a Data Block and shall be set to zero to indicate that this is a Chunk response containing a Chunk. Except for Chunk zero, 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. When the ***Request Chunk*** bit is set to one the ***Data Size*** field shall be zero.

6.2.1.2.4 Data Size

The ***Data Size*** field shall indicate how many bytes of data in total are in Data Block being returned. The total number of data bytes in the Message shall not exceed ***MaxExtendedMsgLen***.

If the ***Data Size*** field is 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 zeros (0x00).

6.2.1.2.5 Extended Message Examples

The following examples illustrate the transmission of Extended Messages both Chunked (***Chunked*** bit is one) and Unchunked (***Chunked*** bit is zero). The examples use a ***Security_Request*** Message of ***Data Size*** 7 bytes which is responded to by a ***Security_Response*** Message of ***Data Size*** 30 bytes. The sizes of these Messages are arbitrary and are used to illustrate Message transmission; they are not intended to correspond to genuine security related Messages.

During negotiation of the Explicit Contract after connection, the Port Partners use the Unchunked Extended Messages Supported fields in the ***Source_Capabilities*** Message and the ***Request*** Message to determine the value of the ***Chunked***

Microchip Ex. 1020, p. 101
Microchip v. Aptiv
IPR2024-00496

bit (see Table 6-4). When both Port Partners support Unchunked Messages then the ***Chunked*** bit is one otherwise the ***Chunked*** bit is zero.

The ***Chunked*** bit is used to determine whether or not:

- The Chunk request/response mechanism is used
- Extended Messages are Chunked
- Padding is applied
- The ***Number of Data Objects*** field is used

The following examples illustrate the expected usage in each case.

Table 6-4 Use of Unchunked Message Supported bit

		Source: ***Source_Capabilities*** Message	
		Unchunked Message Supported bit = 0	Unchunked Message Supported bit = 1
Sink: ***Request*** Message	Unchunked Message Supported bit = 0	***Chunked*** bit = 1	***Chunked*** bit = 1
	Unchunked Message Supported bit = 1	***Chunked*** bit = 1	***Chunked*** bit = 0

6.2.1.2.5.1 Security_Request/Security_Response Unchunked Example

Figure 6-4 illustrates a typical sequence for a ***Security_Request*** Message responded to by a ***Security_Response*** Message using Unchunked Extended Messages (***Chunked*** bit is zero) between a USB Host and a power brick. The entire Data Block is returned in one Message. The Chunk request/response mechanism is not used.

Figure 6-4 Example Security_Request sequence Unchunked (Chunked bit = 0)

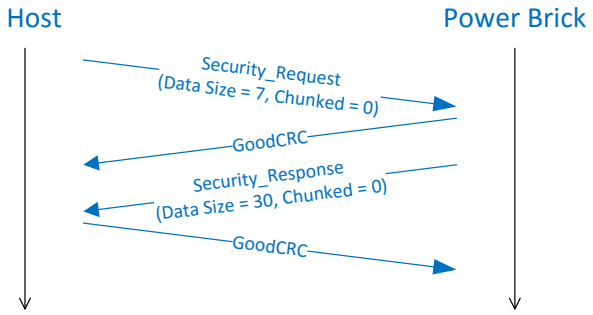


Figure 6-5 details the ***Security_Request*** Message shown in Figure 6-4. The figure shows the byte ordering on the bus as well as the fact that there is no padding in this case. The ***Number of Data Objects*** field has a value of 0 since it is ***Reserved*** when the ***Chunked*** bit is zero. The ***Data Size field*** indicates the length of the Extended Message when the ***Chunked*** bit is set to 0, which in this case is 7 bytes.

Microchip Ex. 1020, p. 102
Microchip v. Aptiv
IPR2024-00496

Figure 6-5 Example byte transmission for Security_Request Message of Data Size 7 (Chunked bit is set to 0)

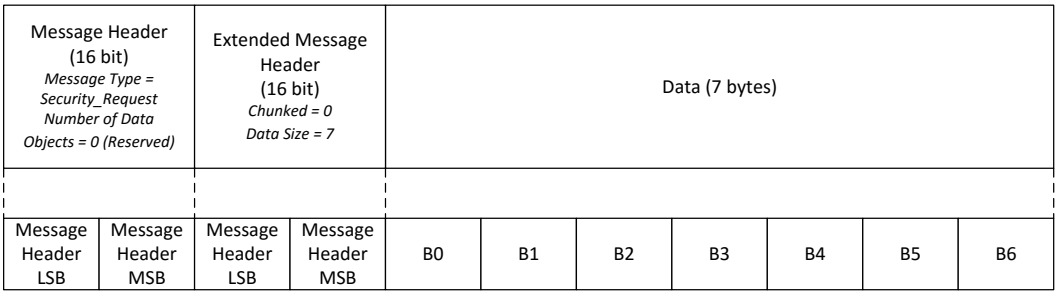
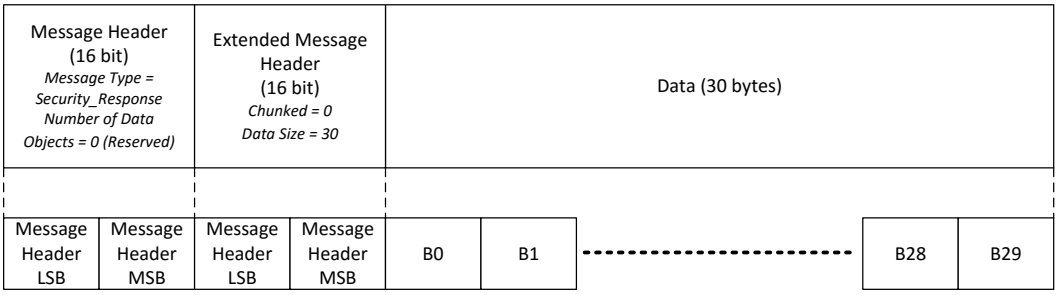


Figure 6-6 details the ***Security_Response*** Message shown in Figure 6-4. The figure shows the byte ordering on the bus as well as the fact that there is no padding in this case. The ***Number of Data Objects*** field has a value of 0 since it is ***Reserved*** when the ***Chunked*** bit is zero. The ***Data Size field*** indicates the length of the Extended Message when the ***Chunked*** bit is set to 0, which in this case is 30 bytes.

Figure 6-6 Example byte transmission for Security_Response Message of Data Size 7 (Chunked bit is set to 0)



6.2.1.2.5.1 Security_Request/Security_Response Chunked Example

Figure 6-7 illustrates a typical sequence for a ***Security_Request*** Message responded to by a ***Security_Response*** Message using Chunked Extended Messages (***Chunked*** bit is one) between a USB Host and a power brick. Note that ***Chunk Number*** zero in every Extended Message is sent without the need for a Chunk Request, but ***Chunk Number*** one and following need to be requested with a Chunk request.

Microchip Ex. 1020, p. 103
Microchip v. Aptiv
IPR2024-00496

Figure 6-7 Example Security_Request sequence Chunked (Chunked bit = 1)

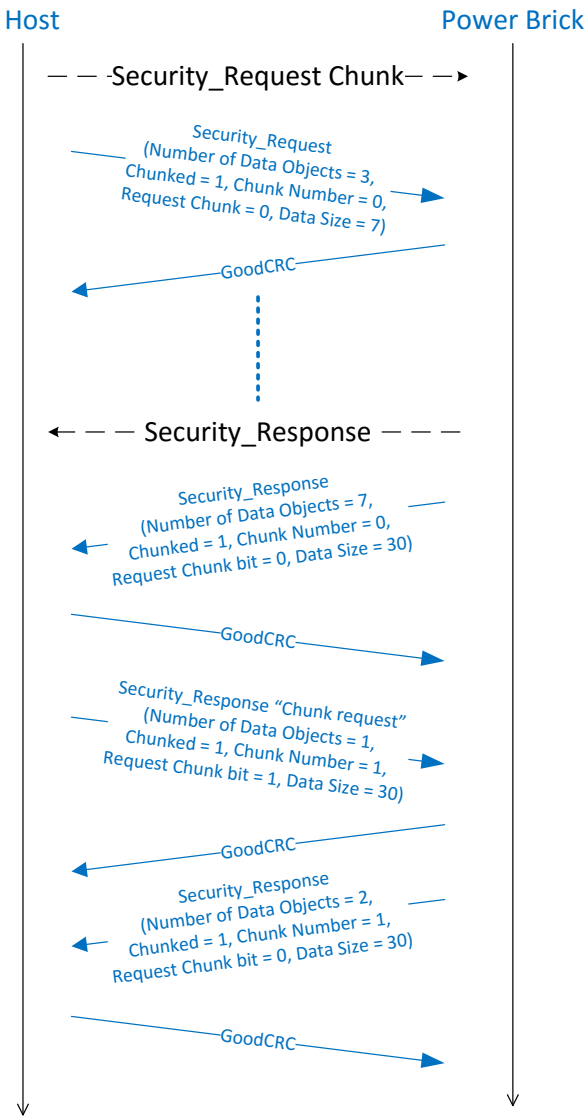


Figure 6-8 shows the ***Security_Request*** Message shown in Figure 6-7 in more detail including the byte ordering on the bus and padding. Three bytes of padding have been added to the Message so that the total number of bytes is a multiple of 32-bits, corresponding to 3 Data Objects. The ***Number of Data Objects*** field is set to 3 to indicate the length of this Chunk. The ***Chunk Number*** is set to zero and the ***Data Size*** field is set to 7 to indicate the length of the whole Extended Message.

Microchip Ex. 1020, p. 104
Microchip v. Aptiv
IPR2024-00496

Figure 6-8 Example Security_Request Message of Data Size 7 (Chunked bit set to 1)

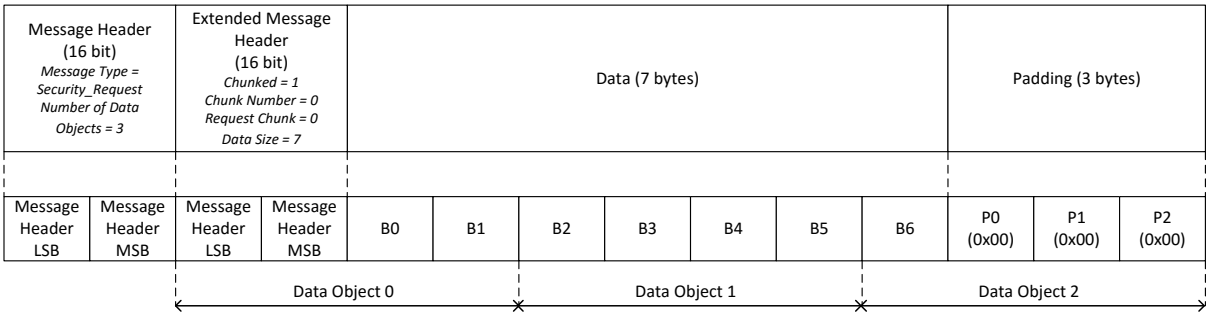


Figure 6-10 shows ***Chunk Number*** zero of the ***Security_Response*** Message shown in Figure 6-7 in more detail including the byte ordering on the bus and padding. No padding is need for this Chunk since the full 26-byte payload plus 2-byte Extended Message Header is a multiple of 32-bits, corresponding to 7 Data Objects. The ***Number of Data Objects*** field is set to 7 to indicate the length of this Chunk and the ***Data Size*** field is set to 30 to indicate the length of the whole Extended Message.

Figure 6-9 Example Chunk 0 of Security_Response Message of Data Size 30 (Chunked bit set to 1)

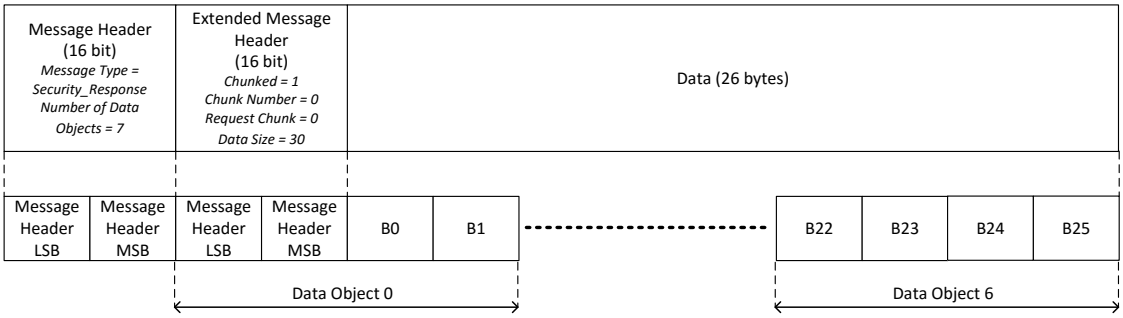


Figure 6-10 shows an example of the Message format, byte ordering and padding for the***Security_Response*** Message Chunk request for ***Chunk Number*** one shown in Figure 6-7. In the Chunk request the ***Number of Data Objects*** field in the Message is set to 1 to indicate that the payload is 32 bits equivalent to 1 data object. Since the ***Chunked*** bit is set to 1 the Chunk request/Chunk response mechanism is used. The Message is a Chunk request so the ***Request Chunk*** bit is set to one, and in this case Chunk one is being requested so ***Chunk Number*** is set to one. ***Data Size*** is set to 30 indicating the length of the Data Block being transferred. Two bytes of padding are added to ensure that the payload is a multiple of 32 bits.

Figure 6-10 Example byte transmission for a Security_Request Message Chunk request (Chunked bit is set to 1)

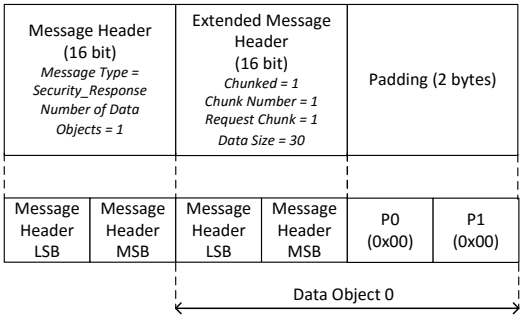
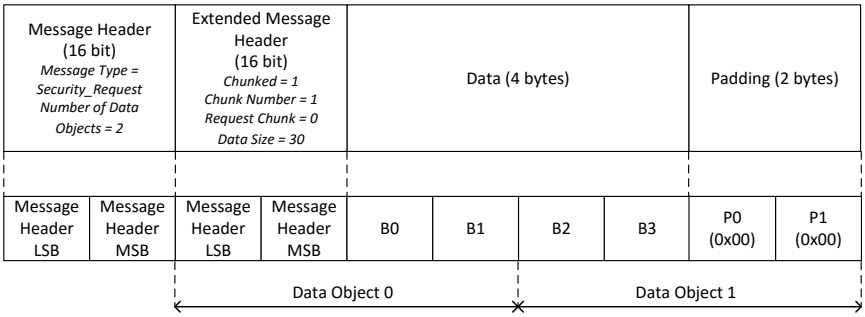


Figure 6-11 shows ***Chunk Number*** one of the ***Security_Response*** Message shown in Figure 6-7 in more detail including the byte ordering on the bus and padding. Two bytes of padding are added to ensure that the payload is a multiple of 32 bits, corresponding to 2 Data Objects. The ***Number of Data Objects*** field is set to 2 to indicate the length of this Chunk and the ***Data Size*** field is set to 30 to indicate the length of the whole Extended Message.

Microchip Ex. 1020, p. 105
Microchip v. Aptiv
IPR2024-00496

Figure 6-11 Example Chunk 1 of Security_Response Message of Data Size 30 (Chunked bit set to 1)



6.3 Control Message

A Message is defined as a Control Message when the ***Number of Data Objects*** field in the Message Header is set to 0. The Control Message consists only of a Message Header and a CRC. The Protocol Layer originates the Control Messages (i.e. ***Accept*** Message, ***Reject*** Message etc.).

The Control Message types are specified in the Message Header's ***Message Type*** field (bits 4...0) and are summarized in Table 6-5. The Sent by column indicates entities which may send the given Message (Source, Sink or Cable Plug); entities not listed shall not issue the corresponding Message. The "Valid Start of Packet" column indicates the Messages which shall only be issued in SOP Packets and the Messages which may be issued in SOP* Packets.

Table 6-5 Control Message Types

Bits 4..0	Message Type	Sent by	Description	Valid Start of Packet
0 0000	***Reserved***	N/A	All values not explicitly defined are ***Reserved*** and shall not be used.	
0 0001	***GoodCRC***	Source, Sink or Cable Plug	See Section 6.3.1.	SOP*
0 0010	***GotoMin***	Source only	See Section 6.3.2.	SOP only
0 0011	***Accept***	Source, Sink or Cable Plug	See Section 6.3.3.	SOP*
0 0100	***Reject***	Source or Sink	See Section 6.3.4.	SOP only
0 0101	***Ping***	Source only	See Section 6.3.5.	SOP only
0 0110	***PS_RDY***	Source or Sink	See Section 6.3.6.	SOP only
0 0111	***Get_Source_Cap***	Sink or DRP	See Section 6.3.7.	SOP only
0 1000	***Get_Sink_Cap***	Source or DRP	See Section 6.3.8.	SOP only
0 1001	***DR_Swap***	Source or Sink	See Section 6.3.9	SOP only
0 1010	***PR_Swap***	Source or Sink	See Section 6.3.10	SOP only
0 1011	***VCONN_Swap***	Source or Sink	See Section 6.3.11	SOP only
0 1100	***Wait***	Source or Sink	See Section 6.3.12	SOP only
0 1101	***Soft_Reset***	Source or Sink	See Section 6.3.13	SOP*
0 1110-0 1111	***Reserved***	N/A	All values not explicitly defined are ***Reserved*** and shall not be used.	
1 0000	***Not_Supported***	Source, Sink or Cable Plug	See Section 6.3.14	SOP*
1 0001	***Get_Source_Cap_Extended***	Sink or DRP	See Section 6.3.15	SOP only
1 0010	***Get_Status***	Source or Sink	See Section 6.3.16	SOP only

Microchip Ex. 1020, p. 106
Microchip v. Aptiv
IPR2024-00496

Bits 4..0	Message Type	Sent by	Description	Valid Start of Packet
1 0011	***FR_Swap***	Sink[1]	See Section 6.3.17	SOP only
1 0100-1 1111	***Reserved***	N/A	All values not explicitly defined are ***Reserved*** and shall not be used.	
Note 1: In this case the Port is providing ***vSafe5V*** however it will have Rd asserted rather than Rp and sets the ***Port Power Role*** field to Sink, until the Fast Role Swap AMS has completed.				

6.3.1 GoodCRC Message

The ***GoodCRC*** Message shall be sent by the receiver to acknowledge that the previous Message was correctly received (i.e. had a good CRC). The ***GoodCRC*** Message shall return the Message's ***MessageID*** so the transmitter can determine that the correct Message is being acknowledged. The first bit of the ***GoodCRC*** Message shall be returned within ***tTransmit*** after receipt of the last bit of the previous Message.

BIST does not send the ***GoodCRC*** Message while in a Continuous BIST Mode (see Section 6.4.3).

6.3.2 GotoMin Message

The ***GotoMin*** Message applies only to those Sinks that have requested power with the GiveBack capable flag set in the Sink Request Data Object.

It is a directive to the Sink Port to reduce its operating power level to the amount specified in the Minimum Operating Current field of its latest Sink Request Data Object.

The GotoMin process is designed to allow the Source to temporarily reallocate power to meet a short term requirement. For example, a Source may reduce a Sink's power consumption for 10-20 seconds to allow another Sink (e.g. an HDD to spin up).

The Source sends this Message as a means to harvest power in order to meet a request for power that it cannot otherwise meet. The Device Policy Manager determines which Port or ports will receive the Message.

The Sink shall respond to a ***GotoMin*** Message by reducing its power consumption to less than or equal to the pre-negotiated value (Minimum Operating Current) within ***tSnkNewPower*** time.

The Source sends a ***GotoMin*** Message as a shortcut in the power negotiation process since the Source and Sink have already made a Contract with respect to the power to be returned. In essence, the Source does not have to advertise its Capabilities and the Sink does not have to make a Request based on them. The Source simply sends the ***GotoMin*** Message in place of the ***Accept*** Message normally sent during the power negotiation process (see step 19 in Figure 8-5). The power negotiation process then completes from this point in the normal manner with the Source sending a ***PS_RDY*** Message once the power supply transition is complete. The steps of the GotoMin process are fully described in Figure 8-6.

The Source shall return power to the Sink(s) it has 'borrowed' from using the GotoMin mechanism before it can allocate any 'new' power to other devices.

6.3.3 Accept Message

The ***Accept*** Message is a valid response in the following cases:

- It shall be sent by the Source to signal the Sink that the Source is willing to meet the ***Request*** Message.
- It 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 sequence.
- It 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 sequence.
- It 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 sequence.
- It shall be sent by the recipient of the ***FR_Swap*** Message to indicate that it has begun the Fast Role Swap sequence.

Microchip Ex. 1020, p. 107
Microchip v. Aptiv
IPR2024-00496

- It shall be sent by the recipient of the ***Soft_Reset*** Message to indicate that it has completed its Soft Reset.

The ***Accept*** Message shall be sent within ***tReceiverResponse*** of the receipt of the last bit of the Message (see Section 6.6.2).

6.3.4 Reject Message

The ***Reject*** Message is a valid response in the following cases:

- It shall be sent to signal the Sink 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.
- It shall be sent by the recipient of a ***PR_Swap*** Message to indicate it is unable to do a Power Role Swap.
- It shall be sent by the recipient of a ***DR_Swap*** Message to indicate it is unable to do a Data Role Swap.
- It shall be sent by the recipient of a ***VCONN_Swap*** Message, that is not presently the VCONN Source, to indicate it is unable to do a VCONN Swap.

The ***Reject*** Message shall be sent within ***tReceiverResponse*** of the receipt of the last bit of Message (see Section 6.6.2).

Note: the ***Reject*** Message is not a valid response when a Message is not supported. In this case the ***Not_Supported*** Message is returned (see Section 6.3.14).

6.3.5 Ping Message

The ***Ping*** Message was previously used on USB Type-A and USB Type-B connectors to determine the continued presence of the Sink when no other messaging was taking place. USB Type-C connectors have a mechanism to determine Sink presence so when the Port Partners are both connected using USB Type-C connectors the ***Ping*** Message is not necessary but may be sent by a Source if desired. A Sink using a USB Type-C connector shall not expect to receive ***Ping*** Messages but shall not treat ***Ping*** Messages as an error if they are received.

6.3.6 PS_RDY Message

The ***PS_RDY*** Message shall be sent by the Source (or by both the new Sink and new Source during the Power Role Swap sequence or Fast Role Swap sequence) to indicate its power supply has reached the desired operating condition (see Section 8.3.2.2).

6.3.7 Get_Source_Cap Message

The ***Get_Source_Cap*** (Get Source Capabilities) Message may be sent by a Port to request the Source Capabilities and Dual-Role Power capability of its Port Partner (e.g. Dual-Role Power capable). The Port shall respond by returning a ***Source_Capabilities*** Message (see Section 6.4.1.1.1).

6.3.8 Get_Sink_Cap Message

The ***Get_Sink_Cap*** (Get Sink Capabilities) Message may be sent by a Port to request the Sink Capabilities and Dual-Role Power capability of its Port Partner (e.g. Dual-Role Power capable). The Port shall respond by returning a ***Sink_Capabilities*** Message (see Section 6.4.1.1.2).

6.3.9 DR_Swap Message

The ***DR_Swap*** Message is used to exchange DFP and UFP operation between Port Partners while maintaining the direction of power flow over V_{BUS}. The DR_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 Type-C 1.2] DRPs 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

Microchip Ex. 1020, p. 108
Microchip v. Aptiv
IPR2024-00496

Port Partners when a ***DR_Swap*** Message is a received then a Hard Reset shall be performed (see Section 6.4.4.3.4). 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 DR Swap request.

The Source of V_{BUS} and V_{CONN} 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, ***Reject*** Message or ***Wait*** Message.

- If an ***Accept*** Message is sent, the Source and Sink shall exchange operational roles.
- If a ***Reject*** Message is sent, the requester is informed that the recipient is unable, or unwilling, to do a Data Role Swap and no action shall be taken.
- If a ***Wait*** Message is sent, the requester is informed that a Data Role Swap might be possible in the future but that no immediate action shall be taken.

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.

After a successful Data Role Swap the DFP/Host shall become the UFP/Device and vice-versa; the new DFP shall have its ***Port Data Role*** bit set to DFP, and the new UFP shall have its ***Port Data Role*** bit set to UFP. Where USB Communication is supported by both Port Partners a USB data connection should be established according to the new data roles.

If the Data Role Swap, after having been accepted by the Port Partner, is subsequently not successful, in order to attempt a re-establishment of the connection on the CC Wire, USB Type-C Error Recovery actions, such as disconnect, as defined in ***[USB Type-C 1.2]*** will be necessary.

See Section 8.3.2.6, Section 8.3.3.15.1 and Section 8.3.3.15.2 for further details.

6.3.10 PR_Swap Message

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, ***Reject*** Message or ***Wait*** Message.

- If an ***Accept*** Message is sent, the Source and Sink shall do a Power Role Swap.
- If a ***Reject*** Message is sent, the requester is informed that the recipient is unable, or unwilling, to do a Power Role Swap and no action shall be taken.
- If a ***Wait*** Message is sent, the requester is informed that a Power Role Swap might be possible in the future but that no immediate action shall be taken.

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 an Explicit Contract. At this point the Source shall also reset its ***CapsCounter***.

Since a UFP Source can attempt to send a ***Discover Identity*** Command using SOP' to a Cable Plug prior to the establishment of an Explicit Contract, a DFP Sink shall disable the receiving of SOP' Messages until an Explicit Contract has been established. This ensures that only the Cable Plug responds with a ***GoodCRC*** Message to the ***Discover Identity*** Command.

The Source shall have Rp asserted on the CC wire and the Sink shall have Rd asserted on the CC wire as defined in ***[USB Type-C 1.2]***. 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) roles and V_{CONN} Source shall remain unchanged during the Power Role Swap process.

Note: during the Power Role Swap process the initial Sink does not disconnect even though V_{BUS} drops below ***vSafe5V***.

For more information regarding the Power Role Swap, refer to Section 7.3.9 and Section 7.3.10 in the Power Supply chapter, Section 8.3.2.6, Section 8.3.3.15.3 and Section 8.3.3.15.4 in the Device Policy chapter and Section 9.1.2 for V_{BUS} mapping to USB states.

Microchip Ex. 1020, p. 109
Microchip v. Aptiv
IPR2024-00496

6.3.11 VCONN_Swap Message

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 or ***Wait*** Message.

- If an ***Accept*** Message is sent, the Port Partners shall perform a VCONN Swap. 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.
- If a ***Reject*** Message is sent, the requester is informed that the recipient is unable, or unwilling, to do a VCONN Swap and no action shall be taken. A ***Reject*** Message shall only be sent by the Port that is not presently the Vconn Source in response to a ***VCONN_Swap*** Message. The Port that is presently the Vconn Source shall not send a ***Reject*** Message in response to ***VCONN_Swap*** Message.
- If a ***Wait*** Message is sent, the requester is informed that a VCONN Swap might be possible in the future but that no immediate action shall be taken. A ***Wait*** Message shall only be sent by the Port that is not presently the Vconn Source in response to a ***VCONN_Swap*** Message. The Port that is presently the Vconn Source shall not send a ***Wait*** Message in response to ***VCONN_Swap*** Message.

The DFP (Host), UFP (Device) roles and Source of V_{BUS} shall remain unchanged as well as the Rp/Rd resistors on the CC wire during the VCONN Swap process.

Note: VCONN shall be continually sourced during the VCONN Swap process in order 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.

Note: even when it is presently the VCONN Source, the Sink is not permitted to initiate an AMS with a Cable Plug unless Rp is set to ***SinkTxOk*** (see Section 6.9).

6.3.12 Wait Message

The ***Wait*** Message is a valid response to a ***Request***, a ***PR_Swap***, ***DR_Swap*** or ***VCONN_Swap*** Message.

- It shall be sent to signal the Sink that the Source is unable to meet the request at this time.
- It shall be sent by the recipient of a ***PR_Swap*** Message to indicate it is unable to do a Power Role Swap at this time.
- It shall be sent by the recipient of a ***DR_Swap*** Message to indicate it is unable to do a Data Role Swap at this time.

The ***Wait*** Message shall be sent within ***tReceiverResponse*** of the receipt of the last bit of the Message (see Section 6.6.2).

6.3.12.1 Wait in response to a Request Message

The ***Wait*** Message is used by the Source when a Sink that has reserved power, requests it. The ***Wait*** Message allows the Source time to recover the power it requires to meet the request through the GotoMin process. A Source shall 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*** to ensure that there is ***tSinkRequest*** between requests.

6.3.12.2 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 Capabilities from the originator of the ***PR_Swap*** Message. Once it has completed

Microchip Ex. 1020, p. 110
Microchip v. Aptiv
IPR2024-00496

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 ***[USBTypeCBridge 1.0]***).

6.3.12.3 Wait in response to a DR_Swap Message

The ***Wait*** Message is used when responding to a ***DR_Swap*** Message to indicate that a Date 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.

6.3.13 Soft Reset Message

A ***Soft_Reset*** Message may be initiated by either the Source or Sink to its Port Partner requesting a Soft Reset. The ***Soft_Reset*** Message shall cause a Soft Reset of the connected Port Pair (see Section 6.8.1). 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 is used to recover from Protocol Layer errors; putting the Message counters to a known state in order to regain Message synchronization. The ***Soft_Reset*** Message has no effect on the Source or Sink; that is the previously negotiated direction. Voltage and current remain unchanged. Modal Operation is unaffected by Soft Reset. However 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 8.3.3.4).

A ***Soft_Reset*** Message may be sent by either the Source or Sink when there is a Message synchronization error. If the error is not corrected by the Soft Reset, ***Hard Reset*** Signaling shall be issued (see Section 6.8).

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'/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 in order 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.

6.3.14 Not_Supported Message

The ***Not_Supported*** Message shall be sent by a Port in response to any Message it does not support. Returning a ***Not_Supported*** Message is assumed in this specification and has not been called out explicitly except in Section 6.12 which defines cases where the ***Not_Supported*** Message is returned.

6.3.15 Get_Source_Cap_Extended Message

The ***Get_Source_Cap_Extended*** (Get Source Capabilities Extended) Message is sent by a Port to request additional information about a Port's Source Capabilities. The Port should respond by returning a ***Source_Capabilities_Extended*** Message (see Section 6.5.1).

6.3.16 Get_Status Message

The ***Get_Status*** Message is sent by a Port to request the Port Partner's present status. The Source or Sink should respond by returning a ***Status*** Message (see Section 6.5.2). A Port that receives an ***Alert*** Message (see Section 6.4.6) indicates that the Source or Sink's Status has changed and should be re-read using a ***Get_Status*** Message.

Microchip Ex. 1020, p. 111
Microchip v. Aptiv
IPR2024-00496

6.3.17 FR_Swap Message

The ***FR_Swap*** Message should be sent by the new Source ***tFRSwapInit*** after it has detected a Fast Role Swap signal (see Section 5.8.6.3 and Section 6.6.16). The Fast Role Swap AMS is necessary to apply Rp to the new Source and Rd to the new Sink and to re-synchronize the state machines.

The recipient of the ***FR_Swap*** Message shall respond by sending an ***Accept*** Message.

After a successful Fast 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 an Explicit Contract. At this point the Source shall also reset its ***CapsCounter***.

Since a UFP Source can attempt to send a ***Discover Identity*** Command using SOP' to a Cable Plug prior to the establishment of an Explicit Contract, a DFP Sink shall disable the receiving of SOP' Messages until an Explicit Contract has been established. This ensures that only the Cable Plug responds with a ***GoodCRC*** Message to the ***Discover Identity*** Command.

Prior to the Fast Role Swap AMS the new Source shall have Rd asserted on the CC wire and the new Sink shall have Rp asserted on the CC wire. Note that this is an incorrect assignment of Rp/Rd (since Rp follows the Source and Rd follows the Sink as defined in ***[USB Type-C 1.2]***) that is corrected by the Fast Role Swap AMS.

During the Fast Role Swap AMS the new Source shall change its CC Wire resistor from Rd to Rp and the new Sink shall change its CC Wire resistor from Rp to Rd. The DFP (Host), UFP (Device) roles and VCONN Source shall remain unchanged during the Fast Role Swap process.

The initial Source should avoid being the VCONN source (by using the VCONN Swap process) whenever not actively communicating with the cable, since it is difficult for the initial Source to maintain VCONN power during the Fast Role Swap process.

Note: a Fast Role Swap is a "best effort" solution to a situation where a PDUSB Device has lost its external power. This process can occur at any time, even during a Non-interruptible AMS in which case error handling such as Hard Reset or ***[USB Type-C 1.2]*** Error Recovery will be triggered.

Note: during the Fast Role Swap process the initial Sink does not disconnect even though V_{BUS} drops below ***vSafe5V***.

For more information regarding the Fast Role Swap process, refer to Section 7.1.14 and Section 7.2.9.2 in the Power Supply chapter, Section 8.3.3.15.5 and Section 8.3.3.15.6 in the Device Policy chapter and Section 9.1.2 for V_{BUS} mapping to USB states.

6.4 Data Message

A Data Message shall consist of a Message Header and be followed by one or more Data Objects. Data Messages are easily identifiable because the ***Number of Data Objects*** field in the Message Header is a non-zero value.

There are several types of Data Objects:

- BIST Data Object (BDO) used for PHY Layer compliance testing
- 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 a Contract
- Vendor Defined Data Object (VDO) used to convey vendor specific information
- Battery Status Data Object (BSDO) used to convey Battery status information
- Source Alert Data Object (SADO) used to indicates events occurring on the Source

The type of Data Object being used in a Data Message is defined by the Message Header's ***Message Type*** field and is summarized in Table 6-6. The Sent by column indicates entities which may send the given Message (Source, Sink or Cable Plug); entities not listed shall not issue the corresponding Message. The Valid Start of Packet column indicates the Messages which shall only be issued in SOP Packets and the Messages which may be issued in SOP* Packets.

Microchip Ex. 1020, p. 112
Microchip v. Aptiv
IPR2024-00496

Table 6-6 Data Message Types

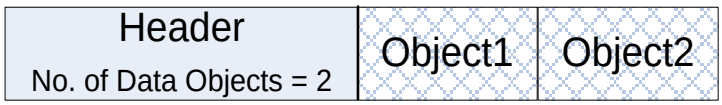
Bits 4..0	Type	Sent by	Description	Valid Start of Packet
0 0000	***Reserved***		All values not explicitly defined are ***Reserved*** and shall not be used.	
0 0001	***Source_Capabilities***	Source or Dual-Role Power	See Section 6.4.1.2	SOP only
0 0010	***Request***	Sink only	See Section 6.4.2	SOP only
0 0011	***BIST***	Tester, Source or Sink	See Section 6.4.3	SOP*
0 0100	***Sink_Capabilities***	Source or Dual-Role Power	See Section 6.4.1.3	SOP only
0 0101	***Battery_Status***	Source or Sink	See Section 6.4.5	SOP only
0 0110	***Alert***	Source or Sink	See Section 6.4.6	SOP only
0 0111 -0 1110	***Reserved***		All values not explicitly defined are ***Reserved*** and shall not be used.	
0 1111	***Vendor_Defined***	Source, Sink or Cable Plug	See Section 6.4.4	SOP*
1 0000-1 1111	***Reserved***		All values not explicitly defined are ***Reserved*** and shall not be used.	

6.4.1 Capabilities Message

A Capabilities Message (***Source_Capabilities*** Message or ***Sink_Capabilities*** Message) shall have at least one Power Data Object for ***vSafe5V***. The Capabilities Message shall also contain the sending Port's information followed by up to 6 additional Power Data Objects. Power Data Objects in a Capabilities Message shall be sent in the following order:

1. The ***vSafe5V*** Fixed Supply Object shall always be the first object.
2. The remaining Fixed Supply Objects, if present, shall be sent in voltage order; lowest to highest.
3. The Battery Supply Objects, if present shall be sent in Minimum Voltage order; lowest to highest.
4. The Variable Supply (non-Battery) Objects, if present, shall be sent in Minimum Voltage order; lowest to highest.

Figure 6-12 Example Capabilities Message with 2 Power Data Objects



In Figure 6-12, the ***Number of Data Objects*** field is 2: ***vSafe5V*** plus one other voltage.

Power Data Objects (PDO) are identified by the Message Header's Type field. They are used to form ***Source_Capabilities*** Messages and ***Sink_Capabilities*** Messages.

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 the most commonly used to expose well-regulated fixed voltage power supplies.
- Variable power supply is used to expose very poorly regulated power supplies.
- Battery is used to expose batteries than can be directly connected to V_{BUS}.

Power Data Objects are also used to expose additional capabilities that may be utilized; such as in the case of a Power Role Swap.

Microchip Ex. 1020, p. 113
Microchip v. Aptiv
IPR2024-00496

A list of one or more Power Data Objects shall be sent by the Source in order 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, in order to negotiate a mutually agreeable 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 Type-C 1.2]***). 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 4.4.

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 a number of 32-bit Power Data Objects (see Table 6-7).

Table 6-7 Power Data Object

Bit(s)	Description	
B31..30	**Value**	**Parameter**
	00b	Fixed supply (Vmin = Vmax)
	01b	Battery
	10b	Variable Supply (non-Battery)
	11b	***Reserved***
B29..0	Specific Power Capabilities are described by the PDOs in the following sections.	

6.4.1.1 Use of the Capabilities Message

6.4.1.1.1 Use by Sources

Sources send a ***Source_Capabilities*** Message (see Section 6.4.1) either as part of advertising Port capabilities, or in response to a ***Get_Source_Cap*** Message.

Following a Hard Reset, a power-on event or plug insertion event, a Source Port shall send a ***Source_Capabilities*** Message after every ***SourceCapabilityTimer*** timeout as an advertisement that shall be interpreted by the Sink Port on Attachment. The Source shall continue sending a minimum of ***nCapsCount Source_Capabilities*** Messages until a ***GoodCRC*** Message is received.

Additionally, a ***Source_Capabilities*** Message shall only be sent in the following cases:

- By the Source Port from the ***PE_SRC_Ready*** state upon a change in its ability to supply power
- By a Source Port or Dual-Role Power Port in response to a ***Get_Source_Cap*** Message

6.4.1.1.2 Use by Sinks

Sinks send a ***Sink_Capabilities*** Message (see Section 6.4.1.3) in response to a ***Get_Sink_Cap*** Message.

A USB Power Delivery capable Sink, upon detecting ***vSafe5V*** on V_{BUS} and after a ***SinkWaitCapTimer*** timeout without seeing a ***Source_Capabilities*** Message, shall send a Hard Reset. If the Attached Source is USB Power Delivery capable, it responds by sending ***Source_Capabilities*** Messages thus allowing power negotiations to begin.

6.4.1.1.3 Use by Dual-Role Power devices

Dual-Role Power devices send a ***Source_Capabilities*** Message (see Section 6.4.1) as part of advertising Port capabilities when operating in Source role. Dual-Role Power devices send a ***Source_Capabilities*** Message (see Section 6.4.1) in response to a ***Get_Source_Cap*** Message regardless of their present operating role. Similarly Dual-Role Power devices send a ***Sink_Capabilities*** Message (see Section 6.4.1.3) in response to a ***Get_Sink_Cap*** Message regardless of their present operating role.

Microchip Ex. 1020, p. 114
Microchip v. Aptiv
IPR2024-00496

6.4.1.2 Source_Capabilities Message

A Source Port shall report its capabilities in a series of 32-bit Power Data Objects (see Table 6-7) as part of a ***Source_Capabilities*** Message (see Figure 6-12). Power Data Objects are used to convey a Source Port's capabilities to provide power including Dual-Role Power ports presently operating as a Sink.

Each Power Data Object shall describe a specific Source capability such as a Battery (e.g. 2.8-4.1V) or a fixed power supply (e.g. 12V) at a maximum allowable current. The ***Number of Data Objects*** field in the Message Header shall define the number of Power Data Objects that follow the Message Header in a Data Message. All Sources shall minimally offer one Power Data Object that reports ***vSafe5V***. A Source shall not offer multiple Power Data Objects of the same type (fixed, variable, Battery) and the same voltage but shall instead offer one Power Data Object with the highest available current for that Source capability and voltage. Sinks with Accessory Support do not source V_{BUS} (see ***[USB Type-C 1.2]***) however when sourcing V_{CONN} they shall advertise ***vSafe5V*** with the Maximum Current set to 0mA in the first Power Data Object.

A Sink shall evaluate every ***Source_Capabilities*** Message it receives and shall respond with a ***Request*** Message. If its power consumption exceeds the Source's capabilities it shall re-negotiate so as not to exceed the Source's most recently advertised capabilities.

6.4.1.2.1 Management of the Power Reserve

A Power Reserve may be allocated to a Sink when it makes a request from Source Capabilities which includes a Maximum Operating Current/Power. The size of the Power Reserve for a particular Sink is calculated as the difference between its Maximum Operating Current/Power field and its Operating Current/Power field. For a Hub with multiple ports this same Power Reserve may be shared between several Sinks. The Power Reserve may also be temporarily used by a Sink which has indicated it can give back power by setting the GiveBack flag.

Where a Power Reserve has been allocated to a Sink the Source shall indicate the Power Reserve as part of every ***Source_Capabilities*** Message it sends. When the same Power Reserve is shared between several Sinks the Source shall indicate the Power Reserve as part of every ***Source_Capabilities*** Message it sends to every Sink. Every time a Source sends capabilities including the Power Reserve capability and then accepts a request from a Sink including the Power Reserve indicated by its Maximum operating Current/Power it is confirming that the Power Reserve is part of the Explicit Contract with the Sink.

When the Reserve is being temporarily used by a giveback capable Sink the Source shall indicate the Power Reserve as available in every ***Source_Capabilities*** Message it sends. However in this situation, when the Power Reserve is requested by a Sink, the Source shall return a ***Wait*** Message while it retrieves this power using a ***GotoMin*** Message. Once the additional power has been retrieved the Source shall send a new ***Source_Capabilities*** Message in order to trigger a new request from the Sink requesting the Power Reserve.

The Power Reserve may be de-allocated by the Source at any time, but the de-allocation shall be indicated to the Sink or Sinks using the Power Reserve by sending a new ***Source_Capabilities*** Message.

6.4.1.2.2 Fixed Supply Power Data Object

Table 6-8 describes the Fixed Supply (00b) PDO. See Section 7.1.3 for the electrical requirements of the power supply.

Since all USB Providers support ***vSafe5V***, the required ***vSafe5V*** Fixed Supply Power Data Object is also used to convey additional information that is returned in bits 29 through 25. All other Fixed Supply Power Data Objects shall set bits 29...22 to zero.

For a Source offering no capabilities, the Voltage (B19..10) shall be set to 5V and the Maximum Current shall be set to 0mA. This is used in cases such as a Dual-Role Power device which offers no capabilities in its default role or when external power is required in order to offer power.

When a Source wants a Sink, consuming power from V_{BUS}, to go to its lowest power state, the Voltage (B19..10) shall be set to 5V and the Maximum Current shall be set to 0mA. This is used in cases where the Source wants the Sink to draw ***pSnkSusp***.

Microchip Ex. 1020, p. 115
Microchip v. Aptiv
IPR2024-00496

Table 6-8 Fixed Supply PDO - Source

Bit(s)	Description
B31..30	Fixed supply
B29	Dual-Role Power
B28	USB Suspend Supported
B27	Externally Powered
B26	USB Communications Capable
B25	Dual-Role Data
B24	Unchunked Extended Messages Supported
B23..22	***Reserved*** – shall be set to zero.
B21..20	Peak Current
B19..10	Voltage in 50mV units
B9..0	Maximum Current in 10mA units

6.4.1.2.2.1 Dual-Role Power

The Dual-Role Power bit shall be set when the Port is Dual-Role Power capable i.e. supports the ***PR_Swap*** Message. This is a static capability which shall remain fixed for a given device.

6.4.1.2.2.2 USB Suspend Supported

Prior to a Contract or when the USB Communications Capable bit is set to zero, this flag is undefined and Sinks shall follow the rules for suspend as defined in ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]***. After a Contract has been negotiated:

- If the USB Suspend Supported flag is set, then the Sink shall follow the ***[USB 2.0]*** or ***[USB 3.1]*** rules for suspend and resume. A PDUSB Peripheral may draw up to ***pSnkSusp*** during suspend; a PDUSB Hub may draw up to ***pHubSusp*** during suspend (see Section 7.2.3).
- If the USB Suspend Supported flag is cleared, then the Sink shall not apply the ***[USB 2.0]*** or ***[USB 3.1]*** rules for suspend and may continue to draw the negotiated power. Note that 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 6.4.2.5).

6.4.1.2.2.3 Externally Powered

The Externally Powered bit shall be set when an external source of power is available.

The Externally Powered bit shall be set when either:

- There is an AC supply, e.g. a Wall Wart, directly connected to the Source.
- Or, in the case of a PDUSB Hub:
    - The Hub is receiving power from a PD Source with its Externally Powered bit set.
    - The Hub is receiving power from multiple PD Sources all with their Externally Powered bits set.

When the Externally Powered bit is set the Source Inputs field in the ***Source_Capabilities_Extended*** Message contains further details of the available supply.

6.4.1.2.2.4 USB Communications Capable

The USB Communications Capable bit shall only be set for devices capable of communication over the USB data lines (e.g. D+/- or SS Tx/Rx).

Microchip Ex. 1020, p. 116
Microchip v. Aptiv
IPR2024-00496

6.4.1.2.2.5 Dual-Role Data

The Dual-Role Data bit shall be set when the Port is Dual-Role data capable i.e. it supports the ***DR_Swap*** Message. This is a static capability which shall remain fixed for a given device.

6.4.1.2.2.6 Unchunked Extended Messages Supported

The Unchunked Extended Messages Supported bit shall be set when the Port can send and receive Extended Messages with ***Data Size*** > ***MaxExtendedMsgLegacyLen*** bytes in a single, Unchunked Message.

6.4.1.2.2.7 Peak Current

The USB Power Delivery Fixed Supply is only required to deliver the amount of current requested in the Operating Current (I_{OC}) field of an RDO. In some usages however, for example computer systems, where there are short bursts of activity, it may be desirable to overload the power 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 7.2.8) period needed to maintain such average power. The Peak Current field allows a power 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 6-9). 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 6.5.1) 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 6-9 Fixed Power Source Peak Current Capability

Bits 21..20	Description
00	Peak current equals I_{OC} (default) or look at extended Source capabilities (send ***Get_Source_Cap_Extended*** Message)
01	Overload Capabilities: 1. Peak current equals 150% I_{OC} for 1ms @ 5% duty cycle (low current equals 97% I_{OC} for 19ms) 2. Peak current equals 125% I_{OC} for 2ms @ 10% duty cycle (low current equals 97% I_{OC} for 18ms) 3. Peak current equals 110% I_{OC} for 10ms @ 50% duty cycle (low current equals 90% I_{OC} for 10ms)
10	Overload Capabilities: 1. Peak current equals 200% I_{OC} for 1ms @ 5% duty cycle (low current equals 95% I_{OC} for 19ms) 2. Peak current equals 150% I_{OC} for 2ms @ 10% duty cycle (low current equals 94% I_{OC} for 18ms) 3. Peak current equals 125% I_{OC} for 10ms @ 50% duty cycle (low current equals 75% I_{OC} for 10ms)
11	Overload Capabilities: 1. Peak current equals 200% I_{OC} for 1ms @ 5% duty cycle (low current equals 95% I_{OC} for 19ms) 2. Peak current equals 175% I_{OC} for 2ms @ 10% duty cycle (low current equals 92% I_{OC} for 18ms) 3. Peak current equals 150% I_{OC} for 10ms @ 50% duty cycle (low current equals 50% I_{OC} for 10ms)

6.4.1.2.3 Variable Supply (non-Battery) Power Data Object

Table 6-9 describes a Variable Supply (non-Battery) (10b) PDO for a Source. See Section 7.1.3 for the electrical requirements of the power supply.

The voltage fields shall define the range that output voltage shall fall within. This does not indicate the voltage that will actually be supplied, except it shall fall within that range. The absolute voltage, including any voltage variation, shall not fall below the Minimum Voltage and shall not exceed the Maximum Voltage.

Microchip Ex. 1020, p. 117
Microchip v. Aptiv
IPR2024-00496

Table 6-10 Variable Supply (non-Battery) PDO - Source

Bit(s)	Description
B31..30	Variable Supply (non-Battery)
B29..20	Maximum Voltage in 50mV units
B19..10	Minimum Voltage in 50mV units
B9..0	Maximum Current in 10mA units

6.4.1.2.4 Battery Supply Power Data Object

Table 6-11 describes a Battery (01b) PDO for a Source. See Section 7.1.3 for the electrical requirements of the power supply.

The voltage fields shall represent the Battery's voltage range. The Battery shall be capable of supplying the Power value over the entire voltage range. The absolute voltage, including any voltage variation, shall not fall below the Minimum Voltage and shall not exceed the Maximum Voltage. Note, only the Battery PDO uses power instead of current.

The Sink may monitor the Battery voltage.

Table 6-11 Battery Supply PDO - Source

Bit(s)	Description
B31..30	Battery
B29..20	Maximum Voltage in 50mV units
B19..10	Minimum Voltage in 50mV units
B9..0	Maximum Allowable Power in 250mW units

6.4.1.3 Sink Capabilities Message

A Sink Port shall report power levels it is able to operate at in a series of 32-bit Power Data Objects (see Table 6-7). These are returned as part of a ***Sink_Capabilities*** Message in response to a ***Get_Sink_Cap*** Message (see Figure 6-12). This is similar to that used for Source Port capabilities with equivalent Power Data Objects for Fixed, Variable and Battery Supplies as defined in this section. Power Data Objects are used to convey the Sink Port's operational power requirements including Dual-Role Power Ports presently operating as a Source.

Each Power Data Object shall describe a specific Sink operational power level, such as a Battery (e.g. 2.8-4.1V) or a fixed power supply (e.g. 12V). The ***Number of Data Objects*** field in the Message Header shall define the number of Power Data Objects that follow the Message Header in a Data Message.

All Sinks shall minimally offer one Power Data Object with a power level at which the Sink can operate. A Sink shall not offer multiple Power Data Objects of the same type (fixed, variable, Battery) and the same voltage but shall instead offer one Power Data Object with the highest available current for that Sink capability and voltage.

All Sinks shall include one Power Data Object that reports ***vSafe5V*** even if they require additional power to operate fully. In the case where additional power is required for full operation the Higher Capability bit shall be set.

6.4.1.3.1 Sink Fixed Supply Power Data Object

Table 6-12 describes the Sink Fixed Supply (00b) PDO. See Section 7.1.3 for the electrical requirements of the power supply. The Sink shall set Voltage to its required voltage and Operational Current to its required operating current. Required operating current is defined as the amount of current a given device needs to be functional. This value could be the maximum current the Sink will ever require or could be sufficient to operate the Sink in one of its modes of operation.

Since all USB Consumers support ***vSafe5V***, the required ***vSafe5V*** Fixed Supply Power Data Object is also used to convey additional information that is returned in bits 29 through 20. All other Fixed Supply Power Data Objects shall set bits 29...20 to zero.

Microchip Ex. 1020, p. 118
Microchip v. Aptiv
IPR2024-00496

For a Sink requiring no power from the Source, the Voltage (B19..10) shall be set to 5V and the Operational Current shall be set to 0mA.

Table 6-12 Fixed Supply PDO - Sink

<table>
<tr><th>Bit(s)</th><th>Description</th></tr>
<tr><td>B31..30</td><td>Fixed supply</td></tr>
<tr><td>B29</td><td>Dual-Role Power</td></tr>
<tr><td>B28</td><td>Higher Capability</td></tr>
<tr><td>B27</td><td>Externally Powered</td></tr>
<tr><td>B26</td><td>USB Communications Capable</td></tr>
<tr><td>B25</td><td>Dual-Role Data</td></tr>
<tr><td>B24..23</td><td>Fast Role Swap required USB Type-C Current (see also [USB Type-C 1.2]):

<table><tr><th>Value</th><th>Description</th></tr><tr><td>00b</td><td>Fast Swap not supported (default)</td></tr><tr><td>01b</td><td>Default USB Power</td></tr><tr><td>10b</td><td>1.5A @ 5V</td></tr><tr><td>11b</td><td>3.0A @ 5V</td></tr></table></td></tr>
<tr><td>B22..20</td><td>Reserved – shall be set to zero.</td></tr>
<tr><td>B19..10</td><td>Voltage in 50mV units</td></tr>
<tr><td>B9..0</td><td>Operational Current in 10mA units</td></tr>
</table>

6.4.1.3.1.1 Dual-Role Power

The Dual-Role Power bit shall be set when the Port is Dual-Role Power capable i.e. supports the ***PR_Swap*** Message. This is a static capability which shall remain fixed for a given device.

6.4.1.3.1.2 Higher Capability

In the case that the Sink needs more than ***vSafe5V*** (e.g. 12V) to provide full functionality, then the Higher Capability bit shall be set.

6.4.1.3.1.3 Externally Powered

The Externally Powered bit shall be set when an external source of power is available.

The Externally Powered bit shall be set when either:

- There is an AC supply, e.g. a Wall Wart, directly connected to the Sink.
- Or, in the case of a PDUSB Hub:
  - The Hub is receiving power from a PD Source with its Externally Powered bit set.
  - The Hub is receiving power from multiple PD Sources all with their Externally Powered bits set.

6.4.1.3.1.4 USB Communications Capable

The USB Communications Capable bit shall only be set for devices capable of communication over the USB data lines (e.g. D+/- or SS Tx/Rx).

6.4.1.3.1.5 Dual-Role Data

The Dual-Role Data bit shall be set when the Port is Dual-Role data capable i.e. it supports the ***DR_Swap*** Message. This is a static capability which shall remain fixed for a given device.

Microchip Ex. 1020, p. 119
Microchip v. Aptiv
IPR2024-00496

6.4.1.3.1.6 Fast Role Swap USB Type-C Current

The Fast Role Swap USB Type-C Current field shall indicate the current level the Sink will require after a Fast Role Swap has been performed.

Initially when the new Source applies ***vSafe5V*** it will have Rd asserted but shall provide the USB Type-C Current indicated by the new Sink in this field. If the new Source is not able to supply this level of current it shall not perform a Fast Role Swap. When Rp is asserted by the new Source during the Fast Role Swap AMS (see Section 6.3.17), the value of USB Type-C Current indicated by Rp shall be the same or greater than that indicated in the Fast Role Swap USB Type-C Current field.

6.4.1.3.2 Variable Supply (non-Battery) Power Data Object

Table 6-13 describes a Variable Supply (non-Battery) (10b) PDO used by a Sink. See Section 7.1.3 for the electrical requirements of the power supply.

The voltage fields shall be set to the output voltage range that the Sink requires to operate. The Operational Current field shall be set to the operational current that the Sink requires at the given voltage range. The absolute voltage, including any voltage variation, shall not fall below the Minimum Voltage and shall not exceed the Maximum Voltage. Required operating current is defined as the amount of current a given device needs to be functional. This value could be the maximum current the Sink will ever require or could be sufficient to operate the Sink in one of its modes of operation.

Table 6-13 Variable Supply (non-Battery) PDO - Sink

Bit(s)	Description
B31..30	Variable Supply (non-Battery)
B29..20	Maximum Voltage in 50mV units
B19..10	Minimum Voltage in 50mV units
B9..0	Operational Current in 10mA units

6.4.1.3.3 Battery Supply Power Data Object

Table 6-14 describes a Battery (01b) PDO used by a Sink. See Section 7.1.3 for the electrical requirements of the power supply.

The voltage fields shall be set to the output voltage range that the Sink requires to operate. The Operational Power field shall be set to the operational power that the Sink requires at the given voltage range. The absolute voltage, including any voltage variation, shall not fall below the Minimum Voltage and shall not exceed the Maximum Voltage. Note, only the Battery PDO uses power instead of current. Required operating power is defined as the amount of power a given device needs to be functional. This value could be the maximum power the Sink will ever require or could be sufficient to operate the Sink in one of its modes of operation.

Table 6-14 Battery Supply PDO - Sink

Bit(s)	Description
B31..30	Battery
B29..20	Maximum Voltage in 50mV units
B19..10	Minimum Voltage in 50mV units
B9..0	Operational Power in 250mW units

6.4.2 Request Message

A ***Request*** Message shall be sent by a Sink to request power, typically during the request phase of a 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 (see Section 8.3.2.1.3). A ***Request*** Message shall return one and only one Sink Request Data Object that shall identify the Power Data Object being requested.

Microchip Ex. 1020, p. 120
Microchip v. Aptiv
IPR2024-00496

The ***Request*** Message includes the requested power level. For example, if the ***Source_Capabilities*** Message includes a Fixed Supply PDO that offers 12V @ 1.5A and if the Sink only wants 12V @ 0.5A, it will set the Operating Current field to 50 (i.e. 10mA * 50 = 0.5A). The ***Request*** Message requests the highest current the Sink will ever require in the Maximum Operating Current Field (in this example it would be 100 (100 * 10mA = 1.0A)).

The request takes one of two forms depending on the kind of power requested. The Fixed Power Data Object and Variable Power Data Object share a common format (see Table 6-15 and Table 6-16). The Battery Power Data Object uses a different format (see Table 6-17 and Table 6-18).

Table 6-15 Fixed and Variable Request Data Object

Bits	Description
B31	***Reserved*** – shall be set to zero
B30..28	Object position (000b is ***Reserved*** and shall not be used)
B27	GiveBack flag = 0
B26	Capability Mismatch
B25	USB Communications Capable
B24	No USB Suspend
B23	Unchunked Extended Messages Supported
B22..20	***Reserved*** - shall be set to zero.
B19..10	Operating current in 10mA units
B9..0	Maximum Operating Current 10mA units

Table 6-16 Fixed and Variable Request Data Object with GiveBack Support

Bits	Description
B31	***Reserved*** – shall be set to zero
B30..28	Object position (000b is ***Reserved*** and shall not be used)
B27	GiveBack flag =1
B26	Capability Mismatch
B25	USB Communications Capable
B24	No USB Suspend
B23	Unchunked Extended Messages Supported
B22..20	***Reserved*** - shall be set to zero.
B19..10	Operating Current in 10mA units
B9..0	Minimum Operating Current 10mA units

Table 6-17 Battery Request Data Object

Bits	Description
B31	***Reserved*** – shall be set to zero
B30..28	Object position (000b is ***Reserved*** and shall not be used)
B27	GiveBackFlag = 0
B26	Capability Mismatch
B25	USB Communications Capable
B24	No USB Suspend
B23	Unchunked Extended Messages Supported
B22..20	***Reserved*** - shall be set to zero.
B19..10	Operating Power in 250mW units
B9..0	Maximum Operating Power in 250mW units

Microchip Ex. 1020, p. 121
Microchip v. Aptiv
IPR2024-00496

Table 6-18 Battery Request Data Object with GiveBack Support

Bits	Description
B31	***Reserved*** – shall be set to zero
B30..28	Object position (000b is ***Reserved*** and shall not be used)
B27	GiveBackFlag = 1
B26	Capability Mismatch
B25	USB Communications Capable
B24	No USB Suspend
B23	Unchunked Extended Messages Supported
B22..20	***Reserved*** - shall be set to zero.
B19..10	Operating Power in 250mW units
B9..0	Minimum Operating Power in 250mW units

6.4.2.1 Object Position

The value in the Object Position field shall indicate which object in the ***Source_Capabilities*** Message the RDO refers. The value 1 always indicates the 5V Fixed Supply PDO as it is the first object following the ***Source_Capabilities*** Message Header. The number 2 refers to the next PDO and so forth.

6.4.2.2 GiveBack Flag

The GiveBack flag shall be set to indicate that the Sink will respond to a ***GotoMin*** Message by reducing its load to the Minimum Operating Current. It will typically be used by a USB Device while charging its Battery because a short interruption of the charge will have minimal impact on the user and will allow the Source to manage its load better.

6.4.2.3 Capability Mismatch

A Capability Mismatch occurs when the Sink cannot satisfy its power requirements from the capabilities offered by the Source. In this case the Sink shall make a valid request from the offered capabilities and shall set the Capability Mismatch bit (see Section 8.2.5.2).

When a Sink returns a Request Data Object in response to advertised capabilities with this bit set, it indicates that the Sink wants power that the Source cannot provide. This may be due to either a voltage that is not available or the amount of available current. At this point the Source can use the information in the ***Request*** Message combined with the contents of the ***Sink_Capabilities*** Message to ascertain the Voltage and Current required by the Sink for full operation.

In this context a valid ***Request*** Message means the following:

- The Object position field shall contain a reference to an object in the last received ***Source_Capabilities*** Message.
- The Operating Current/Power field shall contain a value which is less than or equal to the maximum current/power offered in the ***Source_Capabilities*** Message.
- If the GiveBack flag is set to zero i.e. there is a Maximum Operating Current/Power field:
    - If the Capability Mismatch bit is set to one
        - The Maximum Operating Current/Power field may contain a value larger than the maximum current/power offered in the ***Source_Capabilities*** Message's PDO as referenced by the Object position field. This enables the Sink to indicate that it requires more current/power than is being offered. If the Sink requires a different voltage this will be indicated by its ***Sink_Capabilities*** Message.
    - Else if the Capability Mismatch bit is set to zero
        - The Maximum Operating Current/Power field shall contain a value less than or equal to the maximum current/power offered in the ***Source_Capabilities*** Message's PDO as referenced by the Object position field.

Microchip Ex. 1020, p. 122
Microchip v. Aptiv
IPR2024-00496

- Else if the GiveBack flag is set to one i.e. there is a Minimum Operating Current/Power field:
    - The Minimum Operating Current/Power field shall contain a value less than the Operating Current/Power field.

6.4.2.4 USB Communications Capable

The USB Communications Capable flag shall be set to one when the Sink has USB data lines and is capable of communicating using either ***[USB 2.0]*** or ***[USB 3.1]*** protocols. The USB Communications Capable flag shall be set to zero when the Sink does not have USB data lines or is otherwise incapable of communicating using either ***[USB 2.0]*** or ***[USB 3.1]*** protocols. This is used by the Source to determine operation in certain cases such as USB suspend. If the USB Communications Capable flag has been set to zero by a Sink then the Source needs to be aware that USB Suspend rules cannot be observed by the Sink.

6.4.2.5 No USB Suspend

The No USB Suspend flag may be set by the Sink to indicate to the Source that this device is requesting to continue its 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 flag cleared.

6.4.2.6 Unchunked Extended Messages Supported

The Unchunked Extended Messages Supported bit shall be set when the Port can send and receive Extended Messages with ***Data Size*** > ***MaxExtendedMsgLegacyLen*** bytes in a single, Unchunked Message.

6.4.2.7 Operating Current

The Operating Current field in the Request Data Object shall be set to the actual amount of current the Sink needs to operate at a given time. A new ***Request*** Message, with an updated Operating Current value, shall be issued whenever the Sink's power needs change e.g. from Maximum Operating Current down to a lower current level. In conjunction with the Maximum Operating Current field or Minimum Operating Current field, it provides the Source with additional information that allows it to better manage the distribution of its power. This field shall apply to the Fixed and Variable RDO.

6.4.2.8 Maximum Operating Current

The Maximum Operating Current field in the ***Request*** Message shall be set to the highest current the Sink will ever require. The difference between the Operating Current and Maximum Operating Current fields (when the GiveBack Flag is cleared) is used by the Device Policy Manager in the Source to calculate the size of the Power Reserve to be maintained (see Section 8.2.5.1). The Operating Current value shall be less than or equal to the Maximum Operating Current value.

When the Capabilities Mismatch bit is set to zero the requested Maximum Operating Current shall be less than or equal to the current in the offered Source Capabilities since the Source will need to reserve this power for future use. The Maximum Operating Current field shall continue to be set to the highest current needed in order to maintain the allocation of the Power Reserve. If Maximum Operating Current is requested when the Power Reserve is being used by a GotoMin capable device then the resulting Message will be a ***Wait*** Message to enable the Source to reclaim the additional current (see Section 6.3.12.1 and Section 8.2.5.1).

When the Capabilities Mismatch bit is set to one the requested Maximum Operating Current may be greater than the current in the offered Source Capabilities since the Source will need this information to ascertain the Sink's actual needs.

See Section 6.4.2.3 for more details of the usage of the Capabilities Mismatch bit.

This field shall apply to the Fixed and Variable RDO.

Microchip Ex. 1020, p. 123
Microchip v. Aptiv
IPR2024-00496

6.4.2.9 Minimum Operating Current

The Minimum Operating Current field in the ***Request*** Message shall be set to the lowest current the Sink requires to maintain operation. The difference between the Operating Current and Minimum Operating Current fields (when the GiveBack Flag is set) is used by the Device Policy Manager to calculate the amount of power which can be reclaimed using a ***GotoMin*** Message. The Operating Current value shall be greater than the Minimum Operating Current value.

This field shall apply to the Fixed and Variable RDO.

6.4.2.10 Operating Power

The Operating Power field in the Request Data Object shall be set to the actual amount of power the Sink wants at this time. In conjunction with the Maximum Operating Power field, it provides the Source with additional information that allows it to better manage the distribution of its power.

This field shall apply to the Battery RDO.

6.4.2.11 Maximum Operating Power

The Maximum Operating Power field in the ***Request*** Message shall be set to the highest power the Sink will ever require. This allows a Source with a power supply shared amongst multiple ports to intelligently distribute power.

When the Capabilities Mismatch bit is set to zero the requested Maximum Operating Power shall be less than or equal to the power in the offered Source Capabilities since the Source will need to reserve this power for future use. The Maximum Operating Power field shall continue to be set to the highest power needed in order to maintain the allocation of the Power Reserve. If Maximum Operating Power is requested when the Power Reserve is being used by a GotoMin capable device then the resulting Message will be a ***Wait*** Message to enable the Source to reclaim the additional power (see Section 6.3.12.1 and Section 8.2.5.1).

When the Capabilities Mismatch bit is set to one the requested Maximum Operating Power may be greater than the current in the offered Source Capabilities since the Source will need this information to ascertain the Sink's actual needs

See Section 6.4.2.3 for more details of the usage of the Capabilities Mismatch bit.

This field shall apply to the Battery RDO.

6.4.2.12 Minimum Operating Power

The Minimum Operating Power field in the ***Request*** Message shall be set to the lowest current the Sink requires to maintain operation. When combined with the Operating Power, it gives a Source with a power supply shared amongst multiple ports information about how much power it can temporarily get back so it can to intelligently distribute power.

This field shall apply to the Battery RDO.

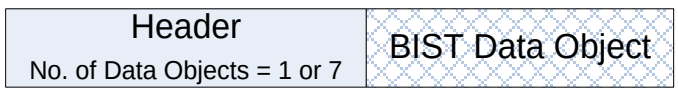
6.4.3 BIST Message

The ***BIST*** Message is sent to request the Port to enter a Physical Layer test mode (see Section 5.9) that performs one of the following functions:

- Enter a Continuous BIST Mode to send a continuous stream of test data to the Tester
- Send BIST test data to the UUT

The Message format is as follows:

Figure 6-13 BIST Message



Microchip Ex. 1020, p. 124
Microchip v. Aptiv
IPR2024-00496

All ports shall be able to be a Unit Under Test (UUT) only when operating at ***vSafe5V***. All of the following BIST Modes shall be supported:

- Process reception of a ***BIST Carrier Mode*** BIST Data Object that shall result in the generation of the appropriate carrier signal.
- Process reception of a ***BIST Test Data*** BIST Data Object that shall result in the Message being ***Ignored***.

It is ***Optional*** for a Port to take on the role of a Tester.

When a Port receives a ***BIST*** Message BIST Data Object for a BIST Mode when Power Role swapped or not operating at ***vSafe5V***, the ***BIST*** Message shall be ***Ignored***.

When a Port receives a ***BIST*** Message BIST Data Object for a BIST Mode it does not support the ***BIST*** Message shall be ***Ignored***.

When a Port or Cable Plug receives a ***BIST*** Message BIST Data Object for a Continuous BIST Mode that it supports, the Port or Cable Plug enters the requested BIST Mode and shall remain in that BIST Mode for ***tBISTContMode*** and then shall return to normal operation (see Section 6.6.7.2).

The usage model of the PHY Layer BIST modes generally assumes that some controlling agent will request a test of its Port Partner. A UUT Port minimally has to process a request to enter test mode and return error counters. A Tester Port shall have a means to place the UUT Port into receiver test mode and retrieve the error counters from the UUT. A Port, that is not part of a Tester, is not expected to be the initiator of a receiver test operation, but is not precluded from doing so.

In Section 8.3.2.14 there is a sequence description of the test sequences used for compliance testing.

The fields in the BIST Data Object are defined in the Table 6-19.

Table 6-19 BIST Data Object

Bit(s)	Value	Parameter	Description	Reference
B31..28	0000b..0100b	***Reserved***	Shall not be used	Section 1.4.2.9
	0101b	***BIST Carrier Mode***	Request Transmitter to enter BIST Carrier Mode	Section 6.4.3.1
	0110b..0111b	***Reserved***	Shall not be used	Section 1.4.2.9
	1000b	***BIST Test Data***	Sends a Test Data Frame.	Section 6.4.3.2
	1001b..1111b	***Reserved***	Shall not be used	Section 1.4.2.9
B27..0		***Reserved***	Shall be set to zero.	Section 1.4.2.9

6.4.3.1 BIST Carrier Mode

Upon receipt of a ***BIST*** Message, with a ***BIST Carrier Mode*** BIST Data Object, the UUT shall send out a continuous string of alternating "1"s and “0”s. Note: that in the case that the BMC Signaling Scheme is used the “1”s and “0”s will in addition be BMC encoded.

The UUT shall exit the Continuous BIST Mode within ***tBISTContMode*** of this Continuous BIST Mode being enabled (see Section 6.6.7.2).

6.4.3.2 BIST Test Data

Upon receipt of a ***BIST*** Message, with a ***BIST Test Data*** BIST Data Object, the UUT shall return a ***GoodCRC*** Message and shall enter a test mode in which it sends no further Messages except for ***GoodCRC*** Messages in response to received Messages. See Section 5.9.2 for the definition of the Test Data Frame.

The test shall be ended by sending ***Hard Reset*** Signaling to reset the UUT.

Microchip Ex. 1020, p. 125
Microchip v. Aptiv
IPR2024-00496

6.4.4 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, the VDM Header, and may contain up to a maximum of six additional VDM Objects (VDO).

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. For example a ***Vendor_Defined*** Message could be used to enable the Source to offer additional power via a ***Source_Capabilities*** Message.

The Message format shall be as shown in Figure 6-14.

Figure 6-14 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 6-20. 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 6-21.

Both Unstructured and Structured VDMs shall only be sent and received after an Explicit Contract has been established. The only exception to this is the ***Discover Identity*** Command which may be sent by Source when no Contract or an Implicit Contract (in place after a Power Role Swap or Fast Role Swap) is in place in order to discover Cable capabilities (see Section 8.3.3.21.3). A VDM Message sequence shall not interrupt any other PD Message Sequence. A VDM Message sequence shall be interruptible by any other PD Message Sequence.

6.4.4.1 Unstructured VDM

The Unstructured VDM does not define the contents of bits B14..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.

The following rules apply to the use of Unstructured VDM Messages:

- Unstructured VDMs shall only be used when an Explicit Contract is in place.
- 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 Unstructured VDM.
- Unstructured VDMs shall not be initiated or responded to under any other circumstances.
- A “command” sequence shall be interruptible e.g. due to the need for a power related AMS.
- Unstructured VDMs shall only be used during Modal Operation in the context of an Active Mode.
- Unstructured VDMs may be used with SOP* Packets.
- When a DFP or UFP does not support Structured VDMs or does not recognize the VID it shall return a ***Not_Supported*** Message.
-

Microchip Ex. 1020, p. 126
Microchip v. Aptiv
IPR2024-00496

Table 6-20 illustrates the VDM Header bits.

Table 6-20 Unstructured VDM Header

Bit(s)	Parameter	Description
B31..16	Vendor ID (VID)	Unique 16-bit unsigned integer. Assigned by the USB-IF to the Vendor.
B15	VDM Type	0 = Unstructured VDM
B14..0	Available for Vendor Use	Content of this field is defined by the vendor.

6.4.4.1.1 USB Vendor ID

The Vendor ID 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.

6.4.4.1.2 VDM Type

The VDM Type field shall be set to zero indicating that this is an Unstructured VDM.

6.4.4.2 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 6-21.

The following rules apply to the use of Structured VDM Messages:

- Structured VDMs shall only be used when an Explicit Contract is in place with the following exception:
  - Prior to establishing an Explicit Contract a Source may issue ***Discover Identity*** Messages, to a Cable Plug using SOP' Packets, as an Initiator (see Section 8.3.3.21.3).
- Either Port may be an Initiator of Structured VDMs except for the ***Enter Mode*** and ***Exit Mode*** Commands which shall only be initiated by the DFP.
- A Cable Plug shall only be a Responder to Structured VDMs.
- ***Enter Mode*** and ***Exit Mode*** Commands shall only be responded to by a UFP or Cable Plug.
- Structured VDMs shall not be initiated or responded to under any other circumstances.
- When a DFP or UFP does not support Structured VDMs any Structured VDMs received shall return a ***Not_Supported*** Message.
- When a Cable Plug does not support Structured VDMs any Structured VDMs received shall be ***Ignored***.
- A DFP, UFP or Cable Plug which supports Structured VDMs and receiving a Structured VDM for a SVID that it does not recognize shall reply with a NAK Command.
- A Structured VDM Command sequence shall be interruptible e.g. due to the need for a power related AMS.

Table 6-21 Structured VDM Header

Bit(s)	Field	Description
B31..16	Standard or Vendor ID (SVID)	Unique 16 bit unsigned integer, assigned by the USB-IF
B15	VDM Type	1 = Structured VDM
B14..13	Structured VDM Version	Version Number of the Structured VDM (not this specification Version): • Version 1.0 = 00b (shall not be used) • Version 2.0 = 01b • Values 2-3 are ***Reserved*** and shall not be used
B12..11	***Reserved***	For Commands 0..15 shall be set to 0 and shall be ***Ignored*** SVID Specific Commands (16..31) defined by the SVID.

Microchip Ex. 1020, p. 127
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Description
B10..8	Object Position	For the ***Enter Mode, Exit Mode*** and ***Attention*** Commands: • 000b = ***Reserved*** and shall not be used. • 001b..110b = Index into the list of VDOs to identify the desired Mode VDO • 111b = Exit all Active Modes (equivalent of a power on reset). Shall only be used with the ***Exit Mode*** Command. Commands 0..3, 7..15: • 000b • 001b..111b = ***Reserved*** and shall not be used. SVID Specific Commands (16..31) defined by the SVID.
B7..6	Command Type	00b = Initiator 01b = Responder ACK 10b = Responder NAK 11b = Responder BUSY
B5	***Reserved***	Shall be set to 0 and shall be ***Ignored***
B4..0	Command[1]	0 = ***Reserved***, shall not be used 1 = ***Discover Identity*** 2 = ***Discover SVIDs*** 3 = ***Discover Modes*** 4 = ***Enter Mode*** 5 = ***Exit Mode*** 6 = ***Attention*** 7-15 = ***Reserved***, shall not be used 16..31 = SVID Specific Commands
Note 1: In the case where a SID is used the modes are defined by a standard. When a VID is used the modes are defined by the Vendor.		

Table 6-22 shows the Commands, which SVID to use with each Command and the ***SOP**** values which shall be used.

Table 6-22 Structured VDM Commands

Command	VDM Header SVID Field	SOP* used
Discover Identity	Shall only use the ***PD SID***.	Shall only use ***SOP***/***SOP'***.
Discover SVIDs	Shall only use the ***PD SID***.	Shall only use ***SOP***/***SOP'***.
Discover Modes	Valid with any SVID.	Shall only use ***SOP***/***SOP'***.
Enter Mode	Valid with any SVID.	Valid with ***SOP****.
Exit Mode	Valid with any SVID.	Valid with ***SOP****.
Attention	Valid with any SVID.	Valid with ***SOP.***
SVID Specific Commands	Valid with any SVID.	Valid with ***SOP**** (defined by SVID).

6.4.4.2.1 SVID

The SVID field shall contain either a 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.

Table 6-23 lists specific SVID values referenced by this specification.

Table 6-23 SVID Values

Parameter	Value	Description
PD SID	0xFF00	Standard ID allocated to this specification.

Microchip Ex. 1020, p. 128
Microchip v. Aptiv
IPR2024-00496

6.4.4.2.2 VDM Type

The VDM Type field shall be set to one indicating that this is a Structured VDM.

6.4.4.2.3 Structured VDM Version

The Structured VDM Version field indicates 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. This field shall be set to 01b to indicate Version 2.0.

To ensure interoperability with existing USBPD Products, USBPD 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 that supported, a Port shall respond using the highest Version number it supports.

The Structured VDM Version field of the ***Discover Identity*** Command sent and received during VDM discovery shall be used to determine the lowest common Structured VDM Version supported by the Port Partners or Cable Plug and shall continue to operate using this Specification Revision until they are Detached. After discovering the Structure VDM Version, the Structured VDM Version field shall match the agreed common Structured VDM Version.

6.4.4.2.4 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 a Mode. The value in Object Position field is an index into that list that indicates which VDO (e.g. Mode) in the list the ***Enter Mode*** and ***Exit Mode*** Command refers to. The Object Position shall start with one for the first Mode in the list. If the SVID is a VID, the content of the VDO for the Mode shall be defined by the vendor. If the SVID is a SID, the content shall be defined by the Standard. The VDO's content may be as simple as a numeric value or as complex as bit mapped description of capabilities of the Mode. In all cases, the Responder is responsible for deciphering the contents to know whether or not it supports the Mode at the Object Position.

This field shall be set to zero when not required by the Command.

6.4.4.2.5 Command Type

This Command Type field shall be used to indicate the type of Command request/response being sent.

An Initiator shall set the field to "Initiator" to indicate that this is a Command request from an Initiator.

"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:

- has an ***Invalid*** parameter (e.g. ***Invalid*** SVID or Mode)
- cannot not be acted upon because the configuration is not correct (e.g. a Mode which has a dependency on another Mode or a request to exit a Mode which is not Active)
- is not recognized

The handling of "Responder NAK" is left up to the Initiator.

"Responder BUSY" shall be sent in the response to a VDM when the Responder is unable to respond to the Command request immediately, but the Command request may be retried. The Initiator shall wait ***tVDMBusy*** after a "Responder BUSY" response is received before retrying the Command request.

6.4.4.2.6 Command

This field contains the value for the VDM Command being sent. The Commands explicitly listed in this 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.

Microchip Ex. 1020, p. 129
Microchip v. Aptiv
IPR2024-00496

A Structured VDM Command consists of a Command request and a Command response (ACK, NAK or BUSY). A Structured VDM Command is deemed to be completed (and if applicable, the transition to the requested functionality is made) when the ***GoodCRC*** Message has been successfully sent by the Initiator in reply to the Responder's Command response.

If the Structured VDM Command request is not recognized it shall be NAKed.

6.4.4.3 Use of Commands

The VDM Header for a Structured VDM Message defines Commands used to retrieve a list of SVIDs the device supports, to discover the Modes associated with each SVID, and to enter/exit the Modes. The Commands include:

- ***Discover Identity***
- ***Discover SVIDs***
- ***Discover Modes***
- ***Enter Mode***
- ***Exit Mode***
- ***Attention***

Additional Command space is also reserved for Standard and Vendor use and for future extensions.

The Command sequences use the terms Initiator and Responder to identify messaging roles the ports are taking on relative to each other. This role is independent of the Port's power capability (Provider, Consumer etc.) or its present power role (Source or Sink). The Initiator is the Port sending the initial Command request and the Responder is the Port replying with the Command response. See Section 6.4.4.3.6.

All Ports that support Modes shall support the ***Discover Identity***, ***Discover SVIDs***, the ***Discover Modes***, the ***Enter Mode*** and ***Exit Mode*** Commands.

Table 6-24 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. A NAK response should be taken as an indication not to retry that particular Command.

Table 6-24 Commands and Responses

Command	Allowed Response	Reference
Discover Identity	ACK, NAK, BUSY	Section 6.4.4.3.1
Discover SVIDs	ACK, NAK, BUSY	Section 6.4.4.3.2
Discover Modes	ACK, NAK, BUSY	Section 6.4.4.3.3
Enter Mode	ACK, NAK	Section 6.4.4.3.4
Exit Mode	ACK, NAK	Section 6.4.4.3.5
Attention	None	Section 6.4.4.3.6

Examples of Command usage can be found in Appendix G.

6.4.4.3.1 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). The Discovery Identity Command is also used to determine whether a Cable Plug is PD-Capable by looking for a GoodCRC Message Response.

The ***Discover Identity*** Command shall be used to determine whether a given Cable Plug is PD Capable (see Section 8.3.3.17.1 and Section 8.3.3.21.3). In this case a ***Discover Identity*** Command request sent to SOP' shall not cause a Soft Reset if a ***GoodCRC*** Message response is not returned since this can indicate a non-PD Capable cable. Note that a Cable Plug will not be ready for PD Communication until tVCONNStable after VCONN has been applied (see ***[USB Type-C 1.2]***). During Cable Plug discovery, when there is an Explicit Contract, ***Discover Identity*** Commands are sent at a rate

Microchip Ex. 1020, p. 130
Microchip v. Aptiv
IPR2024-00496

defined by the ***DiscoverIdentityTimer*** (see Section 6.6.14) up to a maximum of ***nDiscoverIdentityCount*** times (see Section 6.7.5).

A PD-Capable Cable Plug shall return a ***Discover Identity*** Command ACK in response to a ***Discover Identity*** Command request sent to SOP'.

A PD-Capable UFP that supports Modal Operation shall return a ***Discover Identity*** Command ACK in response to a ***Discover Identity*** Command request sent to SOP.

The SVID in the ***Discover Identity*** Command request shall be set to the ***PD SID*** (see Table 6-23).

The ***Number of Data Objects*** field in the Message Header in the ***Discover Identity*** Command request shall be set to 1 since the ***Discover Identity*** Command request shall not contain any VDOs.

The ***Discover Identity*** Command ACK sent back by the Responder shall contain an ID Header VDO, a Cert Stat VDO, a Product VDO and the Product Type VDOs defined by the Product Type as shown in Figure 6-15. This specification defines the following Product Type VDOs:

- Cable VDO (see Section 6.4.4.3.1.2).
- Alternate Mode Adapter VDO (see Section 6.4.4.3.1.3)

No VDOs other than those defined in this specification shall be sent as part of the ***Discover Identity*** Command response. 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. Any additional VDOs received by the initiator shall be ***Ignored***.

Figure 6-15 Discover Identity Command response



[1] Only Data objects defined in this specification can be sent as part of the ***Discover Identity*** Command.

[2] The following sections define the number and content of the VDOs for each Product Type.

The ***Number of Data Objects*** field in the Message Header in the ***Discover Identity*** Command NAK and BUSY responses shall be set to 1 since they shall not contain any VDOs.

6.4.4.3.1.1 ID Header VDO

The ID Header VDO contains information corresponding to the Power Delivery Product. The fields in the ID Header VDO shall be as defined in Table 6-25.

Table 6-25 ID Header VDO

Bit(s)	Description	Reference
B31	USB Communications Capable as USB Host: • Shall be set to one if the product is capable of enumerating USB Devices. • Shall be set to zero otherwise	Section 6.4.4.3.1.1.1
B30	USB Communications Capable as a USB Device: • Shall be set to one if the product is capable of being enumerated as a USB Device. • Shall be set to zero otherwise	Section 6.4.4.3.1.1.2
B29..27	Product Type (UFP): • 000b – Undefined • 001b – PDUSB Hub • 010b – PDUSB Peripheral • 011b..100b – ***Reserved***, shall not be used. • 101b – Alternate Mode Adapter (AMA) • 110b..111b – ***Reserved***, shall not be used.	Section 6.4.4.3.1.1.3

Microchip Ex. 1020, p. 131
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Description	Reference
	Product Type (Cable Plug): • 000b – Undefined • 001b..010b – ***Reserved***, shall not be used. • 011b – Passive Cable • 100b – Active Cable • 101b..111b – ***Reserved***, shall not be used.	
B26	Modal Operation Supported: • Shall be set to one if the product supports Modal Operation. • Shall be set to zero otherwise	Section 6.4.4.3.1.1.4
B25..23	Product Type (DFP): • 000b – Undefined • 001b – PDUSB Hub • 010b – PDUSB Host • 011b – Power Brick • 100b - Alternate Mode Controller (AMC) • 101b..111b – ***Reserved***, shall not be used.	
B22..16	***Reserved***. Shall be set to zero.	
B15..0	16-bit unsigned integer. USB Vendor ID	***[USB 2.0]*/*[USB 3.1]***

6.4.4.3.1.1.1 Data Capable as a USB Host

The Data Capable as a USB Host field is used to indicate whether or not the Port has a USB Host Capability.

6.4.4.3.1.1.2 Data Capable as a USB Device

The Data Capable as a USB Device field is used to indicate whether or not the Port has a USB Device Capability.

6.4.4.3.1.1.3 Product Type (UFP)

The 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. For DRD Products this field shall indicate the capability regardless of the present Data Role. Table 6-26 defines the Product Type VDOs which shall be returned.

Table 6-26 Product Types (UFP)

Product Type	Description	Product Type VDO	Reference
Undefined	Shall be used where no other Product Type value is appropriate.	None	
PDUSB Hub	Shall be used when the Product is a PDUSB Hub.	None	
PDUSB Peripheral	Shall be used when the Product is a PDUSB Device other than a PDUSB Hub.	None	
Alternate Mode Adapter	Shall be used when the Product is a PDUSB Device that supports one or more Alternate Modes.	AMA VDO	Section 6.4.4.3.1.3

6.4.4.3.1.1.4 Product Type (Cable Plug)

The Product Type (Cable Plug) field indicates the type of Product when the Product is a Cable Plug, whether a VDO will be returned and if so the type of VDO to be returned. Table 6-27 defines the Product Type VDOs which shall be returned.

Microchip Ex. 1020, p. 132
Microchip v. Aptiv
IPR2024-00496

Table 6-27 Product Types (Cable Plug)

Product Type	Description	Product Type VDO	Reference
Undefined	Shall be used where no other Product Type value is appropriate.	None	
Active Cable	Shall be used when the Product is a cable that incorporates signal conditioning circuits.	Active Cable VDO	Section 6.4.4.3.1.2.2
Passive Cable	Shall be used when the Product is a cable that does not incorporate signal conditioning circuits.	Passive Cable VDO	Section 6.4.4.3.1.2.1

6.4.4.3.1.1.5 Modal Operation Supported

The Modal Operation Supported bit is used to indicate whether or the not the Product supports Modes.

6.4.4.3.1.1.6 Product Type (DFP)

The 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. For DRD Products this field shall indicate the capability regardless of the present Data Role. Table 6-28 defines the Product Type VDOs which shall be returned.

Table 6-28 Product Types (DFP)

Product Type	Description	Product Type VDO	Reference
Undefined	Shall be used where no other Product Type value is appropriate.	None	
PDUSB Hub	Shall be used when the Product is a PDUSB Hub.	None	
PDUSB Host	Shall be used when the Product is a PDUSB Host.	None	
Power Brick	Shall be used when the Product is a Power Brick/Wall Wart.	None	
Alternate Mode Controller	Shall be used when the Product is a PDUSB Host or DFP that supports one or more Alternate Modes.	AMA VDO	Section 6.4.4.3.1.3

6.4.4.3.1.1.7 Vendor ID

Manufacturers shall set the Vendor ID field to the value of the Vendor ID assigned to them by USB-IF. For USB Devices or Hubs which support USB communications the Vendor ID field shall be identical to the Vendor ID field defined in the product's USB Device Descriptor (see ***[USB 2.0]*** and ***[USB 3.1]***).

6.4.4.3.1.1.8 Cert Stat VDO

The Cert Stat VDO shall contain the XID assigned by USB-IF to the product before certification in binary format. The fields in the Cert Stat VDO shall be as defined in Table 6-29.

Table 6-29 Cert Stat VDO

Bit(s)	Description	Reference
B31..0	32-bit unsigned integer, XID	Assigned by USB-IF

6.4.4.3.1.1.9 Product VDO

The Product VDO contains identity information relating to the product. The fields in the Product VDO shall be as defined in Table 6-30.

Microchip Ex. 1020, p. 133
Microchip v. Aptiv
IPR2024-00496

Table 6-30 Product VDO

Bit(s)	Description	Reference
B31..16	16-bit unsigned integer. USB Product ID	*[USB 2.0]*/*[USB 3.1]*
B15..0	16-bit unsigned integer. bcdDevice	*[USB 2.0]*/*[USB 3.1]*

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. For USB Devices or Hubs which support USB communications the Product ID and bcdDevice fields shall be identical to the Product ID and bcdDevice fields defined in the product's USB Device Descriptor (see *[USB 2.0]* and *[USB 3.1]*).

6.4.4.3.1.2 Cable VDO

The Cable VDO defined in this section shall be sent when the Product Type is given as Passive or Active Cable. Table 6-31 and Table 6-32 define the Cable VDOs which shall be sent in each case.

6.4.4.3.1.2.1 Passive Cable VDO

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 V_{BUS} 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 as shown in Table 6-31.

Table 6-31 Passive Cable VDO

Bit(s)	Field	Description
B31..28	HW Version	0000b..1111b assigned by the VID owner
B27..24	Firmware Version	0000b..1111b assigned by the VID owner
B23..21	VDO Version	Version Number of the VDO (not this specification Version): • Version 1.0 = 000b Values 001b..111b are ***Reserved*** and shall not be used
B20	***Reserved***	Shall be set to zero.
B19..18	USB Type-C plug to USB Type-C/Captive	00b = ***Reserved***, shall not be used 01b = ***Reserved***, shall not be used 10b = USB Type-C 11b = Captive
B17	***Reserved***	Shall be set to zero.
B16..13	Cable Latency	0000b – ***Reserved***, 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 ***Reserved***, shall not be used Includes latency of electronics in Active Cable
B12..11	Cable Termination Type	00b = V_{CONN} not required. Cable Plugs that only support ***Discover Identity*** Commands shall set these bits to 00b. 01b = V_{CONN} required 10b..11b = ***Reserved***, shall not be used

Microchip Ex. 1020, p. 134
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Description
B10..9	Maximum V_{BUS} Voltage	Maximum Cable V_{BUS} Voltage: 00b – 20V 01b – 30V 10b – 40V 11b – 50V
B8..7	***Reserved***	Shall be set to zero.
B6..5	V_{BUS} Current Handling Capability	00b = ***Reserved***, shall not be used. 01b = 3A 10b = 5A 11b = ***Reserved***, shall not be used.
B4..3	***Reserved***	Shall be set to zero.
B2..0	USB SuperSpeed Signaling Support	000b = USB 2.0 only, no SuperSpeed support 001b = ***[USB 3.1]*** Gen1 010b = ***[USB 3.1]*** Gen1 and Gen2 011b.. 111b = ***Reserved***, shall not be used See ***[USB Type-C 1.2]*** for definitions.

The HW Version field (B31..28) contains a HW Version assigned by the VID owner.

The FW Version field (B27..24) contains a FW Version assigned by the VID owner.

The VDO Version field (B23..20) contains a VDO version for this VDM version number. This field indicates the expected content for this VDO.

The Connector Type field (B19..18) shall contain a value corresponding to the connector type on the opposite end from the USB Type-C connector.

The Cable Latency field (B16..13) shall contain a value corresponding to the signal latency through the cable which can be used as an approximation for its length.

The Cable Termination Type field (B12..11) shall contain a value indicating whether the Passive Cable needs V_{CONN} only initially in order to support the ***Discover Identity*** Command, after which it can be removed, or the Passive Cable needs V_{CONN} to be continuously applied in order to power some feature of the Cable Plug.

The Maximum V_{BUS} Voltage field (B10..9) shall contain the 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 ***vSrcNew*** max + ***vSrcValid*** max. For example when the Maximum V_{BUS} Voltage 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 21.5V.

The V_{BUS} Current Handling Capability field (B6..5) shall indicate whether the cable is capable of carrying 3A or 5A.

The USB SuperSpeed Signaling Support field (B2..0) shall indicate whether the cable supports only ***[USB 2.0]*** , or in addition Supports ***[USB 3.1]*** Gen1, or Gen1 and Gen2.

6.4.4.3.1.2.2 Active Cable VDO

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 V_{BUS} wire. An Active Cable shall respond to SOP' Communication and may respond to SOP'' Communication. Active Cables shall support the Structured VDM ***Discover Identity*** Command and shall return the Active Cable VDO in a ***Discover Identity*** Command ACK as shown in Table 6-32.

Microchip Ex. 1020, p. 135
Microchip v. Aptiv
IPR2024-00496

Table 6-32 Active Cable VDO

Bit(s)	Field	Description
B31..28	HW Version	0000b..1111b assigned by the VID owner
B27..24	Firmware Version	0000b..1111b assigned by the VID owner
B23..21	VDO Version	Version Number of the VDO (not this specification Version): • Version 1.0 = 000b Values 001b..111b are ***Reserved*** and shall not be used
B20	***Reserved***	Shall be set to zero.
B19..18	USB Type-C plug to USB Type-C/Captive	00b = ***Reserved***, shall not be used 01b = ***Reserved***, shall not be used 10b = USB Type-C 11b = Captive
B17	***Reserved***	Shall be set to zero.
B16..13	Cable Latency	0000b – ***Reserved***, 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) 1011b ….1111b ***Reserved***, shall not be used Includes latency of electronics in Active Cable
B12..11	Cable Termination Type	00b..01b = ***Reserved***, shall not be used 10b = One end Active, one end passive, VCONN required 11b = Both ends Active, VCONN required
B10..9	Maximum VBUS Voltage	Maximum Cable VBUS Voltage: 00b – 20V 01b – 30V 10b – 40V 11b – 50V
B8..7	***Reserved***	Shall be set to zero.
B6..5	VBUS Current Handling Capability	When VBUS Through Cable is “No”, ***Reserved***, shall not be used. When VBUS Though Cable is “Yes”: 00b = ***Reserved***, shall not be used. 01b = 3A 10b = 5A 11b = ***Reserved***, shall not be used.
B4	VBUS Through Cable	0 = No 1 = Yes
B3	SOP” Controller Present	0 = No SOP” controller present 1 = SOP” controller present

Microchip Ex. 1020, p. 136
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Description
B2..0	USB SuperSpeed Signaling Support	000b = ***[USB 2.0]*** only 001b = ***[USB 3.1]*** Gen1 010b = ***[USB 3.1]*** Gen1 and Gen2 011b.. 111b = ***Reserved***, shall not be used

The HW Version field (B31..28) contains a HW Version assigned by the VID owner.

The FW Version field (B27..24) contains a FW Version assigned by the VID owner.

The VDO Version field (B23..20) contains a VDO version for this VDM version number. This field indicates the expected content for this VDO.

The Connector Type field (B19..18) shall contain a value corresponding to the connector type on the opposite end from the USB Type-C connector.

The Cable Latency field (B16..13) shall contain a value corresponding to the signal latency through the cable which can be used as an approximation for its length.

The Cable Termination Type field (B12..11) shall contain a value corresponding to whether the Active Cable has one or two Cable Plugs requiring power from V_{CONN}.

The Maximum V_{BUS} Voltage field (B10..9) shall contain the 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 ***vSrcNew*** max + ***vSrcValid*** max. For example when the Maximum V_{BUS} Voltage 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 21.5V.

The V_{BUS} Current Handling Capability field (B6..5) shall indicate whether the cable is capable of carrying 3A or 5A. The V_{BUS} Current Handling Capability shall only be valid when the V_{BUS} Through Cable field indicates an end to end V_{BUS} wire.

The V_{BUS} Through Cable field (B4) shall indicate whether the cable contains an end to end V_{BUS} wire.

The SOP'' Controller Present field (B3) shall indicate whether one of the Cable Plugs is capable of SOP'' Communication in addition to the ***Normative*** SOP' Communication.

The USB SuperSpeed Signaling Support field (B2..0) shall indicate whether the cable supports only ***[USB 2.0]*** , or in addition Supports ***[USB 3.1]*** Gen1, or Gen1 and Gen2.

6.4.4.3.1.3 Alternate Mode Adapter VDO

The Alternate Mode Adapter (AMA) VDO defined in this section shall be sent when the Product Type is given as Alternate Mode Adapter. Table 6-33 defines the AMA VDO which shall be sent.

Microchip Ex. 1020, p. 137
Microchip v. Aptiv
IPR2024-00496

Table 6-33 AMA VDO

Bit(s)	Field	Description
B31..28	HW Version	0000b..1111b assigned by the VID owner
B27..24	Firmware Version	0000b..1111b assigned by the VID owner
B23..21	VDO Version	Version Number of the VDO (not this specification Version): • Version 1.0 = 000b Values 001b..111b are ***Reserved*** and shall not be used
B20..8	***Reserved***.	Shall be set to zero.
B7..5	V_{CONN} power	When the V_{CONN} required field is set to "Yes" V_{CONN} power needed by adapter for full functionality 000b = 1W 001b = 1.5W 010b = 2W 011b = 3W 100b = 4W 101b = 5W 110b = 6W 111b = ***Reserved***, shall not be used When the V_{CONN} required field is set to "No" ***Reserved***, shall be set to zero.
B4	V_{CONN} required	0 = No 1 = Yes
B3	V_{BUS} required	0 = No 1 = Yes
B2..0	USB SuperSpeed Signaling Support	000b = ***[USB 2.0]*** only 001b = ***[USB 3.1]*** Gen1 and USB 2.0 010b = ***[USB 3.1]*** Gen1, Gen2 and USB 2.0 011b = ***[USB 2.0]*** billboard only 100b.. 111b = ***Reserved***, shall not be used

The HW Version field (B31..28) contains a HW Version assigned by the VID owner.

The FW Version field (B27..24) contains a FW Version assigned by the VID owner.

The VDO Version field (B23..20) contains a VDO version for this VDM version number. This field indicates the expected content for this VDO.

When the V_{CONN} required field indicates that V_{CONN} is required the V_{CONN} power field shall indicate how much power the AMA needs in order to fully operate.

The V_{CONN} required field shall indicate whether V_{CONN} is needed for the AMA to operate.

The V_{BUS} required field shall indicate whether V_{BUS} is needed for the AMA to operate.

The USB SuperSpeed Signaling Support field (B2..0) shall indicate whether the cable supports only ***[USB 2.0]*** , or in addition Supports ***[USB 3.1]*** Gen1, or Gen1 and Gen2 or ***[USB 2.0]*** billboard only.

6.4.4.3.2 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 6-23) 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.

Microchip Ex. 1020, p. 138
Microchip v. Aptiv
IPR2024-00496

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 6-34). 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 Mode.

A Responder that does not support any SVIDs shall return a NAK.

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.

If the Responder supports 12 or more SVIDs then the ***Discover SVIDs*** Command shall be executed multiple times until a Discover SVIDs VDO is returned 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.

Note: that 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.

Figure 6-16 shows an example response to the ***Discover SVIDs*** Command request with two VDOs containing three SVIDs. Figure 6-17 shows an example response with two VDOs containing four SVIDs followed by an empty VDO to terminate the response. Figure 6-18 shows an example response with six VDOs containing twelve SVIDs followed by an additional request that returns an empty VDO indicating there are no more SVIDs to return.

Table 6-34 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.

Figure 6-16 Example Discover SVIDs response with 3 SVIDs

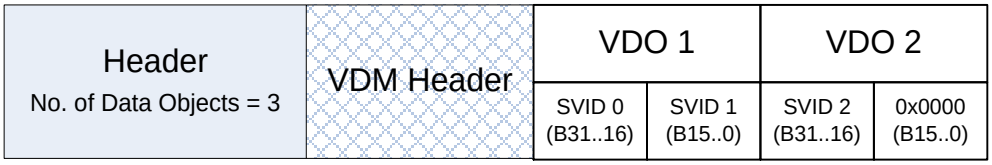
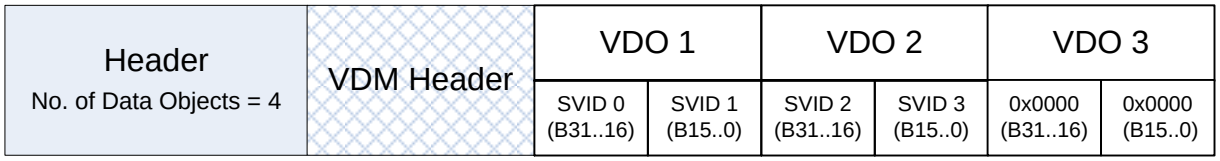
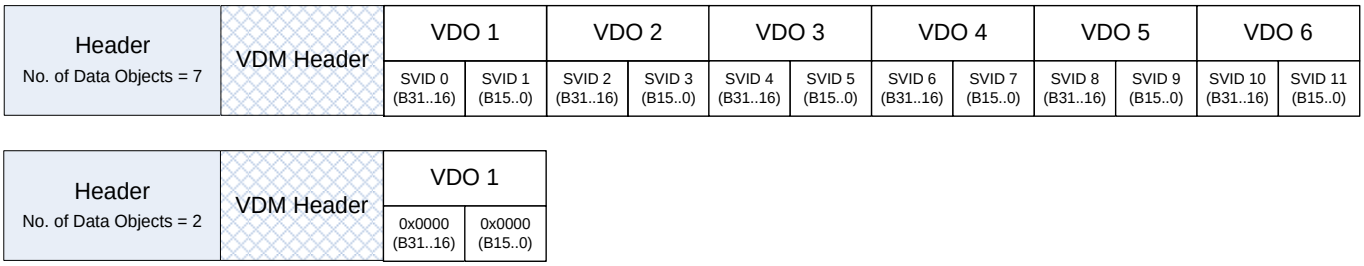


Figure 6-17 Example Discover SVIDs response with 4 SVIDs



Microchip Ex. 1020, p. 139
Microchip v. Aptiv
IPR2024-00496

Figure 6-18 Example Discover SVIDs response with 12 SVIDs followed by an empty response



6.4.4.3.3 Discover Modes

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.

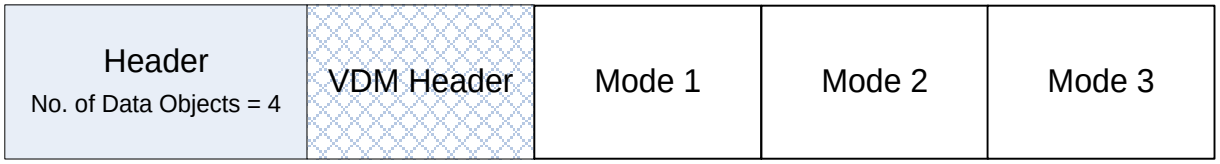
The ***Discover Modes*** Command ACK sent back by the Responder shall contain one or more Modes. The ***Discover Modes*** Command ACK shall contain a Message Header with the ***Number of Data Objects*** field set to a value of 1 to 7 (the actual value is the number of Mode objects plus one). If the ID is a VID, the structure and content of the VDO is left to the Vendor. If the ID is a SID, the structure and content of the VDO is defined by the relevant Standard.

A Responder that does not support any Modes shall return a NAK.

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 6-19 shows an example of a ***Discover Modes*** Command response from a Responder which supports three Modes for a given SVID.

Figure 6-19 Example Discover Modes response for a given SVID with 3 Modes



6.4.4.3.4 Enter Mode Command

The ***Enter Mode*** Command is used by an Initiator (DFP) to command a Responder (UFP or Cable Plug) to enter a specified Mode of operation. Only a DFP is allowed to initiate the Enter Mode Process which it starts after it has successfully completed the Discovery Process.

The value in the Object Position field in the VDM Header shall indicate to which Mode in the ***Discover Modes*** Command the VDO refers (see Figure 6-19). The value 1 always indicates the first Mode as it is the first object following the VDM Header. The value 2 refers to the next 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 Mode.

The ***Number of Data Objects*** field in the Command response shall be set to 1 since an ***Enter Mode*** Command response (ACK, NAK, BUSY) shall not contain any VDOs.

Microchip Ex. 1020, p. 140
Microchip v. Aptiv
IPR2024-00496

Before entering a Mode, by sending the ***Enter Mode*** Command request, that requires the reconfiguring of any pins on entry to that Mode, the Initiator shall ensure that those pins being reconfigured are placed into the USB Safe State. Before entering a 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.

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 Standard or Vendor. Where there are multiple Active Modes at the same time Modal Operation shall start on entry to the first 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.

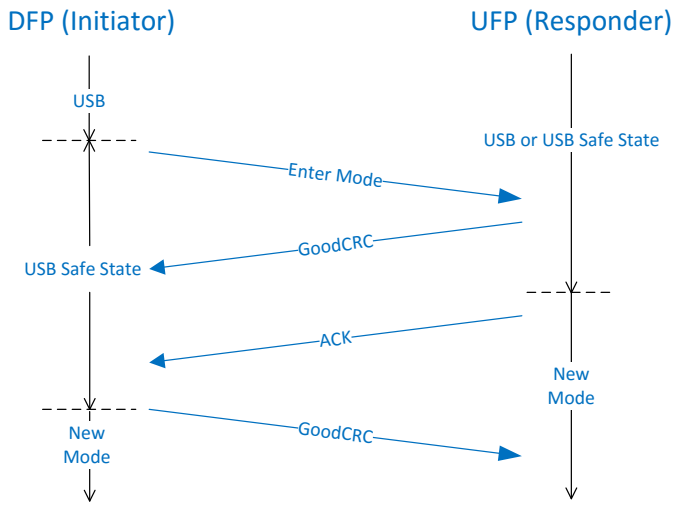
If the Responder responds to the ***Enter Mode*** Command request with an ACK, the Responder shall enter the Mode before sending the ACK. The Initiator shall enter the Mode on reception of the ACK. Receipt of the ***GoodCRC*** Message corresponding to the ACK confirms to the Responder that the Initiator is in an Active Mode and is ready to operate.

If the Responder responds to the ***Enter Mode*** Command request with a NAK, the 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 Mode but return to USB operation.

Figure 6-20 shows the sequence of events during the transition between USB operation and entering a Mode. It illustrates when the Responder's Mode changes and when the Initiator's Mode changes. Figure 6-21 shows a sequence that is Interrupted by a ***Source_Capabilities*** Message, that completes a Contract Negotiation, and then the sequence is Re-run. Figure 6-22 illustrates that when the Responder returns a NAK the transition to a Mode do not take place and the Responder and Initiator remain in their default USB roles.

Figure 6-20 Successful Enter Mode sequence



Microchip Ex. 1020, p. 141
Microchip v. Aptiv
IPR2024-00496

Figure 6-21 Enter Mode sequence Interrupted by Source Capabilities and then Re-run

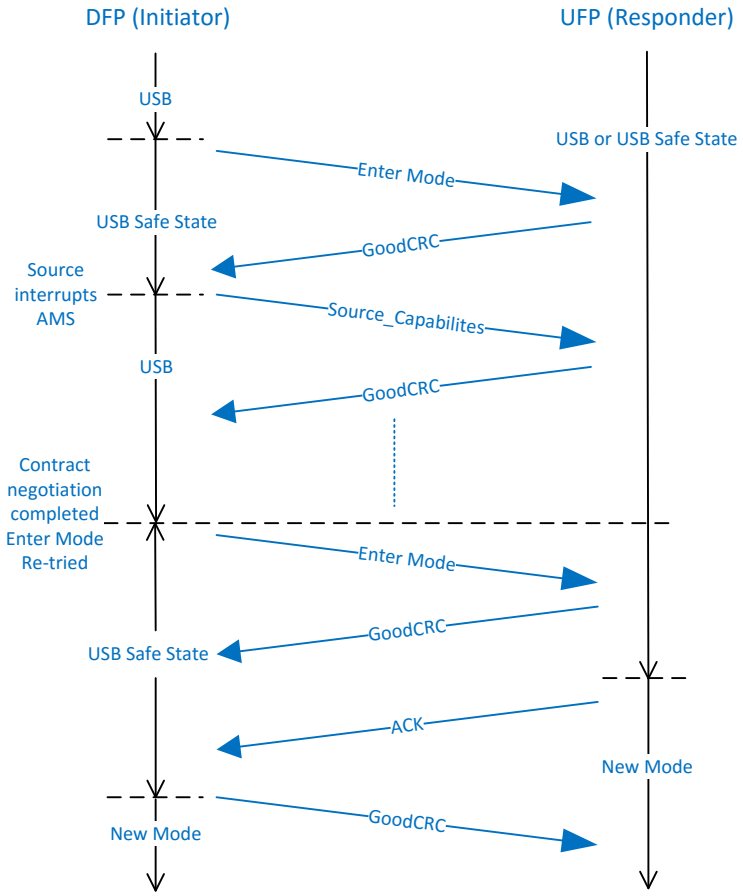
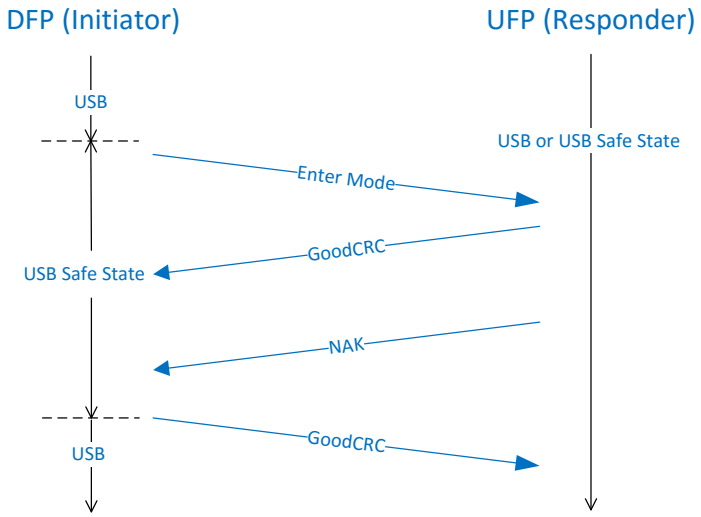


Figure 6-22 Unsuccessful Enter Mode sequence due to NAK



Microchip Ex. 1020, p. 142
Microchip v. Aptiv
IPR2024-00496

Once the Mode is entered, the device shall remain in that Active Mode until the ***Exit Mode*** Command is successful (see Section 6.4.4.3.5).

The following events shall also cause the Port Partners and Cable Plug(s) to exit all Active Modes:

- A PD Hard Reset
- The Port Partners or Cable Plug(s) are Detached
- A Cable Reset (only exits the Cable Plug's Active Modes)

The Initiator shall return to USB Operation within ***tVDMExitMode*** of a disconnect or of ***Hard Reset*** Signaling being detected.

The Responder shall return to either USB operation or USB Safe State within ***tVDMExitMode*** of a disconnect or of ***Hard Reset*** Signaling being detected.

A ***DR_Swap*** Message shall not be sent during Modal Operation between the Port Partners (see Section 6.3.9).

6.4.4.3.5 Exit Mode Command

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 is allowed to initiate the Exit Mode Process.

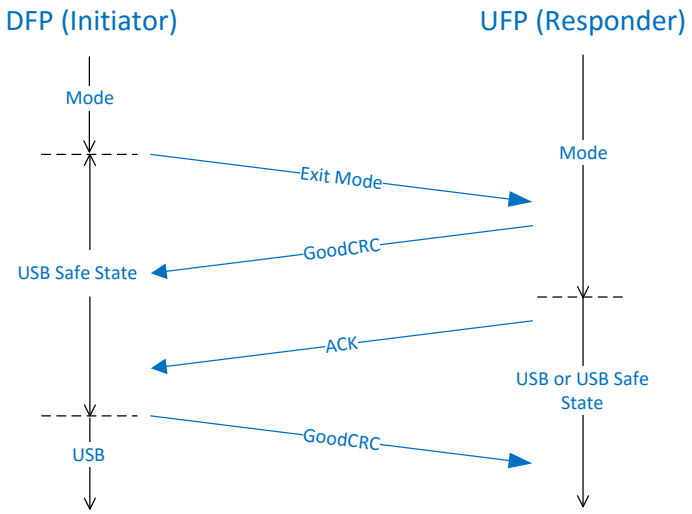
The value in the Object Position field shall indicate to which Mode in the ***Discover Modes*** Command the VDO refers (see Figure 6-19) and shall have been used previously in an ***Enter Mode*** Command request for an Active Mode. The value 1 always indicates the first Mode as it is the first object following the VDM Header. The value 2 refers to the next 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, BUSY) shall be set to 1 since an ***Exit Mode*** Command shall not contain any VDOs.

The Responder shall exit its Active Mode before sending the response Message. The Initiator shall exit its Active Mode before sending ***GoodCRC*** Message in response to the ACK. Receipt of the ***GoodCRC*** Message confirms to the Responder that the Initiator has exited the Mode. The Responder shall not return a BUSY acknowledgement and shall only return a NAK acknowledgement 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 6-23 shows the sequence of events during the transition between exiting an Active Mode and USB operation. It illustrates when the Responder's Mode changes and when the Initiator's Mode changes.

Figure 6-23 Exit Mode sequence



Microchip Ex. 1020, p. 143
Microchip v. Aptiv
IPR2024-00496

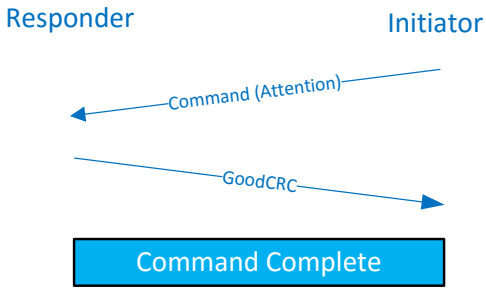
6.4.4.3.6 Attention

The ***Attention*** Command may be used by the Initiator to notify the Responder that it requires service.

The value in the Object Position field shall indicate to which Mode in the ***Discover Modes*** Command the VDO refers (see Figure 6-19) and shall have been used previously in an ***Enter Mode*** Command request for an Active Mode. The value 1 always indicates the first Mode as it is the first object following the VDM Header. The value 2 refers to the next Mode and so forth. A value of 000b or 111b in the Object Position field shall not be used by the ***Attention*** Command.

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 Mode.

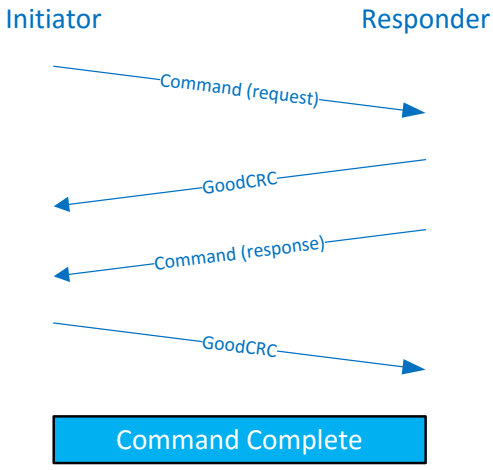
Figure 6-24 Attention Command request/response sequence



6.4.4.4 Command Processes

The Message flow of Commands during a Process is a query followed by a response. Every Command request sent has to be responded to with a ***GoodCRC*** Message. The ***GoodCRC*** Message only indicates the Command request was received correctly; it does not mean that the Responder understood or even supports a particular SVID. Figure 6-25 shows the request/response sequence including the ***GoodCRC*** Messages.

Figure 6-25 Command request/response sequence



Microchip Ex. 1020, p. 144
Microchip v. Aptiv
IPR2024-00496

In order for the Initiator to know that the Command request was actually consumed, it needs an acknowledgement from the Responder. There are three responses that indicate the Responder received and processed the Command request:

- ACK
- NAK
- BUSY

The Responder shall complete:

- Enter Mode requests within ***tVDMEnterMode***
- Exit Mode requests within ***tVDMExitMode***
- Other requests within ***tVDMReceiverResponse***,

An Initiator not receiving a response within the following times shall 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***,

The Responder shall respond with:

- ACK if it recognizes the SVID and can process it at this time
- NAK
    - if it recognizes the SVID but cannot process the Command request
    - or if it does not recognize the SVID
    - or if it does not support the Command
    - or if a VDO has been supplied which is ***Invalid***
- BUSY if it recognizes the SVID and the Command but cannot process the Command request at this time

The ACK, NAK or BUSY response shall contain the same SVID as the Command request.

6.4.4.4.1 Discovery Process

The Initiator (usually the DFP) always begins the Discovery Process. The Discovery Process has two phases. In the first phase, the ***Discover SVIDs*** Command request is sent by the Initiator to get the list of SVIDs the Responder supports. In the second phase, the Initiator sends a ***Discover Modes*** Command request for each SVID supported by both the Initiator and Responder.

6.4.4.4.2 Enter Vendor Mode / Exit Vendor Mode Processes

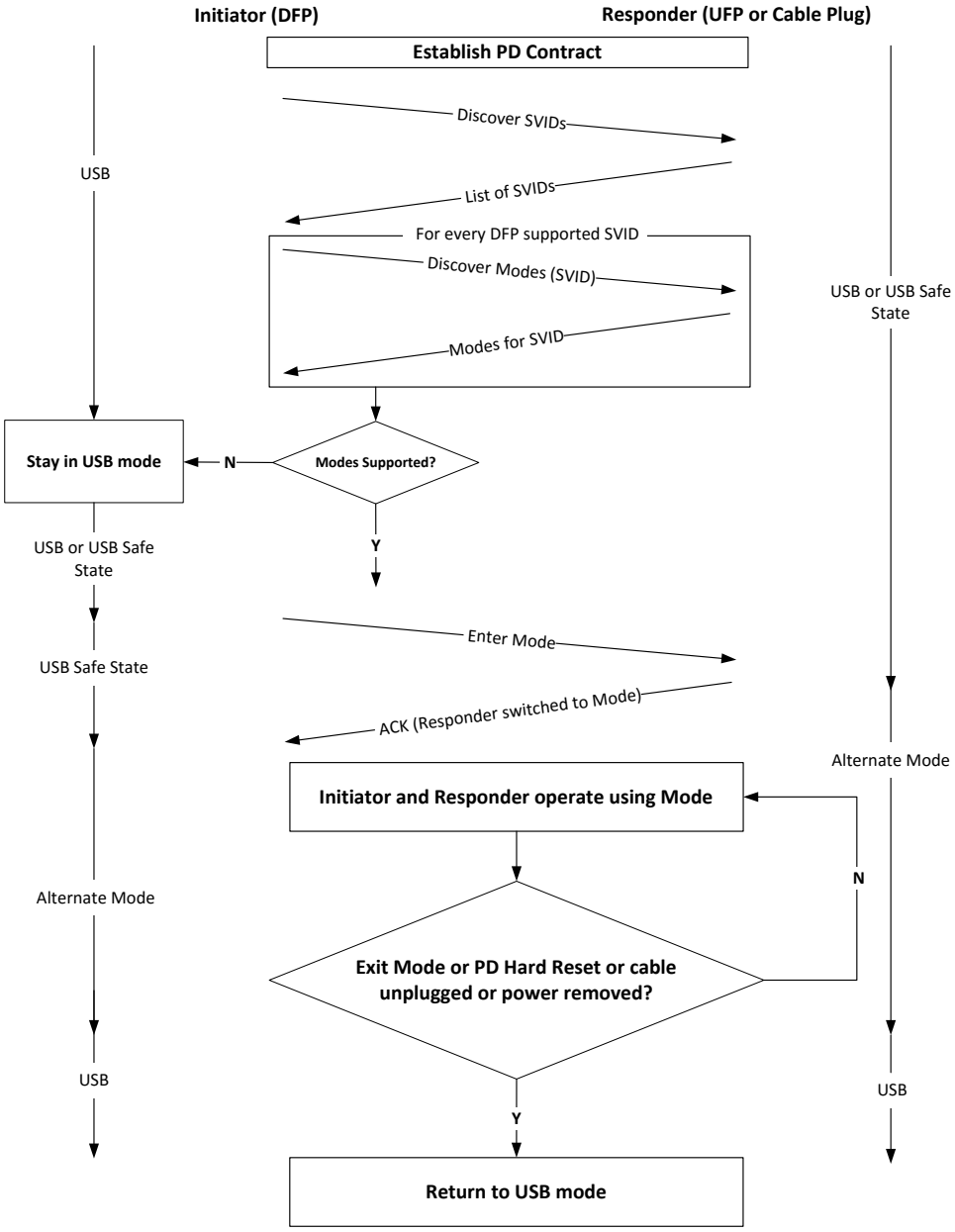
The result of the Discovery Process is that both the Initiator and Responder identify the Modes they mutually support. The Initiator (DFP), upon finding a suitable Mode, uses the ***Enter Mode*** Command to enable the Mode.

The Responder (UFP or Cable Plug) and Initiator continue using the Active Mode until the Active Mode is exited. In a managed termination, using the ***Exit Mode*** Command, the Active Mode shall be exited in a controlled manner as described in Section 6.4.4.3.5. In an unmanaged termination, triggered by a Power Delivery Hard Reset (i.e. ***Hard Reset*** Signaling sent by either Port Partner) or by cable Detach (device unplugged), the Active Mode shall still be exited but there may not be a transition through the USB Safe State. In both the managed and unmanaged terminations the Initiator and Responder return to USB operation as defined in ***[USB Type-C 1.2]*** following an exit from a Mode.

The overall Message flow is illustrated in Figure 6-26.

Microchip Ex. 1020, p. 145
Microchip v. Aptiv
IPR2024-00496

Figure 6-26 Enter/Exit Mode Process



6.4.4.5 VDM Message Timing and Normal PD Messages

Any Command Process or other VDM sequence may be interrupted by any other USB PD Message. The Vendor or Standards defined state operation shall comprehend this and continue to operate as expected when processing any other USB PD Messages.

The timing and interspersing of VDMs between regular PD Messages shall be done without perturbing the PD Message sequences. This requirement shall apply to both Unstructured VDMs and Structured VDMs.

The use of Structured VDMs by an Initiator shall not interfere with the normal PD Message timing requirements nor shall either the Initiator or Responder interrupt a PD Message sequence (e.g. Power Negotiation, Power Role Swap, Data Role Swap etc.). The use of Unstructured VDMs shall not interfere with normal PD Message timing.

Microchip Ex. 1020, p. 146
Microchip v. Aptiv
IPR2024-00496

VDM sequences shall be interruptible after the return of a ***GoodCRC*** Message has been completed. In the case where there is an error in transmission of the ***Vendor_Defined*** Message, as for any other PD Message, the ***Vendor_Defined*** Message will not be retried, but instead the incoming Message will be processed by the Policy Engine. This means that the ***Vendor_Defined*** Message sequence will need to be Re-run after the USB PD Message sequence has completed.

6.4.5 Battery_Status Message

The ***Battery_Status*** Message shall be sent in response to a ***Get_Battery_Status*** Message. The ***Battery_Status*** Message contains one Battery Status Data Object (BSDO) for one of the Batteries it supports as reported by Battery field in the ***Source_Capabilities_Extended*** Message. The returned BSDO shall correspond to the Battery requested in the ***Battery Status Ref*** field contained in the ***Get_Battery_Status*** Message.

The ***Battery_Status*** Message returns a BSDO whose format shall be as shown in Figure 6-27 and Table 6-35. The ***Number of Data Objects*** field in the ***Battery_Status*** Message shall be set to 1.

Figure 6-27 Battery_Status Message

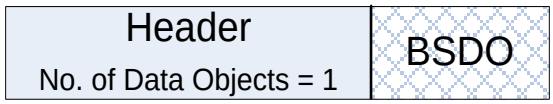


Table 6-35 Battery Status Data Object (BSDO)

Bit(s)	Field	Size	Value	Description
B31..16	Battery PC	2	1/10 WH	Battery's State of Charge (SOC) Note: 0xFFFF = Battery's SOC unknown
B15..8	Battery Info	1	Bit Field	<table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>Invalid Battery reference</td></tr><tr><td>1</td><td>Battery is present when set</td></tr><tr><td>3..2</td><td>When Battery is present shall contain the Battery charging status: 00b: Battery is Charging 01b: Battery is Discharging 10b: Battery is Idle 11b: ***Reserved***, shall not be used When Battery is not present: 11b..00b: ***Reserved***, shall not be used</td></tr><tr><td>7..4</td><td>***Reserved*** and shall be set to zero</td></tr></table>
B7..0	***Reserved***	2	Numeric	Shall be set to zero

6.4.5.1 Battery's State of Charge

The Battery Present Capacity field shall return either the Battery's present capacity in tenths of WH or indicate that the Battery's present State of Charge (SOC) is unknown.

6.4.5.2 Battery Info

The Battery Info field shall be used to report additional information about the Battery's present status. The Battery Info field's bits shall reflect the present conditions under which the Battery is operating in the systems.

Microchip Ex. 1020, p. 147
Microchip v. Aptiv
IPR2024-00496

6.4.5.2.1 Invalid Battery Reference

The Invalid Battery Reference bit shall be set when the ***Get_Battery_Status*** Message contains a reference to a Battery that does not exist.

6.4.5.2.2 Battery is Present

The Battery is Present bit shall be set whenever the Battery is present. It shall always be set for Batteries that are not Hot Swappable Batteries. For Hot Swappable Batteries, Battery is Present bit shall indicate whether the Battery is attached or detached.

6.4.5.2.3 Battery Charging Status

The Battery charging status bits indicate whether the Battery is being charged, discharged or is idle (neither charging nor discharging). These bits shall be set when the Battery is present bit is set. Otherwise when the Battery is present bit is zero the Battery charging status bits shall also be zero.

6.4.6 Alert 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 informative such as change in a Battery's status from charging to neither charging nor discharging.

The ***Alert*** Message shall only be sent when the Source or Sink detects a status change.

The ***Alert*** Message shall contain exactly one Alert Data Object (ADO) and the format shall be as shown in Figure 6-28 and Table 6-36.

Figure 6-28 Alert Message



Table 6-36 Alert Data Object

Bit(s)	Field	Value	Description
B31..24	***Type of Alert***	Bit Field	<table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>***Reserved*** and shall be set to zero</td></tr><tr><td>1</td><td>Battery Status Change Event(attach/detach/charging/discharging/idle)</td></tr><tr><td>2</td><td>OCP event when set (Source only, for Sink ***Reserved*** and shall be set to zero)</td></tr><tr><td>3</td><td>OTP event when set</td></tr><tr><td>4</td><td>Operating Condition Change when set</td></tr><tr><td>5</td><td>Source Input Change Event when set</td></tr><tr><td>6</td><td>OVP event when set (Sink only, for Source ***Reserved*** and shall be set to zero)</td></tr><tr><td>7</td><td>***Reserved*** and shall be set to zero</td></tr></table>
B23..20	***Fixed Batteries***	Bit Field	When Battery Status Change bit set indicates which Fixed Batteries have had a status change. B20 corresponds to Battery 0 and B23 corresponds to Battery 3.
B19..16	***Hot Swappable Batteries***	Bit Field	When Battery Status Change bit set indicates which Hot Swappable Batteries have had a status change. B16 corresponds to Battery 0 and B19 corresponds to Battery 3.
B15..0	***Reserved***	n/a	Shall be set to zero

Microchip Ex. 1020, p. 148
Microchip v. Aptiv
IPR2024-00496

6.4.6.1 Type of Alert

The ***Type of Alert*** field shall be used to report Source or Sink status changes. Only one ***Alert*** Message shall be generated for each Event or Change; however multiple Type of Alert bits may be set in one ***Alert*** Message. Once the ***Alert*** Message has been sent the ***Type of Alert*** field shall be cleared.

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.

A ***Get_Status*** Message should be sent in response to a non-Battery status change in an ***Alert*** Message from to get the details of the change.

6.4.6.1.1 Battery Status Change

The Battery Status Change bit shall be set when any Battery's power state changes between charging, discharging, neither. For Hot Swappable Batteries, it shall also be set when a Battery is attached or detached.

6.4.6.1.2 Over-Current Protection Event

The Over-Current Protection Event bit shall be set when a Source detects its output current exceeds its limits triggering its protection circuitry. This bit is ***Reserved*** for a Sink.

6.4.6.1.3 Over-Temperature Protection Event

The Over-Temperature Protection Event bit shall be set when a Source or Sink shuts down due to over-temperature triggering its protection circuitry.

6.4.6.1.4 Operating Condition Change

The Operating Condition Change bit shall be set when a Source or Sink detects its Operating Condition enters or exits either the 'orange' or 'red' temperature states.

6.4.6.1.5 Source Input Change Event

The Source Input Event bit shall be set when the Source/Sink's input changes. For example when the AC input is removed and the Source/Sink continues to be powered from one or more of its batteries or when AC returns and the Source/Sink transitions from Battery to AC operation or when the Source/Sink changes operation from one (or more) Battery to another (or more) Battery.

6.4.6.1.1 Over-Voltage Protection Event

The Over-Voltage Protection Event bit shall be set when the Sink detects its output voltage exceeds its limits triggering its protection circuitry. This bit is ***Reserved*** for a Source.

6.4.6.2 Fixed Batteries

The ***Fixed Batteries*** field indicates which Fixed Batteries have had a status change. B20 corresponds to Battery 0 and B23 corresponds to Battery 3.

Once the ***Alert*** Message has been sent the ***Fixed Batteries*** field shall be cleared.

6.4.6.3 Hot Swappable Batteries

The ***Hot Swappable Batteries*** field indicates which Hot Swappable Batteries have had a status change. B16 corresponds to Battery 0 and B19 corresponds to Battery 3.

Once the ***Alert*** Message has been sent the ***Hot Swappable Batteries*** field shall be cleared.

Microchip Ex. 1020, p. 149
Microchip v. Aptiv
IPR2024-00496

6.5 Extended Message

An Extended Message shall contain an Extended Message Header (indicated by the ***Extended*** field in the Message Header being set) and be followed by zero or more data bytes.

The format of the Extended Message is defined by the Message Header's ***Message Type*** field and is summarized in Table 6-37. The Sent by column indicates entities which may send the given Message (Source, Sink or Cable Plug); entities not listed shall not issue the corresponding Message. The Valid Start of Packet column indicates the Messages which shall only be issued in SOP Packets and the Messages which may be issued in SOP* Packets.

Table 6-37 Extended Message Types

Bits 4..0	Type	Sent by	Description	Valid Start of Packet
0 0000	***Reserved***		All values not explicitly defined are ***Reserved*** and shall not be used.	
0 0001	***Source_Capabilities_Extended***	Source or Dual-Role Power	See Section 6.5.1	SOP only
0 0010	***Status***	Source only	See Section 6.5.2	SOP only
0 0011	***Get_Battery_Cap***	Source or Sink	See Section 6.5.3	SOP only
0 0100	***Get_Battery_Status***	Source or Sink	See Section 6.5.4	
0 0101	***Battery_Capabilities***	Source or Sink	See Section 6.5.5	SOP only
0 0110	***Get_Manufacturer_Info***	Source or Sink	See Section6.5.6	SOP*
0 0111	***Manufacturer_Info***	Source, Sink or Cable Plug	See Section 6.5.7	SOP*
0 1000	***Security_Request***	Source or Sink	See Section 6.5.8.1	SOP*
0 1001	***Security_Response***	Source, Sink or Cable Plug	See Section 6.5.8.2	SOP*
0 1010	***Firmware_Update_Request***	Source or Sink	See Section 6.5.9.1	SOP*
0 1011	***Firmware_Update_Response***	Source, Sink or Cable Plug	See Section 6.5.9.2	SOP*
0 1100 - 1 1111	***Reserved***		All values not explicitly defined are ***Reserved*** and shall not be used.	

6.5.1 Source_Capabilities_Extended Message

The ***Source_Capabilities_Extended*** Message should be sent in response to a ***Get_Source_Cap_Extended*** Message. The ***Source_Capabilities_Extended*** Message enables a Source or a DRP to inform the Sink about its capabilities as a Source.

The ***Source_Capabilities_Extended*** Message shall return a 23-byte Source Capabilities Extended Data Block (SCEDB) whose format shall be as shown in Figure 6-29 and Table 6-38.

Figure 6-29 Source_Capabilities_Extended Message



Microchip Ex. 1020, p. 150
Microchip v. Aptiv
IPR2024-00496

Table 6-38 Source Capabilities Extended Data Block (SCEDB)

<table>
<tr><th>Offset</th><th>Field</th><th>Size</th><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>VID</td><td>2</td><td>Numeric</td><td>Vendor ID (assigned by the USB-IF)</td></tr>
<tr><td>2</td><td>PID</td><td>2</td><td>Numeric</td><td>Product ID (assigned by the manufacturer)</td></tr>
<tr><td>4</td><td>XID</td><td>4</td><td>Numeric</td><td>Value provided by the USB-IF to assign to product</td></tr>
<tr><td>8</td><td>FW Version</td><td>1</td><td>Numeric</td><td>Firmware version number</td></tr>
<tr><td>9</td><td>HW Version</td><td>1</td><td>Numeric</td><td>Hardware version number</td></tr>
<tr><td>10</td><td>Voltage Regulation</td><td>1</td><td>Bit Field</td><td><table><tr><th>Bit</th><th>Description</th></tr><tr><td>1..0</td><td>00b: 150mA/us Load Step (default)
11b..01b: Reserved and shall not be used</td></tr><tr><td>2</td><td>0b: 25% IoC (default)
1b: 90% IoC</td></tr><tr><td>3..7</td><td>Reserved and shall be set to zero</td></tr></table></td></tr>
<tr><td>11</td><td>Holdup Time</td><td>1</td><td>Numeric</td><td>Output will stay with regulated limits for this number of milliseconds after removal of the AC from the input.
0x00 = feature not supported
Note: a value of 3ms should be used</td></tr>
<tr><td>12</td><td>Compliance</td><td>1</td><td>Bit Field</td><td><table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>LPS compliant when set</td></tr><tr><td>1</td><td>PS1 compliant when set</td></tr><tr><td>2</td><td>PS2 compliant when set</td></tr><tr><td>3..7</td><td>Reserved and shall be set to zero</td></tr></table></td></tr>
<tr><td>13</td><td>Touch Current</td><td>1</td><td>Bit Field</td><td><table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>Low touch Current EPS when set</td></tr><tr><td>1</td><td>Ground pin supported
when set</td></tr><tr><td>2</td><td>Ground pin intended for protective earth
when set</td></tr><tr><td>3..7</td><td>Reserved and shall be set to zero</td></tr></table></td></tr>
<tr><td>14</td><td>Peak Current1</td><td>2</td><td>Bit field</td><td><table><tr><th>Bit</th><th>Description</th></tr><tr><td>0..4</td><td>Percent overload in 10% increments
Values higher than 25 (11001b) are clipped to 250%.</td></tr><tr><td>5..10</td><td>Overload period in 20ms</td></tr><tr><td>11.14</td><td>Duty cycle in 5% increments</td></tr><tr><td>15</td><td>V_{BUS} Voltage droop</td></tr></table></td></tr>
</table>

Microchip Ex. 1020, p. 151
Microchip v. Aptiv
IPR2024-00496

<table>
<tr><th>Offset</th><th>Field</th><th>Size</th><th>Value</th><th>Description</th></tr>
<tr><td>16</td><td>Peak Current2</td><td>2</td><td>Bit field</td><td><table><tr><th>Bit</th><th>Description</th></tr><tr><td>0..4</td><td>Percent overload in 10% increments
Values higher than 25 (11001b) are clipped to 250%.</td></tr><tr><td>5..10</td><td>Overload period in 20ms</td></tr><tr><td>11.14</td><td>Duty cycle in 5% increments</td></tr><tr><td>15</td><td>V_{BUS} Voltage droop</td></tr></table></td></tr>
<tr><td>18</td><td>Peak Current3</td><td>2</td><td>Bit field</td><td><table><tr><th>Bit</th><th>Description</th></tr><tr><td>0..4</td><td>Percent overload in 10% increments
Values higher than 25 (11001b) are clipped to 250%.</td></tr><tr><td>5..10</td><td>Overload period in 20ms</td></tr><tr><td>11.14</td><td>Duty cycle in 5% increments</td></tr><tr><td>15</td><td>V_{BUS} Voltage droop</td></tr></table></td></tr>
<tr><td>20</td><td>Touch Temp</td><td>1</td><td>Value</td><td>Temperature conforms to:
0 = [IEC 60950-1] (default)
1 = [IEC 62368-1] TS1
2 = [IEC 62368-1] TS2
Note: All other values Reserved</td></tr>
<tr><td>21</td><td>Source Inputs</td><td>1</td><td>Bit field</td><td><table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>0b: No external supply
1b: External supply present</td></tr><tr><td>1</td><td>If bit 0 is set:

0b: External supply is constrained
1b: External supply is unconstrained

If bit 0 is not set Reserved and shall be set to zero</td></tr><tr><td>2</td><td>0b: No internal Battery
1b: Internal Battery present</td></tr><tr><td>3..7</td><td>Reserved and shall be set to zero</td></tr></table></td></tr>
<tr><td>22</td><td>Batteries</td><td>1</td><td>Byte</td><td>Upper Nibble = Number of Hot Swappable Batteries (0..4)
Lower Nibble = Number of Fixed Batteries (0..4)</td></tr>
</table>

6.5.1.1 Vendor ID (VID) Field

The Vendor ID field shall contain the 16-bit Vendor ID (VID) assigned to the Source's vendor by the USB-IF. If the vendor does not have a VID, the Vendor ID field shall be set to zero. Devices that have a USB data interface shall report the same VID as the idVendor in the Standard Device Descriptor (see ***[USB 2.0]*** and ***[USB 3.1]***).

Microchip Ex. 1020, p. 152
Microchip v. Aptiv
IPR2024-00496

6.5.1.2 Product ID (PID) Field

The Product ID field shall contain the 16-bit Product ID (VID) assigned by the Source's vendor. Devices that have a USB data interface shall report the same PID as the idProduct in the Standard Device Descriptor (see ***[USB 2.0]*** and ***[USB 3.1]***).

6.5.1.1 XID Field

The XID field shall contain the 32-bit XID provided by the USB-IF to the vendor who in turns assigns it to a product. If the vendor does not have an XID, then it shall return zero in this field (see ***[USB 2.0]*** and ***[USB 3.1]***).

6.5.1.2 Firmware Version Field

The Firmware Version field shall contain an 8-bit firmware version number assigned to the device by the vendor.

6.5.1.3 Hardware Version Field

The Hardware Version field shall contain an 8-bit hardware version number assigned to the device by the vendor.

6.5.1.4 Voltage Regulation Field

The Voltage Regulation field contains bits covering Load Step Slew Rate and Magnitude as well as the value to be use for ***pSnkStdby***.

See Section 7.1.13.1 for further details.

6.5.1.4.1 Load Step Slew Rate

The Source shall report its load step response capability in bits 0..1 of the Voltage Regulation bit field.

6.5.1.4.2 Load Step Magnitude – Bit 3

The Source shall report its load step magnitude rate as a percentage of IoC in bit 2 of the Voltage Regulation field.

6.5.1.5 Holdup Time Field

The Holdup Time field shall contain the Source's holdup time (see Section 7.1.13.2).

6.5.1.6 Compliance Field

The Compliance field shall contain the standards the Source is compliant with (see Section 7.1.13.3).

6.5.1.7 Touch Current

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.

A Source with a ground pin shall set the Ground pin bit (bit-1).

A Source whose Ground pin is intended to be connected to a protective earth shall set both bit1 and bit 2.

6.5.1.8 Peak Current

The Peak Current field shall contain the combinations of Peak Current that the Source supports (see Section 7.1.13.4).

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

Microchip Ex. 1020, p. 153
Microchip v. Aptiv
IPR2024-00496

defined as PeakCurrent1, PeakCurrent2 and PeakCurrent3 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:

- **Percentage Overload** shall be the maximum peak current reported in 10% increments as a percentage of the negotiated operating current (IoC) offered by the Source. Values higher than 25 (11001b) are clipped to 250%.
- **Overload Period** shall be the minimum rolling average time window in 20ms increments, where a value of 20ms is recommended.
- **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.
- **V_{BUS} Droop** shall be set to one to indicate there is an additional 5% voltage droop on V_{BUS} when the overload conditions occur. However, it is recommended that the Source should provide V_{BUS} in the range of ***vSrcNew*** when overload conditions occur and set this bit to zero.

6.5.1.9 Touch Temp

The Touch Temp field shall report 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]***.

6.5.1.10 Source Inputs

The Source Inputs field shall identify the possible inputs that provide power to the Source. Note some Sources may only be powered by a Battery (e.g. an automobile) rather than the more common mains.

- When bit 0 is set, the Source can be sourced by an external power supply.
- When bits 0 and 1 are set, the Source can be sourced by an external power supply which is assumed to be effectively "infinite" i.e. it won't run down over time.
- When bit 2 is set the Source can be sourced by an internal Battery

Bit 2 may be set independently of bits 0 and 1.

6.5.1.11 Batteries

The Batteries field shall report the number of batteries the source supports. It shall independently report the number of Hot Swappable Batteries and the number of Fixed batteries. The maximum number of each type of Battery shall be no more than 4.

6.5.2 Status Message

The ***Status*** Message shall be sent in response to a ***Get_Status*** Message. The ***Status*** Message enables a Port to inform its Port Partner about the present status of the Source or Sink. Typically a ***Get_Status*** Message will be sent by the Port after receipt of an ***Alert*** Message. Some of the reported events are critical such as OCP, OVP and OTP, while others may informative such as change in a Battery's status from charging to neither charging nor discharging.

The ***Status*** Message returns a 3-byte Status Data Block (SDB) whose format shall be as shown in Figure 6-30 and Table 6-39.

Figure 6-30 Status Message



Microchip Ex. 1020, p. 154
Microchip v. Aptiv
IPR2024-00496

Table 6-39 Status Data Block (SSDB)

<table>
<tr><th>Offset (Byte)</th><th>Field</th><th>Size</th><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>Internal Temp</td><td>1</td><td>Numeric</td><td>Source or Sink's internal temperature in degrees centigrade.
0 = feature not supported
1 = temperature is less than 2°C.
2-255 = temperature in °C.</td></tr>
<tr><td>1</td><td>Present Input</td><td>1</td><td>Bit field</td><td><table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>Reserved and shall be set to zero</td></tr><tr><td>1</td><td>External Power when set</td></tr><tr><td>2</td><td>External Power AC/DC (valid when Bit 1 set)
0: DC
1: AC</td></tr><tr><td>3</td><td>Internal Power from Battery when set</td></tr><tr><td>4</td><td>Internal Power from non-Battery power source when set</td></tr><tr><td>7</td><td>Reserved and shall be set to zero</td></tr></table></td></tr>
<tr><td>2</td><td>Present Battery Input</td><td>1</td><td>Bit field</td><td>When Present Input field bit 3 set shall contain the bit corresponding to the Battery or Batteries providing power:

Upper nibble = Hot Swappable Battery (b7..4)
Lower nibble = Fixed Battery (b3..0)

When Present Source Input field bit 3 is not set this field is Reserved and shall be set to zero.</td></tr>
</table>

6.5.2.1 Internal Temp

The Internal Temp field reports the instantaneous temperature of a portion of the Source.

6.5.2.2 Present Input

The Present Input field indicates which supplies are presently powering the Source or Sink.

The following bits are defined:

- Bit 1 indicates that an external Source is present. This bit shall match the Externally Powered bit in the ***Source_Capabilities*** Message.
- Bit 2 indicates whether the external unconstrained Source is AC or DC.
- Bit 3 indicates that power is being provided from Battery.
- Bit4 indicates an alternative internal source of power that is not a Battery.

6.5.2.3 Present Battery Input

The Present Battery Input field indicates which Battery or Batteries are presently supplying power to the Source or Sink. The Present Battery Input field is only valid when the Present Input field indicates that there is Internal Power from Battery.

The upper nibble of the field indicates which Hot Swappable Battery/Batteries are supplying power with bit 0 in upper nibble corresponding to Battery 0 and bit 0 in the upper nibble corresponding to Battery 4 (see Section 6.5.3 and Section 6.5.4).

Microchip Ex. 1020, p. 155
Microchip v. Aptiv
IPR2024-00496

The lower nibble of the field indicates which Fixed Battery/Batteries are supplying power with bit 0 in lower nibble corresponding to Battery 0 and bit 0 in the lower nibble corresponding to Battery 4 (see Section 6.5.3 and Section 6.5.4).

6.5.3 Get_Battery_Cap Message

The ***Get_Battery_Cap*** (Get Battery Capabilities) Message is used to request the capability of a Battery present in its Port Partner. The Port shall respond by returning a ***Battery_Capabilities*** Message (see Section 6.5.5) containing a Battery Capabilities Data Block (BCDB) for the targeted Battery.

The ***Get_Battery_Cap*** Message contains a 1 byte Get Battery Cap Data Block (GBCDB). This block defines for which Battery the request is being made.

The ***Get_Battery_Cap*** Message returns a GBCDB whose format shall be as shown in Figure 6-31 and Table 6-40. The ***Data Size*** field in the ***Get_Battery_Cap*** Message shall be set to 1.

Figure 6-31 Get_Battery_Cap Message



Table 6-40 Get Battery Cap Data Block (GBCDB)

Offset	Field	Size	Value	Description
0	***Battery Cap Ref***	1	Value	Number of the Battery indexed from zero: • Values 0..3 represent the Fixed Batteries. • Values 4..7 represent the Hot Swappable Batteries.

6.5.4 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. The port shall respond by returning a ***Battery_Status*** Message (see Section 6.4.5) containing a Battery Status Data Object (BSDO) for the targeted Battery.

The ***Get_Battery_Status*** Message contains a 1 byte Get Battery Cap Data Block (GBSDB). This block contains details of the requested Battery.

The ***Get_Battery_Status*** Message returns a GBSDB whose format shall be as shown in Figure 6-32 and Table 6-41. The ***Data Size*** field in the ***Get_Battery_Status*** Message shall be set to 1.

Figure 6-32 Get_Battery_Status Message



Table 6-41 Get Battery Status Data Block (GBSDB)

Offset	Field	Size	Value	Description
0	***Battery Status Ref***	1	Value	Number of the Battery indexed from zero: • Values 0..3 represent the Fixed Batteries. • Values 4..7 represent the Hot Swappable Batteries.

6.5.5 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 its supports as reported by

Microchip Ex. 1020, p. 156
Microchip v. Aptiv
IPR2024-00496

Battery 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.

The ***Battery_Capabilities*** Message returns a 9-byte BCDB whose format shall be as shown in Figure 6-33 and Table 6-39.

Figure 6-33 Battery_Capabilities Message

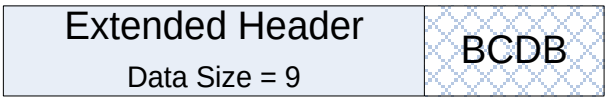


Table 6-42 Battery Capability Data Block (BCDB)

Offset (Byte)	Field	Size	Value	Description
0	VID	2	Numeric	Vendor ID (assigned by the USB-IF)
2	PID	2	Numeric	Product ID (assigned by the manufacturer)
4	Battery Design Capacity	2	1/10 WH	Battery's design capacity Note: 0x0000 = Battery not present 0xFFFF = design capacity unknown
6	Battery Last Full Charge Capacity	2	1/10 WH	Battery's last full charge capacity Note: 0x0000 = Battery not present 0xFFFF = last full charge capacity unknown
8	Battery Type	1	Bit Field	**Bit** / **Description** 0 – Invalid Battery reference 1 – Hot Swappable Battery when set 2-7 – ***Reserved***

6.5.5.1 Battery Design Capacity Field

The Battery Design Capacity field shall return the Battery's design capacity in tenths of WH. If the Battery is Hot Swappable and is not present, the Battery Design Capacity field shall be set to 0. If the Battery is unable to report its Design Capacity, it shall return 0xFFFF.

6.5.5.2 Battery Last Full Charge Capacity Field

The Battery Last Full Charge Capacity field shall return the Battery's last full charge capacity in tenths of WH. If the Battery is Hot Swappable and is not present, the Battery Last Full Charge Capacity field shall be set to 0. If the Battery is unable to report its Design Capacity, the Battery Last Full Charge Capacity field shall be set to 0xFFFF.

6.5.5.3 Battery Type Field

The Battery Type Field is used to report additional information about the Battery's capabilities.

6.5.5.3.1 Invalid Battery Reference

The Invalid Battery Reference bit shall be set when the ***Get_Battery_Cap*** Message contains a reference to a Battery that does not exist.

Microchip Ex. 1020, p. 157
Microchip v. Aptiv
IPR2024-00496

6.5.5.3.2 Hot Swappable Battery

The Hot Swappable Battery bit shall be set to 1 when the system supports a Hot Swappable Battery. The Hot Swappable Battery bit shall be set to 0 if the Battery is Hot Swappable.

6.5.6 Get_Manufacturer_Info Message

The ***Get_Manufacturer_Info*** (Get Manufacturer Info) Message is sent by a Port to request manufacturer specific information relating to its Port Partner or Cable Plug or of a Battery behind a Port. The Port or Cable Plug shall respond by returning a ***Manufacturer_Info*** Message (Section 6.5.7) containing a Manufacturer Info Data Block (MIDB).

The ***Get_Manufacturer_Info*** Message contains a 2-byte Get Manufacturer Info Data Block (GMIDB). This block defines whether it is the Device or Battery manufacturer information being requested and for which Battery the request is being made.

The ***Get_Manufacturer_Info*** Message returns a GMIDB whose format shall be as shown in Figure 6-32 and Table 6-43.

Figure 6-34 Get_Manufacturer_Info Message



Table 6-43 Get Manufacturer Info Data Block (GMIDB)

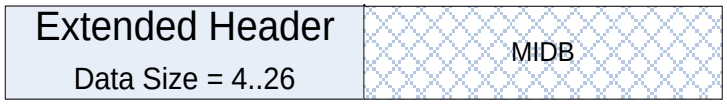
Offset	Field	Size	Value	Description
0	***Manufacturer Info Target***	1	Value	0: Port/Cable Plug 1: Battery 255..2: ***Reserved*** shall not be used.
1	***Manufacturer Info Ref***	1	Value	If ***Manufacturer Info Target*** subfield is Battery (01b) the ***Manufacturer Info Ref*** field shall contain the Battery number reference which is the number of the Battery indexed from zero: • Values 0..3 represent the Fixed Batteries. • Values 4..7 represent the Hot Swappable Batteries. Otherwise this field is **Reserved** and shall be set to zero.

6.5.7 Manufacturer_Info Message

The ***Manufacturer_Info*** Message shall be sent in response to a ***Get_Manufacturer_Info*** Message. The ***Manufacturer_Info*** Message contains the USB VID and the Vendor's PID to identify the device or Battery and the device or Battery's manufacturer byte array in a variable length Data Block of up to ***MaxExtendedMsgLegacyLen*** .

The ***Manufacturer_Info*** Message returns a Manufacturer Info Data Block (MIDB) whose format shall be as shown in Figure 6-33 and Table 6-39.

Figure 6-35 Manufacturer_Info Message



Microchip Ex. 1020, p. 158
Microchip v. Aptiv
IPR2024-00496

Table 6-44 Manufacturer Info Data Block (MIDB)

Offset	Field	Size	Value	Description
0	VID	2	Numeric	Vendor ID (assigned by the USB-IF)
2	PID	2	Numeric	Product ID (assigned by the manufacturer)
4	Manufacturer String	0..22	String	Vendor defined byte array If the Manufacturer Info Target field or Manufacturer Info Ref field in the ***Get_Manufacturer_Info*** Message is unrecognized return zero bytes.

6.5.7.1 Vendor ID (VID)

This field shall contain the device's or Battery's16-bit vendor ID assigned by the USB.

6.5.7.2 Product ID (PID)

This field shall contain the device's or Battery's 16-bit product identifier designated by the vendor.

6.5.7.3 Manufacturer String

This field shall contain the device's or Battery's manufacturer string as defined by the vendor. If the ***Manufacturer Info Target*** field or ***Manufacturer Info Ref*** field in the ***Get_Manufacturer_Info*** Message is unrecognized the field shall return zero bytes.

6.5.8 Security Messages

The authentication process between Port Partners or a Port and Cable Plug is fully described in ***[USBTypeCAuthentication 1.0]***. This specification describes two Extended Messages used by the authentication process when applied to PD.

In the authentication process described in ***[USBTypeCAuthentication 1.0]*** 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.

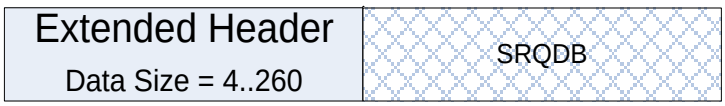
For further details refer to ***[USBTypeCAuthentication 1.0]***.

6.5.8.1 Security_Request

The ***Security_Request*** Message is used by a Port to pass a security data structure to its Port Partner or a Cable Plug.

The ***Security_Request*** Message contains a Security Request Data Block (SRQDB) whose format shall be as shown in Figure 6-36. The contents of the SRQDB and its use are defined in ***[USBTypeCAuthentication 1.0]***.

Figure 6-36 Security_Request Message



Microchip Ex. 1020, p. 159
Microchip v. Aptiv
IPR2024-00496

6.5.8.2 Security_Response

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 ***Security_Response*** Message contains a Security Response Data Block (SRPDB) whose format shall be as shown in Figure 6-37. The contents of the SRPDB and its use are defined in ***[USBTypeCAuthentication 1.0]***.

Figure 6-37 Security_Response Message



6.5.9 Firmware Update Messages

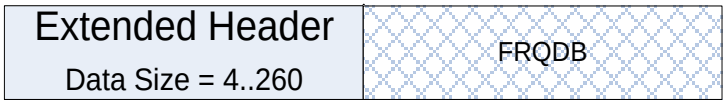
The firmware update process between Port Partners or a Port and Cable Plug is fully described in ***[USBPDFirmwareUpdate 1.0]***. This specification describes two Extended Messages used by the firmware update process when applied to PD.

6.5.9.1 Firmware_Update_Request

The ***Firmware_Update_Request*** Message is used by a Port to pass a firmware update data structure to its Port Partner or a Cable Plug.

The ***Firmware_Update_Request*** Message contains a Firmware Update Request Data Block (FRQDB) whose format shall be as shown in Figure 6-38. The contents of the FRQDB and its use are defined in ***[USBPDFirmwareUpdate 1.0]***.

Figure 6-38 Firmware_Update_Request Message

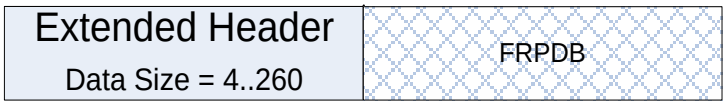


6.5.9.2 Firmware_Update_Response

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_Response*** Message contains a Firmware Update Response Data Block (FRPDB) whose format shall be as shown in Figure 6-39. The contents of the FRPDB and its use are defined in ***[USBPDFirmwareUpdate 1.0]***.

Figure 6-39 Firmware_Update_Response Message



6.6 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.

6.6.1 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 acknowledgement of a Message (a ***GoodCRC*** Message) whether caused by a bad CRC on the receiving end or by a garbled Message within ***tReceive*** is detected when the ***CRCReceiveTimer*** expires.

Microchip Ex. 1020, p. 160
Microchip v. Aptiv
IPR2024-00496

The sender's Protocol Layer response when a ***CRCReceiveTimer*** expires shall be to retry ***nRetryCount*** times. Note: that Cable Plugs do not retry Messages and large Extended Messages that are not Chunked are not retried (see Section 6.7.2). 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 Physical Layer. The ***CRCReceiveTimer*** shall be stopped when the last bit of the ***EOP*** corresponding to the ***GoodCRC*** Message has been received by the Physical 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*** shall be measured from when the last bit of the Message ***EOP*** has been received by the Physical Layer until the first bit of the Preamble of the ***GoodCRC*** Message has been transmitted by the Physical Layer.

6.6.2 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.

The Policy Engine's response when the ***SenderResponseTimer*** expires shall be dependent on the Message sent (see Section 8.3).

The ***SenderResponseTimer*** shall be started from the time the last bit of the ***GoodCRC*** Message ***EOP*** (i.e. the ***GoodCRC*** Message corresponding to the Message requesting a response) has been received by the Physical Layer. The ***SenderResponseTimer*** shall be stopped when the last bit of the expected response Message ***EOP*** has been received by the Physical 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 Message ***EOP*** has been received by the Physical Layer until the first bit of the response Message Preamble has been transmitted by the Physical Layer.

6.6.3 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.

6.6.3.1 SourceCapabilityTimer

Prior to a successful negotiation 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 Physical Layer since a PD connection has been established. At this point the Source waits for a ***Request*** Message or a response timeout.

See Section 8.3.3.2 more details of when ***Source_Capabilities*** Messages are transmitted.

6.6.3.2 SinkWaitCapTimer

The Sink shall support the ***SinkWaitCapTimer***. When a Sink observes an absence of ***Source_Capabilities*** Messages, after V_{BUS} is present, 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 6.7.4).

See Section 8.3.3.3 for more details of when the ***SinkWaitCapTimer*** are run.

Microchip Ex. 1020, p. 161
Microchip v. Aptiv
IPR2024-00496

6.6.3.3 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 V_{BUS} reaching ***vSafe5V***. This ensures that the Sink receives a ***Source_Capabilities*** Message before the Sink's ***SinkWaitCapTimer*** expires.

6.6.4 SinkRequestTimer

The ***SinkRequestTimer*** is used to ensure that the time between Sink ***Request*** Messages, after a ***Wait*** Message has been received from the Source, is a minimum of ***tSinkRequest*** (see Section 6.3.12).

The ***SinkRequestTimer*** shall be started when 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 a ***Wait*** Message, before issuing 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***.

6.6.5 Power Supply Timers

6.6.5.1 PSTransitionTimer

The ***PSTransitionTimer*** is used by the Policy Engine to timeout on a ***PS_RDY*** Message. It is started when a request for a new Capability 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 7.1).

The ***PSTransitionTimer*** shall be started when the last bit of an ***Accept*** or ***GotoMin*** Message ***EOP*** has been received by the Physical Layer. The ***PSTransitionTimer*** shall be stopped when the last bit of the ***PS_RDY*** Message ***EOP*** has been received by the Physical Layer.

6.6.5.2 PSSourceOffTimer

6.6.5.2.1 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 sequence. 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 ***EOP*** of the ***GoodCRC*** Message corresponding to this ***Accept*** Message is received by the Sink, 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 ***EOP*** of this ***Accept*** Message is received by the Sink then the ***PSSourceOffTimer*** shall be started.

The ***PSSourceOffTimer*** shall be stopped when:

- The last bit of the ***EOP*** of the ***PS_RDY*** Message is received.

The ***PSSourceOffTimer*** relates to the time taken for the remote Dual-Role Power Device to stop supplying power (see also Section 7.3.9 and Section 7.3.10). 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.

6.6.5.2.2 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 sequence. This condition leads to USB Type-C Error Recovery.

Microchip Ex. 1020, p. 162
Microchip v. Aptiv
IPR2024-00496

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 ***EOP*** of the ***GoodCRC*** Message corresponding to this ***Accept*** Message is received by the initial Sink, then the ***PSSourceOffTimer*** shall be started.

The ***PSSourceOffTimer*** shall be stopped when:

- The last bit of the ***EOP*** of the ***PS_RDY*** Message is received.

The ***PSSourceOffTimer*** relates to the time taken for the initial Source to stop supplying power and for V_{BUS} to revert to ***vSafe5V*** (see also Section 7.2.10 and Section 7.3.15). The timer shall time out if a ***PS_RDY*** Message has not been received from the initial Source within ***tPSSourceOff*** indicating this has occurred.

6.6.5.3 PSSourceOnTimer

6.6.5.3.1 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 ***EOP*** of the ***GoodCRC*** Message corresponding to the transmitted ***PS_RDY*** Message is received by the Physical Layer

The ***PSSourceOnTimer*** shall be stopped when:

- The last bit of the ***EOP*** of the ***PS_RDY*** Message is received by the Physical Layer

The ***PSSourceOnTimer*** relates to the time taken for the remote Dual-Role Power Device to start sourcing power (see also Section 7.3.9 and Section 7.3.10) and will time out if a ***PS_RDY*** Message indicating this has not been received within ***tPSSourceOn***.

6.6.5.3.2 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 ***EOP*** of the ***GoodCRC*** Message corresponding to the transmitted ***PS_RDY*** Message is received by the Physical Layer

The ***PSSourceOnTimer*** shall be stopped when:

- The last bit of the ***EOP*** of the ***PS_RDY*** Message is received by the Physical Layer

The ***PSSourceOnTimer*** relates to the time taken for the remote Dual-Role Power Device to start sourcing power (see also Section 7.2.10 and Section 7.3.15) and will time out if a ***PS_RDY*** Message indicating this has not been received within ***tPSSourceOn***.

6.6.6 NoResponseTimer

The ***NoResponseTimer*** is used by the Policy Engine in a Source or Sink 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 Sink fails to receive a ***Source_Capabilities*** Message received 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

Microchip Ex. 1020, p. 163
Microchip v. Aptiv
IPR2024-00496

- 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 Sink shall issue additional Hard Resets up to ***nHardResetCount*** times (see Section 6.8.2).

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 6.8.2).

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.

6.6.7 BIST Timers

6.6.7.1 tBISTMode

tBISTMode is used to define the maximum time that a UUT has to enter a BIST mode when requested by a Tester.

A UUT shall enter the appropriate BIST mode within ***tBISTMode*** of the last bit of the ***EOP*** of the ***BIST*** Message used to initiate the test is received by the Physical Layer. In ***BIST Carrier Mode*** when transmitting a continuous carrier signal transmission shall start as soon as the UUT enters BIST mode.

6.6.7.2 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 the last bit of the bit of the ***EOP*** of ***GoodCRC*** Message sent in response to the ***BIST*** Message used to enable the Continuous BIST Mode.

6.6.8 Power Role Swap Timers

6.6.8.1 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.

6.6.9 Soft Reset Timers

6.6.9.1 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.

Microchip Ex. 1020, p. 164
Microchip v. Aptiv
IPR2024-00496

6.6.9.2 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*** sent in response to the Message that caused the Protocol Error.

6.6.10 Hard Reset Timers

6.6.10.1 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.

6.6.10.2 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 V_{BUS}.

When a Hard Reset occurs the Source stops driving V_{CONN}, removes Rp from the V_{CONN} pin and starts to transition the V_{BUS} 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 7.1.6.

6.6.10.3 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*** sent in response to the ***DR_Swap*** Message.

6.6.10.4 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*** sent in response to the Message that caused the Protocol Error.

6.6.11 Structured VDM Timers

6.6.11.1 VDMResponseTimer

The ***VDMResponseTimer*** shall be used by the Initiator's Policy Engine to ensure that a Structured VDM Command request needing a response (e.g. ***Discover Identity*** Command request) is responded to within a bounded time of ***tVDMSenderResponse***. The ***VDMResponseTimer*** shall be applied to all Structured VDM Commands except the ***Enter Mode*** and ***Exit Mode*** Commands which have their own timers (***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 be dependent on the Message sent (see Section 8.3).

The ***VDMResponseTimer*** shall be started from the time the last bit of the ***GoodCRC*** Message ***EOP*** (i.e. the ***GoodCRC*** Message corresponding to the VDM Command requesting a response) has been received by the Physical Layer. The ***VDMResponseTimer*** shall be stopped when the last bit of the expected VDM Command response ***EOP*** has been received by the Physical Layer.

Microchip Ex. 1020, p. 165
Microchip v. Aptiv
IPR2024-00496

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 received by the Physical Layer until the first bit of the response Message Preamble has been transmitted by the Physical Layer.

6.6.11.2 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 Mode has been entered) arrives within a bounded time of ***tVDMWaitModeEntry***. 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 8.3.3.19.1).

The ***VDMModeEntryTimer*** shall be started from the time the last bit of the ***GoodCRC*** Message ***EOP*** (i.e. the ***GoodCRC*** Message corresponding to the VDM Command request) has been received by the Physical Layer. The ***VDMModeEntryTimer*** shall be stopped when the last bit of the expected Structured VDM Command response (ACK, NAK or BUSY) ***EOP*** has been received by the Physical 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 Message ***EOP*** has been received by the Physical Layer until the first bit of the response Message Preamble has been transmitted by the Physical Layer.

6.6.11.3 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 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 8.3.3.19.2).

The ***VDMModeExitTimer*** shall be started from the time the last bit of the ***GoodCRC*** Message ***EOP*** (i.e. the ***GoodCRC*** Message corresponding to the VDM Command requesting a response) has been received by the Physical Layer. The ***VDMModeExitTimer*** shall be stopped when the last bit of the expected Structured VDM Command response ACK ***EOP*** has been received by the Physical 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 Physical Layer until the first bit of the response Message Preamble has been transmitted by the Physical Layer.

6.6.11.4 tVDMBusy

The Initiator shall wait at least ***tVDMBusy***, after receiving a BUSY Command response, before repeating the Structured VDM request again.

6.6.12 VCONN Timers

6.6.12.1 VCONNOnTimer

The ***VCONNOnTimer*** is used during a VCONN Swap.

The ***VCONNOnTimer*** shall be started when:

- The last bit of the ***EOP*** of the ***Accept*** Message is received.

Microchip Ex. 1020, p. 166
Microchip v. Aptiv
IPR2024-00496

- The last bit of the ***EOP*** of ***GoodCRC*** Message corresponding to the ***Accept*** Message is received.

The ***VCONNOnTimer*** shall be stopped when:

- The last bit of the ***EOP*** of the ***PS_RDY*** Message is received.

Prior to sending the ***PS_RDY*** Message, the Port shall have turned VCONN On.

6.6.12.2 tVCONNSourceOff

The ***tVCONNSourceOff*** time applies during a Vconn Swap. The initial VCONN Source shall cease sourcing VCONN within ***tVCONNSourceOff*** of receipt of the last bit of the ***EOP*** of the ***PS_RDY*** Message.

6.6.13 tCableMessage

Ports compliant with this Revision of the specification shall not wait ***tCableMessage*** before sending an SOP' or SOP'' Packet even when communicating using ***[USBPD 2.0]*** 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' or SOP'' Packet when operating at ***[USBPD 2.0]***. When operating at Revisions higher than ***[USBPD 2.0]*** Cable Plugs shall not wait ***tCableMessage*** before sending an SOP' or SOP'' Packet.

6.6.14 DiscoverIdentityTimer

The ***DiscoverIdentityTimer*** is used 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.

6.6.15 Collision Avoidance Timers

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***.

6.6.16 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 5.8.6.3).

6.6.17 Time Values and Timers

Table 6-45 summarizes the values for the timers listed in this section. For each Timer Value, a given implementation shall pick a fixed value within the range specified. Table 6-46 lists the timers.

Microchip Ex. 1020, p. 167
Microchip v. Aptiv
IPR2024-00496

Table 6-45 Time Values

Parameter	Value (min)	Value (max)	Units	Reference
tBISTContMode	30	60	ms	Section 6.6.7.2
tBISTMode		300	ms	Section 6.6.7.1
tCableMessage	750		µs	Section 6.6.13
tDiscoverIdentity	40	50	ms	Section 6.6.13
tDRSwapHardReset		15	ms	Section 6.6.10.3
tFirstSourceCap		250	ms	Section 6.6.3.3
tFRSwapInit		15	ms	Section 6.3.17
tHardReset		5	ms	Section 6.3.13
tHardResetComplete	4	5	ms	Section 6.6.9
tNoResponse	4.5	5.5	s	Section 6.6.6
tProtErrHardReset		5	ms	Section 6.6.10.4
tProtErrSoftReset		15	ms	Section 6.6.9.2
tPSHardReset	25	35	ms	Section 6.6.10.2
tPSSourceOff	750	920	ms	Section 6.6.5.2
tPSSourceOn	390	480	ms	Section 6.6.5.3
tPSTransition	450	550	ms	Section 6.6.5.1
tReceive	0.9	1.1	ms	Section 6.6.1
tReceiverResponse		15	ms	Section 6.6.2
tRetry		75	µs	Section 6.6.1
tSenderResponse	24	30	ms	Section 6.6.2
tSinkRequest	100		ms	Section 6.6.4
tSinkTx	16	20	ms	Section 6.6.15
tSoftReset		15	ms	Section 6.8.1
tSwapSinkReady		15	ms	Section 6.6.8.1
tSwapSourceStart	20		ms	Section 6.6.8.1
tTransmit		195	µs	Section 6.6.1
tTypeCSendSourceCap	100	200	ms	Section 6.6.3.1
tTypeCSinkWaitCap	310	620	ms	Section 6.6.3.2
tVCONNSourceOff		25	ms	Section 6.6.12
tVCONNSourceOn		100	ms	Section 6.6.12
tVDMBusy	50		ms	Section 6.6.11.4
tVDMEnterMode		25	ms	Section 6.6.11.2
tVDMExitMode		25	ms	Section 6.6.11.3
tVDMReceiverResponse		15	ms	Section 6.6.11.1
tVDMSenderResponse	24	30	ms	Section 6.6.11.1
tVDMWaitModeEntry	40	50	ms	Section 6.6.11.2
tVDMWaitModeExit	40	50	ms	Section 6.6.11.3

Microchip Ex. 1020, p. 168
Microchip v. Aptiv
IPR2024-00496

Table 6-46 Timers

Timer	Parameter	Used By	Reference
BISTContModeTimer	*tBISTContMode*	Policy Engine	Section 6.6.7.2
CRCReceiveTimer	*tReceive*	Protocol	Section 6.6.1
DiscoverIdentityTimer	*tDiscoverIdentity*	Policy Engine	Section 6.6.14
HardResetCompleteTimer	*tHardResetComplete*	Protocol	Section 6.6.9
NoResponseTimer	*tNoResponse*	Policy Engine	Section 6.6.6
PSHardResetTimer	*tPSHardReset*	Policy Engine	Section 6.6.10.2
PSSourceOffTimer	*tPSSourceOff*	Policy Engine	Section 6.6.5.2
PSSourceOnTimer	*tPSSourceOn*	Policy Engine	Section 6.6.5.3
PSTransitionTimer	*tPSTransition*	Policy Engine	Section 6.6.5.1
SenderResponseTimer	*tSenderResponse*	Policy Engine	Section 6.6.2
SinkRequestTimer	*tSinkRequest*	Policy Engine	Section 6.6.4
SinkWaitCapTimer	*tTypeCSinkWaitCap*	Policy Engine	Section 6.6.3.2
SourceCapabilityTimer	*tTypeCSendSourceCap*	Policy Engine	Section 6.6.3.1
SinkTxTimer	*tSinkTx*	Protocol Layer	Section 6.6.15
SwapSourceStartTimer	*tSwapSourceStart*	Policy Engine	Section 6.6.8.1
VCONNOnTimer	*tVCONNSourceOn*	Policy Engine	Section 6.6.12.1
VDMModeEntryTimer	*tVDMWaitModeEntry*	Policy Engine	Section 6.6.11.2
VDMModeExitTimer	*tVDMWaitModeExit*	Policy Engine	Section 6.6.11.3
VDMResponseTimer	*tVDMSenderResponse*	Policy Engine	Section 6.6.11.1

Microchip Ex. 1020, p. 169
Microchip v. Aptiv
IPR2024-00496

6.7 Counters

6.7.1 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.

6.7.1.1 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***.

6.7.1.2 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.

6.7.2 Retry Counter

The ***RetryCounter*** is used by a Port whenever there is a Message transmission failure (timeout of ***CRCReceiveTimer***). 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 also Section 6.11.2.1):

- 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) shall be retried
- Extended Messages of ***Data Size*** > ***MaxExtendedMsgLegacyLen*** that are Chunked (***Chunked*** flag set to one) individual Chunks shall be retried

Microchip Ex. 1020, p. 170
Microchip v. Aptiv
IPR2024-00496

6.7.3 Hard Reset Counter

The ***HardResetCounter*** is used to retry the Hard Reset whenever there is no response from the remote device (see Section 6.6.6). Once the Hard Reset has been retried ***nHardResetCount*** times then it shall be assumed that the remote device is non-responsive.

6.7.4 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 ***SinkWaitCapTimer*** times out in order to reset the Source. Resetting the Source shall also reset the ***CapsCounter*** and restart the sending of ***Source_Capabilities*** Messages.

6.7.5 Discover Identity Counter

When sending ***Discover Identity*** Messages to a Cable Plug a Port shall maintain a count of Messages sent (***DiscoverIdentityCounter***). No more than ***nDiscoverIdentityCount Discover Identity*** Messages shall be sent by the Port receiving a ***GoodCRC*** Message response. A Data Role Swap shall reset the ***DiscoverIdentityCounter*** to zero.

6.7.6 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 Type-C 1.2]*** requirements for Mode entry should use an ***nBusyCount*** of 1.

6.7.7 Counter Values and Counters

Table 6-48 lists the counters used in this section and Table 6-47 shows the corresponding parameters.

Table 6-47 Counter parameters

Parameter	Value	Reference
nBusyCount	5	Section 6.7.6
nCapsCount	50	Section 6.7.4
nDiscoverIdentityCount	20	Section 6.7.5
nHardResetCount	2	Section 6.7.3
nMessageIDCount	7	Section 6.7.1
nRetryCount	2	Section 6.7.2

Microchip Ex. 1020, p. 171
Microchip v. Aptiv
IPR2024-00496

Table 6-48 Counters

Counter	Max	Reference
CapsCounter	*nCapsCount*	Section 6.7.4
DiscoverIdentityCounter	*nDiscoverIdentityCount*	Section 6.7.5
HardResetCounter	*nHardResetCount*	Section 6.7.3
MessageIDCounter	*nMessageIDCount*	Section 6.7.1
RetryCounter	*nRetryCount*	Section 6.7.2
VDMBusyCounter	*nBusyCount*	Section 6.7.6

6.8 Reset

Resets are a necessary response to protocol or other error conditions. USB Power Delivery defines two different types of reset; a Soft Reset, that resets protocol, and a Hard Reset which resets both the power supplies and protocol.

6.8.1 Soft Reset and Protocol Error

A *Soft_Reset* Message is used to cause a Soft Reset of protocol communication when this has broken down in some way. It shall not have any impact on power supply operation, but is used to correct a Protocol Error occurring during an Atomic Message Sequence (AMS). The Soft Reset may be triggered by either Port Partner in response to the Protocol Error.

Protocol Errors are any unexpected Message during an AMS. If the first Message in an AMS has been passed to the Protocol Layer by the Policy Engine but has not yet been sent (*GoodCRC* Message not received) when the Protocol Error occurs, the Policy Engine shall not issue a Soft Reset but shall return to the *PE_SNK_Ready* or *PE_SRC_Ready* state and then process the incoming Message. If the Protocol Error occurs during an Interruptible AMS then the Policy Engine shall not issue a Soft Reset but shall return to the *PE_SNK_Ready* or *PE_SRC_Ready* state and then process the incoming Message. If the Protocol Error occurs during a Non-interruptible AMS this shall lead to a Soft Reset in order to re-synchronize the Policy Engine state machines (see Section 8.3.3.4) except when the voltage is transition when a Protocol Error shall lead to a Hard Reset (see Section 6.6.10.4 and Section 8.3.3.2). An unrecognized or unsupported Message (except for a Structured VDM), received in the *PE_SNK_Ready* or *PE_SRC_Ready* states, shall not cause a *Soft_Reset* Message to be generated but instead a *Not_Supported* Message shall be generated. Details of Interruptible and Non-interruptible AMS's can be found in Section 8.3.2.1.3.

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 6.6.9.1).

A Soft Reset shall impact the USB Power Delivery layers in the following ways:

- Physical Layer: Reset not required since the Physical 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: Shall not change

A Soft Reset is performed using a sequence of protocol Messages (see Table 8-7). Message numbers shall be set to zero prior to sending the *Soft_Reset*/*Accept* Message since the issue may be with the counters. The sender of a *Soft_Reset* Message shall reset its *MessageIDCounter* and *RetryCounter*, the receiver of the Message shall reset its *MessageIDCounter* and *RetryCounter* before sending the *Accept* Message response. Any failure in the Soft Reset process will trigger a Hard Reset when SOP Packets are being used or Cable Reset for any other SOP* Packets; for example a *GoodCRC* Message is not received during the Soft Reset process (see Section 6.8.2 and Section 6.8.3).

6.8.2 Hard Reset

Hard Resets are signaled by an ordered set as defined in Section 5.6.4. Both the sender and recipient shall cause their power supplies to return to their default states (see Section 7.3.12 and Section 7.3.13 for details of voltage transitions). In addition their respective Protocol Layers shall be reset as for the Soft Reset. This allows the Attached

Microchip Ex. 1020, p. 172
Microchip v. Aptiv
IPR2024-00496

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 6.6.6 and Section 6.7.3). Note: that even though V_{BUS} 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 Type-C 1.2]***) 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.

In effect the Hard Reset will revert the Ports to their default state based on their CC line resistors. Removing and reapplying VCONN from the Cable Plugs also ensures that they re-establish their configuration as either SOP' or SOP'' based on the location of VCONN (see ***[USB Type-C 1.2]***).

If the Hard Reset is insufficient to clear the error condition then the Port should use Error Recovery mechanisms as defined in ***[USB Type-C 1.2]***.

A Sink shall be able to send ***Hard Reset*** signaling regardless of the value of Rp (see Section 5.7).

6.8.2.1 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 8.3.3.21.2.2). The Cable Plugs shall perform the equivalent of a power cycle returning to their initial power up state. This allows the Attached products to be in a state where they can re-establish USB PD communication.

6.8.2.2 Modal Operation and Hard Reset

A Hard Reset shall cause all Active Modes to be exited by both Port Partners and any Cable Plugs (see Section 6.4.4.3.4).

6.8.3 Cable Reset

Cable Resets are signaled by an ordered set as defined in Section 5.6.5. 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 Attached products to be in a state where they can re-establish USB PD communication.

The DFP has to be supplying VCONN prior to a Cable Reset to ensure that the Cable Plugs correctly configure SOP' and SOP'' after the Cable Reset is complete. 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.

Only a DFP shall generate ***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 6.4.4.3.4).

6.9 Collision Avoidance

In order 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 and shall only send Messages that are part of an AMS the Source has initiated. Note that this restriction applies to SOP* AMS's 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.

Microchip Ex. 1020, p. 173
Microchip v. Aptiv
IPR2024-00496

6.10 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***.

See Table 6-49 for details of the Messages that shall/shall not be ***Discarded***.

Table 6-49 Message discarding

Message pending transmission	Message received	Discard pending transmission?
SOP	SOP	Yes
SOP	SOP'/SOP''	No
SOP'	SOP	Yes
SOP'	SOP'	No
SOP'	SOP''	No
SOP''	SOP	Yes
SOP''	SOP'	No
SOP''	SOP''	No

Microchip Ex. 1020, p. 174
Microchip v. Aptiv
IPR2024-00496

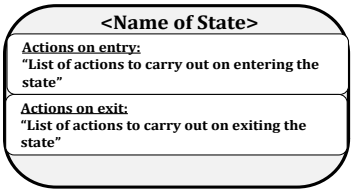
6.11 State behavior

6.11.1 Introduction to state diagrams used in Chapter 6

The state diagrams defined in Section 6.11 are ***Normative*** and shall define the operation of the Power Delivery protocol layer. Note that these state diagrams are not intended to replace a well written and robust design.

Figure 6-40 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 6-40 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 6-41 indicates how such references are made. The reference is indicated with a hatched box. The box contains the name of the referenced state.

Figure 6-41 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 particular 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.)

6.11.2 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 BIST State Machine instances, with their own counter and timer 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).

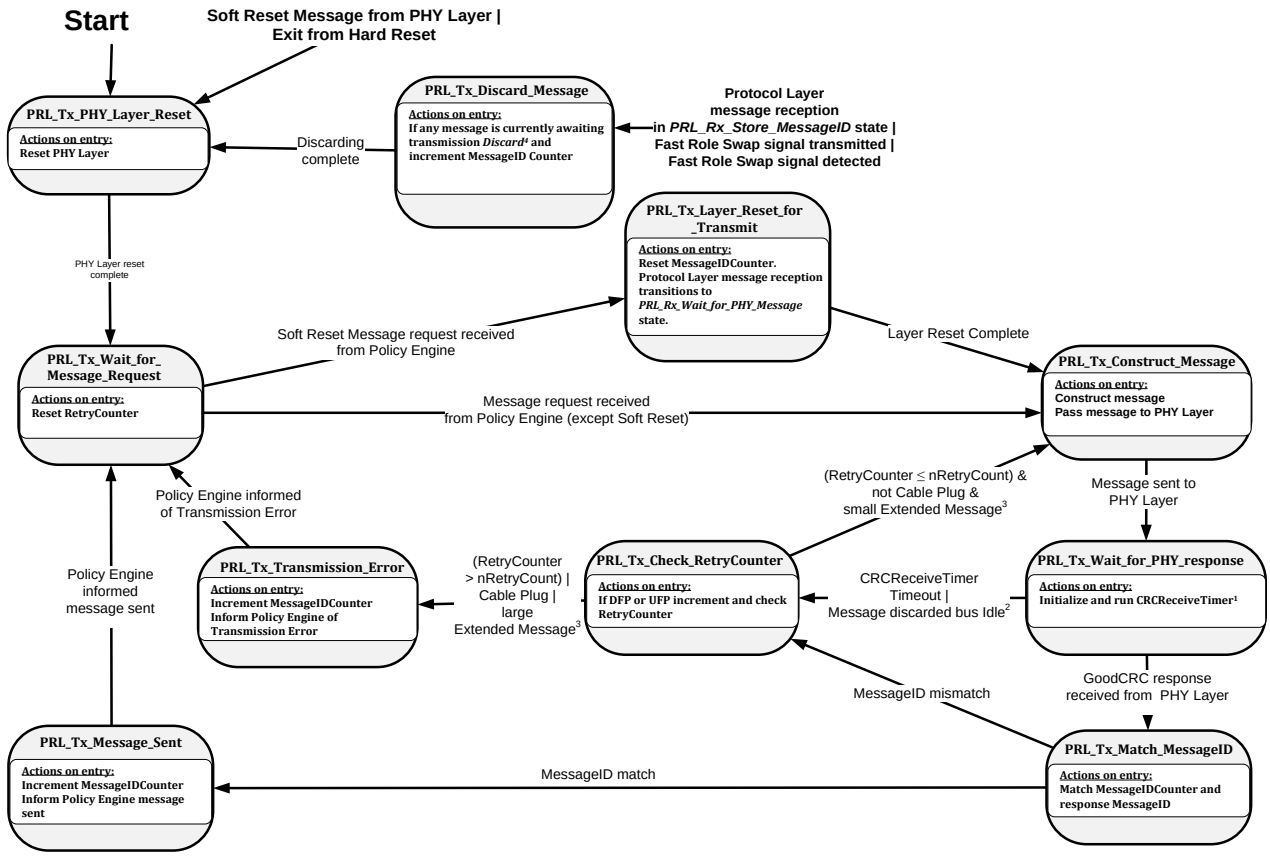
Microchip Ex. 1020, p. 175
Microchip v. Aptiv
IPR2024-00496

6.11.2.1 Protocol Layer Message Transmission

6.11.2.1.1 Common Protocol Layer Message Transmission State Diagram

Figure 6-42 shows the state behavior, common between the Source and the Sink, for the Protocol Layer when transmitting a Message.

Figure 6-42 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 6.6.1).

[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 5.7). 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 6.10 for details of when Messages are ***Discarded***.

6.11.2.1.1.1 PRL_Tx_PHY_Layer_Reset state

The Protocol Layer shall enter the ***PRL_Tx_PHY_Layer_Reset*** state:

Microchip Ex. 1020, p. 176
Microchip v. Aptiv
IPR2024-00496

- 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.

6.11.2.1.1.2 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.

6.11.2.1.1.3 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 6.11.2.2.1) 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.

6.11.2.1.1.4 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.

6.11.2.1.1.5 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 6.6.1).

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 5.7).

6.11.2.1.1.6 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.

Microchip Ex. 1020, p. 177
Microchip v. Aptiv
IPR2024-00496

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.

6.11.2.1.1.7 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.

6.11.2.1.1.8 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 that 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:

- ***RetryCounter*** ≤ ***nRetryCount*** and
- This is not a Cable Plug and
- This is an Extended Message with ***Data Size*** ≤ ***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.

6.11.2.1.1.9 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.

6.11.2.1.1.10 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 signal is being transmitted (see Section 5.8.5.6)
- The Fast Role Swap signal is detected (see Section 5.8.6.3).

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 6.10 and increment the ***MessageIDCounter***.

The Protocol Layer shall transition to the ***PRL_Tx_PHY_Layer_Reset*** state when:

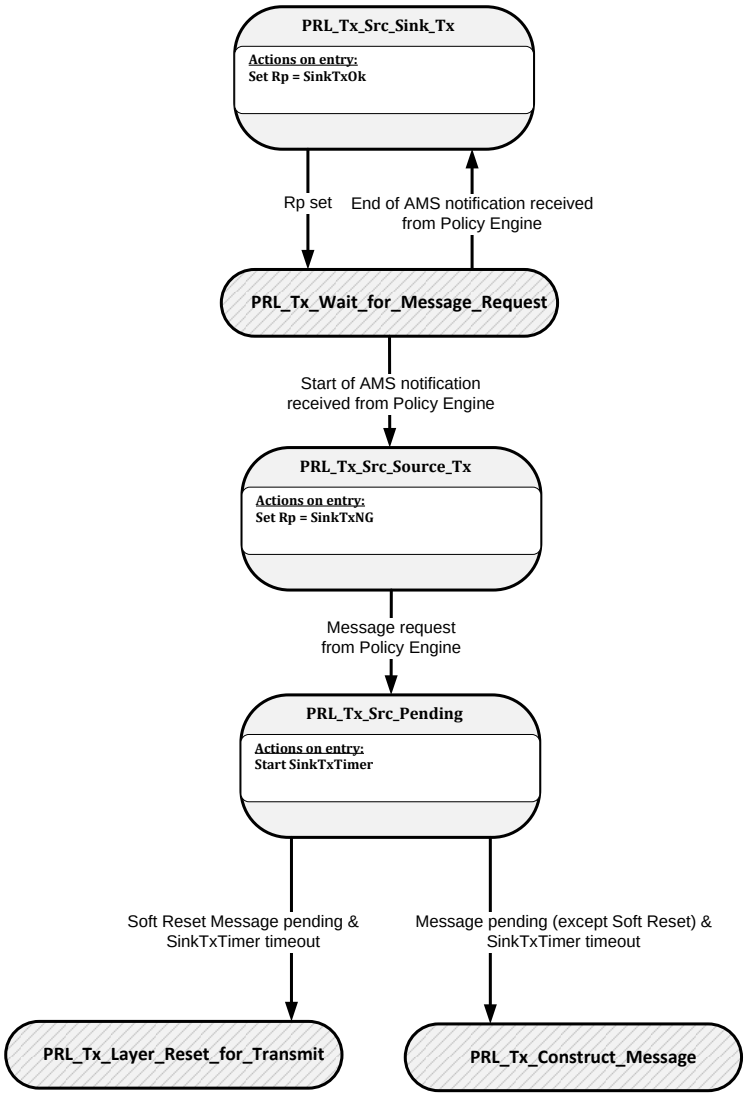
- Discarding is complete i.e. the Message queue is empty.

Microchip Ex. 1020, p. 178
Microchip v. Aptiv
IPR2024-00496

6.11.2.1.2 Source Protocol Layer Message Transmission State Diagram

Figure 6-43 shows the state behavior for the Protocol Layer in a Source when transmitting a Message.

Figure 6-43 Source Protocol Layer Message transmission State Diagram



6.11.2.1.2.1 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***.

Microchip Ex. 1020, p. 179
Microchip v. Aptiv
IPR2024-00496

The Protocol Layer shall transition to the ***PRL_Tx_Wait_for_Message_Request*** state when:

- Rp has been set

6.11.2.1.2.2 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

6.11.2.1.2.3 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:

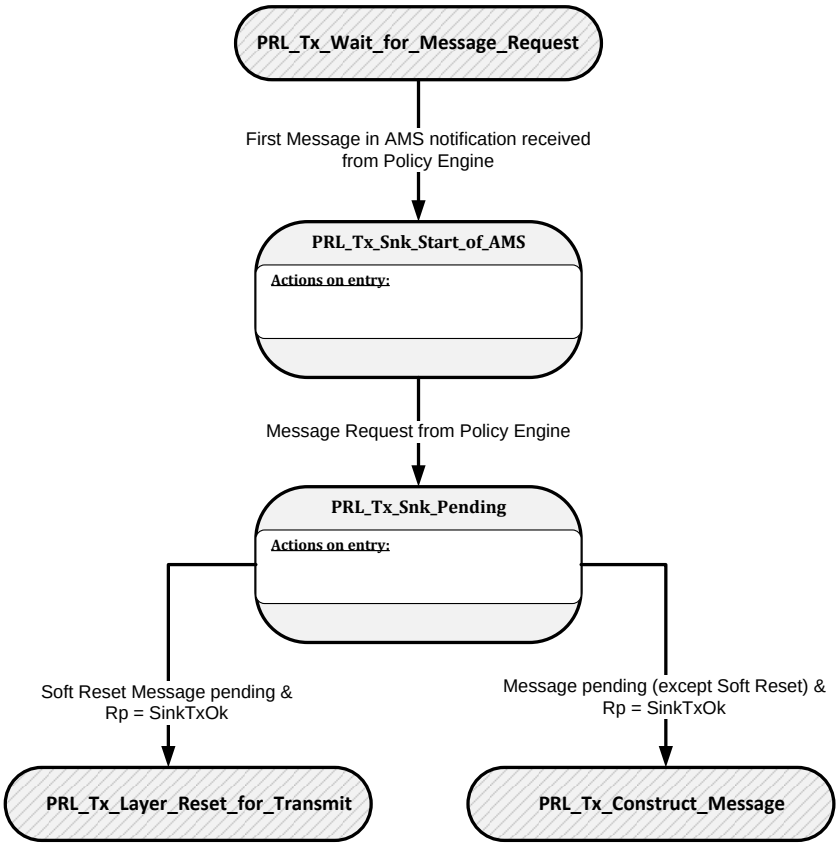
- The pending Message request from the Policy Engine is a ***Soft_Reset*** Message and
- The ***SinkTxTimer*** times out.

Microchip Ex. 1020, p. 180
Microchip v. Aptiv
IPR2024-00496

6.11.2.1.3 Sink Protocol Layer Message Transmission State Diagram

Figure 6-44 shows the state behavior for the Protocol Layer in a Source when transmitting a Message.

Figure 6-44 Sink Protocol Layer Message transmission State Diagram



6.11.2.1.3.1 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_Src_Pending*** state when:

- A Message request is received from the Policy Engine.

6.11.2.1.3.2 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 ***Soft_Reset*** Message is pending 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 and

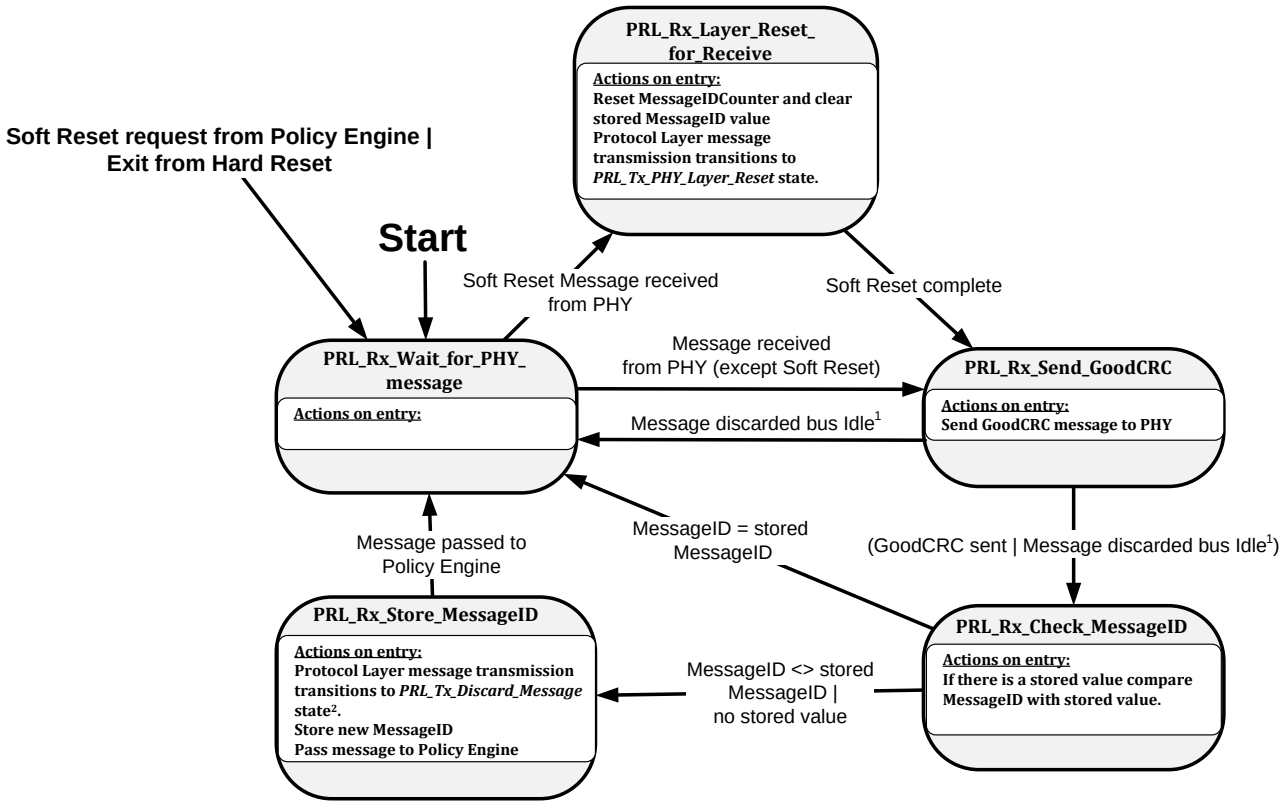
Microchip Ex. 1020, p. 181
Microchip v. Aptiv
IPR2024-00496

- Rp is set to ***SinkTxOk***.

6.11.2.2 Protocol Layer Message Reception

Figure 6-45 shows the state behavior for the Protocol Layer when receiving a Message.

Figure 6-45 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 5.7). 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***.

6.11.2.2.1 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.

Microchip Ex. 1020, p. 182
Microchip v. Aptiv
IPR2024-00496

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.

6.11.2.2.2 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 6.11.2.1.1.1).

The Protocol Layer shall transition to the ***PRL_Rx_Send_GoodCRC*** State when:

- The Soft Reset actions in this state have been completed.

6.11.2.2.3 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 5.7), the Protocol Layer shall either:

- Transition to the ***PRL_Rx_Check_MessageID state*** or
- Transition to the ***PRL_Rx_Wait_for_PHY_Message*** state

6.11.2.2.4 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

6.11.2.2.5 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 (except when a ***Ping*** Message has been received in which case the ***PRL_Tx_Discard_Message*** state should not be entered), 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:

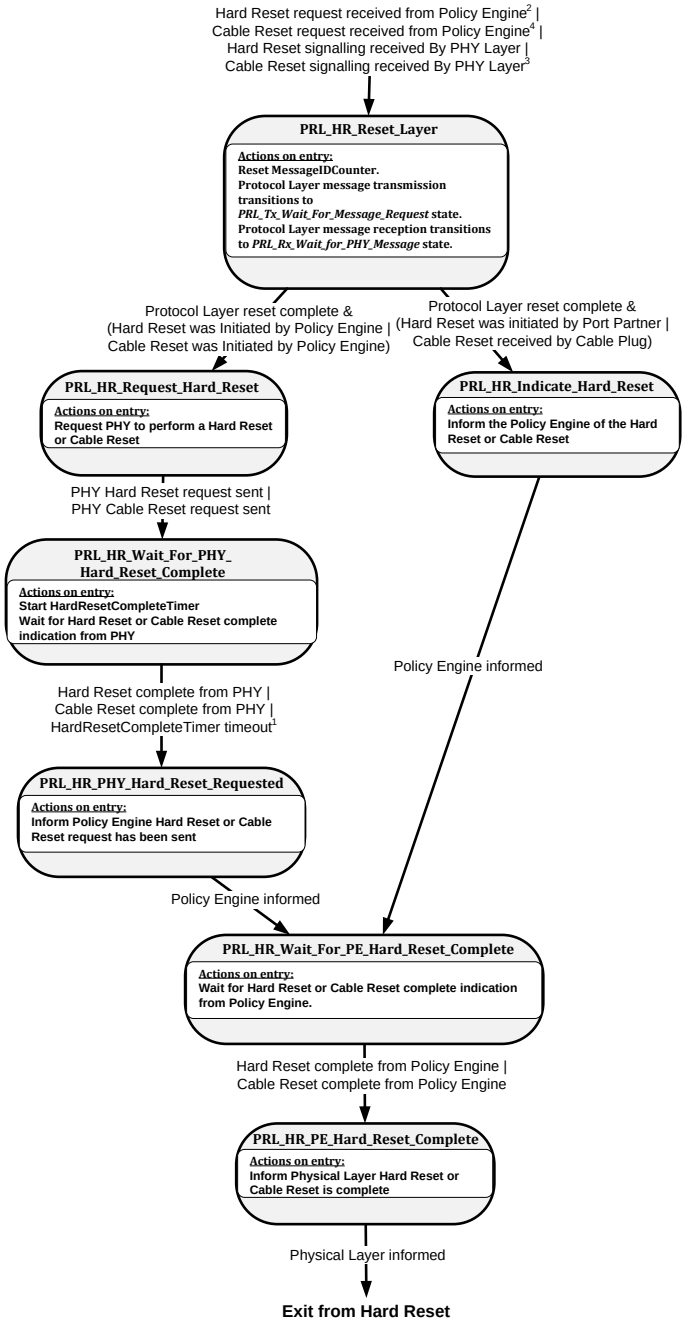
- The Message has been passed up to the Policy Engine.

6.11.2.3 Hard Reset operation

Figure 6-46 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 Physical Layer (see also Section 6.8.2 and Section 6.8.3).

Microchip Ex. 1020, p. 183
Microchip v. Aptiv
IPR2024-00496

Figure 6-46 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

Microchip Ex. 1020, p. 184
Microchip v. Aptiv
IPR2024-00496

6.11.2.3.1 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 Physical 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 Physical Layer or
- A Cable Reset Request is received from the Policy Engine or
- ***Cable Reset*** Signaling is received from the Physical 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 Physical Layer or
    - A Cable Reset request has been passed up from the Physical Layer (Cable Plug only).

6.11.2.3.2 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.

6.11.2.3.3 PRL_HR_Request_Hard_Reset state

On entry to the ***PRL_HR_Request_Hard_Reset*** state the Protocol Layer shall request the Physical 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 Physical Layer ***Hard Reset*** Signaling request has been sent or
- The Physical Layer ***Cable Reset*** Signaling request has been sent.

6.11.2.3.4 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

Microchip Ex. 1020, p. 185
Microchip v. Aptiv
IPR2024-00496

- A Cable Reset complete indication is received from the PHY Layer or
- The ***HardResetCompleteTimer*** times out.

6.11.2.3.5 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.

6.11.2.3.6 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.

6.11.2.3.7 PRL_HR_PE_Hard_Reset_Complete

On entry to the ***PRL_HR_PE_Hard_Reset_Complete*** state the Protocol Layer shall inform the Physical 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 Physical 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 Physical Layer has been informed that the Cable Reset is complete.

Microchip Ex. 1020, p. 186
Microchip v. Aptiv
IPR2024-00496

6.11.3 List of Protocol Layer States

Table 6-50 lists the states used by the various state machines.

Table 6-50 Protocol Layer States

State name	Reference
Protocol Layer Message Transmission	
Common Protocol Layer Message Transmission	
PRL_Tx_PHY_Layer_Reset	Section 6.11.2.1.1.1
PRL_Tx_Wait_for_Message_Request	Section 6.11.2.1.1.2
PRL_Tx_Layer_Reset_for_Transmit	Section 6.11.2.1.1.3
PRL_Tx_Construct_Message	Section 6.11.2.1.1.4
PRL_Tx_Wait_for_PHY_Response	Section 6.11.2.1.1.5
PRL_Tx_Match_MessageID	Section 6.11.2.1.1.6
PRL_Tx_Message_Sent	Section 6.11.2.1.1.7
PRL_Tx_Check_RetryCounter	Section 6.11.2.1.1.8
PRL_Tx_Transmission_Error	Section 6.11.2.1.1.9
PRL_Tx_Discard_Message	Section 6.11.2.1.1.10
Source Protocol Layer Message Transmission	
PRL_Tx_Src_Sink_Tx	Section 6.11.2.1.2.1
PRL_Tx_Src_Source_Tx	Section 6.11.2.1.2.2
PRL_Tx_Src_Pending	Section 6.11.2.1.2.3
Sink Protocol Layer Message Transmission	
PRL_Tx_Snk_Start_of_AMS	Section 6.11.2.1.3.1
PRL_Tx_Snk_Pending	Section 6.11.2.1.3.2
Protocol Layer Message Reception	
PRL_Rx_Wait_for_PHY_Message	Section 6.11.2.2.1
PRL_Rx_Layer_Reset_for_Receive	Section 6.11.2.2.2
PRL_Rx_Send_GoodCRC	Section 6.11.2.2.3
PRL_Rx_Check_MessageID	Section 6.11.2.2.4
PRL_Rx_Store_MessageID	Section 6.11.2.2.5
Hard Reset Operation	
PRL_HR_Reset_Layer	Section 6.11.2.3.1
PRL_HR_Indicate_Hard_Reset	Section 6.11.2.3.2
PRL_HR_Request_Hard_Reset	Section 6.11.2.3.3
PRL_HR_Wait_for_PHY_Hard_Reset_Complete	Section 6.11.2.3.4
PRL_HR_PHY_Hard_Reset_Requested	Section 6.11.2.3.5
PRL_HR_Wait_for_PE_Hard_Reset_Complete	Section 6.11.2.3.6
PRL_HR_PE_Hard_Reset_Complete	Section 6.11.2.3.7

Microchip Ex. 1020, p. 187
Microchip v. Aptiv
IPR2024-00496

6.12 Message Applicability

The following tables outline the Messages supported by a given port, depending on its capability.

When a Message is supported the feature and Message sequence implied by the Message shall also be supported. For example Sinks using power for charging that support the ***GotoMin*** Message shall be able to reduce their current draw when requested via a ***GotoMin*** Message.

The following abbreviations are used:

- N – ***Normative***; shall be supported by this Port/Cable Plug
- CN – ***Conditional Normative***; shall be supported by a given Port/Cable Plug based on features
- R – Recommended; should be supported by this Port/Cable Plug
- O – ***Optional***; may be supported by this Port/Cable Plug
- NS – Not Supported; shall result in a ***Not_Supported*** Message response by this Port/Cable Plug when received.
- I – Ignore; shall be ***Ignored*** by this Port/Cable Plug when received.
- NK – NAK; this Port/Cable Plug shall return Responder NAK to this Command when received
- NA – Not allowed; shall not be transmitted by this Port/Cable Plug.

For the case of ***Conditional Normative*** a note has been added to indicate the condition. "CN/" notation is used to indicate the level of support when the condition is not present.

"R/" and "O/" notation is used to indicate the response when the Recommended or ***Optional*** Message is not supported.

Note: that where NS/RJ/NK is indicated for Received Messages this shall apply to the ***PE_CBL_Ready***, ***PE_SNK_Ready*** or ***PE_SRC_Ready*** states only since unexpected Messages received during a Message sequence are Protocol Errors (see Section 6.8.1).

This section covers Control and Data Message support for Sources, Sink and Cable Plugs. It also covers VDM Command support for DFPs, UFPs and Cable Plugs.

Microchip Ex. 1020, p. 188
Microchip v. Aptiv
IPR2024-00496

6.12.1 Applicability of Control Messages

Table 6-51 details Control Messages that shall/should/shall not be transmitted and received by a Source, Sink or Cable Plug. Requirements for Dual-Role Power Ports and Dual-Role Data Ports shall override any requirements for Source-only or Sink-Only Ports.

Table 6-51 Applicability of Control Messages

Message Type	Source	Sink	Dual-Role Power	Dual-Role Data	Cable Plug
Transmitted Message					
Accept	N	N			N
DR_Swap	O	O		N	NA
FR_Swap	NA	NA	R		NA
Get_Sink_Cap	R	NA	N		NA
Get_Source_Cap	NA	R	N		NA
Get_Source_Cap_Extended	NA	R	R		NA
Get_Status	R	R			NA
GoodCRC	N	N			N
GotoMin	CN[1]/O	NA			NA
Ping	O	NA			NA
PR_Swap	NA	NA	N		NA
PS_RDY	N	NA	N		NA
Reject	N	NA	O	O	NA
Soft_Reset	N	N			NA
VCONN_Swap	R	R			NA
Wait	CN[2]/O	NA	O	O	NA
Received Message					
Accept	N	N	N	N	I
DR_Swap	O/NS	O/NS		N	I
FR_Swap	NS	NS	CN[7]		I
Get_Sink_Cap	NS	N	N		I
Get_Source_Cap	N	NS	N		I
Get_Source_Cap_Extended	CN[5]/NS	NS	CN[5]/NS		I
Get_Status	CN[6]/NS	CN[6]/NS	CN[6]/NS		I
GoodCRC	N	N			N
GotoMin	NS	R[3]			I
Ping	NS	I			I
PR_Swap	NS	NS	N		I
PS_RDY	NS	N	N		I
Reject	NS	N	N	N	I
Soft_Reset	N	N			N
VCONN_Swap	CN[4]/ NS	CN[4]/ NS			I

Microchip Ex. 1020, p. 189
Microchip v. Aptiv
IPR2024-00496

Message Type	Source	Sink	Dual-Role Power	Dual-Role Data	Cable Plug
Wait	NS	N	N	N	I
Note 1: Shall be supported by a Hub with multiple Downstream Ports. Should be supported by a Host with multiple Downstream Ports. Note 2: Shall be supported when transmission of ***GotoMin*** Messages is supported. Note 3: Should be supported by Sinks which use PD power for charging. Note 4: Shall be supported by any Port that can operate as a VCONN Source. Note 5: Shall be supported products that support the ***Source_Capabilities_Extended*** Message. Note 6: Shall be supported by Sources that support the ***Alert*** Message. Note 7: Shall be supported when the Fast Role Swap signal is supported.					

6.12.2 Applicability of Data Messages

Table 6-52 details Data Messages (except for VDM Commands) that shall/should/shall not be transmitted and received by a Source, Sink or Cable Plug. Requirements for Dual-Role Power Ports shall override any requirements for Source-only or Sink-Only Ports.

Table 6-52 Applicability of Data Messages

Message Type	Source	Sink	Dual-Role Power	Cable Plug
Transmitted Message				
Source_Capabilities	N	NA	N	NA
Request	NA	N		NA
BIST	N[1]	N[1]		NA
Sink_Capabilities	NA	N	N	NA
Battery_Status	CN[2]	CN[2]		NA
Alert	R	R		NA
Received Message				
Source_Capabilities	NS	N	N	I
Request	N	NS		I
BIST	N[1]	N[1]		N[1]
Sink_Capabilities	N	NS	N	I
Battery_Status	CN[3]	CN[3]		I
Alert	R/NS	R/NS		I
Note 1: For details of which BIST Modes and Messages shall be supported see Section 5.9 and Section 6.4.3. Note 2: Shall be supported by products that contain batteries. Note 3: Shall be supported by products that support the ***Get_Battery_Status*** Message.				

6.12.3 Applicability of Extended Messages

Table 6-53 details Extended Messages (except for Extended VDM Commands) that shall/should/shall not be transmitted and received by a Source, Sink or Cable Plug. Requirements for Dual-Role Power Ports shall override any requirements for Source-only or Sink-Only Ports.

Microchip Ex. 1020, p. 190
Microchip v. Aptiv
IPR2024-00496

Table 6-53 Applicability of Extended Messages

Message Type	Source	Sink	Dual-Role Power	Cable Plug
Transmitted Message				
Source_Capabilities_Extended	R	NA	R	NA
Status	R	R	R	NA
Get_Battery_Cap	R	R		NA
Get_Battery_Status	R	R		NA
Battery_Capabilities	CN[1]/NA	CN[1]/NA		NA
Get_Manufacturer_Info	R	R		NA
Manufacturer_Info	R	R		R
Security_Request	CN[6]/NA	CN[6]/NA		NA
Security_Response	CN[6]/NA	CN[6]/NA		CN[6]/NA
Firmware_Update_Request	CN[7]/NA	CN[7]/NA		NA
Firmware_Update_Response	CN[7]/NA	CN[7]/NA		CN[7]/NA
Received Message				
Source_Capabilities_Extended	NS	CN[2]	CN[2]	I
Status	CN[3]	CN[3]		I
Get_Battery_Cap	CN[1]/NS	CN[1]/NS		I
Get_Battery_Status	CN[1]/NS	CN[1]/NS		I
Battery_Capabilities	CN[4]	CN[4]		I
Get_Manufacturer_Info	R	R		R
Manufacturer_Info	CN[5]	CN[5]		I
Security_Request	CN[6]/NA	CN[6]/NA		CN[6]/I
Security_Response	CN[6]/NA	CN[6]/NA		I
Firmware_Update_Request	CN[7]/NA	CN[7]/NA		CN[7]/I
Firmware_Update_Response	CN[7]/NA	CN[7]/NA		I

Note 1: Shall be supported by products that contain batteries.
Note 2: Shall be supported by products that can transmit the ***Get_Source_Cap_Extended*** Message.
Note 3: Shall be supported by products that can transmit the ***Get_Status*** Message.
Note 4: Shall be supported by products that can transmit the ***Get_Battery_Cap*** Message.
Note 5: Shall be supported by products that can transmit the ***Get_Manufacturer_Info*** Message
Note 6: Shall be supported by products that support USB security communication as defined in ***[USBTypeCAuthentication 1.0]***
Note 7: Shall be supported by products that support USB firmware update communication as defined in ***[USBPDFirmwareUpdate 1.0]***

6.12.4 Applicability of VDM Commands

Table 6-54 details VDM Commands that shall/should/shall not be transmitted and received by a DFP, UFP or Cable Plug.

Microchip Ex. 1020, p. 191
Microchip v. Aptiv
IPR2024-00496

Table 6-54 Applicability of VDM Commands

Command Type	DFP	UFP	Cable Plug
Transmitted Command Request			
Discover Identity	CN[1]/R	R[2]	NA
Discover SVIDs	CN[1]/ O	O	NA
Discover Modes	CN[1]/ O	O	NA
Enter Mode	CN[1]/NA	NA	NA
Exit Mode	CN[1]/NA	NA	NA
Attention	O	O	NA
Received Command Request/Transmitted Command Response			
Discover Identity	0/NK	CN[1]/R/NK	N
Discover SVIDs	0/NK	CN[1]/NK	CN[1]/NK
Discover Modes	0/NK	CN[1]/NK	CN[1]/NK
Enter Mode	NK	CN[1]/NK	CN[1]/NK
Exit Mode	NK	CN[1]/NK	CN[1]/NK
Attention	0/NK	O/NK	O
Note 1: Shall be supported when Modal Operation is supported. Note 2: May be transmitted by a UFP/Source during discovery (see Section 6.4.4.3.1 and Section 8.3.3.21.3).			

6.12.5 Applicability of Reset Signaling

Table 6-55 details Reset Signaling that shall/should/shall not be transmitted and received by a DFP/UFP or Cable Plug.

Table 6-55 Applicability of Reset Signaling

Signaling Type	DFP	UFP	Cable Plug
Transmitted Message/Signaling			
Soft_Reset	N	N	NA
Hard Reset	N	N	NA
Cable Reset	CN[1]	NA	NA
Received Message/Signaling			
Soft_Reset	N	N	N
Hard Reset	N	N	N
Cable Reset			N
Note 1: Shall be supported when transmission of SOP' Packets are supported.			

6.12.6 Applicability of Fast Role Swap signal

Table 6-55 details the Fast Role Swap signal that shall/should/shall not be transmitted and received by a Source or Sink.

Microchip Ex. 1020, p. 192
Microchip v. Aptiv
IPR2024-00496

Table 6-56 Applicability of Fast Role Swap signal

Command Type	Source	Sink	Dual-Role Power
Transmitted Message/Signaling			
Fast Role Swap	NA	NA	R
Received Message/Signaling			
Fast Role Swap	NA	NA	R

Microchip Ex. 1020, p. 193
Microchip v. Aptiv
IPR2024-00496

6.13 Value Parameters

Table 6-57 contains value parameters used in this section.

Table 6-57 Value Parameters

Parameter	Description	Value	Unit	Reference
MaxExtendedMsgLen	Maximum length of an Extended Message as expressed in the ***Data Size*** field.	260	Byte	Section 6.5
MaxExtendedMsgChunkLen		26	Byte	Section 6.5
MaxExtendedMsgLegacyLen		26	Byte	Section 6.5

Microchip Ex. 1020, p. 194
Microchip v. Aptiv
IPR2024-00496

7. Power Supply

7.1 Source Requirements

7.1.1 Behavioral Aspects

A USB PD Source exhibits the following behaviors.

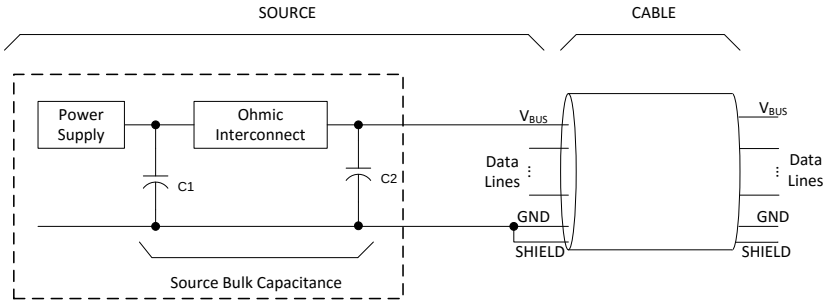
- Shall be backward compatible with legacy V_{BUS} ports.
- Shall supply the default ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** voltage and current to V_{BUS} when the USB cable is Attached (USB Default Operation).
- Shall supply the default ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** voltage and current to V_{BUS} when a Contract does not exist (USB Default Operation).
- Shall return ***vSafe0V*** for some time then return to ***vSafe5V*** when ***Hard Reset*** Signaling is received.
- Shall control V_{BUS} voltage transitions as bound by undershoot, overshoot and transition time requirements.

7.1.2 Source Bulk Capacitance

The Source bulk capacitance shall not be placed between the transceiver isolation impedance and the USB receptacle. The Source bulk capacitance consists of C1 and C2 as shown in Figure 7-1. The Ohmic Interconnect may consist of PCB traces for power distribution or power switching devices. The capacitance may be a single capacitor, a capacitor bank or distributed capacitance. If the power supply is shared across multiple ports, the bulk capacitance is defined as ***cSrcBulkShared***. If the power supply is dedicated to a single Port, the minimum bulk capacitance is defined as ***cSrcBulk***.

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 Default 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 7-1 Placement of Source Bulk Capacitance



7.1.3 Types of Sources

Consistent with the Power Data Objects discussed in Section 6.4.1, the three possible power supply types that are available as Sources in a USB Power Delivery System are:

- The Fixed Supply PDO exposes well-regulated fixed voltage power supplies. Sources shall support at least one Fixed Supply capable of supplying ***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 7-16 and described in Section 7.1.9.

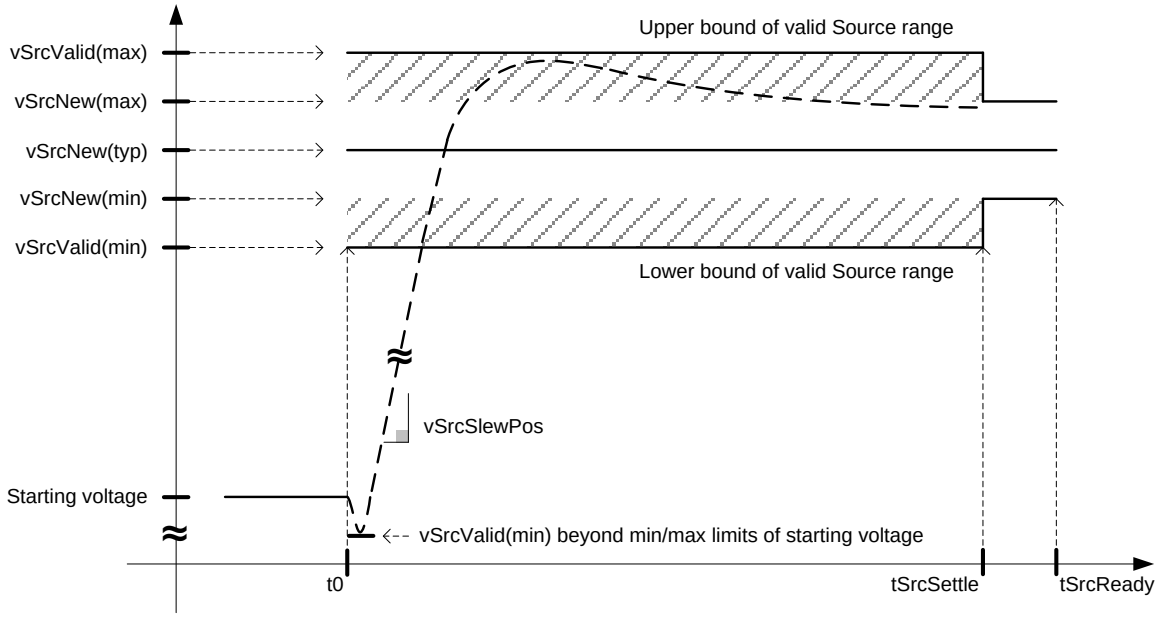
Microchip Ex. 1020, p. 195
Microchip v. Aptiv
IPR2024-00496

- The Variable Supply (non-Battery) PDO exposes very poorly 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 than can be connected directly as a Source to V_{BUS}. 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.

7.1.4 Positive Voltage Transitions

The Source shall transition V_{BUS} from the starting voltage to the higher new voltage in a controlled manner. The negotiated new voltage (e.g., 12V or 20V) defines the nominal value for ***vSrcNew***. During the positive transition the Source shall be able to supply the Sink standby power and the transient current to charge the total bulk capacitance on V_{BUS}. The slew rate of the positive transition shall not exceed ***vSrcSlewPos***. 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***. The positive voltage transition shall remain monotonic while the transitioning voltage is below ***vSrcValid*** min and shall remain within the ***vSrcValid*** range upon crossing ***vSrcValid*** min as shown in Figure 7-2. The starting time, t0, in Figure 7-2 starts ***tSrcTransition*** after the last bit of the ***EOP*** of the ***GoodCRC*** Message has been received by the Source.

Figure 7-2 Transition Envelope for Positive Voltage Transitions



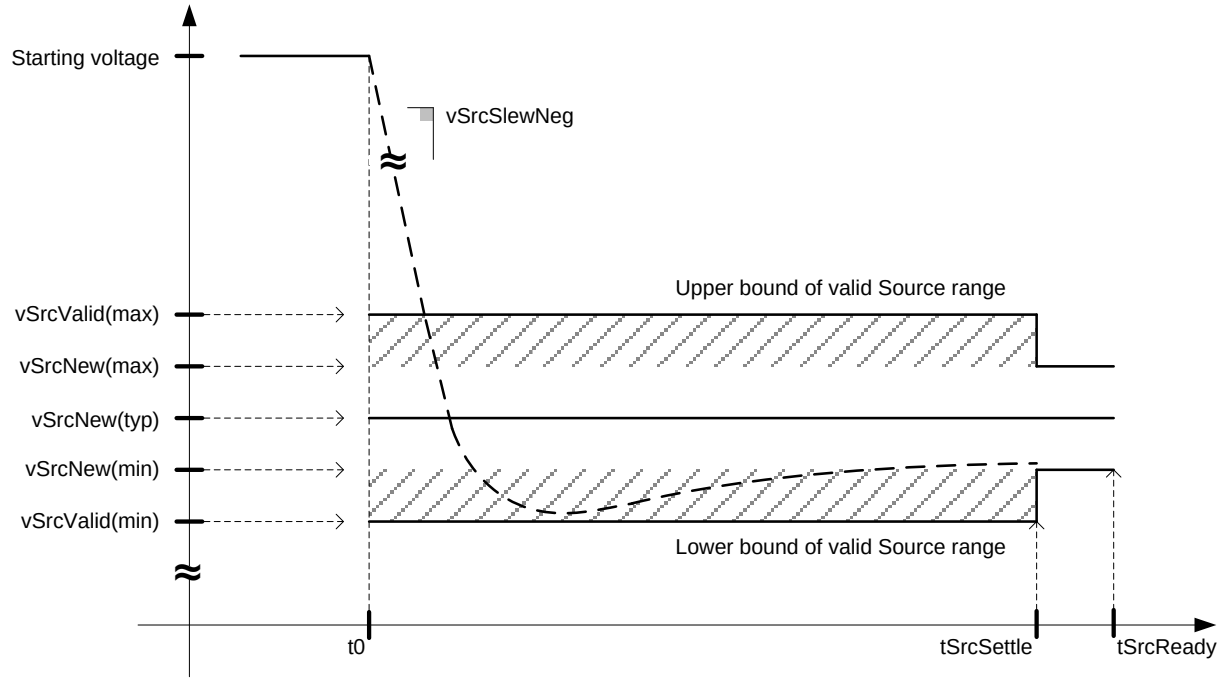
At the start of the positive voltage transition the V_{BUS} voltage level shall not droop ***vSrcValid*** min below either ***vSrcNew*** (i.e., if the starting V_{BUS} voltage level is not ***vSafe5V***) or ***vSafe5V*** as applicable.

7.1.5 Negative Voltage Transitions

Negative voltage transitions are defined as shown in Figure 7-3 and are specified in a similar manner to positive voltage transitions. Figure 7-3 does not apply to ***vSafe0V*** transitions. The slew rate of the negative transition shall not exceed ***vSrcSlewNeg***. The negative voltage transition shall remain monotonic while the transitioning voltage is above ***vSrcValid*** max and shall remain within the ***vSrcValid*** range upon crossing ***vSrcValid*** max as shown in Figure 7-3. The starting time, t0, in Figure 7-3 starts ***tSrcTransition*** after the last bit of the ***EOP*** of the ***GoodCRC*** Message has been received by the Source.

Microchip Ex. 1020, p. 196
Microchip v. Aptiv
IPR2024-00496

Figure 7-3 Transition Envelope for Negative Voltage Transitions



If the newly negotiated voltage is ***vSafe5V***, then the ***vSrcValid*** limits shall determine the transition window and the transitioning Source shall settle within the ***vSafe5V*** limits by ***tSrcSettle***.

7.1.6 Response to Hard Resets

Hard Reset Signaling indicates a communication failure has occurred and the Source shall stop driving VCONN, shall remove Rp from the VCONN pin and shall drive V_{BUS} to ***vSafe0V*** as shown in Figure 7-4. The USB connection may reset during a Hard Reset since the V_{BUS} voltage will be less than ***vSafe5V*** for an extended period of time. After establishing the ***vSafe0V*** voltage condition on V_{BUS}, the Source shall wait ***tSrcRecover*** before re-applying VCONN and restoring V_{BUS} to ***vSafe5V***. A Source shall conform to the VCONN timing as specified in ***[USB Type-C 1.2]***.

Device operation during and after a Hard Reset is defined as follows:

- Self-powered devices should not disconnect from USB during a Hard Reset (see Section 9.1.2).
- 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.

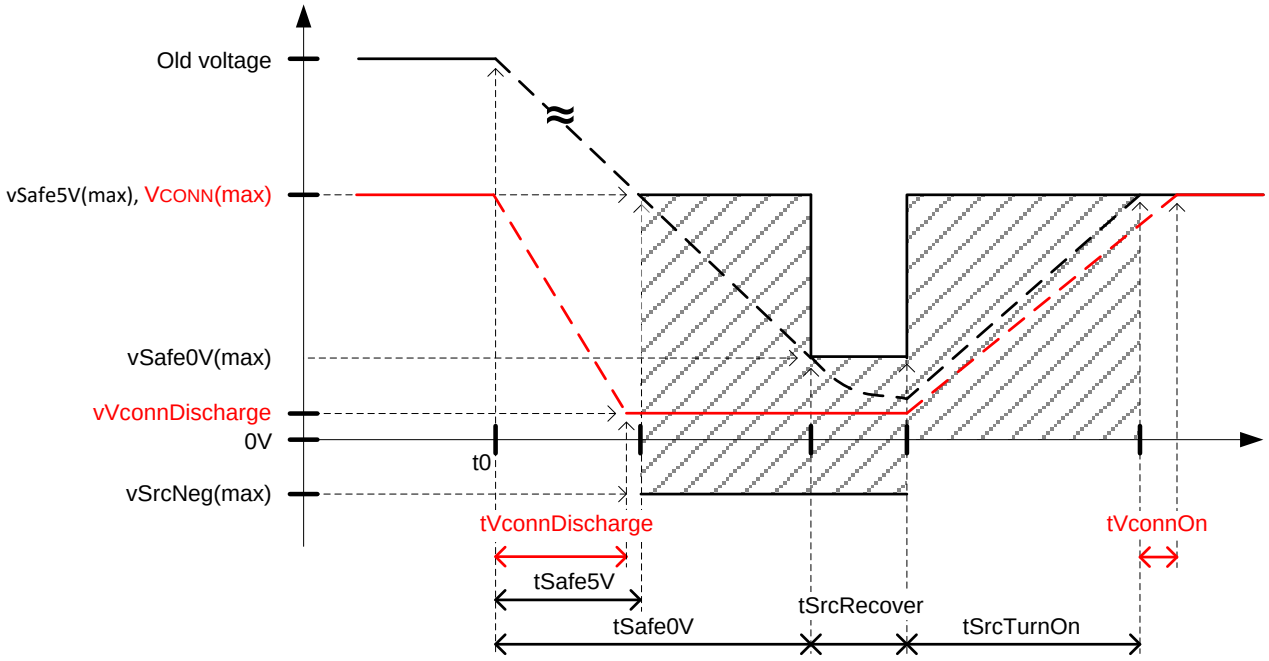
When a Hard Reset occurs the Source shall stop driving VCONN, shall remove Rp from the VCONN pin and shall start to transition the V_{BUS} voltage to ***vSafe0V*** either:

- ***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

The Source shall meet both ***tSafe5V*** and ***tSafe0V*** relative to the start of the voltage transition as shown in Figure 7-4.

Microchip Ex. 1020, p. 197
Microchip v. Aptiv
IPR2024-00496

Figure 7-4 Source V_{BUS} Response to Hard Reset



V_{CONN} will meet ***tVconnDischarge*** relative to the start of the voltage transition as shown in Figure 7-4 due to the discharge circuitry in the Cable Plug. V_{CONN} shall meet ***tVconnOn*** relative to V_{BUS} reaching ***vSafe5V***. Note ***tVconnOn*** and ***tVconnDischarge*** are defined in ***[USB Type-C 1.2]***.

7.1.7 Changing the Output Power Capability

Some USB Power Delivery negotiations will require the Source to adjust its output power capability without changing the output voltage. In this case the Source shall be able to supply a higher or lower load current within ***tSrcReady***.

7.1.8 Robust Source Operation

7.1.8.1 Output Over Current Protection

Sources shall implement output over current protection to prevent damage from output current that exceeds the current handling capability of the Source. The definition of current handling capability is left to the discretion of the Source implementation and shall take into consideration the current handling capability of the connector contacts. The response to over current shall not interfere with the negotiated V_{BUS} current level.

Sources should attempt to send a ***Hard Reset*** message when over current protection engages followed by an ***Alert*** Message indicating an OCP event once 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 default operation after determining that the cause of over current is no longer present and may latch off to protect the port or system. The definition of how to detect if the cause of over current is still present is left to the discretion of the Source implementation.

The Source shall renegotiate with the Sink (or Sinks) after choosing to resume default operation. The decision of how to renegotiate after an over current event is left to the discretion of the Source implementation.

Microchip Ex. 1020, p. 198
Microchip v. Aptiv
IPR2024-00496

The Source shall prevent continual system or port cycling if over current protection continues to engage after initially resuming either default operation or renegotiation. Latching off the port or system is an acceptable response to recurring over current.

During the over current response and subsequent system or port shutdown, all affected Source ports operating with V_{BUS} greater than ***vSafe5V*** shall discharge V_{BUS} to ***vSafe5V*** by the time ***tSafe5V*** and ***vSafe0V*** by the time ***tSafe0V***.

7.1.8.2 Over Temperature Protection

Sources shall implement over temperature protection 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 over temperature protection are left to the discretion of the Source implementation.

Sources should attempt to send a ***Hard Reset*** message when over temperature protection engages followed by an ***Alert*** Message indicating an OTP event once an Explicit Contract has been established. The over temperature protection response may engage at either the port or system level. Systems or ports that have engaged over temperature protection should attempt to resume default operation and may latch off to protect the port or system.

The Source shall renegotiate with the Sink (or Sinks) after choosing to resume default operation. The decision of how to renegotiate after an over temperature event is left to the discretion of the Source implementation.

The Source shall prevent continual system or port cycling if over temperature protection continues to engage after initially resuming either default operation or renegotiation. Latching off the port or system is an acceptable response to recurring over temperature.

During the over temperature response and subsequent system or port shutdown, all affected Source ports operating with V_{BUS} greater than ***vSafe5V*** shall discharge V_{BUS} to ***vSafe5V*** by the time ***tSafe5V*** and ***vSafe0V*** by the time ***tSafe0V***.

7.1.8.3 vSafe5V Externally Applied to Ports Supplying vSafe5V

Safe operation mandates that Power Delivery Sources shall be tolerant of ***vSafe5V*** being present on V_{BUS} when simultaneously applying power to V_{BUS}. Normal USB PD communication shall be supported when this ***vSafe5V*** to ***vSafe5V*** connection exists.

7.1.8.4 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*** by ***tSafe0V*** relative to when the Detach event occurred. During the transition to ***vSafe0V*** the V_{BUS} voltage shall be below ***vSafe5V*** max by ***tSafe5V*** relative to when the Detach event occurred and shall not exceed ***vSafe5V*** max after this time.

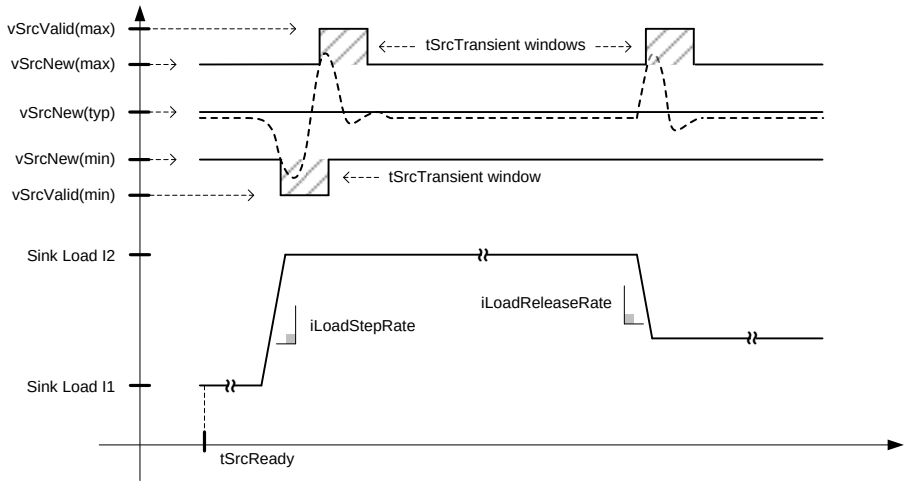
7.1.9 Output Voltage Tolerance and Range

After a voltage transition is complete (i.e. after ***tSrcReady***) and during static load conditions the Source output voltage shall remain within the ***vSrcNew*** or ***vSafe5V*** limits as applicable.. The ranges defined by ***vSrcNew*** and ***vSafe5V*** account for DC regulation accuracy, line regulation, load regulation and output ripple. After a voltage transition is complete (i.e. after ***tSrcReady***) and during transient load conditions the Source output voltage shall not go beyond the range specified by ***vSrcValid***. The amount of time the Source output voltage can be in the band between either ***vSrcNew*** or ***vSafe5V*** and ***vSrcValid*** shall not exceed ***tSrcTransient***. Refer to Table 7-16 for the output voltage tolerance specifications. Figure 7-5 illustrates the application of ***vSrcNew*** and ***vSrcValid*** after the voltage transition is complete.

The ***vSrcNew*** and ***vSrcValid*** limits shall not apply to V_{BUS} during the V_{BUS} discharge and switchover that occurs during a Fast Role Swap as described in Section 7.1.14.

Microchip Ex. 1020, p. 199
Microchip v. Aptiv
IPR2024-00496

Figure 7-5 Application of vSrcNew and vSrcValid limits after tSrcReady



The Source output voltage shall be measured at the connector receptacle. The stability of the Source shall be tested in 25% load step increments from minimum load to maximum load and also from maximum load to minimum load. The transient behavior of the load current is defined in Section 7.2.6. The time between each step shall be sufficient to allow for the output voltage to settle between load steps. In some systems it may 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.

7.1.10 Charging and Discharging the Bulk Capacitance on V_{BUS}

The Source shall charge and discharge the bulk capacitance on V_{BUS} whenever the Source voltage is negotiated to a different value. The charging or discharging occurs during the voltage transition and shall not interfere with the Source's ability to meet ***tSrcReady***.

7.1.11 Swap Standby for Sources

Sources and Sinks of a Dual-Role Power Port shall support Swap Standby. Swap Standby occurs for the Source after the Source power supply has discharged the bulk capacitance on V_{BUS} to ***vSafe0V*** as part of the Power Role Swap transition.

While in Swap Standby:

- The Source shall not drive V_{BUS} that is therefore expected to remain at ***vSafe0V***.
- Any discharge circuitry that was used to achieve ***vSafe0V*** shall be removed from V_{BUS}.
- The Dual-Role Power Port shall be configured as a Sink
- The USB connection shall not reset even though ***vSafe5V*** is no longer present on V_{BUS} (see Section 9.1.2).

The ***PS_RDY*** Message associated with the Source being in Swap Standby shall be sent after the V_{BUS} drive is removed. The time for the Source to transition to Swap Standby shall not exceed ***tSrcSwapStdby***. Upon entering Swap Standby the Source has relinquished its role as Source and is ready to become the new Sink. The transition time from Swap Standby to being the new Sink shall be no more than ***tNewSnk***. The new Sink may start using power after the new Source sends the ***PS_RDY*** Message.

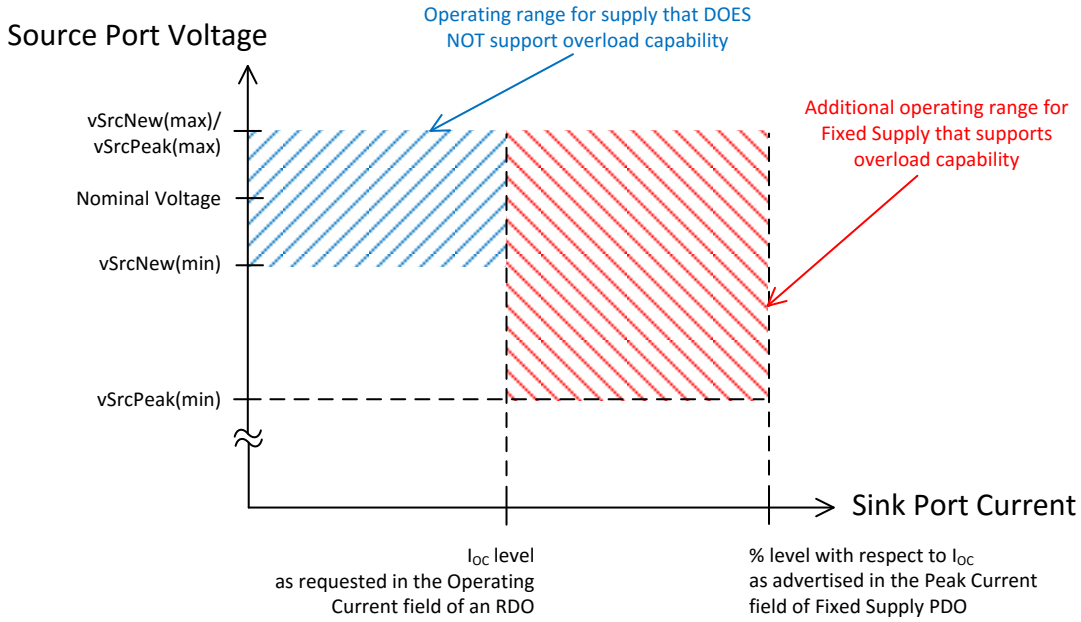
7.1.12 Source Peak Current Operation

A Source that has the Fixed Supply PDO Peak Current bits set to 01b, 10b and 11b shall be designed to support one of the overload capabilities defined in Table 6-9. The overload conditions are bound in magnitude, duration and duty cycle as listed in Table 6-9. Sources are not required to support continuous overload operation. When overload conditions occur, the Source is allowed the range of ***vSrcPeak*** (instead of ***vSrcNew)*** relative to the nominal value (see

Microchip Ex. 1020, p. 200
Microchip v. Aptiv
IPR2024-00496

Figure 7-6). When the overload capability is exceeded, the Source is expected take whatever action is necessary to prevent electrical or thermal damage to the Source. The Source may send a new ***Source_Capabilities*** Message with the Fixed Supply PDO Peak Current bits set to 00b to prohibit overload operation even if an overload capability was previously negotiated with the Sink.

Figure 7-6 Source Peak Current Overload



Microchip Ex. 1020, p. 201
Microchip v. Aptiv
IPR2024-00496

7.1.13 Source Capabilities Extended Parameters

Implementers can choose to make available certain characteristics of a USB PD Source 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 are described in full in Section 6.5.1 and listed in Figure 6-29. The subset of parameters listed below directly represent Source capabilities and are described in the rest of this section.

- Voltage Regulation
- Holdup Time
- Compliance
- Peak Current
- Source Inputs
- Batteries

7.1.13.1 Voltage Regulation Field

The power consumption of a device may change dynamically. The ability of the Source to regulate its voltage output may be important if the device is sensitive to fluctuations in voltage. The Voltage Regulation bit field is used to convey information about the Sources output regulation and tolerance to various load steps.

7.1.13.1.1 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 Extended Source Capabilities:

- The Source shall maintain V_{BUS} regulation within the ***vSrcValid*** 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 5.8.2 under all load conditions.

7.1.13.1.2 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.

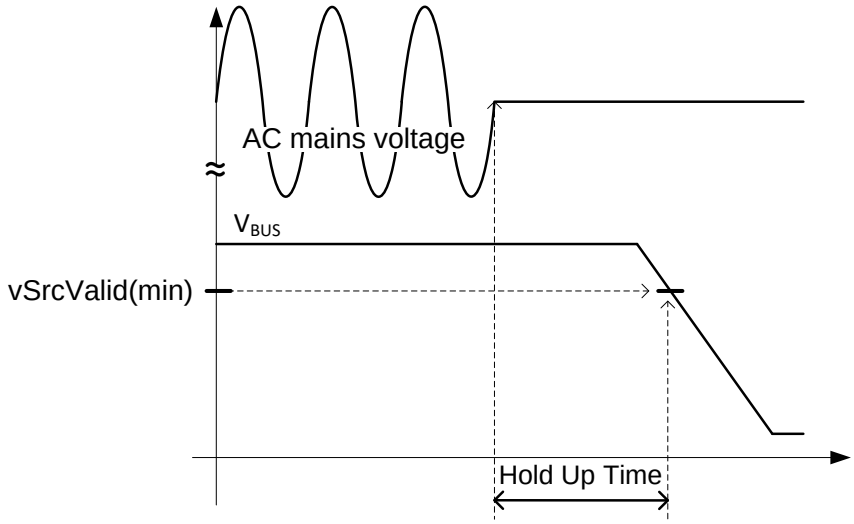
7.1.13.2 Holdup Time Field

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 AC mains.

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

Microchip Ex. 1020, p. 202
Microchip v. Aptiv
IPR2024-00496

Figure 7-7 Holdup Time Measurement



7.1.13.3 Compliance Field

A Source claiming LPS, PS1 or PS2 compliance shall report its capabilities in the Compliance field. Since the Source may have several potential output voltage and current settings, every Source supply (indicated by a PDO) shall be compliant to LPS requirements.

Note: according to the requirements of ***[IEC 60950-1]***, a device tested and certified with an LPS Source is prohibited to use a non-LPS Source. Alternatively, ***[IEC 62368-1]***, classifies power sources according to their maximum, constrained power output (15watts or 100watts).

7.1.13.4 Peak Current

The Source reports its ability to source peak current delivery in excess of the negotiated amount in the Peak Current field. The duration of peak current must be followed by a current consumption below the Operating Current (IoC) in order to maintain average power delivery below the IoC current.

A Source may have greater capability to source peak current than can be reported using the Peak Current field in the Fixed Supply PDO. In this case the Source shall report its additional capability in the Peak Current field in the ***Source_Capabilities_Extended*** Message.

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 6.5.1.8) shall not exceed the negotiated current. This is calculated as:

*Period of reduced current = (1 – value in Duty Cycle field/100) * value in Overload Period field*

7.1.13.5 Source Inputs

The Source Inputs field identifies the possible inputs that provide power to the Source. Note some Sources are only powered by a Battery (e.g., an automobile) rather than the more common mains.

7.1.13.6 Batteries

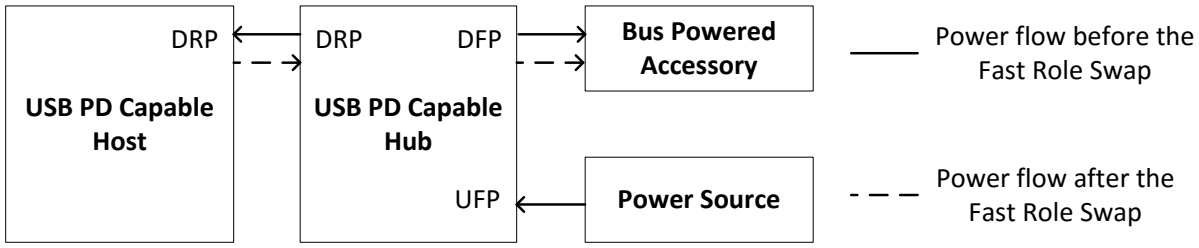
The Batteries field shall report the number of Batteries the Source supports. The Source shall independently report the number of Hot Swappable Batteries and the number of Fixed batteries.

Microchip Ex. 1020, p. 203
Microchip v. Aptiv
IPR2024-00496

7.1.14 Fast Role Swap

A Fast Role Swap limits the interruption of V_{BUS} power to a bus powered accessory connected to a Hub DFP that has a UFP attached to a power source and a DRP attached to a Host port supporting DRP as shown in Figure 7-8 V_{BUS} Power during Fast Role Swap

Figure 7-8 V_{BUS} Power during Fast Role Swap



When the power source connected to the Hub UFP stops sourcing power and V_{BUS} at the Hub DRP connector discharges below ***vSrcValid***(min), if V_{BUS} has been negotiated to a higher voltage than***vSafe5V***, or ***vSafe5V*** (min) the Fast Role Swap signal shall be sent from the Hub DRP to the Host DRP and the Hub DRP shall sink power. In the Fast Role Swap use case, the Hub DRP behaves like a bidirectional power path. The Hub DRP shall not enable V_{BUS} discharge circuitry when changing operation from initial Source to new Sink.

The new Sink shall be limited to USB Type-C Current (see ***[USB Type-C 1.2]***) until a new Explicit Contract is negotiated. All Sink requirements shall apply to the new Sink after the Fast Role Swap is complete. The Fast Role Swap response of the Host DRP is described in Section 7.2.10 since the Host DRP is operating as the initial Sink prior to the Fast Role Swap.

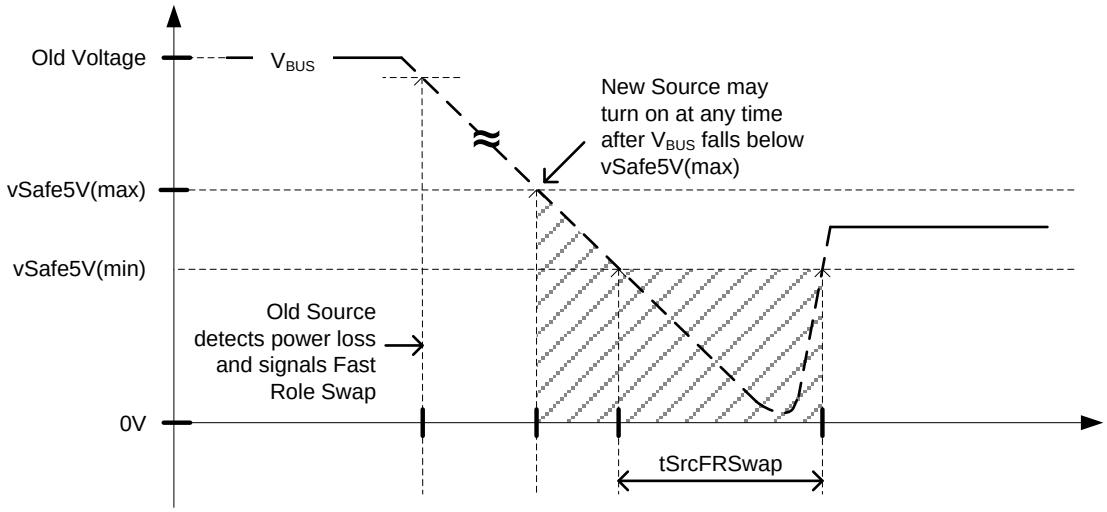
After the V_{BUS} voltage level at the Hub DRP connector drops below ***vSafe5V*** a ***PS_RDY*** Message shall be sent to the Host DRP as shown in the Fast Role Swap transition diagram of Section 7.3.15.

Figure 7-9 shows the V_{BUS} detection and timing for the new Source during a Fast Role Swap after the Fast Role Swap signal has been received. The new Source may turn on the V_{BUS} output switch once V_{BUS} is below ***vSafe5V*** (max). In this case, the new Source prevents V_{BUS} from falling below ***vSafe5V*** (min). The new source shall turn on the V_{BUS} output switch within ***tSrcFRSwap*** of falling below ***vSafe5V*** (min).

Note: V_{BUS} might have started at ***vSafe5V*** or at higher voltage. When the Fast Role Swap Signal is detected, V_{BUS} could therefore be either above ***vSafe5V*** (max), within the ***vSafe5V*** range, or below ***vSafe5V*** (min).

Microchip Ex. 1020, p. 204
Microchip v. Aptiv
IPR2024-00496

Figure 7-9 V_{BUS} detection and timing during Fast Role Swap



Microchip Ex. 1020, p. 205
Microchip v. Aptiv
IPR2024-00496

7.2 Sink Requirements

7.2.1 Behavioral Aspects

A USB PD Sink exhibits the following behaviors.

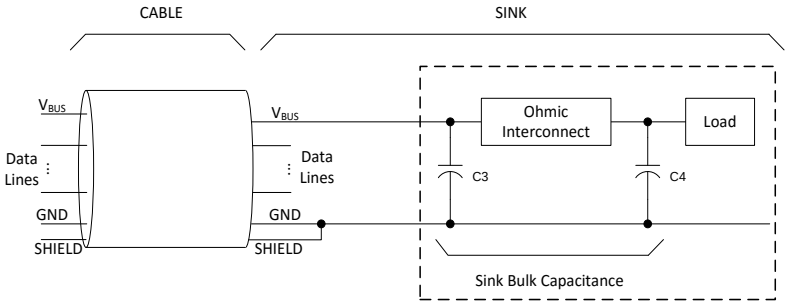
- Shall be backward compatible with legacy V_{BUS} ports.
- Shall draw the default ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** V_{BUS} current when the USB cable is Attached (USB Default Operation).
- Shall draw the default ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** V_{BUS} current when a Contract does not exist (USB Default Operation).
- Shall return to the default ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** V_{BUS} when responding to a Hard Reset (USB Default Operation).
- Shall control V_{BUS} in-rush current when increasing current consumption.

7.2.2 Sink Bulk Capacitance

The Sink bulk capacitance consists of C3 and C4 as shown in Figure 7-10. The Ohmic Interconnect may consist of PCB traces for power distribution or power switching devices. The capacitance may be a single capacitor, a capacitor bank or distributed capacitance. An upper bound of ***cSnkBulkPd*** shall not be exceeded so that the transient charging, or discharging, of the total bulk capacitance on V_{BUS} can be accounted for during voltage transitions.

The Sink bulk capacitance that is within the ***cSnkBulk*** max or ***cSnkBulkPd*** max limits is allowed to change to support a newly negotiated power level. The capacitance can be changed when the Sink enters Sink Standby or during a voltage transition or when the Sink begins to operate at the new power level. Regardless of when the change occurs, the capacitance change shall occur in such a manner that does not introduce a V_{BUS} transient current greater than ***iCapChange***. 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.

Figure 7-10 Placement of Sink Bulk Capacitance



7.2.3 Sink Standby

The Sink shall transition to Sink Standby before a positive or negative voltage transition of V_{BUS}. During Sink Standby the Sink shall reduce its power draw to ***pSnkStdby.*** 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***. The ***pSnkStdby*** requirement shall only apply if the Sink power draw is higher than this level.

See Section 7.3 for details of when ***pSnkStdby*** shall be applied for any given transition.

Microchip Ex. 1020, p. 206
Microchip v. Aptiv
IPR2024-00496

7.2.4 Suspend Power Consumption

When Source has set its USB Suspend Supported flag (see Section 6.4.1.2.2.2), 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.

7.2.5 Zero Negotiated Current

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.

7.2.6 Transient Load Behavior

When a Sink's operating current changes due to a load step, load release or any other change in load level, the positive or negative overshoot of the new load current shall not exceed the range defined by ***iOvershoot***. For the purposes of measuring ***iOvershoot*** the new load current value is defined as the average steady state value of the load current after the load step has settled. The rate of change of any shift in Sink load current during normal operation shall not exceed ***iLoadStepRate*** (for load steps) and ***iLoadReleaseRate*** (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 Load Step Slew Rate field and shall ensure that PD Communications meet the transmit and receive masks as specified in Section 5.8.2.

7.2.7 Swap Standby for Sinks

The Sink capability in a Dual-Role Power Port shall support Swap Standby. Swap Standby occurs for the Sink after evaluating the ***Accept*** Message from the Source during a Power Role Swap negotiation. While in Swap Standby the Sink's current draw shall not exceed ***iSnkSwapStdby*** from V_{BUS} and the Dual-Role Power Port shall be configured as a Source after V_{BUS} has been discharged to ***vSafe0V*** by the existing Initial Source. The Sink's USB connection should not be reset even though ***vSafe5V*** is not present on the V_{BUS} conductor (see Section 9.1.2). 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 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***.

7.2.8 Sink Peak Current Operation

Sinks shall only make use of a Source overload capability when the corresponding Fixed Supply PDO Peak Current bits are set to 01b, 10b and 11b (see Section 6.4.1.2.2.7). Sinks shall manage thermal aspects of the overload event by not exceeding the average negotiated output of a Fixed Supply 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.

7.2.9 Robust Sink Operation

7.2.9.1 Sink Bulk Capacitance Discharge at Detach

When a Source is Detached from a Sink, the Sink shall continue to draw power from its input bulk capacitance until V_{BUS} is discharged to ***vSafe5V*** or lower by no longer than ***tSafe5V*** from the Detach event. This safe Sink requirement shall apply to all Sinks operating with a negotiated V_{BUS} 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 V_{BUS} voltage level droop, the Sink power consumption must be high enough so that V_{BUS} will decay below ***vSrcValid***(min) well within ***tSafe5V*** after the Source bulk capacitance is removed due to the Detach. Once adequate V_{BUS} droop has been achieved, a discharge circuit can be enabled to meet the safe Sink requirement.

Microchip Ex. 1020, p. 207
Microchip v. Aptiv
IPR2024-00496

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

- Negotiated V_{BUS} = 20V
- Maximum allowable supplied V_{BUS} voltage = 21.5V
- 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 V_{BUS} voltage from the worst-case maximum level of 21.5V to 17V in approximately 205ms. At this point, with V_{BUS} 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 V_{BUS} has decayed to ***vSafe5V*** or lower.

7.2.9.2 Input Over Voltage Protection

Sinks shall implement input over voltage protection to prevent damage from input voltage that exceeds the voltage handling capability of the Sink. The definition of voltage handling capability is left to the discretion of the Sink implementation. The response to over voltage shall not interfere with the negotiated V_{BUS} voltage level.

Sinks should attempt to send a ***Hard Reset*** message when over voltage protection engages followed by an ***Alert*** Message indicating an OVP event once an Explicit Contract has been established. The over voltage protection response may engage at either the port or system level. Systems or ports that have engaged over voltage protection shall resume default operation when the Source has re-established ***vSafe5V*** on V_{BUS}.

The Sink shall be able to renegotiate with the Source after resuming default operation. The decision of how to respond to renegotiation after an over voltage event is left to the discretion of the Sink implementation.

The Sink shall prevent continual system or port cycling if over voltage protection continues to engage after initially resuming either default operation or renegotiation. Latching off the port or system is an acceptable response to recurring over voltage.

7.2.9.3 Over Temperature Protection

Sinks shall implement over temperature protection to prevent damage from temperature that exceeds the thermal capability of the Sink. 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.

Sinks shall attempt to send a ***Hard Reset*** message when over temperature protection engages followed by an ***Alert*** Message indicating an OTP event once an Explicit Contract has been established. The over temperature protection response may engage at either the port or system level. Systems or ports that have engaged over temperature protection should attempt to resume default operation after sufficient cooling is achieved and may latch off to protect the port or system. The definition of sufficient cooling is left to the discretion of the Sink implementation.

The Sink shall be able to renegotiate with the Source after resuming default operation. The decision of how to respond to renegotiation after an over temperature event is left to the discretion of the Sink implementation.

The Sink shall prevent continual system or port cycling if over temperature protection continues to engage after initially resuming either default operation or renegotiation. Latching off the port or system is an acceptable response to recurring over temperature.

Microchip Ex. 1020, p. 208
Microchip v. Aptiv
IPR2024-00496

7.2.10 Fast Role Swap

As described in Section 7.1.14 a Fast Role Swap limits the interruption of V_{BUS} power to a bus powered accessory connected to a Hub DFP that has a UFP attached to a power source and a DRP attached to a Host port that supports DRP. This configuration is shown in Figure 7-8 V_{BUS} Power during Fast Role Swap

When the Host DRP that supports Fast Role Swap detects the Fast Role Swap signal, the Host DRP shall stop sinking current and shall be ready and able to source ***vSafe5V*** if the residual V_{BUS} voltage level at the Host DRP connector is greater than ***vSafe5V***. When the residual V_{BUS} voltage level at the Host DRP connector discharges below ***vSafe5V***(min) the Host DRP as the new Source shall supply ***vSafe5V*** to the Hub DRP within ***tSrcFRSwap***. The Host DRP shall not enable V_{BUS} discharge circuitry when changing roles from initial Sink to new Source.

The new Source shall supply ***vSafe5V*** at USB Type-C Current (see ***[USB Type-C 1.2]***) at the value advertised in the Fast Role Swap USB Type-C Current field (see Section 6.4.1.3.1.6). All Source requirements shall apply to the new Source after the Fast Role Swap is complete The Fast Role Swap response of the Hub DRP is described in Section 7.1.14 since the Hub DRP is operating as the initial Source prior to the Fast Role Swap.

After the Host DRP is providing V_{BUS} power to the Hub DRP, a ***PS_RDY*** Message shall be sent to the Hub DRP as defined by the Fast Role Swap signaling and messaging sequence detailed in Section 7.3.15.

Microchip Ex. 1020, p. 209
Microchip v. Aptiv
IPR2024-00496

7.3 Transitions

The following sections illustrate the power supply's response to various types of negotiations. The negotiation cases take into consideration for the examples are as follows:

- Higher Power Transitions
    - Increase the current
    - Increase the voltage
    - Increase the voltage and the current
- Relatively Constant Power Transitions
    - Increase the voltage and decrease the current
    - Decrease the voltage and increase the current
- Lower Power Transitions
    - Decrease the current
    - Decrease the voltage
    - Decrease the voltage and the current
- Power Role Swap Transitions
    - Source requests a Power Role Swap
    - Sink requests a Power Role Swap
- Go To Minimum Current Transition
- Response to ***Hard Reset*** Signaling
    - Source issues ***Hard Reset*** Signaling
    - Sink issues ***Hard Reset*** Signaling
- No change in Current or Voltage.

The transition from ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** operation into Power Delivery Mode can also lead to a Power Transition since this is the initial Contract negotiation. The following types of Power Transitions shall also be applied when moving from ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** operation into Power Delivery Mode:

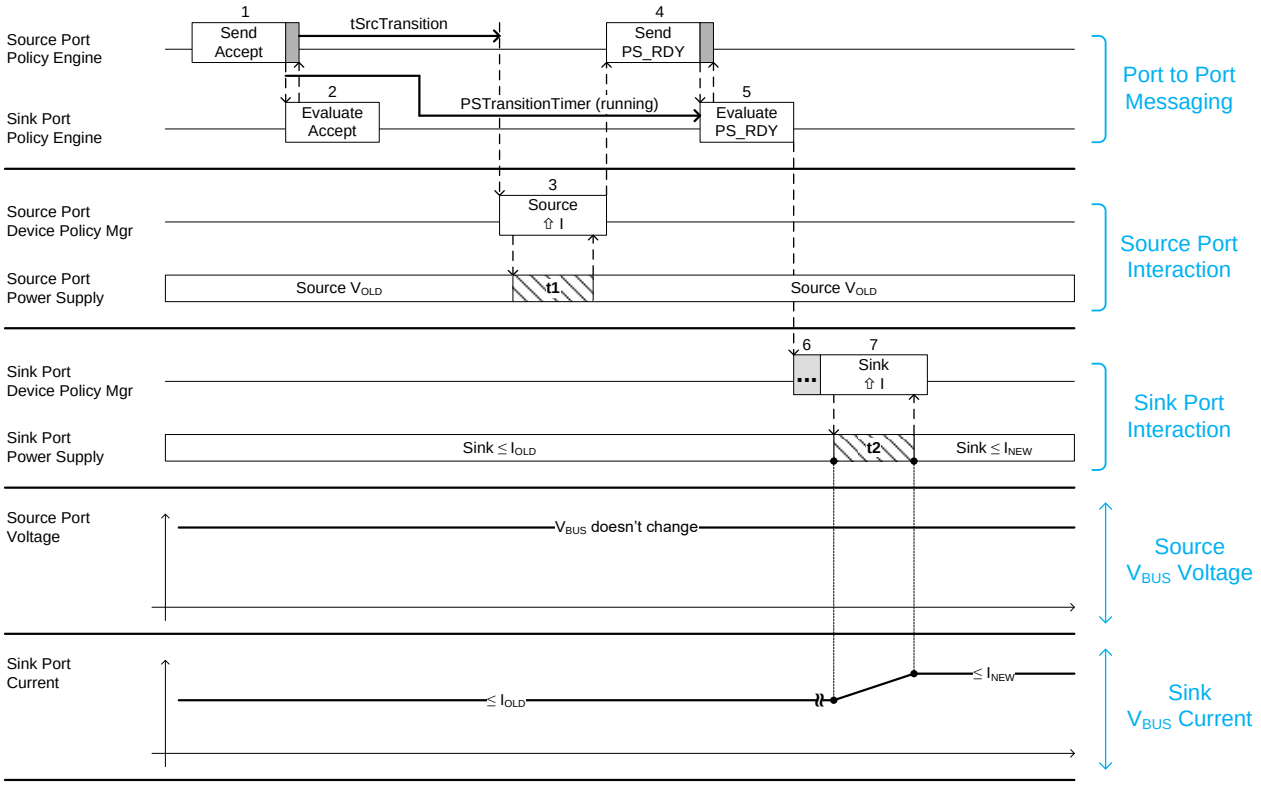
- High Power
- Relatively Constant Power
- Lower Power Transitions
- No change in Current or Voltage.

Microchip Ex. 1020, p. 210
Microchip v. Aptiv
IPR2024-00496

7.3.1 Increasing the Current

The interaction of the System Policy, Device Policy, and power supply that shall be followed when increasing the current is shown in Figure 7-11. The sequence that shall be followed is described in Table 7-1. The timing parameters that shall be followed are listed in Table 7-16 and Table 7-17. Note in this figure, the Sink has previously sent a ***Request*** Message to the Source.

Figure 7-11 Transition Diagram for Increasing the Current



Microchip Ex. 1020, p. 211
Microchip v. Aptiv
IPR2024-00496

Table 7-1 Sequence Description for Increasing the Current

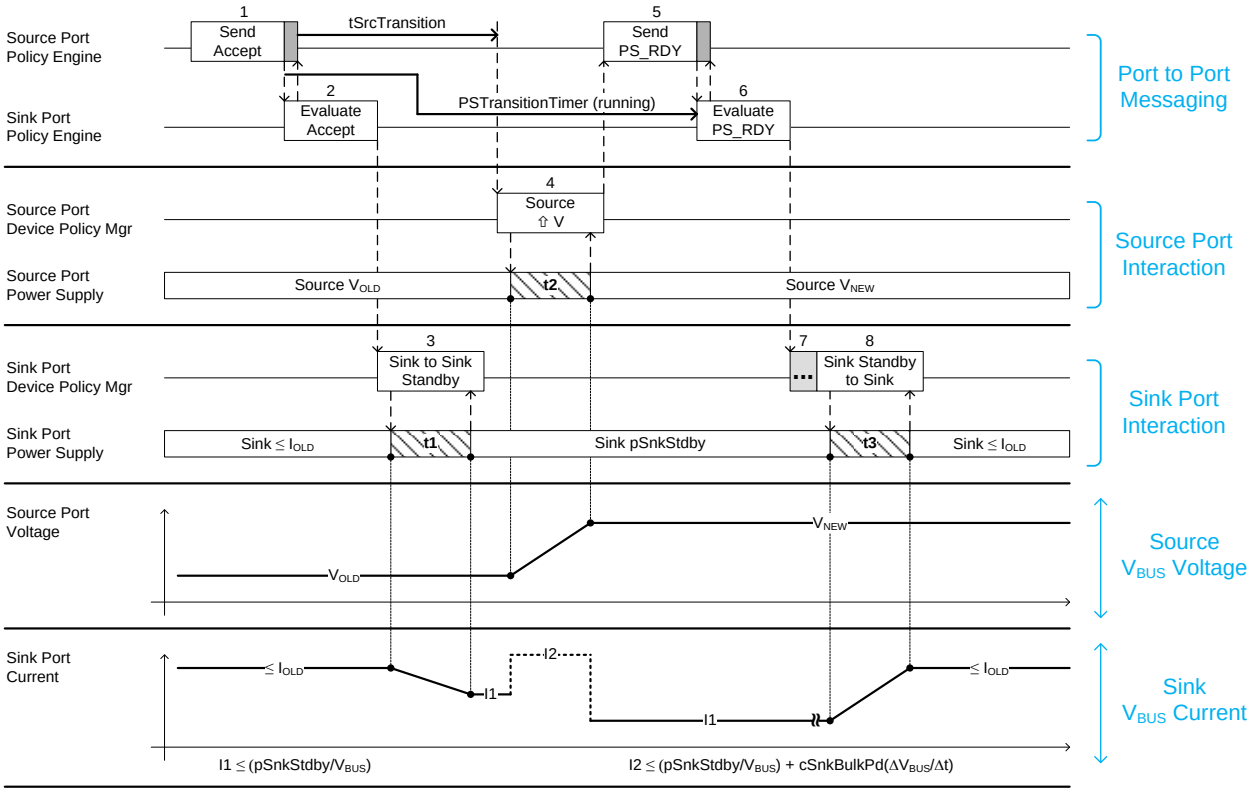
Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine receives the ***Accept*** Message and starts the ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***Accept*** Message.
3	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t1). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
4	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message from the Source.
5	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***PS_RDY*** Message from the Source and tells the Device Policy Manager it is okay to operate at the new power level.
6		The Sink may begin operating at the new power level any time after evaluation of the ***PS_RDY*** Message. This time duration is indeterminate.
7		The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t2) depends on the magnitude of the load change.

Microchip Ex. 1020, p. 212
Microchip v. Aptiv
IPR2024-00496

7.3.2 Increasing the Voltage

The interaction of the System Policy, Device Policy, and power supply that shall be followed when increasing the voltage is shown in Figure 7-12. The sequence that shall be followed is described in Table 7-2. The timing parameters that shall be followed are listed in Table 7-16, Table 7-17 and Table 7-18. Note in this figure, the Sink has previously sent a ***Request*** Message to the Source.

Figure 7-12 Transition Diagram for Increasing the Voltage



Microchip Ex. 1020, p. 213
Microchip v. Aptiv
IPR2024-00496

Table 7-2 Sequence Description for Increasing the Voltage

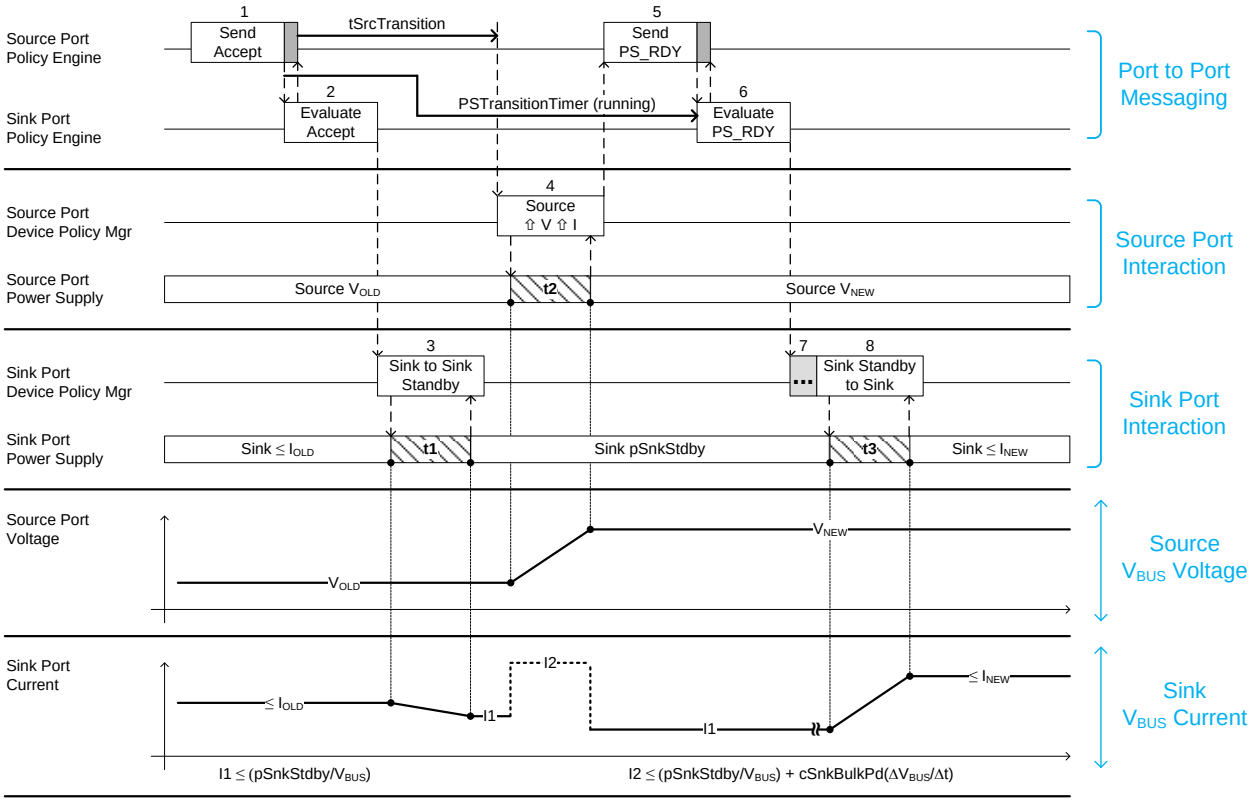
Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine receives the ***Accept*** Message and starts the ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine. Policy Engine then evaluates the ***Accept*** Message.
3		Policy Engine tells the Device Policy Manager to instruct the power supply to reduce power consumption to ***pSnkStdby*** within ***tSnkStdby*** (t1); t1 shall complete before ***tSrcTransition***. The Sink shall not violate transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t2). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
5	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message from the Source.
6	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***PS_RDY*** Message from the Source and tells the Device Policy Manager it is okay to operate at the new power level.
7		The Sink may begin operating at the new power level any time after evaluation of the ***PS_RDY*** Message. This time duration is indeterminate.
8		The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t3) depends on the magnitude of the load change.

Microchip Ex. 1020, p. 214
Microchip v. Aptiv
IPR2024-00496

7.3.3 Increasing the Voltage and Current

The interaction of the System Policy, Device Policy, and power supply that shall be followed when increasing the voltage and current is shown in Figure 7-13. The sequence that shall be followed is described in Table 7-3. The timing parameters that shall be followed are listed in Table 7-16, Table 7-17 and Table 7-18. Note in this figure, the Sink has previously sent a ***Request*** Message to the Source.

Figure 7-13 Transition Diagram for Increasing the Voltage and Current



Microchip Ex. 1020, p. 215
Microchip v. Aptiv
IPR2024-00496

Table 7-3 Sequence Diagram for Increasing the Voltage and Current

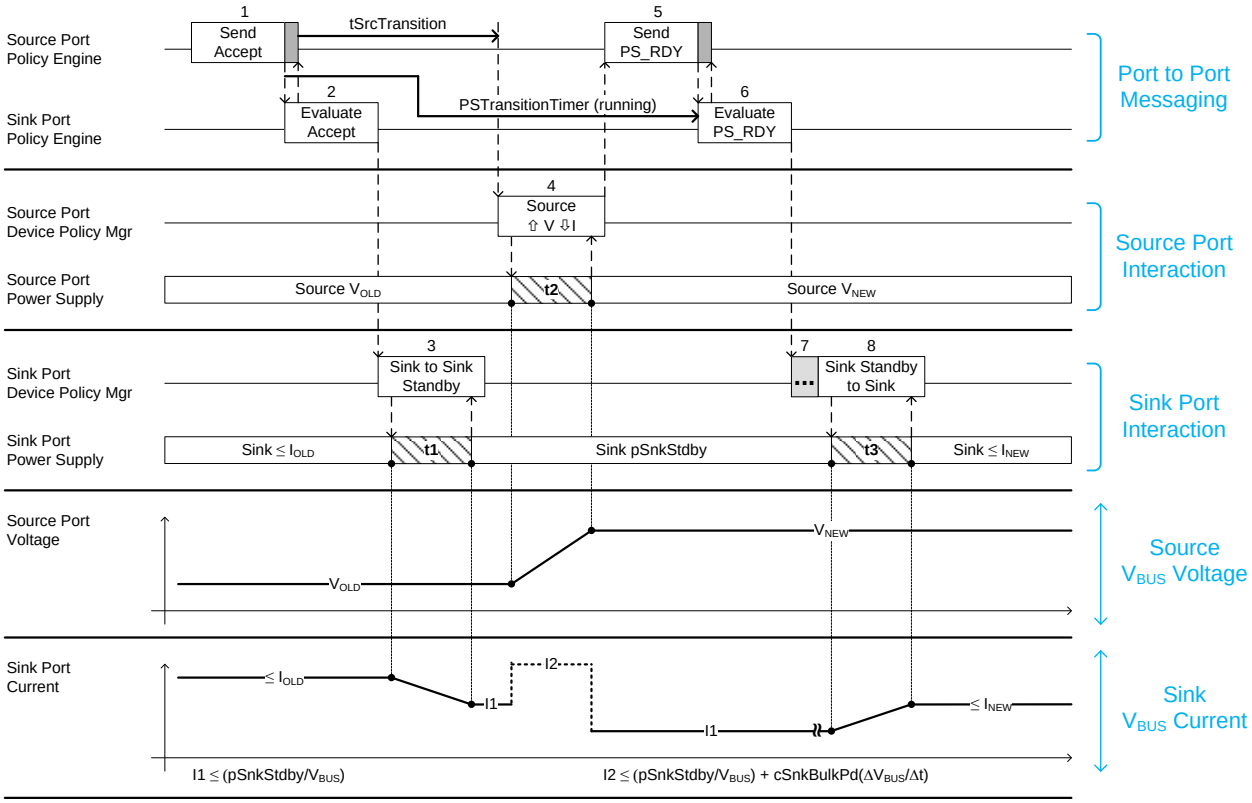
Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine receives the ***Accept*** Message and starts the ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***Accept*** Message.
3		Policy Engine tells the Device Policy Manager to instruct the power supply to reduce power consumption to ***pSnkStdby*** within ***tSnkStdby*** (t1); t1 shall complete before ***tSrcTransition***. The Sink shall not violate transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t2). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
5	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message from the Source.
6	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***PS_RDY*** Message from the Source and tells the Device Policy Manager it is okay to operate at the new power level.
7		The Sink may begin operating at the new power level any time after evaluation of the ***PS_RDY*** Message. This time duration is indeterminate.
8		The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t3) depends on the magnitude of the load change.

Microchip Ex. 1020, p. 216
Microchip v. Aptiv
IPR2024-00496

7.3.4 Increasing the Voltage and Decreasing the Current

The interaction of the System Policy, Device Policy, and power supply that shall be followed when increasing the voltage and decreasing the current is shown in Figure 7-14. The sequence that shall be followed is described in Table 7-4. The timing parameters that shall be followed are listed in Table 7-16, Table 7-17 and Table 7-18. Note in this figure, the Sink has previously sent a ***Request*** Message to the Source.

Figure 7-14 Transition Diagram for Increasing the Voltage and Decreasing the Current



Microchip Ex. 1020, p. 217
Microchip v. Aptiv
IPR2024-00496

Table 7-4 Sequence Description for Increasing the Voltage and Decreasing the Current

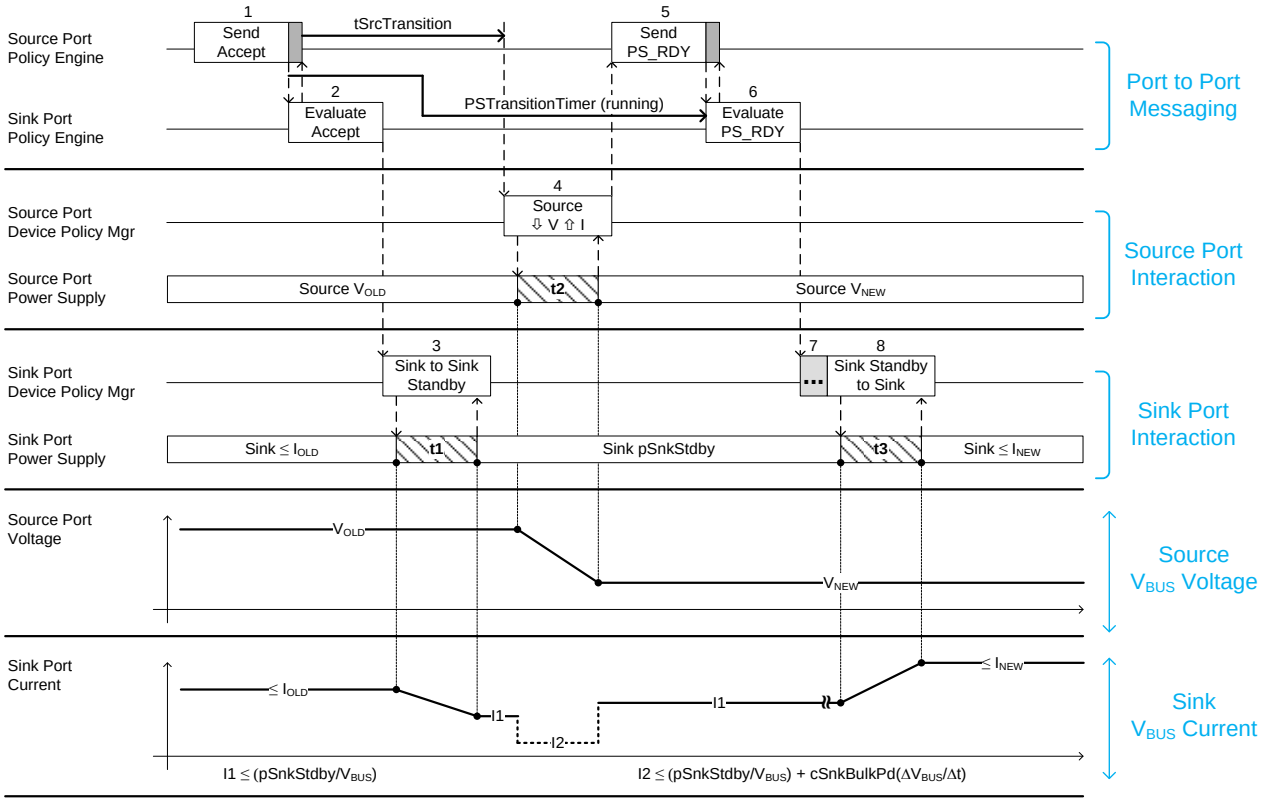
Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine evaluates the ***Accept*** Message and starts the ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***Accept*** Message.
3		Policy Engine tells the Device Policy Manager to instruct the power supply to reduce power consumption to ***pSnkStdby*** within ***tSnkStdby*** (t1); t1 shall complete before ***tSrcTransition***. The Sink shall not violate transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. T The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t2). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
5	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message from the Source.
6	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***PS_RDY*** Message from the Source and tells the Device Policy Manager it is okay to operate at the new power level.
7		The Sink may begin operating at the new power level any time after evaluation of the ***PS_RDY*** Message. This time duration is indeterminate.
8		The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t3) depends on the magnitude of the load change.

Microchip Ex. 1020, p. 218
Microchip v. Aptiv
IPR2024-00496

7.3.5 Decreasing the Voltage and Increasing the Current

The interaction of the System Policy, Device Policy, and power supply that shall be followed when decreasing the voltage and increasing the current is shown in Figure 7-15. The sequence that shall be followed is described in Table 7-5. The timing parameters that shall be followed are listed in Table 7-16, Table 7-17 and Table 7-18. Note in this figure, the Sink has previously sent a ***Request*** Message to the Source.

Figure 7-15 Transition Diagram for Decreasing the Voltage and Increasing the Current



Microchip Ex. 1020, p. 219
Microchip v. Aptiv
IPR2024-00496

Table 7-5 Sequence Description for Decreasing the Voltage and Increasing the Current

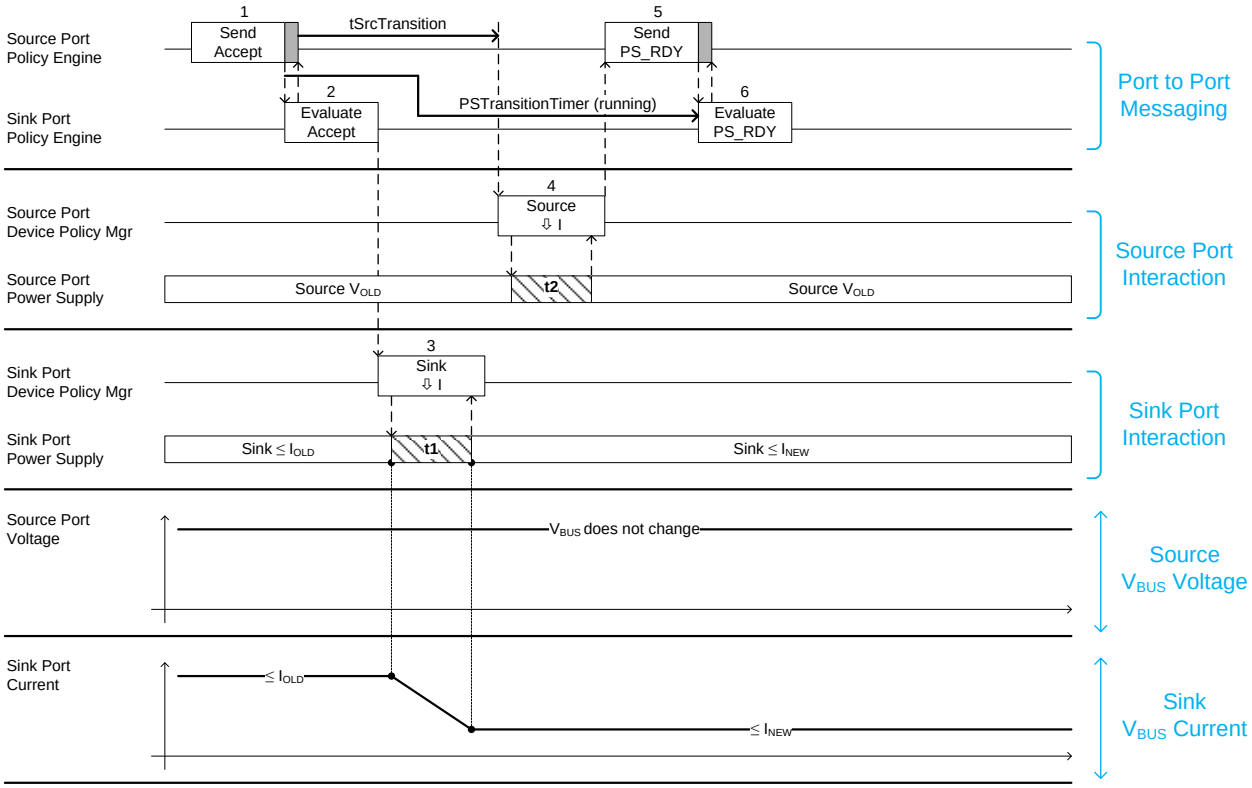
Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine receives the ***Accept*** Message and starts the ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***Accept*** Message.
3		Policy Engine tells the Device Policy Manager to instruct the power supply to reduce power consumption to ***pSnkStdby*** within ***tSnkStdby*** (t1); t1 shall complete before ***tSrcTransition***. The Sink shall not violate transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t2). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
5	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message from the Source.
6	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***PS_RDY*** Message from the Source and tells the Device Policy Manager it is okay to operate at the new power level.
7		The Sink may begin operating at the new power level any time after evaluation of the ***PS_RDY*** Message. This time duration is indeterminate.
8		The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t3) depends on the magnitude of the load change.

Microchip Ex. 1020, p. 220
Microchip v. Aptiv
IPR2024-00496

7.3.6 Decreasing the Current

The interaction of the System Policy, Device Policy, and power supply that shall be followed when decreasing the current is shown in Figure 7-16. The sequence that shall be followed is described in Table 7-6. The timing parameters that shall be followed are listed in Table 7-16, Table 7-17 and Table 7-18. Note in this figure, the Sink has previously sent a ***Request*** Message to the Source.

Figure 7-16 Transition Diagram for Decreasing the Current



Microchip Ex. 1020, p. 221
Microchip v. Aptiv
IPR2024-00496

Table 7-6 Sequence Description for Decreasing the Current

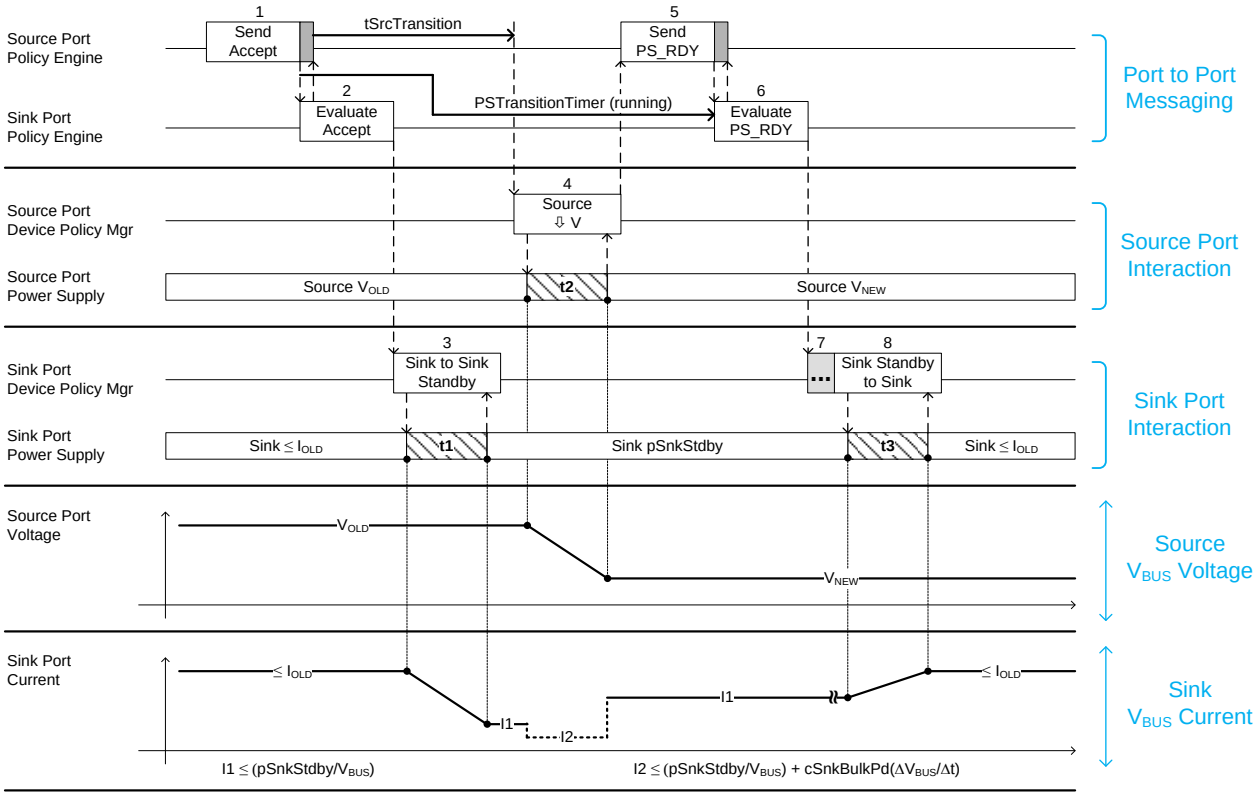
Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine receives the ***Accept*** Message starts ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***Accept*** Message. Policy Engine tells the Device Policy Manager to instruct the power supply to reduce power consumption.
3		The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The Sink shall be able to operate with lower current within ***tSnkNewPower*** (t1); t1 shall complete before ***tSrcTransition***.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t2). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
5	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message from the Source.
6	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine evaluates the ***PS_RDY*** Message from the Source. The Sink is already operating at the new power level so no further action is required.

Microchip Ex. 1020, p. 222
Microchip v. Aptiv
IPR2024-00496

7.3.7 Decreasing the Voltage

The interaction of the System Policy, Device Policy, and power supply that shall be followed when decreasing the voltage is shown in Figure 7-17. The sequence that shall be followed is described in Table 7-7. The timing parameters that shall be followed are listed in Table 7-16, Table 7-17 and Table 7-18. Note in this figure, the Sink has previously sent a ***Request*** Message to the Source.

Figure 7-17 Transition Diagram for Decreasing the Voltage



Microchip Ex. 1020, p. 223
Microchip v. Aptiv
IPR2024-00496

Table 7-7 Sequence Description for Decreasing the Voltage

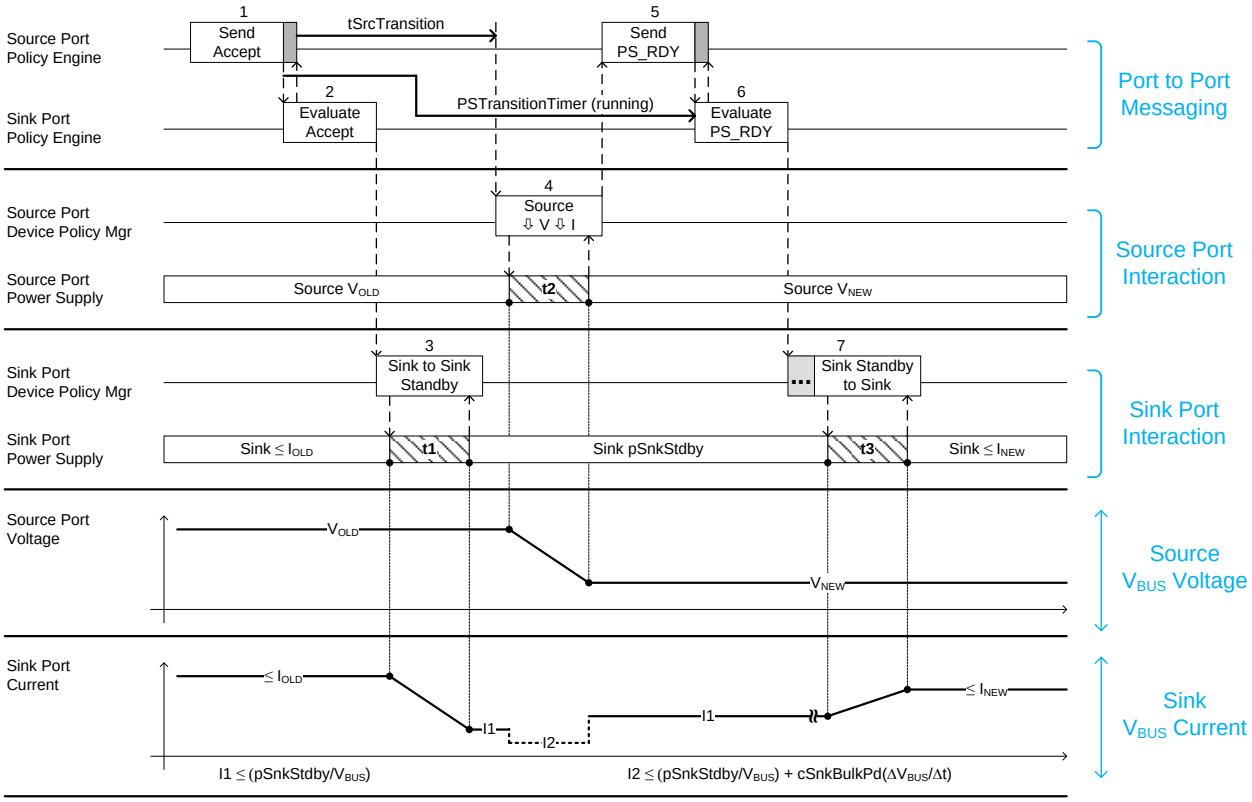
Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine receives the ***Accept*** Message and starts the ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***Accept*** Message.
3		Policy Engine tells the Device Policy Manager to instruct the power supply to reduce power consumption to ***pSnkStdby*** within ***tSnkStdby*** (t1); t1 shall complete before ***tSrcTransition***. The Sink shall not violate transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t2). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
5	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message from the Source.
6	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***PS_RDY*** Message from the Source and tells the Device Policy Manager it is okay to operate at the new power level.
7		The Sink may begin operating at the new power level any time after evaluation of the ***PS_RDY*** Message. This time duration is indeterminate.
8		The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t3) depends on the magnitude of the load change.

Microchip Ex. 1020, p. 224
Microchip v. Aptiv
IPR2024-00496

7.3.8 Decreasing the Voltage and the Current

The interaction of the System Policy, Device Policy, and power supply that shall be followed when decreasing the voltage and current is shown in Figure 7-18. The sequence that shall be followed is described in Table 7-8. The timing parameters that shall be followed are listed in Table 7-16, Table 7-17 and Table 7-18. Note in this figure, the Sink has previously sent a ***Request*** Message to the Source.

Figure 7-18 Transition Diagram for Decreasing the Voltage and the Current



Microchip Ex. 1020, p. 225
Microchip v. Aptiv
IPR2024-00496

Table 7-8 Sequence Description for Decreasing the Voltage and the Current

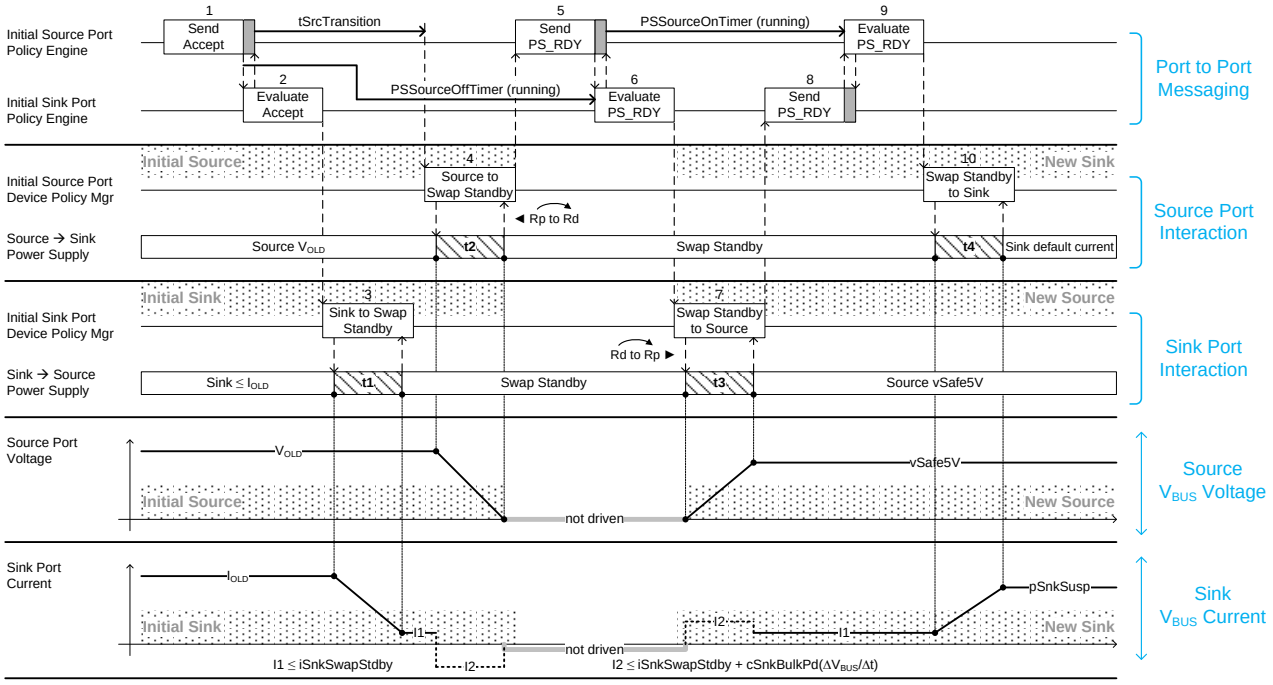
Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine receives the ***Accept*** Message and starts the ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***Accept*** Message.
3		Policy Engine tells the Device Policy Manager to instruct the power supply to reduce power consumption to ***pSnkStdby*** within ***tSnkStdby*** (t1); t1 shall complete before ***tSrcTransition***. The Sink shall not violate transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t2). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
5	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message from the Source.
6	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***PS_RDY*** Message from the Source and tells the Device Policy Manager it is okay to operate at the new power level.
7		The Sink may begin operating at the new power level any time after evaluation of the ***PS_RDY*** Message. This time duration is indeterminate.
8		The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t3) depends on the magnitude of the load change.

Microchip Ex. 1020, p. 226
Microchip v. Aptiv
IPR2024-00496

7.3.9 Sink Requested Power Role Swap

The interaction of the System Policy, Device Policy, and power supply that shall be followed during a Sink requested Power Role Swap is shown in Figure 7-19. The sequence that shall be followed is described in Table 7-9. The timing parameters that shall be followed are listed in Table 7-17. Note in this figure, the Sink has previously sent a ***PR_Swap*** Message to the Source.

Figure 7-19 Transition Diagram for a Sink Requested Power Role Swap



Microchip Ex. 1020, p. 227
Microchip v. Aptiv
IPR2024-00496

Table 7-9 Sequence Description for a Sink Requested Power Role Swap

Step	Initial Source Port → New Sink Port	Initial Sink Port → New Source Port
1	Policy Engine sends the ***Accept*** Message to the Initial Sink.	Policy Engine receives the ***Accept*** and starts the ***PSSourceOffTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Initial Source. Policy Engine then evaluates the ***Accept*** Message.
3		Policy Engine tells the Device Policy Manager to instruct the power supply to transition to Swap Standby within ***tSnkStdby*** (t1); t1 shall complete before ***tSrcTransition***. When in Sink Standby the Initial Sink shall not draw more than ***iSnkSwapStdby*** (I1). The Sink shall not violate transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability to Swap Standby (see Section 7.1.11). The power supply shall complete the transition to Swap Standby within ***tSrcSwapStdby*** (t2). The power supply informs the Device Policy Manager that it is ready to operate as the new Sink. . The CC termination is changed from Rp to Rd (see ***[USB Type-C 1.2]***). The power supply status is passed to the Policy Engine.	
5	The power supply is ready and the Policy Engine sends the ***PS_RDY*** Message to the device that will become the new Source.	
6	Protocol Layer receives the ***GoodCRC*** Message from the device that will become the new Source. Policy Engine starts the ***PSSourceOnTimer***. Upon sending the ***PS_RDY*** Message and receiving the ***GoodCRC*** Message the Initial Source is ready to be the new Sink.	Policy Engine stops the ***PSSourceOffTimer.*** The Protocol Layer sends the ***GoodCRC*** Message to the new Sink. Policy Engine tells the Device Policy to instruct the power supply to operate as the new Source.
7		The CC termination is changed from Rd to Rp (see ***[USB Type-C 1.2]***). The power supply as the new Source transitions from Swap Standby to sourcing default ***vSafe5V*** within ***tNewSrc*** (t3). The power supply informs the Device Policy Manager that it is operating as the new Source.
8	Policy Engine receives the ***PS_RDY*** Message from the Source.	Device Policy Manager informs the Policy Engine the power supply is ready and the Policy Engine sends the ***PS_RDY*** Message to the new Sink.
9	Policy Engine stops the ***PSSourceOnTimer***. Protocol Layer sends the ***GoodCRC*** Message to the new Source. Policy Engine evaluates the ***PS_RDY*** Message from the new Source and tells the Device Policy Manager to instruct the power supply to draw current as the new Sink.	Protocol Layer receives the ***GoodCRC*** Message from the new Sink.

Microchip Ex. 1020, p. 228
Microchip v. Aptiv
IPR2024-00496

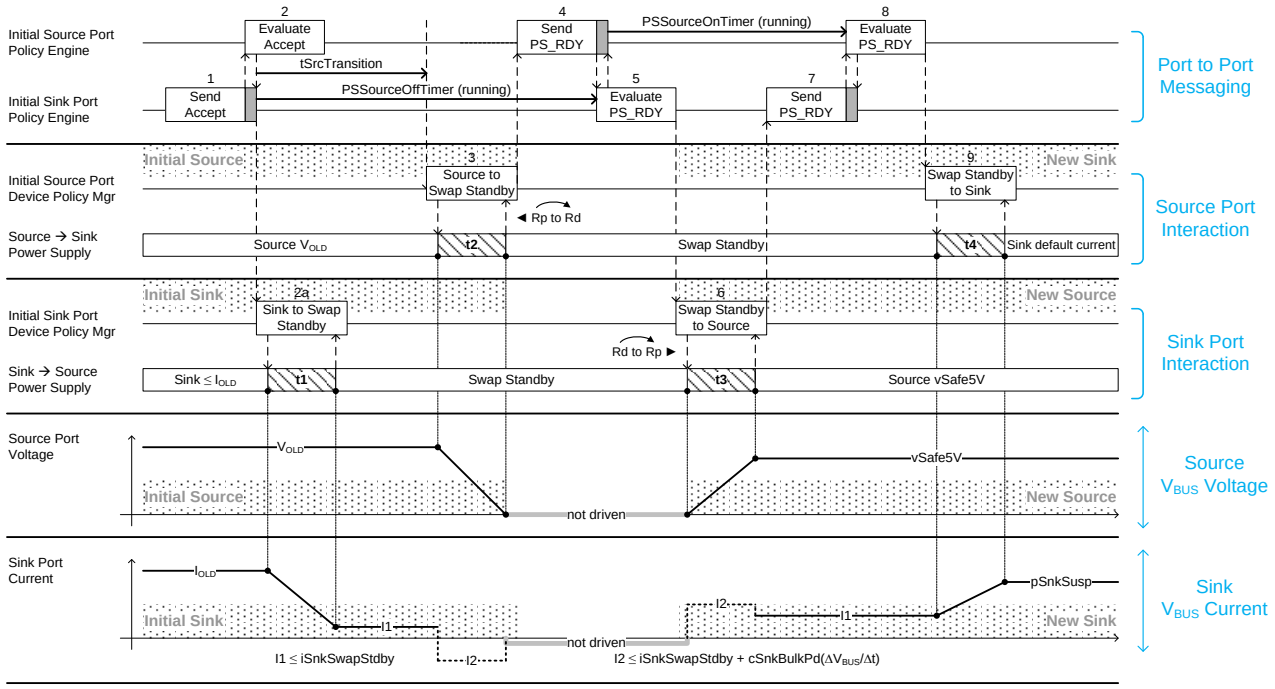
Step	Initial Source Port → New Sink Port	Initial Sink Port → New Source Port
10	The power supply as the new Sink transitions from Swap Standby to drawing ***pSnkSusp*** within ***tNewSnk*** (t4). The power supply informs the Device Policy Manager that it is operating as the new Sink. At this point subsequent negotiations between the new Source and the new Sink may proceed as normal. The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t4) depends on the magnitude of the load change.	

Microchip Ex. 1020, p. 229
Microchip v. Aptiv
IPR2024-00496

7.3.10 Source Requested Power Role Swap

The interaction of the System Policy, Device Policy, and power supply that shall be followed during a Source requested Power Role Swap is shown in Figure 7-20. The sequence that shall be followed is described in Table 7-10. The timing parameters that shall be followed are listed in Table 7-16. Note in this figure, the Sink has previously sent a ***PR_Swap*** Message to the Source.

Figure 7-20 Transition Diagram for a Source Requested Power Role Swap



Microchip Ex. 1020, p. 230
Microchip v. Aptiv
IPR2024-00496

Table 7-10 Sequence Description for a Source Requested Power Role Swap

Step	Initial Source Port→ New Sink Port	Initial Sink Port → New Source Port
1	Policy Engine receives the ***Accept*** Message.	Policy Engine sends the ***Accept*** Message to the Initial Source.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer receives the ***GoodCRC*** Message from the Initial Source. Policy Engine starts the ***PSSourceOffTimer***.
2a		The Policy Engine tells the Device Policy Manager to instruct the power supply to transition to Swap Standby. The power supply shall complete the transition to Swap Standby within ***tSnkStdby*** (t1); t1 shall complete before ***tSrcTransition***. The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. Policy Engine starts ***PSSourceOffTimer***. When in Sink Standby the Initial Sink shall not draw more than ***iSnkSwapStdby*** (I1).
3	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability to Swap Standby (see Section 7.1.11). The power supply shall complete the transition to Swap Standby within ***tSrcSwapStdby*** (t2). The power supply informs the Device Policy Manager that it is ready to operate as the new Sink. The CC termination is changed from Rp to Rd (see ***[USB Type-C 1.2]***). T The power supply status is passed to the Policy Engine.	
4	The Policy Engine sends the ***PS_RDY*** Message to the soon to be new Source.	Policy Engine receives the ***PS_RDY*** Message and stops the ***PSSourceOffTimer***.
5	Protocol Layer receives the ***GoodCRC*** Message from the soon to be new Source. Policy Engine starts the ***PSSourceOnTimer***. At this point the Initial Source is ready to be the new Sink.	Protocol Layer sends the ***GoodCRC*** Message to the new Sink. Upon evaluating the ***PS_RDY*** Message the Initial Sink is ready to operate as the new Source. Policy Engine tells the Device Policy to instruct the power supply to operate as the new Source.
6		The CC termination is changed from Rd to Rp (see ***[USB Type-C 1.2]***). The power supply as the new Source transitions from Swap Standby to sourcing default ***vSafe5V*** within ***tNewSrc*** (t3). The power supply informs the Device Policy Manager that it is operating as the new Source.
7	Policy Engine receives the ***PS_RDY*** Message and stops the ***PSSourceOnTimer***.	Device Policy Manager informs the Policy Engine the power supply is ready and the Policy Engine sends the ***PS_RDY*** Message to the new Sink.
8	Protocol Layer sends the ***GoodCRC*** Message to the new Source. Policy Engine evaluates the ***PS_RDY*** Message from the new Source and tells the Device Policy Manager to instruct the power supply to draw current as the new Sink.	Protocol Layer receives the ***GoodCRC*** Message from the new Sink.

Microchip Ex. 1020, p. 231
Microchip v. Aptiv
IPR2024-00496

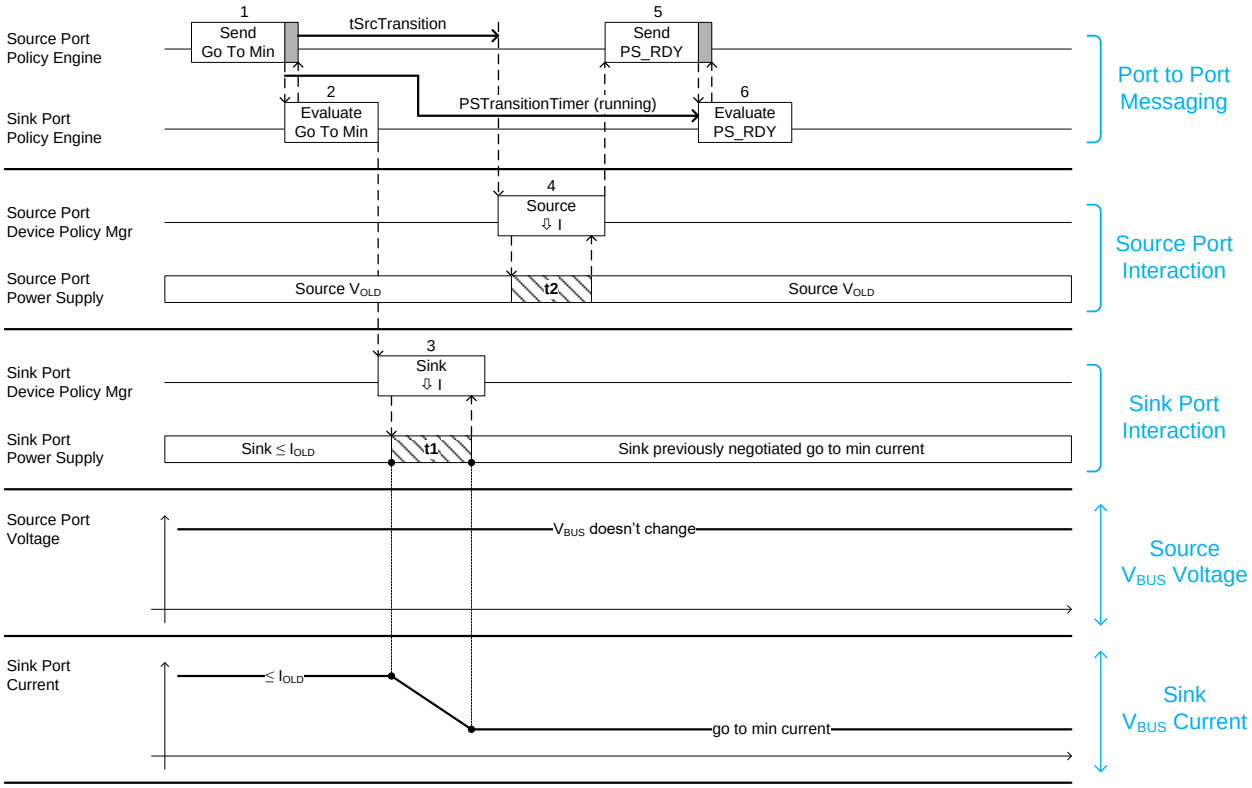
Step	Initial Source Port→ New Sink Port	Initial Sink Port → New Source Port
9	The power supply as the new Sink transitions from Swap Standby to drawing ***pSnkSusp*** within ***tNewSnk*** (t4). The power supply informs the Device Policy Manager that it is operating as the new Sink. At this point subsequent negotiations between the new Source and the new Sink may proceed as normal. The new Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level. The time duration (t4) depends on the magnitude of the load change.	

Microchip Ex. 1020, p. 232
Microchip v. Aptiv
IPR2024-00496

7.3.11 GotoMin Current Decrease

The interaction of the System Policy, Device Policy, and power supply that shall be followed during a GotoMin current decrease is shown in Figure 7-21. The sequence that shall be followed is described in Table 7-11. The timing parameters that shall be followed are listed in Table 7-16 and Table 7-11.

Figure 7-21 Transition Diagram for a GotoMin Current Decrease



Microchip Ex. 1020, p. 233
Microchip v. Aptiv
IPR2024-00496

Table 7-11 Sequence Description for a GotoMin Current Decrease

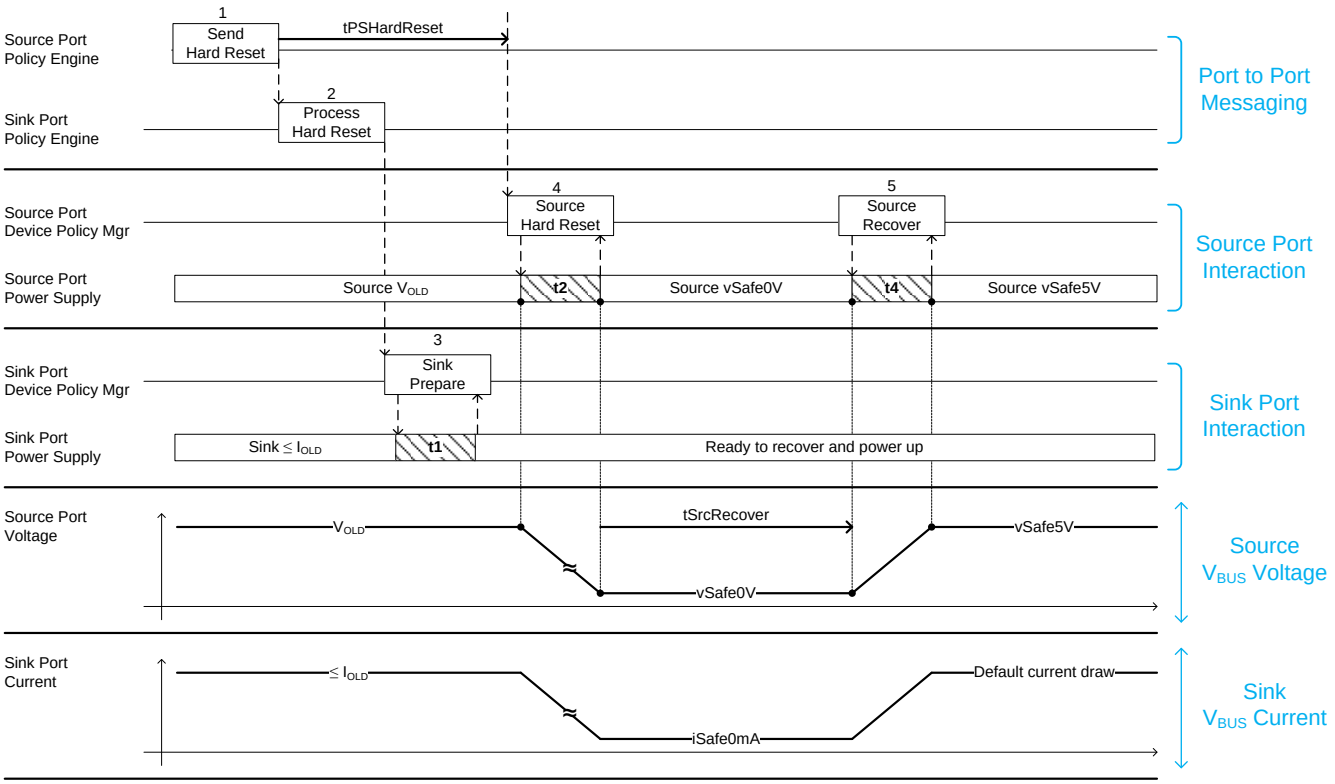
Step	Source Port	Sink Port
1	Policy Engine sends the ***GotoMin*** Message to the Sink.	Policy Engine receives the ***GotoMin*** Message and starts the ***PSTransitionTimer.***
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink. The Policy Engine tells the Device Policy Manager to instruct the power supply to modify its output power.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***GotoMin*** Message.
3		Policy Engine tells the Device Policy Manager to instruct the power supply to reduce power consumption, within ***tSnkNewPower*** (t1), to the pre-negotiated go to reduced power level); t1 shall complete before ***tSrcTransition***. The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.
4	***tSrcTransition*** after the ***GoodCRC*** Message was received the power supply starts to change its output power capability. The power supply shall be ready to operate at the new power level within ***tSrcReady*** (t2). The power supply informs the Device Policy Manager that it is ready to operate at the new power level. The power supply status is passed to the Policy Engine.	
5	The Policy Engine sends the ***PS_RDY*** Message to the Sink.	The Policy Engine receives the ***PS_RDY*** Message.
6	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine evaluates the ***PS_RDY*** Message from the Source and no further action is required.

Microchip Ex. 1020, p. 234
Microchip v. Aptiv
IPR2024-00496

7.3.12 Source Initiated Hard Reset

The interaction of the System Policy, Device Policy, and power supply that shall be followed during a Source Initiated Hard Reset is shown in Figure 7-22. The sequence that shall be followed is described in Table 7-12. The timing parameters that shall be applied are listed in Table 7-16 and Table 7-17.

Figure 7-22 Transition Diagram for a Source Initiated Hard Reset



Microchip Ex. 1020, p. 235
Microchip v. Aptiv
IPR2024-00496

Table 7-12 Sequence Description for a Source Initiated Hard Reset

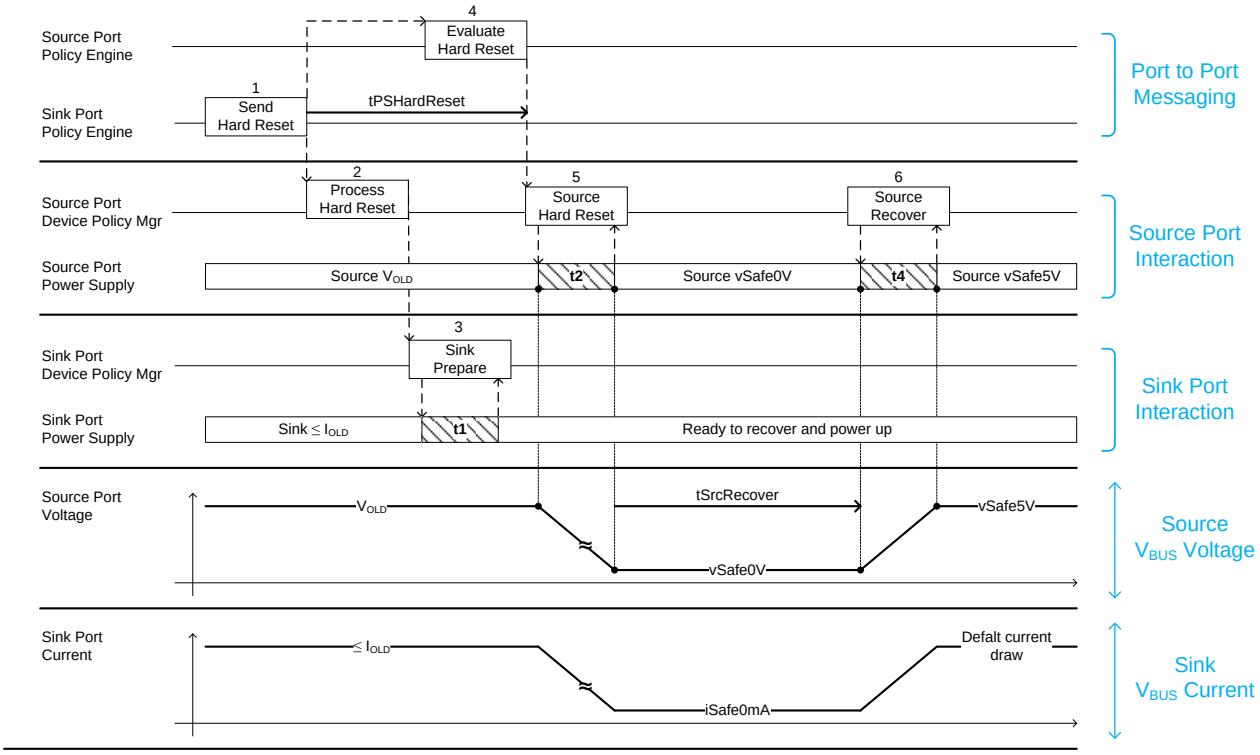
Step	Source Port	Sink Port
1	Policy Engine sends ***Hard Reset*** Signaling to the Sink.	Sink receives ***Hard Reset*** Signaling.
2		Policy Engine is informed of the Hard Reset. Policy Engine tells the Device Policy Manager to instruct the power supply to prepare for a Hard Reset.
3		The Sink prepares for the Hard Reset within ***tSnkHardResetPrepare*** (t1)) and passes an indication to the Device Policy Manger The Sink shall not draw more than ***iSafe0mA*** when V_{BUS} is driven to ***vSafe0V***.
4	Policy Engine waits ***tPSHardReset*** after sending ***Hard Reset*** Signaling and then tells the Device Policy Manager to instruct the power supply to perform a Hard Reset. The transition to ***vSafe0V*** shall occur within ***tSafe0V*** (t2).	
5	After ***tSrcRecover*** the Source applies power to V_{BUS} in an attempt to re-establish communication with the Sink and resume USB Default Operation. The transition to ***vSafe5V*** shall occur within ***tSrcTurnOn*** (t4).	The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.

Microchip Ex. 1020, p. 236
Microchip v. Aptiv
IPR2024-00496

7.3.13 Sink Initiated Hard Reset

The interaction of the System Policy, Device Policy, and power supply that shall be followed during a Sink Initiated Hard Reset is shown in Figure 7-23. The sequence that shall be followed is described in Table 7-13. The timing parameters that shall be followed are listed in Table 7-16 and Table 7-17.

Figure 7-23 Transition Diagram for a Sink Initiated Hard Reset



Microchip Ex. 1020, p. 237
Microchip v. Aptiv
IPR2024-00496

Table 7-13 Sequence Description for a Sink Initiated Hard Reset

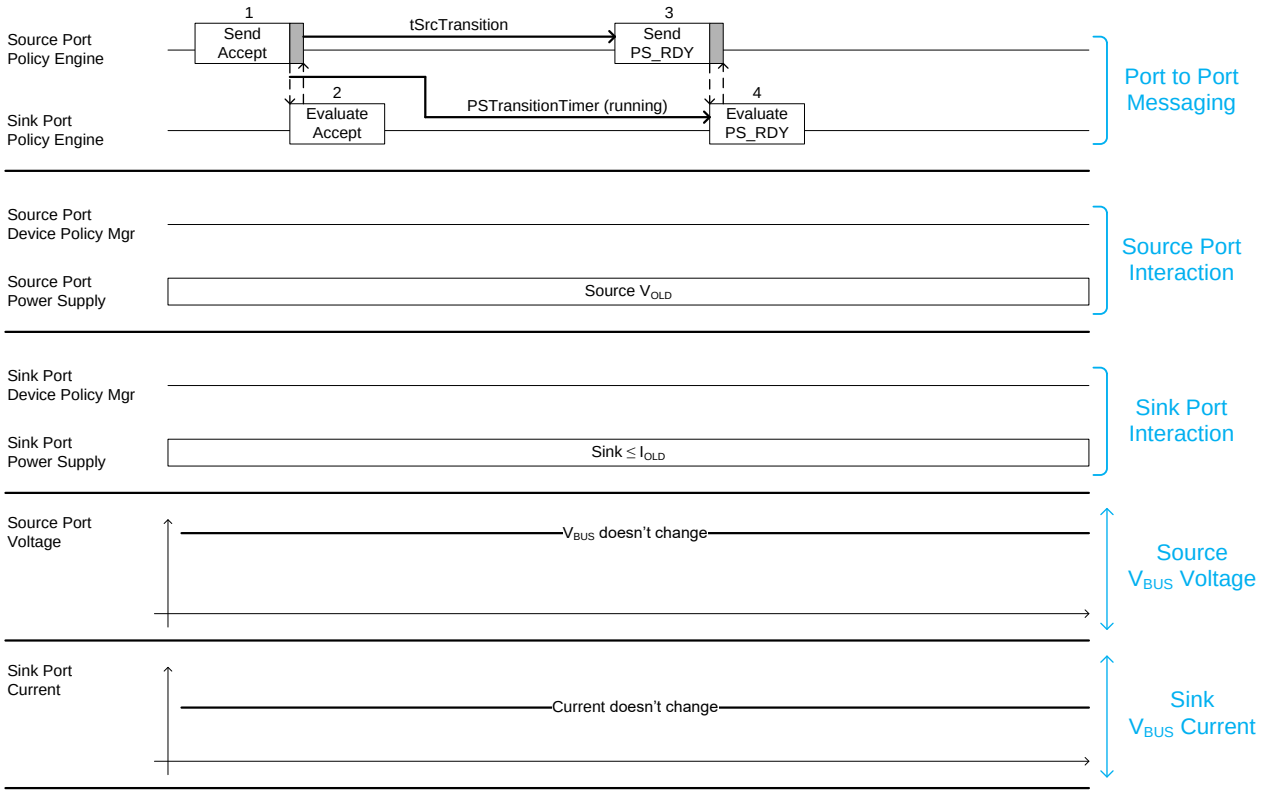
Step	Source Port	Sink Port
1		Policy Engine sends ***Hard Reset*** Signaling to the Source.
2		Policy Engine tells the Device Policy Manager to instruct the power supply to prepare for a Hard Reset.
3		The Sink prepares for the Hard Reset within ***tSnkHardResetPrepare*** (t1) and passes an indication to the Device Policy Manger. The Sink shall not draw more than ***iSafe0mA*** when V_{BUS} is driven to ***vSafe0V***.
4	Policy Engine is informed of the Hard Reset.	
5	Policy Engine waits ***tPSHardReset*** after receiving ***Hard Reset*** Signaling and then tells the Device Policy Manager to instruct the power supply to perform a Hard Reset. The transition to ***vSafe0V*** shall occur within ***tSafe0V*** (t2).	
6	After ***tSrcRecover*** the Source applies power to V_{BUS} in an attempt to re-establish communication with the Sink and resume USB Default Operation. The transition to ***vSafe5V*** shall occur within ***tSrcTurnOn*** (t4).	The Sink shall not violate the transient load behavior defined in Section 7.2.6 while transitioning to and operating at the new power level.

Microchip Ex. 1020, p. 238
Microchip v. Aptiv
IPR2024-00496

7.3.14 No change in Current or Voltage

The interaction of the System Policy, Device Policy, and power supply that shall be followed when the Sink requests the same Voltage and Current as it is currently operating at is shown in Figure 7-24. The sequence that shall be followed is described in Table 7-14. The timing parameters that shall be followed are listed in Table 7-16 and Table 7-17.

Figure 7-24 Transition Diagram for no change in Current or Voltage



Microchip Ex. 1020, p. 239
Microchip v. Aptiv
IPR2024-00496

Table 7-14 Sequence Description for no change in Current or Voltage

Step	Source Port	Sink Port
1	Policy Engine sends the ***Accept*** Message to the Sink.	Policy Engine receives the ***Accept*** Message and starts the ***PSTransitionTimer***.
2	Protocol Layer receives the ***GoodCRC*** Message from the Sink.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine then evaluates the ***Accept*** Message.
3	The Policy Engine waits ***tSrcTransition*** then sends the ***PS_RDY*** Message to the Sink.	Policy Engine receives the ***PS_RDY*** Message.
4	Policy Engine receives the ***GoodCRC*** Message from the Sink. Note: the decision that no power transition is required could be made either by the Device Policy Manager or the power supply depending on implementation.	Protocol Layer sends the ***GoodCRC*** Message to the Source. Policy Engine evaluates the ***PS_RDY*** Message.

Microchip Ex. 1020, p. 240
Microchip v. Aptiv
IPR2024-00496

7.3.15 Fast Role Swap

The interaction of the System Policy, Device Policy, and power supply that shall be followed during a Fast Role Swap is shown in Figure 7-25. The sequence that shall be followed is described in Table 7-15. The timing parameters that shall be followed are listed in Table 7-16 and Table 7-17. Negotiations between the new Source and the new Sink may occur after the new Source sends the final ***PS_RDY*** Message. Note: in Figure 7-25 and Table 7-15 numbers are used to indicate Message related steps and letters are used to indicate other events.

Figure 7-25 Transition Diagram for Fast Role Swap

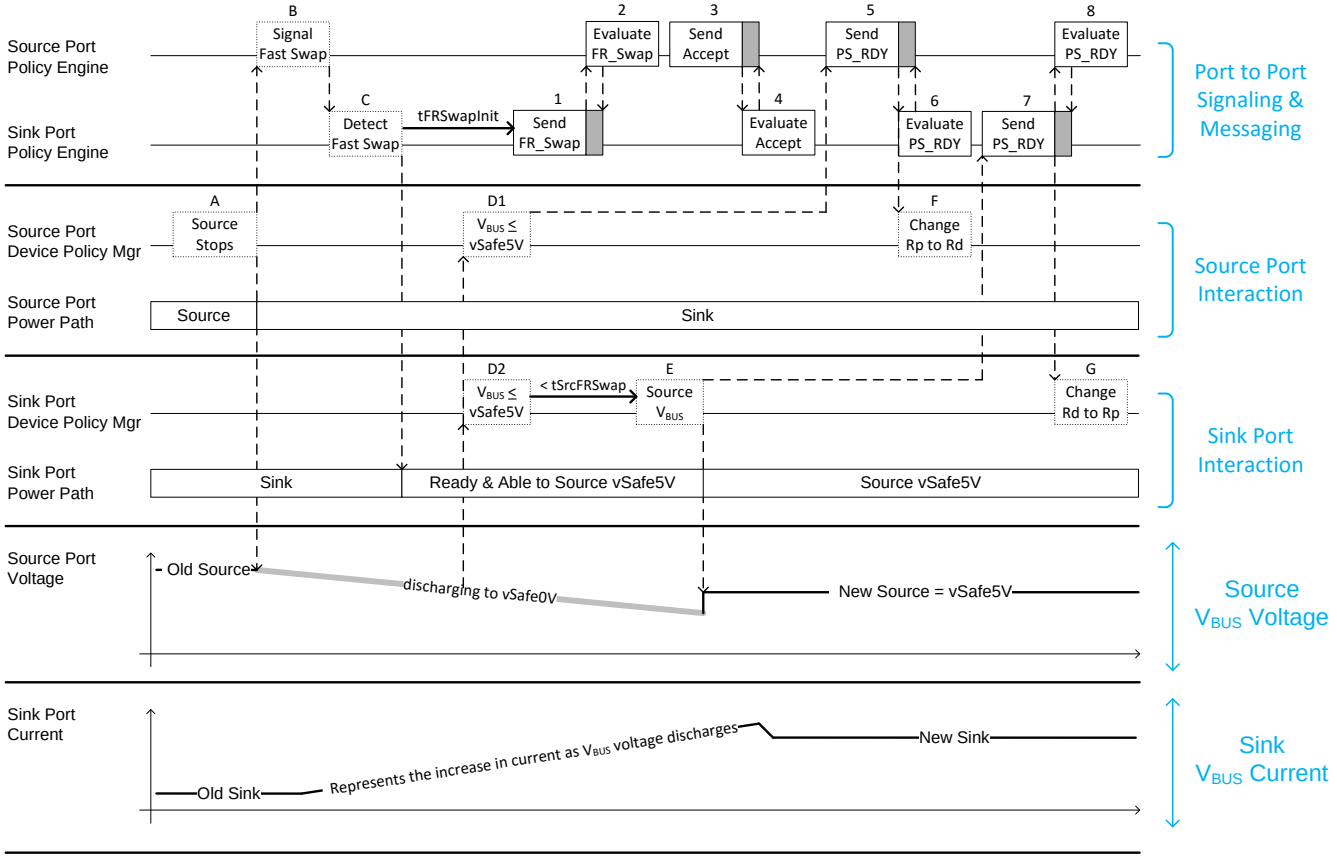


Table 7-15 Sequence Description for Fast Role Swap

Step	Initial Source Port→ New Sink Port	Initial Sink Port → New Source Port
A	The Source connected to the Hub UFP (see Figure 7-8) stops sourcing V_{BUS}.	
B	Policy Engine signals the Fast Role Swap to the initial Sink on the CC wire.	
C		Policy Engine detects the Fast Role swap signal on the CC wire from the initial Source and shall send the ***FR_Swap*** Message back to the initial Source (that is no longer powering V_{BUS}) within time ***tFRSwapInit***.
D1	The Policy engine monitors for V_{BUS} ≤ ***vSafe5V*** so that a ***PS_RDY*** Message can be sent to the new Source at Step 5 of the messaging sequence.	
D2		The Policy engine monitors for V_{BUS} ≤ ***vSafe5V*** so the initial Sink can assume the role of new Source and begin to source V_{BUS}.

Microchip Ex. 1020, p. 241
Microchip v. Aptiv
IPR2024-00496

Step	Initial Source Port→ New Sink Port	Initial Sink Port → New Source Port
E		When V_{BUS} = ***vSafe5V*** the new Source may provide power to $V_{BUS.}$ When V_{BUS} < ***vSafe5V*** the new Source shall provide power to V_{BUS} within ***tSrcFRSwap*** and the ***PS_RDY*** Message can be sent to the new Sink at Step 7 of the messaging sequence.
F	The CC termination is changed from Rp to Rd (see ***[USB Type-C 1.2]***) when the new Sink sends the ***PS_RDY*** Message of Step 5 to the new Source.	
G		The CC termination is changed from Rd to Rp (see ***[USB Type-C 1.2]***) when the new Source sends the ***PS_RDY*** Message of Step 7 to the new Sink.
1	Protocol Layer receives the ***GoodCRC*** Message from the initial Sink.	Policy Engine sends the ***FR_Swap*** Message to the initial Source(that is no longer powering V_{BUS}) after detecting the Fast Role Swap signal of Step C.
2	Policy Engine receives the ***FR_Swap*** Message from the initial Sink that is transitioning to be the new Source.	
3	Policy Engine sends an ***Accept*** Message to the initial Sink that is transitioning to be the new Source.	Protocol Layer receives the ***GoodCRC*** Message from the initial Source that is transitioning to be the new Sink
4		Policy Engine receives the ***Accept*** Message from the initial Source that is transitioning to be the new Sink.
5	Policy Engine sends a ***PS_RDY*** Message to the initial Sink that is transitioning to be the new Source. The Policy Engine shall wait for Step D1 before sending the ***PS_RDY*** Message.	Protocol Layer receives the ***GoodCRC*** Message from the initial Source that has completed the transition to new Sink.
6		Policy Engine receives the ***PS_RDY*** Message from the new Sink.
7	Protocol Layer receives the ***GoodCRC*** Message from the new Source.	Policy Engine sends a ***PS_RDY*** Message to the new Sink. The Policy Engine shall wait for Step E before sending the ***PS_RDY*** Message.
8	Policy Engine receives the ***PS_RDY*** Message from the new Source.	

Microchip Ex. 1020, p. 242
Microchip v. Aptiv
IPR2024-00496

7.4 Electrical Parameters

7.4.1 Source Electrical Parameters

The Source Electrical Parameters that shall be followed are specified in Table 7-16.

Table 7-16 Source Electrical Parameters

Parameter	Description	MIN	TYP	MAX	UNITS	Reference
***cSrcBulk*[1]**	Source bulk capacitance when a Port is powered from a dedicated supply.	10			µF	Section 7.1.2
***cSrcBulkShared*[1]**	Source bulk capacitance when a Port is powered from a shared supply.	120			µF	Section 7.1.2
tNewSnk	Time allowed for an initial Source in Swap Standby to transition new Sink operation.			15	ms	Figure 7-19, Figure 7-20
tSrcFRSwap	Time from the initial Sink detecting that V_{BUS} has dropped below ***vSafe5V*** until the initial Sink/new Source is able to supply USB Type-C Current (see ***[USB Type-C 1.2]***)			150	µs	Section 7.1.14
tSrcReady	Time from positive/negative transition start (t0) to when the Source is ready to provide the newly negotiated power level.			285	ms	Figure 7-2, Figure 7-3
tSrcRecover	Time allotted for the Source to recover.	0.66		1	s	Section 7.1.6
tSrcSettle	Time from positive/negative transition start (t0) to when the transitioning voltage is within the range ***vSrcNew.***			275	ms	Figure 7-2
tSrcSwapStdby	The maximum time for the Source to transition to Swap Standby.			650	ms	Table 7-9 Table 7-10
tSrcTransient	The maximum time for the Source output voltage to be between ***vSrcNew*** and ***vSrcValid*** in response to a load transient.			5	ms	Section 7.1.9
tSrcTransition	The time the Source shall wait before transitioning the power supply to ensure that the Sink has sufficient time to prepare.	25		35	ms	Section 7.3
tSrcTurnOn	Transition time from ***vSafe0V*** to ***vSafe5V***.			275	ms	Table 7-12 Table 7-13
vSrcNeg	Most negative voltage allowed during transition.			-0.3	V	Figure 7-4

Microchip Ex. 1020, p. 243
Microchip v. Aptiv
IPR2024-00496

Parameter	Description	MIN	TYP	MAX	UNITS	Reference
vSrcNew	Fixed Supply output measured at the Source receptacle.	PDO Voltage *0.95	PDO Voltage	PDO Voltage *1.05	V	Figure 7-2 Figure 7-3
	Variable Supply output measured at the Source receptacle.	PDO Minimum Voltage		PDO Maximum Voltage	V	
	Battery Supply output measured at the Source receptacle.	PDO Minimum Voltage		PDO Maximum Voltage	V	
vSrcPeak	The range that a Fixed Supply in Peak Current operation is allowed when overload conditions occur.	PDO Voltage *0.90		PDO Voltage *1.05	V	Table 6-9 Figure 7-6
vSrcSlewNeg	Maximum slew rate allowed for negative voltage transitions. Limits current based on a 3 A connector rating and maximum Sink bulk capacitance of 100 µF.			-30	mV/µs	Section 7.1.5 Figure 7-3
vSrcSlewPos	Maximum slew rate allowed for positive voltage transitions. Limits current based on a 3 A connector rating and maximum Sink bulk capacitance of 100 µF.			30	mV/µs	Section 7.1.4 Figure 7-2
vSrcValid	The range in addition to *vSrcNew* which a newly negotiated voltage is considered valid during and after a transition. This range also applies to *vSafe5V*.	-0.5		0.5	V	Figure 7-2 Figure 7-3
Note 1: The Source shall charge and discharge the total bulk capacitance to meet the transition time requirements.						

Microchip Ex. 1020, p. 244
Microchip v. Aptiv
IPR2024-00496

7.4.2 Sink Electrical Parameters

The Sink Electrical Parameters that shall be followed are specified in Table 7-17.

Table 7-17 Sink Electrical Parameters

Parameter	Description	MIN	TYP	MAX	UNITS	Reference
cSnkBulk[1]	Sink bulk capacitance on V_{BUS} at Attach.	1		10	µF	Section 7.2.2
cSnkBulkPd	Bulk capacitance on V_{BUS} a Sink is allowed after a successful negotiation.	1		100	µF	Section 7.2.2
iCapChange	Transient current allowed to flow when the Sink changes its bulk capacitance.			10	mA	Section 7.2.2
iLoadReleaseRate	Load release di/dt . Refer to ***[USB Type-C 1.2]*** Section 3.7.3.3.2 for cable details.	-150			mA/µs	Section 7.2.6
iLoadStepRate	Load step di/dt . Refer to ***[USB Type-C 1.2]*** Section 3.7.3.3.2 for cable details.			150	mA/µs	Section 7.2.6
iOvershoot	Positive or negative overshoot when a load change occurs less than or equal to ***iLoadStepRate***; relative to the settled value after the load change. Refer to USB ***[USB Type-C 1.2]*** Section 3.7.3.3.2 for cable details.	-230		230	mA	Section 7.2.6
iSafe0mA	Maximum current a Sink is allowed to draw when V_{BUS} is driven to ***vSafe0V***.			1.0	mA	Figure 7-22 Figure 7-23
iSnkSwapStdby	Maximum current a Sink can draw during Swap Standby. Ideally this current is very near to 0 mA largely influenced by Port leakage current.			2.5	mA	Section 7.2.7
pHubSusp	Suspend power consumption for a hub. 25mW + 25mW per downstream Port for up to 4 ports.			125	mW	Section 7.2.3
pSnkStdby	Maximum power consumption while in Sink Standby.			2.5	W	Section 7.2.3
pSnkSusp	Suspend power consumption for a peripheral device.			25	mW	Section 7.2.3
tNewSrc	Maximum time allowed for an initial Sink in Swap Standby to transition to new Source operation.			275	ms	Section 7.2.7 Table 7-9 Table 7-10
tSnkHardResetPrepare	Time allotted for the Sink power electronics to prepare for a Hard Reset.			15	ms	Table 7-13

Microchip Ex. 1020, p. 245
Microchip v. Aptiv
IPR2024-00496

Parameter	Description	MIN	TYP	MAX	UNITS	Reference
tSnkNewPower	Maximum transition time between power levels.			15	ms	Section 7.2.3
tSnkRecover	Time for the Sink to resume USB Default Operation.			150	ms	Table 7-12
tSnkStdby	Time to transition to Sink Standby from Sink.			15	ms	Section 7.2.3
tSnkSwapStdby	Maximum time for the Sink to transition to Swap Standby.			15	ms	Section 7.2.7
Note 1: If more bypass capacitance than *cSnkBulk* max or *cSnkBulkPd* max is required in the device, then the device must incorporate some form of V_{BUS} surge current limiting as described in *[USB 3.1]* Section 11.4.4.1.						

7.4.3 Common Electrical Parameters

Electrical Parameters that are common to both the Source and the Sink that shall be followed are specified in Table 7-18.

Table 7-18 Common Source/Sink Electrical Parameters

Parameter	Description	MIN	TYP	MAX	UNITS	Reference
tSafe0V	Time to reach *vSafe0V* max.			650	ms	Section 7.1.6 Figure 7-4 Table 7-12 Table 7-13
tSafe5V	Time to reach *vSafe5V* max.			275	ms	Section 7.1.5 Figure 7-4
vSafe0V	Safe operating voltage at "zero volts".	0		0.8	V	Section 7.1.6
vSafe5V	Safe operating voltage at 5V. See *[USB 2.0]* and *[USB 3.1]* for allowable V_{BUS} voltage range.	*[USB 2.0]*/ *[USB 3.1]*		*[USB 2.0]*/ *[USB 3.1]*	V	Section 7.1.6

Microchip Ex. 1020, p. 246
Microchip v. Aptiv
IPR2024-00496

8. Device Policy

8.1 Overview

This section describes the Device Policy and Policy Engine that implements it. For an overview of the architecture and how the Device Policy Manager fits into this architecture, please see Section 2.7.

8.2 Device Policy Manager

The Device Policy Manager is responsible for managing the power used by one or more USB Power Delivery ports. In order to have sufficient knowledge to complete this task it needs relevant information about the device it resides in. Firstly it has a priori knowledge of the device including the capabilities of the power supply and the receptacles on each Port since these will for example have specific current ratings. It also has to know information from the USB-C Port Control module regarding cable insertion, type and rating of cable etc. It also has to have information from the power supply about changes in its capabilities as well as being able to request power supply changes. With all of this information the Device Policy Manager is able to provide up to date information regarding the capabilities available to a specific Port and to manage the power resources within the device.

When working out the capabilities for a given Source Port the Device Policy Manager will take into account firstly the current rating of the Port's receptacle and whether the inserted cable is PD or non-PD rated and if so what is the capability of the plug. This will set an upper bound for the capabilities which may be offered. After this the Device Policy Manager will consider the available power supply resources since this will bound which voltages and currents may be offered. Finally, the Device Policy Manager will consider what power is currently allocated to other ports, which power is in the Power Reserve and any other amendments to Policy from the System Policy Manager. The Device Policy Manager will offer a set of capabilities within the bounds detailed above.

When selecting a capability for a given Sink Port the Device Policy Manager will look at the capabilities offered by the Source. This will set an upper bound for the capabilities which may be requested. The Device Policy Manager will also consider which capabilities are required by the Sink in order to operate. If an appropriate match for Voltage and Current can be found within the limits of the receptacle and cable then this will be requested from the Source. If an appropriate match cannot be found then a request for an offered voltage and current will be made, along with an indication of a capability mismatch.

For Dual-Role Power Ports the Device Policy Manager manages the functionality of both a Source and a Sink. In addition it is able to manage the Power Role Swap process between the two. In terms of power management this could mean that a Port which is initially consuming power as a Sink is able to become a power resource as a Source. Conversely, Attached Sources may request that power be provided to them.

The functionality within the Device Policy Manager (and to a certain extent the Policy Engine) is scalable depending on the complexity of the device, including the number of different power supply capabilities and the number of different features supported for example System Policy Manager interface or Capability Mismatch, and the number of ports being managed. Within these parameters it is possible to implement devices from very simple power supplies to more complex power supplies or devices such as USB hubs or Hard Drives. Within multiport devices it is also permitted to have a combination of USB Power Delivery and non-USB Power Delivery ports which should all be managed by the Device Policy Manager.

As noted in Section 2.7 the logical architecture used in the PD specification will vary depending on the implementation. This means that different implementations of the Device Policy Manager may be relative small or large depending on the complexity of the device, as indicated above. It is also possible to allocate different responsibilities between the Policy Engine and the Device Policy Manager, which will lead to different types of architectures and interfaces.

The Device Policy Manager is responsible for the following:

- Maintaining the Local Policy for the device
- For a Source, monitoring the present capabilities and triggering notifications of the change.
- For a Sink, evaluating and responding to capabilities related requests from the Policy Engine for a given Port.

Microchip Ex. 1020, p. 247
Microchip v. Aptiv
IPR2024-00496

- Control of the Source/Sink in the device.
- Control of the USB-C Port Control module for each Port.
- Interface to the Policy Engine for a given Port.

The Device Policy Manager is responsible for the following ***Optional*** features when implemented:

- Communications with the System Policy over USB.
- For Sources with multiple ports monitoring and balancing power requirements across these ports.
- Monitoring of batteries and AC power supplies.
- Managing Modes in its Port Partner and Cable Plug(s)

8.2.1 Capabilities

The Device Policy Manager in a Provider shall know the power supplies available in the device and their capabilities. In addition it shall be aware of any other PD Sources of power such as batteries and AC inputs. The available power sources and existing demands on the device shall be taken into account when presenting capabilities to a Sink.

The Device Policy Manager in a Consumer shall know the requirements of the Sink and use this to evaluate the capabilities offered by a Source. It shall be aware of its own power sources e.g. Batteries or AC supplies where these have a bearing on its operation as a Sink.

The Device Policy Manager in a Dual-Role Power Device shall combine the above capabilities and shall also be able to present the dual-role nature of the device to an Attached PD Capable device.

8.2.2 System Policy

A given PD Capable device may have no USB capability, or PD may have been added to a USB device in such a way that PD is not integrated with USB. In these two cases there shall be no requirement for the Device Policy Manager to interact with the USB interface of the device. The following requirements shall only apply to PD devices that expose PD functionality over USB.

The Device Policy Manager shall communicate over USB with the System Policy Manager according to the requirements detailed in ***[USBTypeCBridge 1.0]***. Whenever requested the Device Policy Manager shall implement a Local Policy according to that requested by the System Policy Manager. For example the System Policy Manager may request that a battery powered Device temporarily stops charging so that there is sufficient power for a HDD to spin up.

Note: that due to timing constraints, a PD Capable device shall be able to respond autonomously to all time-critical PD related requests.

8.2.3 Control of Source/Sink

The Device Policy Manager for a Provider shall manage the power supply for each PD Source Port and shall know at any given time what the negotiated power is. It shall request transitions of the supply and inform the Policy Engine whenever a transition completes.

The Device Policy Manager for a Consumer shall manage the Sink for each PD Sink Port and shall know at any given time what the negotiated power is.

The Device Policy Manager for a Dual-Role Power Device shall manage the transition between Source/Sink roles for each PD Dual-Role Power Port and shall know at any given time what operational role the Port is in.

8.2.4 Cable Detection

8.2.4.1 Device Policy Manager in a Provider

The Device Policy Manager in the Provider shall control the USB-C Port Control module and shall be able to use the USB-C Port Control module to determine the Attachment status.

Microchip Ex. 1020, p. 248
Microchip v. Aptiv
IPR2024-00496

Note: that it may be necessary for the Device Policy Manager to also initiate additional discovery using the ***Discover Identity*** Command in order to determine the full capabilities of the cabling (see Section 6.4.4.2).

8.2.4.2 Device Policy Manager in a Consumer

The Device Policy Manager in a Consumer controls the USB-C Port Control module and shall be able to use the USB-C Port Control module to determine the Attachment status.

8.2.4.3 Device Policy Manager in a Consumer/Provider

The Device Policy Manager in a Consumer/Provider inherits characteristics of Consumers and Providers and shall control the USB-C Port Control module in order to support the Dead Battery back-powering case to determine the following for a given Port:

- Attachment of a USB Power Delivery Provider/Consumer which supports Dead Battery back-powering
- Presence of V_{BUS}.

8.2.4.4 Device Policy Manager in a Provider/Consumer

The Device Policy Manager in a Provider/Consumer inherits characteristics of Consumers and Providers and may control the USB-C Port Control module in order to support the Dead Battery back-powering case to determine the following for a given Port:

- Presence of V_{BUS}.

8.2.5 Managing Power Requirements

The Device Policy Manager in a Provider shall be aware of the power requirements of all devices connected to its Source Ports. This includes being aware of any reserve power that may be required by devices in the future and ensuring that power is shared optimally amongst Attached PD Capable devices. This is a key function of the Device Policy Manager, whose implementation is critical to ensuring that all PD Capable devices get the power they require in a timely fashion in order to facilitate smooth operation. This is balanced by the fact that the Device Policy Manager is responsible for managing the sources of power that are, by definition, finite.

The Consumer's Device Policy Manager shall ensure that it takes no more power than is required to perform its functions and gives back unneeded power whenever possible (in such cases the Provider shall maintain a Power Reserve to ensure future operation is possible).

8.2.5.1 Managing the Power Reserve

There may be some products where a Device has certain functionality at one power level and a greater functionality at another, for example a Printer/Scanner that operates only as a printer with one power level and as a scanner if it can get more power. Visibility of the linkage between power and functionality will only be apparent at the USB Host; however the Device Policy Manager provides the mechanisms to manage the power requirements of such Devices.

Devices with the GiveBack flag cleared report Operating Current and Maximum Operating Current (see Section 6.4.2). For many Devices the Operating Current and the Maximum Operating Current will be the same. Devices with highly variable loads, such as Hard Disk Drives, may use Maximum Operating Current.

Devices with the GiveBack flag set report Operating Current and Minimum Operating Current (see Section 6.4.2). For many Devices the Operating Current and the Minimum Operating Current will be the same. Devices that charge their own batteries may use the Minimum Operating Current and GiveBack flag.

For example in the first case, a mobile device may require 500mA to operate, but would like an additional 1000mA to charge its Battery. The mobile device would set the GiveBack flag (see Section 6.4.2.2) and request 500mA in the Minimum Operating Current field and 1500mA in the Operating Current field (provided that 1500mA was offered by the Source) indicating to the Provider that it could temporarily recover the 1000mA to meet a transitory request.

In the second case, a Hard Disk Drive (HDD) may require 2A to spin-up, but only 1A to operate. At startup the HDD would request Maximum Operating Current of 2A and an Operating Current of 2A. After the drive is spun-up and

Microchip Ex. 1020, p. 249
Microchip v. Aptiv
IPR2024-00496

ready to operate it would make another request of 1A for its Operating Current and 2A for its Maximum Operating Current. Over time, its inactivity timers may expire and the HDD will go to a lower power state. When the HDD is next accessed, it has to spin-up again. So it will request an Operating Current of 2A and a Maximum Operating Current of 2A. The Provider may have the extra power available immediately and can immediately honor the request. If the power is not available, the Provider may have to harvest power, for example use the ***GotoMin*** Message to get back some power before honoring the HDD's request. In such a case, the HDD would be told to wait using a ***Wait*** Message. The HDD continues to Request additional power until the request is finally granted.

It shall be the Device Policy Manager's responsibility to allocate power and maintain a Power Reserve so as not to over-subscribe its available power resource. A Device with multiple ports such as a Hub shall always be able to meet the incremental demands of the Port requiring the highest incremental power from its Power Reserve.

The ***GotoMin*** Message is designed to allow the Provider to reclaim power from one Port to support a Consumer on another Port that temporarily requires additional power to perform some short term operation. In the example above, the mobile device that is being charged reduces its charge rate to allow a Device Policy Manager to meet a request from an HDD for start-up current required to spin-up its platters. Any power which is available to be reclaimed using a ***GotoMin*** Message may be counted as part of the Power Reserve.

A Consumer requesting power shall take into account its operational requirements when advertising its ability to temporarily return power. For example, a mobile device with a Dead Battery that is being used to make a call should make a request that retains sufficient power to continue the call. When the Consumer's requirements change, it shall re-negotiate its power to reflect the changed requirements.

8.2.5.2 Power Capability Mismatch

A capability mismatch occurs when a Consumer cannot obtain required power from a Provider (or the Source is not PD Capable) and the Consumer requires such capabilities to operate. Different actions are taken by the Device Policy Manager and the System Policy Manager in this case.

8.2.5.2.1 Local device handling of mismatch

The Consumer's Device Policy Manager shall cause a Message to be displayed to the end user that a power capability mismatch has occurred. Examples of such feedback can include:

- For a simple Device an LED may be used to indicate the failure. For example, during connection the LED could be solid amber. If the connection is successful the LED could change to green. If the connection fails it could be red or alternately blink amber.
- A more sophisticated Device with a user interface, e.g., a mobile device or monitor, should provide notification through the user interface on the Device.

The Provider's Device Policy Manager may cause a Message to be displayed to the user of the power capability mismatch.

8.2.5.2.2 Device Policy Manager Communication with System Policy

In a USB Power Delivery aware system with an active System Policy manager (see Section 8.2.2), the Device Policy Manager shall notify the System Policy Manager of the mismatch. This information shall be passed back to the System Policy Manager using the mechanisms described in Chapter 8.3.3. The System Policy Manager should ensure that the user is informed of the condition. When another Port in the system could satisfy the Consumer's power requirements the user should be directed to move the Device to the alternate Port.

In order to identify a more suitable Source Port for the Consumer the System Policy Manager shall communicate with the Device Policy Manager in order to determine the Consumer's requirements. The Device Policy Manager shall use a ***Get_Sink_Cap*** Message (see Section 6.3.8) to discover which power levels can be utilized by the Consumer.

8.2.6 Use of "Externally Powered" bit with Batteries and AC supplies

The Device Policy Manager in a Provider or Consumer may monitor the status of any variable sources of power that could have an impact on its capabilities as a Source such as Batteries and AC supplies and reflect this in the "Externally

Microchip Ex. 1020, p. 250
Microchip v. Aptiv
IPR2024-00496

Powered" bit (see Section 6.4.1.3.1.3) provided as part of the Source or Sink Capabilities Message (see Section 6.4.1). When monitored, and a USB interface is supported, the External Power status (see ***[USBTypeCBridge 1.0]***) and the Battery state (see Section 9.4.1) shall also be reported to the System Policy Manager using the USB interface.

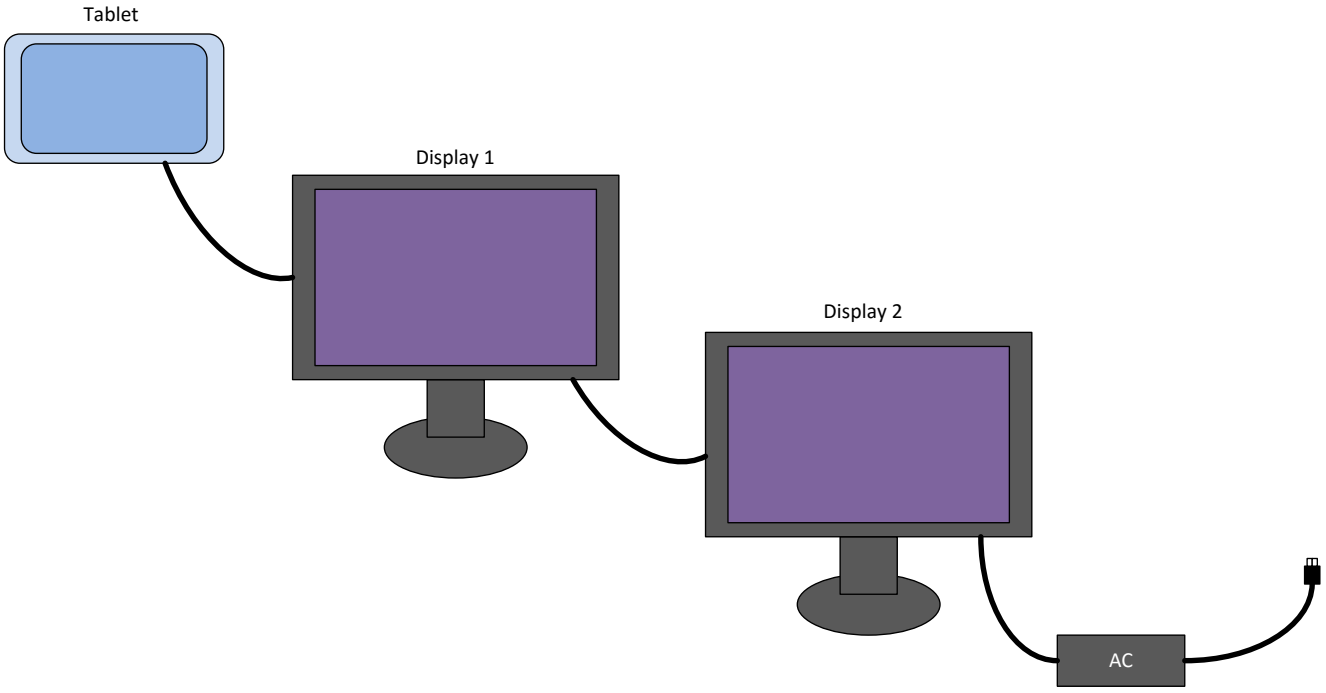
8.2.6.1 AC Supplies

The Externally Powered bit provided by Sources and Sinks (see Section 6.4.1.2.2.3 and Section 6.4.1.3.1.3) refers to the presence of an external AC power supply (i.e. from a Wall Wart) that is providing 100% of the power to a given device. This means that the device is solely getting its power from this power supply and is not aggregating power from other, non-external, power sources. Logically this power can either be from an AC supply directly connected to the device or from an AC supply connected to an Attached device, which is also getting 100% of its power from this power supply. The Externally Powered bit is in this way communicated through a PD system indicating that the origin of the power is from a single or multiple AC supplies:

- If the "Externally Powered" bit is set then that power is originally sourced from an AC supply.
- This concept applies as the power is routed through multiple provider and Consumer tiers, so, as an example. Power provided out of a monitor that is connected to a monitor that gets power from an AC supply, will claim it is "Externally Powered" even though it is not directly connected to the AC supply.

An example use case is a Tablet computer that is used with two USB A/V displays that are daisy chained (see Figure 8-1). The tablet and 1st display are not externally powered, (meaning, they have no source of power outside of USB PD). The 2nd display has an external supply Attached which may either be a USB PD based supply or some other form of external supply. When the displays are connected as shown, the power adapter Attached to the 2nd display is able to power both the 1st display and the tablet. In this case the 2nd display will indicate the presence of the Wall Wart, to the 1st display, by setting its "Externally Powered" bit. The 1st display will then in turn indicate the presence of an external supply to the tablet by setting its "Externally Powered" bit. Power is transmitted through the system to all devices, provided that there is sufficient power available from the external supply.

Figure 8-1 Example of daisy chained displays



8.2.6.2 Battery Supplies

When monitored, and a USB interface is supported, the Battery state shall be reported to the System Policy Manager using the USB interface.

Microchip Ex. 1020, p. 251
Microchip v. Aptiv
IPR2024-00496

A simplified algorithm is detailed below to ensure that Battery powered devices will get charge from non-Battery powered devices when possible, and also to ensure that devices do not constantly Power Role Swap back and forth. When two devices are connected that are not Externally Powered, they should define their own policies so as to prevent constant Power Role Swapping.

This algorithm uses the "Externally Powered" bit (see Section 6.4.1.2.2.3 and Section 6.4.1.3.1.3), thus the decisions are based on the availability of an external supply, not the full capabilities of a system or device or product.

Recommendations:

1. Provider/Consumers using external sources ("Externally powered" bit set) should always deny Power Role Swap requests from Consumer/Providers not using external sources ("Externally Powered" bit cleared).
2. Provider/Consumers not using external sources ("Externally Powered" bit cleared) should always accept a Power Role Swap request from a Consumer/Provider using external power sources ("Externally Powered" bit set) unless the requester is not able to provide the requirements of the present Provider/Consumer.

8.2.7 Interface to the Policy Engine

The Device Policy Manager shall maintain an interface to the Policy Engine for each Port in the device.

8.2.7.1 Device Policy Manager in a Provider

The Device Policy Manager in a Provider shall also provide the following functions to the Policy Engine:

- Inform the Policy Engine of changes in cable/ device Attachment status for a given cable.
- Inform the Policy Engine whenever the Source capabilities available for a Port change.
- Evaluate requests from an Attached Consumer and provide responses to the Policy Engine.
- Respond to requests for power supply transitions from the Policy Engine.
- Indication to Policy Engine when power supply transitions are complete.
- Maintain a Power Reserve for devices operating on a Port at less than maximum power.

8.2.7.2 Device Policy Manager in a Consumer

The Device Policy Manager in a Consumer shall also provide the following functions to the Policy Engine:

- Inform the Policy Engine of changes in cable/device Attachment status.
- Inform the Policy Engine whenever the power requirements for a Port change.
- Evaluate Source capabilities and provide suitable responses:
    - Request from offered capabilities
    - Indicate whether additional power is required
- Respond to requests for Sink transitions from the Policy Engine.

8.2.7.3 Device Policy Manager in a Dual-Role Power Device

The Device Policy Manager in a Dual-Role Power Device shall provide the following functions to the Policy Engine:

- Provider Device Policy Manager
- Consumer Device Policy Manager
- Interface for the Policy Engine to request power supply transitions from Source to Sink and vice versa.
- Indications to Policy Engine during Power Role Swap transitions.

8.2.7.4 Device Policy Manager in a Dual-Role Power Device Dead Battery handling

The Device Policy Manager in a Dual-Role Power Device with a Dead Battery should:

- switch Ports to Sink-only or Sinking DFP operation to obtain power from the next Attached Source
- use V_{BUS} from the Attached Source to power the USB Power Delivery communications as well as charging to enable the negotiation of higher input power.

Microchip Ex. 1020, p. 252
Microchip v. Aptiv
IPR2024-00496

8.3 Policy Engine

8.3.1 Introduction

There is one Policy Engine instance per Port that interacts with the Device Policy Manager in order to implement the present Local Policy for that particular Port. This section includes:

- Message sequences for various operations
- State diagrams covering operation of Sources, Sinks and Cable Plugs

8.3.2 Atomic Message Sequence Diagrams

8.3.2.1 Introduction

The Device Policy Engine drives the Message sequences and responses based on both the expected Message sequences and the present Local Policy.

An AMS shall be defined as a Message sequence that starts and/or ends in either the ***PE_SRC_Ready***, ***PE_SNK_Ready*** or ***PE_CBL_Ready*** states (see Section 8.3.3.2, Section 8.3.3.3 and Section 8.3.3.21).

In addition the Cable Plug discovery sequence specified in Section 8.3.3.21.3 shall be defined as an AMS.

The Source and Sink indicate to the Protocol Layer when an AMS starts and ends on entry to/exit from ***PE_SRC_Ready*** or ***PE_SNK_Ready*** (see Section 8.3.3.2 and Section 8.3.3.3).

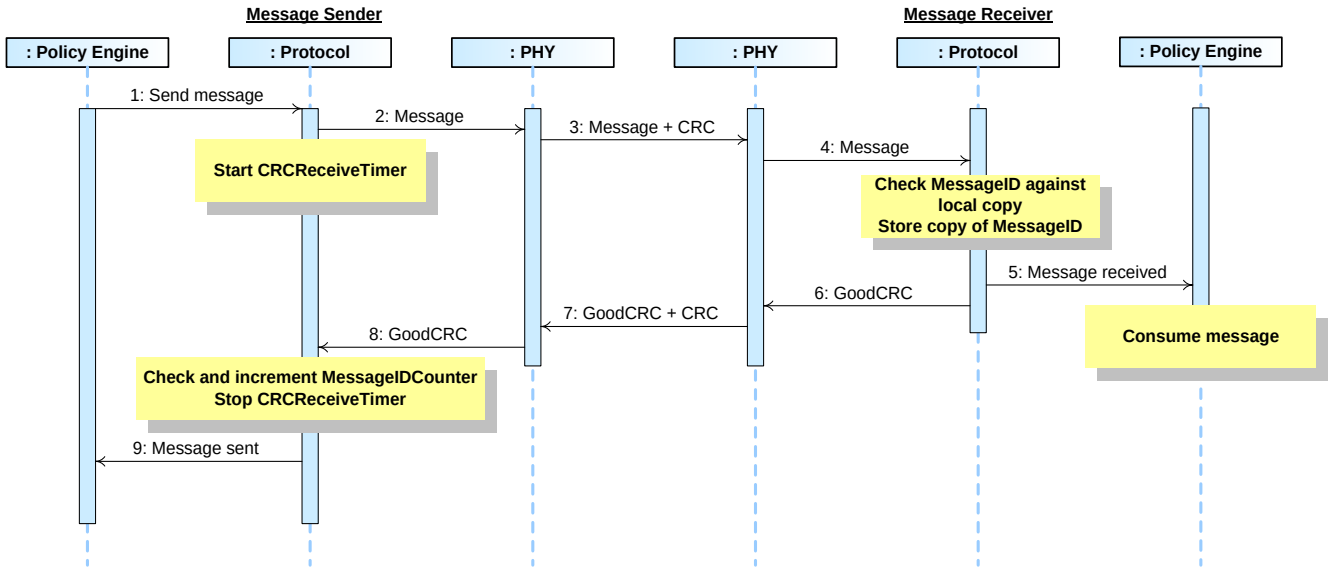
Section 8.3.2.1.3 gives details of which of these AMS's are interruptible or non-interruptible.

This section contains sequence diagrams that highlight some of the more interesting transactions. It is by no means a complete summary of all possible combinations, but is illustrative in nature.

8.3.2.1.1 Basic Message Exchange

Figure 8-2 Basic Message Exchange (Successful) below illustrates how a Message is sent. Note that the sender may be either a Source or Sink while the receiver may be either a Sink or Source. The basic Message sequence is the same. It starts when the Message Sender's Protocol Layer at the behest of its Policy Engine forms a Message that it passes to the Physical Layer.

Figure 8-2 Basic Message Exchange (Successful)



Microchip Ex. 1020, p. 253
Microchip v. Aptiv
IPR2024-00496

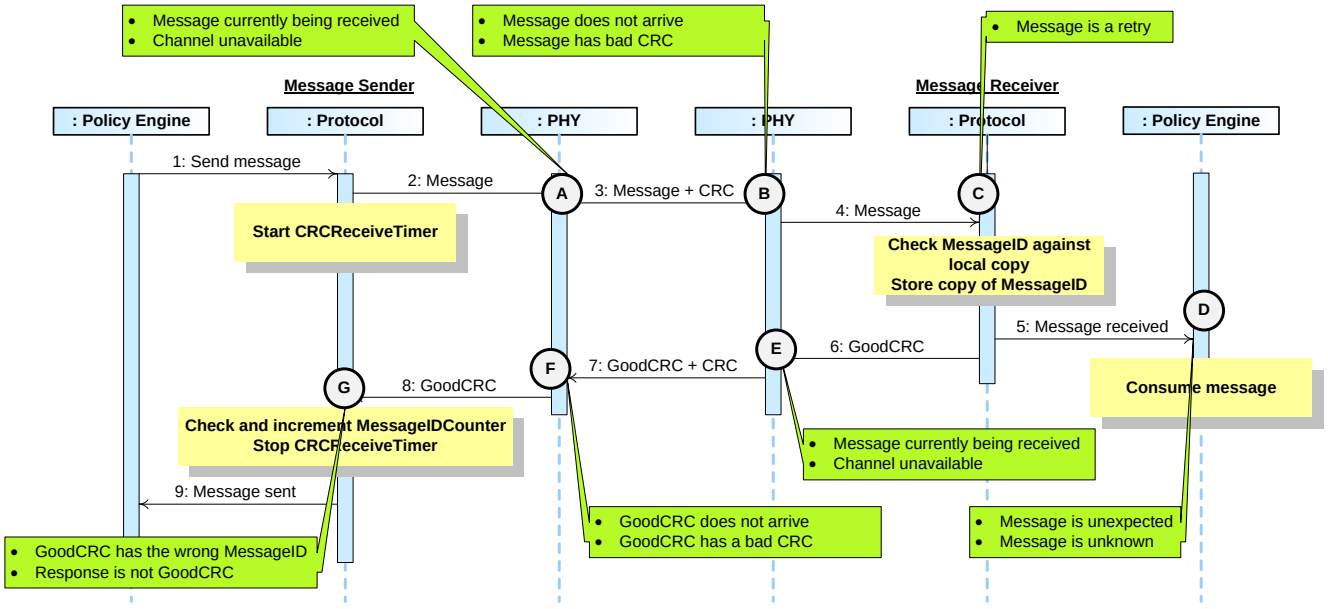
Table 8-1 Basic Message Flow

Step	Message Sender	Message Receiver
1	Policy Engine directs Protocol Layer to send a Message.	
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends a CRC and sends the Message.	Physical Layer receives the Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. Protocol Layer forwards the received Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it to the Physical Layer.
7	Physical Layer receives the Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer. Protocol Layer checks and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***.	
9	Protocol Layer informs the Policy Engine that the Message was successfully sent.	

8.3.2.1.2 Errors in Basic Message flow

There are various points during the Message flow where failures in communication or other issues can occur. Figure 8-3 is an annotated version of Figure 8-2 indicating at which point issues can occur.

Figure 8-3 Basic Message flow indicating possible errors



Microchip Ex. 1020, p. 254
Microchip v. Aptiv
IPR2024-00496

Table 8-2 Potential issues in Basic Message Flow

Point	Possible issues
A	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	1. Message does not arrive at the Physical Layer due to noise on the channel. 2. Message arrives but has been corrupted and has a bad CRC. 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.
C	1. ***MessageID*** of received Message matches stored ***MessageID*** so this is a retry. Message is not passed up to the Policy Engine.
D	1. Policy Engine receives a known Message that it was not expecting. 2. Policy Engine receives an unknown (unrecognized) Message. These cases are errors in the protocol which leads to the generation of a ***Soft_Reset*** Message.
E	Same as point A but at the Message Receiver side.
F	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. A ***GoodCRC*** Message is not received by the Message Sender's Protocol Layer. This leads to a ***CRCReceiveTimer*** timeout in the Message Sender.
G	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). 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).

Figure 8-4 illustrates one of these cases; the basic Message flow with a retry due to a bad CRC at the Message Receiver. It starts when the Message Sender's Protocol Layer at the behest of its Policy Engine forms a Message that it passes to the Physical Layer. The Protocol Layer is responsible for retries on a "'n' strikes and you are out" basis (***nRetryCount***).

Microchip Ex. 1020, p. 255
Microchip v. Aptiv
IPR2024-00496

Figure 8-4 Basic Message Flow with Bad CRC followed by a Retry

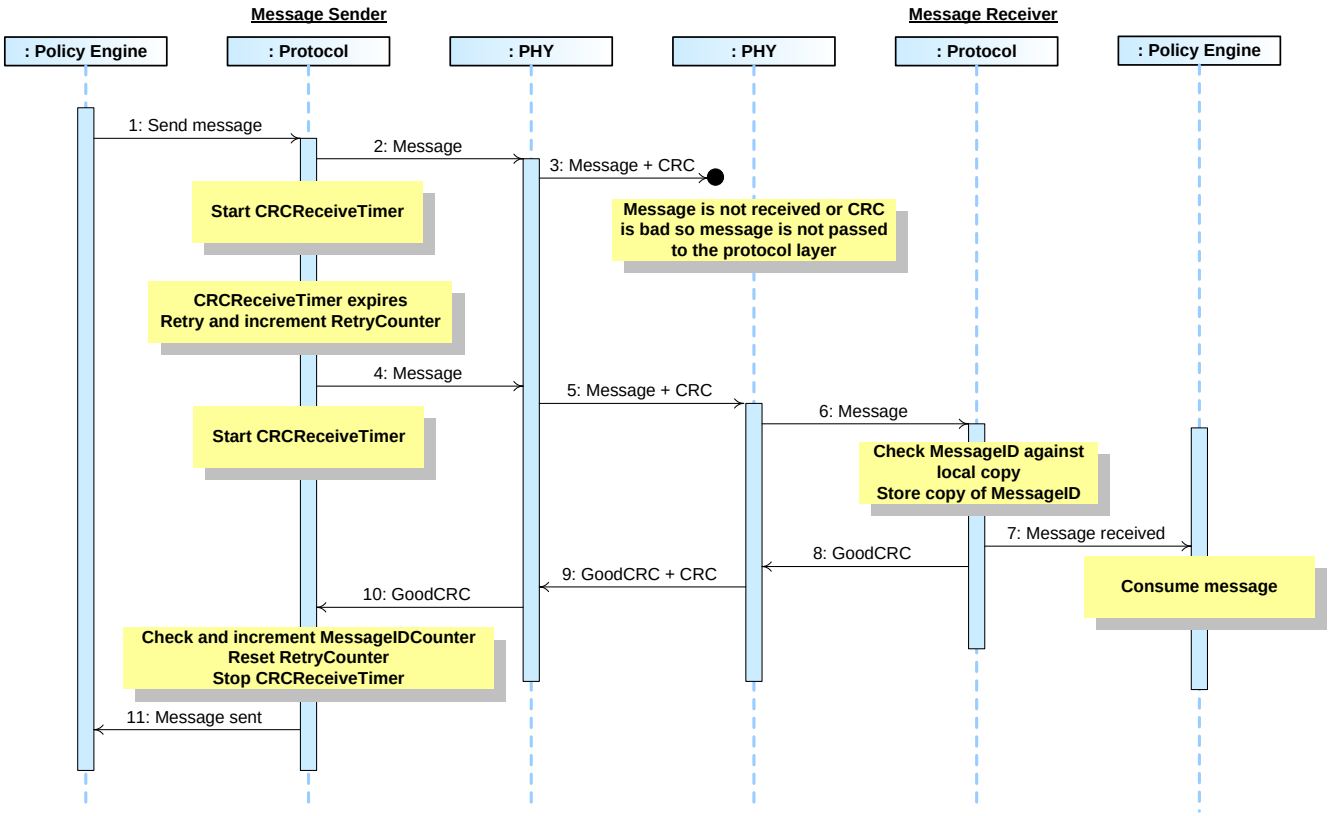


Table 8-3 Basic Message Flow with CRC failure

Step	Message Sender	Message Receiver
1	Policy Engine directs Protocol Layer to send a Message.	
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends a CRC and sends the Message.	Physical Layer receives no Message or a Message with an incorrect CRC. Nothing is passed to Protocol Layer.
4	Since no response is received, the ***CRCReceiveTimer*** will expire and trigger the first retry by the Protocol Layer. The ***RetryCounter*** is incremented. Protocol Layer passes the Message to the Physical Layer. Starts ***CRCReceiveTimer***.	
5	Physical Layer appends a CRC and sends the Message.	Physical Layer receives the Message and checks the CRC to verify the Message.
6		Physical Layer removes the CRC and forwards the Message to the Protocol Layer.
7		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. Protocol Layer forwards the received Message information to the Policy Engine that consumes it.
8		Protocol Layer generates a ***GoodCRC*** Message and passes it to the Physical Layer.

Microchip Ex. 1020, p. 256
Microchip v. Aptiv
IPR2024-00496

Step	Message Sender	Message Receiver
9	Physical Layer receives the Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
10	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
11	Protocol Layer verifies the ***MessageID***, stops ***CRCReceiveTimer*** and resets the ***RetryCounter***. Protocol Layer informs the Policy Engine that the Message was successfully sent.	

8.3.2.1.3 Interruptible and Non-interruptible Atomic Message Sequences

Table 8-4 details which AMS (as defined in Section 8.3.2) shall be treated as Interruptible or Non-interruptible during the sequence. Every AMS which starts with the same Message shall obey the Interruptible/Non-interruptible requirement. Note that every AMS is Interruptible until the first Message in the sequence has been successfully sent (***GoodCRC*** Message received). Any Sequence of VDMs shall be Interruptible. After the AMS that caused the interruption has completed, if the original AMS is still needed the interrupted AMS shall be Re-run.

Table 8-4 Interruptible and Non-interruptible AMS

AMS	Interruptible	Reference
Power Negotiation	No	Section 8.3.3.2, Section 8.3.3.3
GotoMin	No	Section 8.3.3.2, Section 8.3.3.3
Soft Reset	No	Section 8.3.3.4
Hard Reset	No	Section 8.3.3.2, Section 8.3.3.3
Cable Reset	No	Section 8.3.3.21.2.3
Get Source Capabilities	No	Section 8.3.3.2, Section 8.3.3.3
Get Sink Capabilities	No	Section 8.3.3.2, Section 8.3.3.3
Power Role Swap	No	Section 8.3.3.15.3, Section 8.3.3.15.4
Fast Role Swap	No	Section 8.3.3.15.5, Section 8.3.3.15.6
Data Role Swap	No	Section 8.3.3.15.1, Section 8.3.3.15.2
VCONN Swap	No	Section 8.3.3.16
Source Alert	N/A	Section 8.3.3.7
Getting Source Extended Capabilities	No	Section 8.3.3.8
Getting Source Status	No	Section 8.3.3.9
Getting Battery Capabilities	No	Section 8.3.3.10
Getting Battery Status	No	Section 8.3.3.11
Getting Manufacturer Information	No	Section 8.3.3.12
Security	Yes	Section 8.3.3.13
Firmware Update	Yes	Section 8.3.3.14
Discover Identity	Yes	Section 8.3.3.17.1, Section 8.3.3.18.1
Source startup Cable Plug Discover Identity	Yes	Section 8.3.3.17.1, Section8.3.3.21.3
Discover SVIDs	Yes	Section 8.3.3.17.2, Section 8.3.3.18.2
Discover Modes	Yes	Section 8.3.3.17.3, Section 8.3.3.18.3
DFP to UFP Enter Mode	Yes	Section 8.3.3.19.1, Section 8.3.3.20.1
DFP to UFP Exit Mode	Yes	Section 8.3.3.19.2, Section 8.3.3.20.2
DFP to Cable Plug Enter Mode	Yes	Section 8.3.3.19.1, Section 8.3.3.21.4.1
DFP to Cable Plug Exit Mode	Yes	Section 8.3.3.19.1, Section 8.3.3.21.4.2
Attention	N/A	Section 8.3.3.17.4
Built in Self-Test (BIST)	No	Section 8.3.2.14

Microchip Ex. 1020, p. 257
Microchip v. Aptiv
IPR2024-00496

AMS	Interruptible	Reference
Sequence of Unstructured VDMs	Yes	Section 6.4.4.1
Sequence of Structured VDMs using Vendor Commands	Yes	Section 6.4.4.2

8.3.2.2 Power Negotiation

Figure 8-5 illustrates an example of a successful Message flow during Power Negotiation. The negotiation goes through 5 distinct phases:

- The Source sends out its power capabilities in a ***Source_Capabilities*** Message.
- The Sink evaluates these capabilities and in the request phase selects one power level by sending a ***Request*** Message.
- The Source evaluates the request and accepts the request with an ***Accept*** Message.
- The Source transitions to the new power level and then informs the Sink by sending a ***PS_RDY*** Message.
- The Sink starts using the new power level.

Microchip Ex. 1020, p. 258
Microchip v. Aptiv
IPR2024-00496

Figure 8-5 Successful Power Negotiation

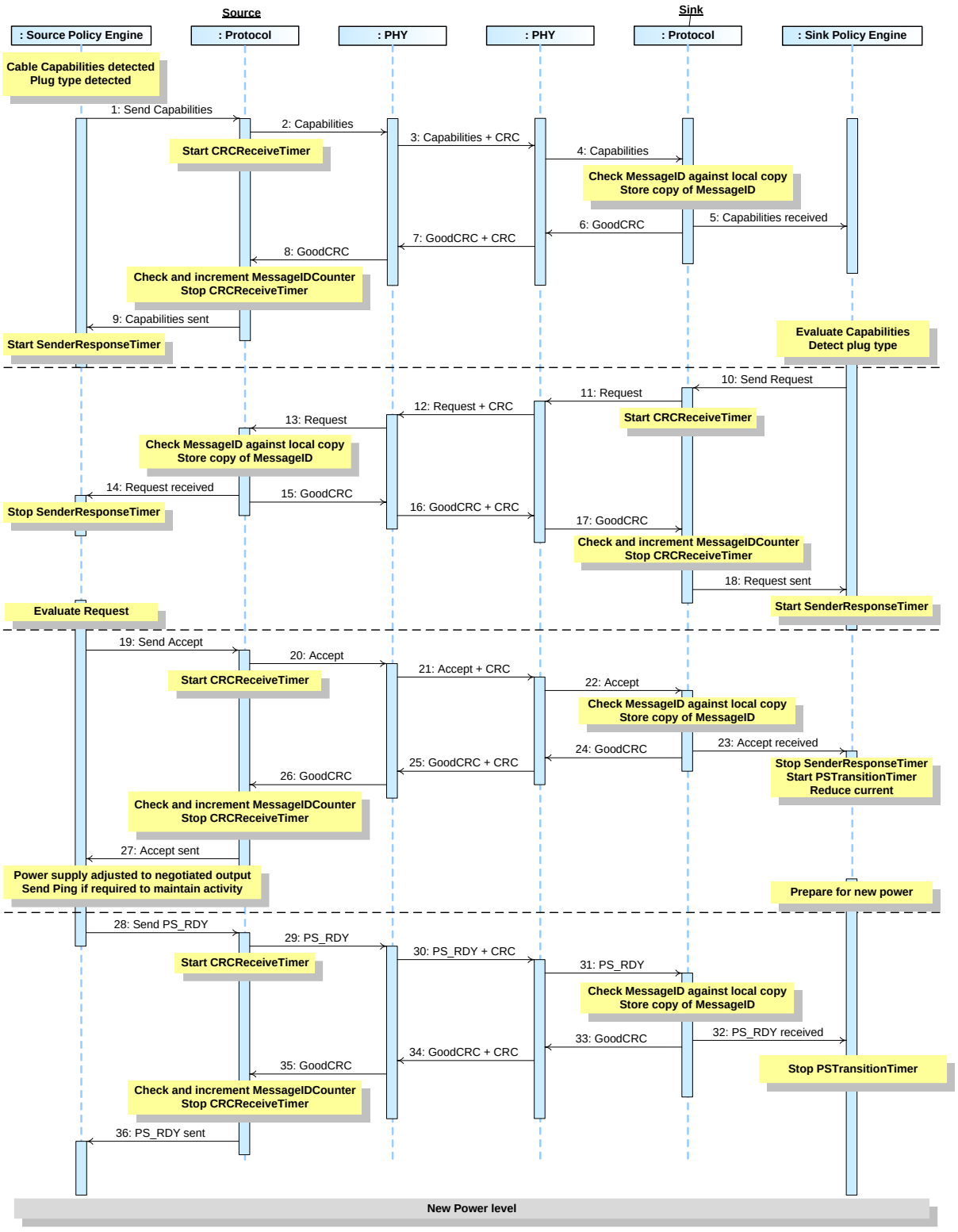


Table 8-5 below provides a detailed explanation of what happens at each labeled step in Figure 8-5 above.

Microchip Ex. 1020, p. 259
Microchip v. Aptiv
IPR2024-00496

Table 8-5 Steps for a successful Power Negotiation

Step	Source	Sink
1	The Cable Capabilities or Plug Type are detected if these are not already known (see Section 4.4). Policy Engine directs the Protocol Layer to send a ***Source_Capabilities*** Message that represents the power supply's present capabilities.	
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Source_Capabilities*** Message.	Physical Layer receives the ***Source_Capabilities*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Source_Capabilities*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Source_Capabilities*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Source_Capabilities*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine evaluates the ***Source_Capabilities*** Message sent by the Source, detects the plug type if this is necessary (see Section 4.4) and selects which power it would like. It tells the Protocol Layer to form the data (e.g. Power Data Object) that represents its Request into a Message.
11		Protocol Layer creates the ***Request*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Request*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Request*** Message.
13	Physical Layer removes the CRC and forwards the ***Request*** Message to the Protocol Layer.	
14	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer passes the Request information to the Policy Engine. Policy Engine stops ***SenderResponseTimer***.	
15	The Protocol Layer generates a ***GoodCRC*** Message and passes it to its Physical Layer.	
16	Physical Layer appends CRC and sends the Message.	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.

Microchip Ex. 1020, p. 260
Microchip v. Aptiv
IPR2024-00496

Step	Source	Sink
17		Physical Layer forwards the ***GoodCRC*** Message to the Protocol Layer.
18		The protocol Layer verifies and increments the ***MessageIDCounter***. It informs the Policy Engine that the ***Request*** Message was successfully sent. The Protocol Layer stops the ***CRCReceiveTimer***. The Policy Engine starts ***SenderResponseTimer***.
19	Policy Engine evaluates the ***Request*** Message sent by the Sink and decides if it can meet the request. It tells the Protocol Layer to form an ***Accept*** Message.	
20	The Protocol Layer forms the ***Accept*** Message that is passed to the Physical Layer and starts the ***CRCReceiveTimer***.	
21	Physical Layer appends CRC and sends the ***Accept*** Message.	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.
22		Physical Layer forwards the ***Accept*** Message to the Protocol Layer.
23		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. Protocol Layer informs the Policy Engine that an ***Accept*** Message has been received. The Policy Engine stops ***SenderResponseTimer***, starts the ***PSTransitionTimer*** and reduces its current draw. The Device Policy Manager prepares the Power supply for transition to the new power level.
24		The Protocol Layer generates a ***GoodCRC*** Message and passes it to its Physical Layer.
25	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends CRC and sends the Message.
26	Physical Layer forwards the ***GoodCRC*** Message to the Protocol Layer. The Protocol Layer verifies and increments the ***MessageIDCounter*** and stops the ***CRCReceiveTimer***.	
27	The Protocol Layer informs the Policy Engine that an ***Accept*** Message was successfully sent.	
power supply Adjusts its Output to the Negotiated Value		
28	The Device Policy Manager informs the Policy Engine that the power supply has settled at the new operating condition and tells the Protocol Layer to send a ***PS_RDY*** Message.	
29	The Protocol Layer forms the ***PS_RDY*** Message and starts the ***CRCReceiveTimer***.	
30	Physical Layer appends CRC and sends the ***PS_RDY*** Message.	Physical Layer receives the ***PS_RDY*** Message and compares the CRC it calculated with the one sent to verify the Message.
31		Physical Layer forwards the ***PS_RDY*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 261
Microchip v. Aptiv
IPR2024-00496

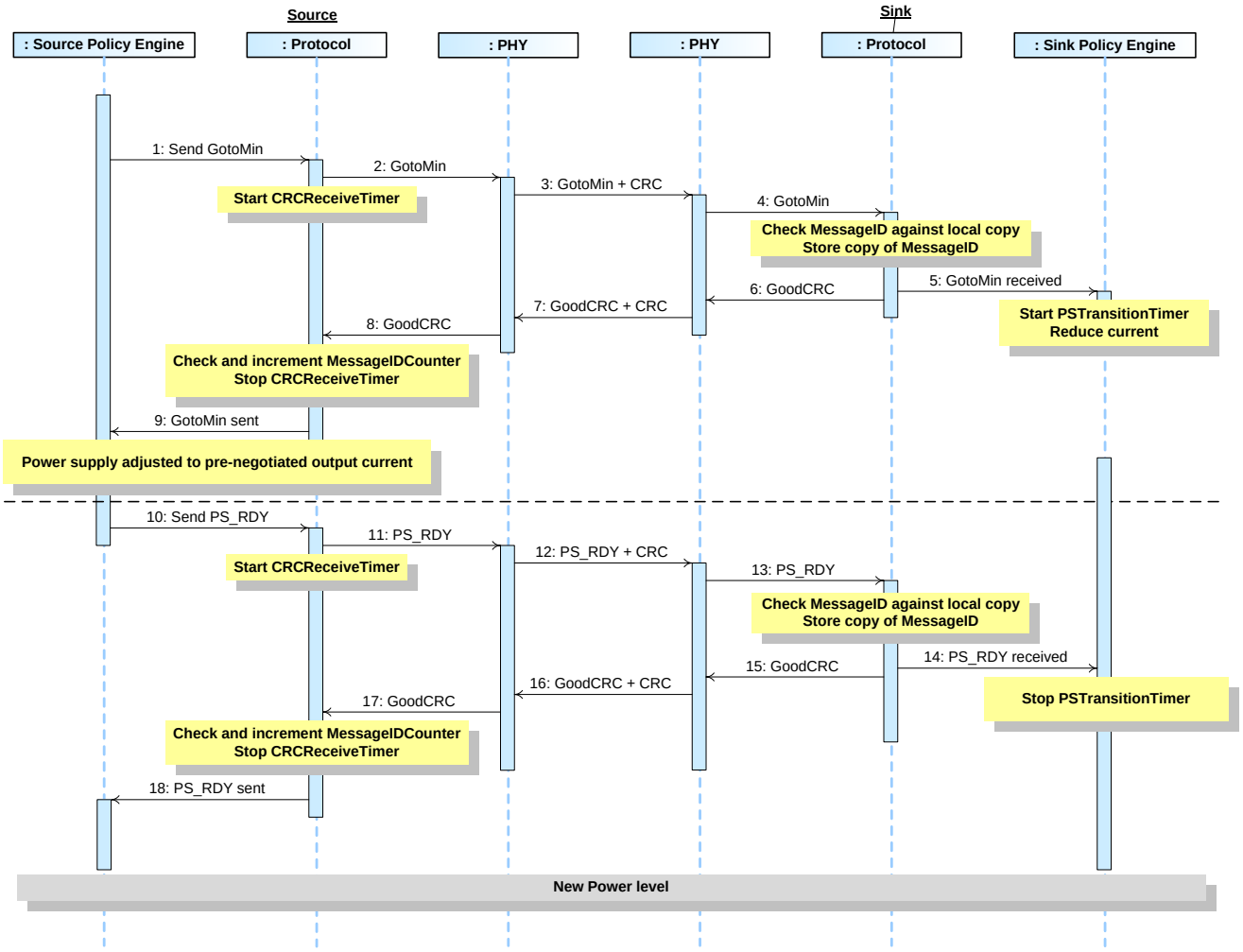
Step	Source	Sink
32		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. Protocol Layer informs the Policy Engine that a RS_RDY has been received. The Policy Engine stops the ***PSTransitionTimer***.
33		The Protocol Layer generates a ***GoodCRC*** Message and passes it to its Physical Layer.
34	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends CRC and sends the Message.
35	Physical Layer forwards the ***GoodCRC*** Message to the Protocol Layer. The Protocol Layer verifies and increments the ***MessageIDCounter***. Stops the ***CRCReceiveTimer***.	
36	The Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent.	

Microchip Ex. 1020, p. 262
Microchip v. Aptiv
IPR2024-00496

8.3.2.3 Reclaiming Power with GotoMin Message

This is an example of a GotoMin operation. Figure 8-6 shows the Messages as they flow across the bus and within the devices to accomplish the GotoMin.

Figure 8-6 Successful GotoMin operation



The table below provides a detailed explanation of what happens at each labeled step in Figure 8-6 above.

Table 8-6 Steps for a GotoMin Negotiation

Step	Source	Sink
1	Policy Engine tells the Protocol Layer to form a ***GotoMin*** Message.	
2	The Protocol Layer forms the ***GotoMin*** Message that is passed to the Physical Layer and starts the ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***GotoMin*** Message.	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.
4		Physical Layer forwards the ***GotoMin*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 263
Microchip v. Aptiv
IPR2024-00496

Step	Source	Sink
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. Protocol Layer informs the Policy Engine that a ***GotoMin*** Message has been received. The Policy starts the ***PSTransitionTimer*** and reduces its current draw. The Policy Engine prepares the Power supply for transition to the new power level.
6		The Protocol Layer generates a ***GoodCRC*** Message and passes it to its Physical Layer.
7	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends CRC and sends the Message.
8	Physical Layer forwards the ***GoodCRC*** Message to the Protocol Layer. The Protocol Layer verifies and increments the ***MessageIDCounter*** and stops the ***CRCReceiveTimer***.	
9	The Protocol Layer informs the Policy Engine that a ***GotoMin*** Message was successfully sent.	
power supply Adjusts its Output to the Negotiated Value		
10	Policy Engine sees the power supply has settled at the new operating condition and tells the Protocol Layer to send a ***PS_RDY*** Message.	
11	The Protocol Layer forms the ***PS_RDY*** Message and starts the ***CRCReceiveTimer***.	
12	Physical Layer appends CRC and sends the ***PS_RDY*** Message.	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.
13		Physical Layer forwards the ***PS_RDY*** Message to the Protocol Layer.
14		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. Protocol Layer informs the Policy Engine that a ***PS_RDY*** Message has been received. The Policy Engine stops the ***PSTransitionTimer***.
15		The Protocol Layer generates a ***GoodCRC*** Message and passes it to its Physical Layer.
16	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends CRC and sends the Message.
17	Physical Layer forwards the ***GoodCRC*** Message to the Protocol Layer. The Protocol Layer verifies and increments the ***MessageIDCounter*** and stops the ***CRCReceiveTimer***.	
18	The Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent.	

8.3.2.4 Soft Reset

This is an example of a Soft Reset operation. Figure 8-7 shows the Messages as they flow across the bus and within the devices to accomplish the Soft Reset.

Microchip Ex. 1020, p. 264
Microchip v. Aptiv
IPR2024-00496

Figure 8-7 Soft Reset

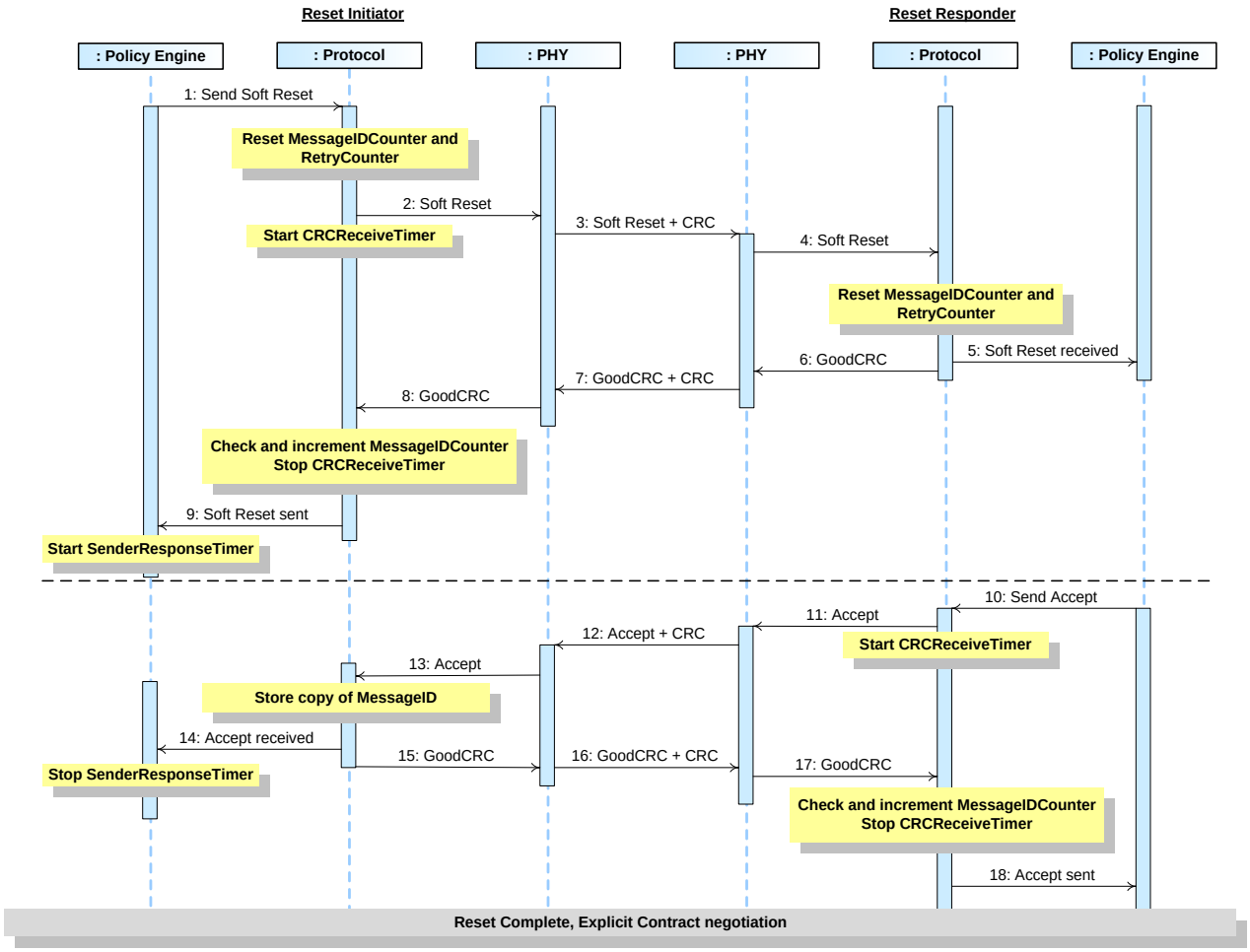


Table 8-7 below provides a detailed explanation of what happens at each labeled step in Figure 8-7 above.

Table 8-7 Steps for a Soft Reset

Step	Reset Initiator	Reset Responder
1	The Policy Engine directs the Protocol Layer to generate a ***Soft_Reset*** Message to request a Soft Reset.	
2	Protocol Layer resets ***MessageIDCounter*** and ***RetryCounter***. Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Soft_Reset*** Message.	Physical Layer receives the ***Soft_Reset*** Message and compares the CRC it calculated with the one sent to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Soft_Reset*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 265
Microchip v. Aptiv
IPR2024-00496

Step	Reset Initiator	Reset Responder
5		Protocol Layer does not check the ***MessageID*** in the incoming Message and resets ***MessageIDCounter*** and ***RetryCounter***. The Protocol Layer forwards the received ***Soft_Reset*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Soft_Reset*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine tells the Protocol Layer to form an ***Accept*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the Message.
13	Protocol Layer stores the ***MessageID*** of the incoming Message.	
14	The Protocol Layer forwards the received ***Accept*** Message information to the Policy Engine that consumes it.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent.
	The reset is complete and protocol communication can restart. Port Partners perform an Explicit Contract negotiation to re-synchronize their state machines.	

Microchip Ex. 1020, p. 266
Microchip v. Aptiv
IPR2024-00496

8.3.2.5 Hard Reset

The following sections describe the steps required for a USB Power Delivery Hard Reset. The Hard Reset returns the operation of the USB Power Delivery to default role and operating voltage/current. During the Hard Reset USB Power Delivery PHY Layer communications shall be disabled preventing communication between the Port partners.

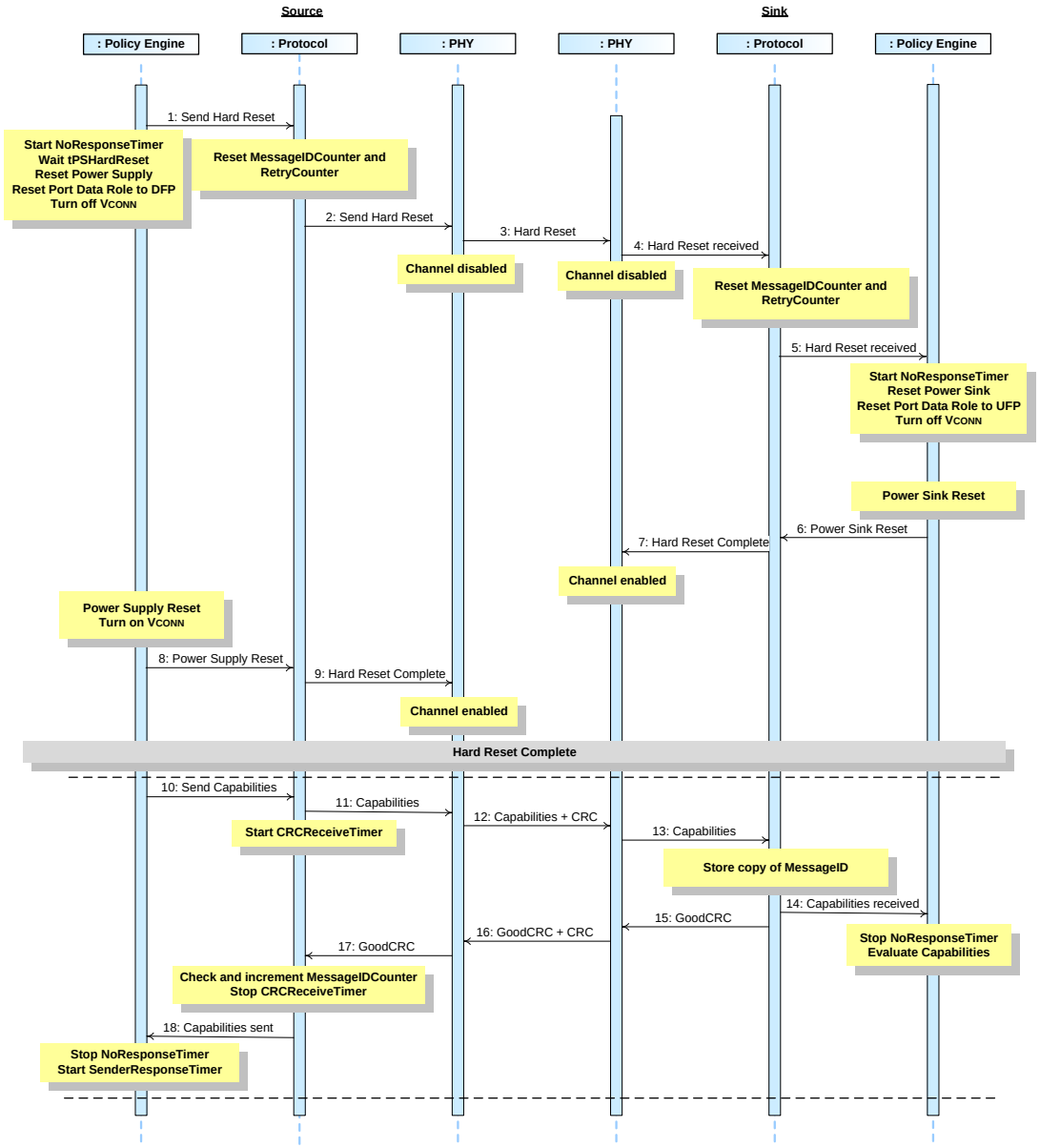
Note: Hard Reset, in this case, is applied to the USB Power Delivery capability of an individual Port on which the Hard Reset is requested. A side effect of the Hard Reset is that it might reset other functions on the Port such as USB.

8.3.2.5.1 Source Initiated Hard Reset

This is an example of a Hard Reset operation when initiated by a Source. Figure 8-8 shows the Messages as they flow across the bus and within the devices to accomplish the Hard Reset.

Microchip Ex. 1020, p. 267
Microchip v. Aptiv
IPR2024-00496

Figure 8-8 Source initiated Hard Reset



Microchip Ex. 1020, p. 268
Microchip v. Aptiv
IPR2024-00496

Table 8-8 Steps for Source initiated Hard Reset

Step	Source	Sink
1	The Policy Engine directs the Protocol Layer to generate ***Hard Reset*** Signaling. The Policy Engine starts the ***NoResponseTimer*** and requests the Device Policy Manager to reset the power supply to USB Default Operation. The Policy Engine requests the Device Policy Manager to reset the Port Data Role to DFP and to turn off VCONN if this is on.	
2	Protocol Layer resets ***MessageIDCounter*** and ***RetryCounter.*** Protocol Layer requests the Physical Layer send ***Hard Reset*** Signaling.	
3	Physical Layer sends ***Hard Reset*** Signaling and then disables the PHY Layer communications channel for transmission and reception.	Physical Layer receives the ***Hard Reset*** Signaling and disables the PHY Layer communications channel for transmission and reception.
4		Physical Layer informs the Protocol Layer of the Hard Reset. Protocol Layer resets ***MessageIDCounter*** and ***RetryCounter***.
5		The Protocol Layer informs the Policy Engine of the Hard Reset. The Policy Engine starts the ***NoResponseTimer*** and requests the Device Policy Manager to reset the Power Sink to default operation. The Policy Engine requests the Device Policy Manager to reset the Port Data Role to UFP and to turn off VCONN if this is on.
6		The Power Sink returns to default operation. The Policy Engine informs the Protocol Layer that the Power Sink has been reset.
7		The Protocol Layer informs the PHY Layer that the Hard Reset is complete. The PHY Layer enables the PHY Layer communications channel for transmission and reception.
8	The power supply is reset to default operation and VCONN is turned on. The Policy Engine informs the Protocol Layer that the power supply has been reset.	
9	The Protocol Layer informs the PHY Layer that the Hard Reset is complete. The PHY Layer enables the PHY Layer communications channel for transmission and reception.	
	The reset is complete and protocol communication can restart.	
10	Policy Engine directs the Protocol Layer to send a ***Source_Capabilities*** Message that represents the power supply's present capabilities.	
11	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
12	Physical Layer appends CRC and sends the ***Source_Capabilities*** Message.	Physical Layer receives the ***Source_Capabilities*** Message and checks the CRC to verify the Message.
13		Physical Layer removes the CRC and forwards the ***Source_Capabilities*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 269
Microchip v. Aptiv
IPR2024-00496

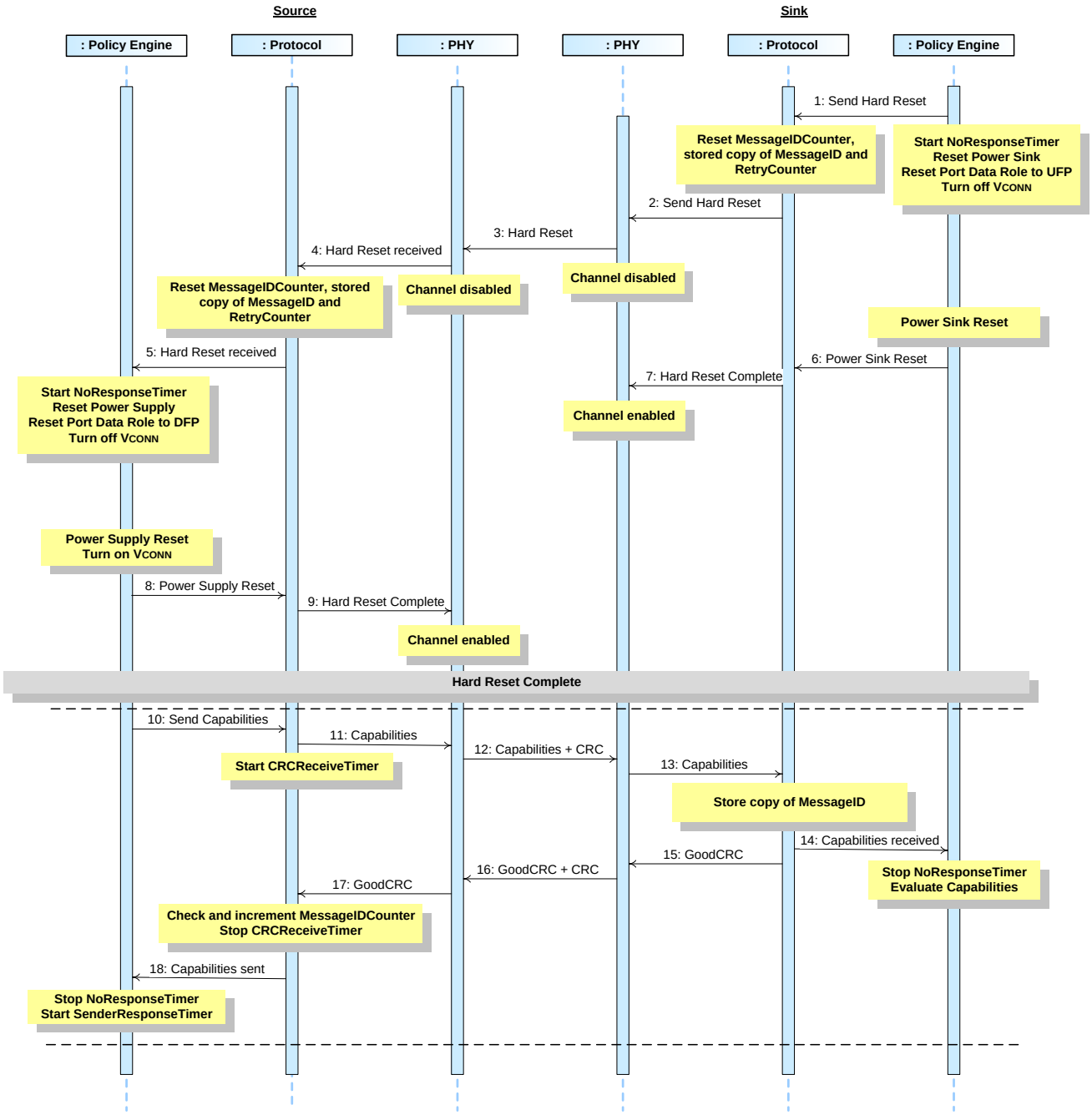
Step	Source	Sink
14		Protocol Layer stores the ***MessageID*** of the incoming Message. The Protocol Layer forwards the received ***Source_Capabilities*** Message information to the Policy Engine that consumes it. The Policy Engine stops the ***NoResponseTimer***.
15		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
16	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
17	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
18	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Source_Capabilities*** Message was successfully sent. Policy Engine stops the ***NoResponseTimer*** and starts the ***SenderResponseTimer***.	
	USB Power Delivery communication is re-established.	

Microchip Ex. 1020, p. 270
Microchip v. Aptiv
IPR2024-00496

8.3.2.5.2 Sink Initiated Hard Reset

This is an example of a Hard Reset operation when initiated by a Sink. Figure 8-9 shows the Messages as they flow across the bus and within the devices to accomplish the Hard Reset.

Figure 8-9 Sink Initiated Hard Reset



Microchip Ex. 1020, p. 271
Microchip v. Aptiv
IPR2024-00496

Table 8-9 Steps for Sink initiated Hard Reset

Step	Source	Sink
1		The Policy Engine directs the Protocol Layer to generate ***Hard Reset*** Signaling. The Policy Engine starts the ***NoResponseTimer*** and requests the Device Policy Manager to reset the power supply to USB Default Operation. The Policy Engine requests the Device Policy Manager to reset the Port Data Role to UFP and to turn off VCONN if this is on.
2		Protocol Layer resets ***MessageIDCounter***, stored copy of ***MessageID*** and ***RetryCounter***. Protocol Layer requests the Physical Layer send ***Hard Reset*** Signaling.
3	Physical Layer receives the ***Hard Reset*** Signaling and disables the PHY Layer communications channel for transmission and reception.	Physical Layer sends the ***Hard Reset*** Signaling and then disables the PHY Layer communications channel for transmission and reception.
4	Physical Layer informs the Protocol Layer of the Hard Reset. Protocol Layer resets ***MessageIDCounter***, stored copy of ***MessageID*** and ***RetryCounter***.	
5	The Protocol Layer Informs the Policy Engine of the Hard Reset. The Policy Engine starts the ***NoResponseTimer*** and requests the Device Policy Manager to reset the Power Sink to default operation. The Policy Engine requests the Device Policy Manager to reset the Port Data Role to DFP and to turn off VCONN if this is on.	
6		The Power Sink returns to USB Default Operation. The Policy Engine informs the Protocol Layer that the Power Sink has been reset.
7		The Protocol Layer informs the PHY Layer that the Hard Reset is complete. The PHY Layer enables the PHY Layer communications channel for transmission and reception.
8	The power supply is reset to USB Default Operation and VCONN is turned on. The Policy Engine informs the Protocol Layer that the power supply has been reset.	
9	The Protocol Layer informs the PHY Layer that the Hard Reset is complete. The PHY Layer enables the PHY Layer communications channel for transmission and reception.	
	The reset is complete and protocol communication can restart.	
10	Policy Engine directs the Protocol Layer to send a ***Source_Capabilities*** Message that represents the power supply's present capabilities.	
11	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
12	Physical Layer appends CRC and sends the ***Source_Capabilities*** Message.	Physical Layer receives the ***Source_Capabilities*** Message and checks the CRC to verify the Message.
13		Physical Layer removes the CRC and forwards the ***Source_Capabilities*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 272
Microchip v. Aptiv
IPR2024-00496

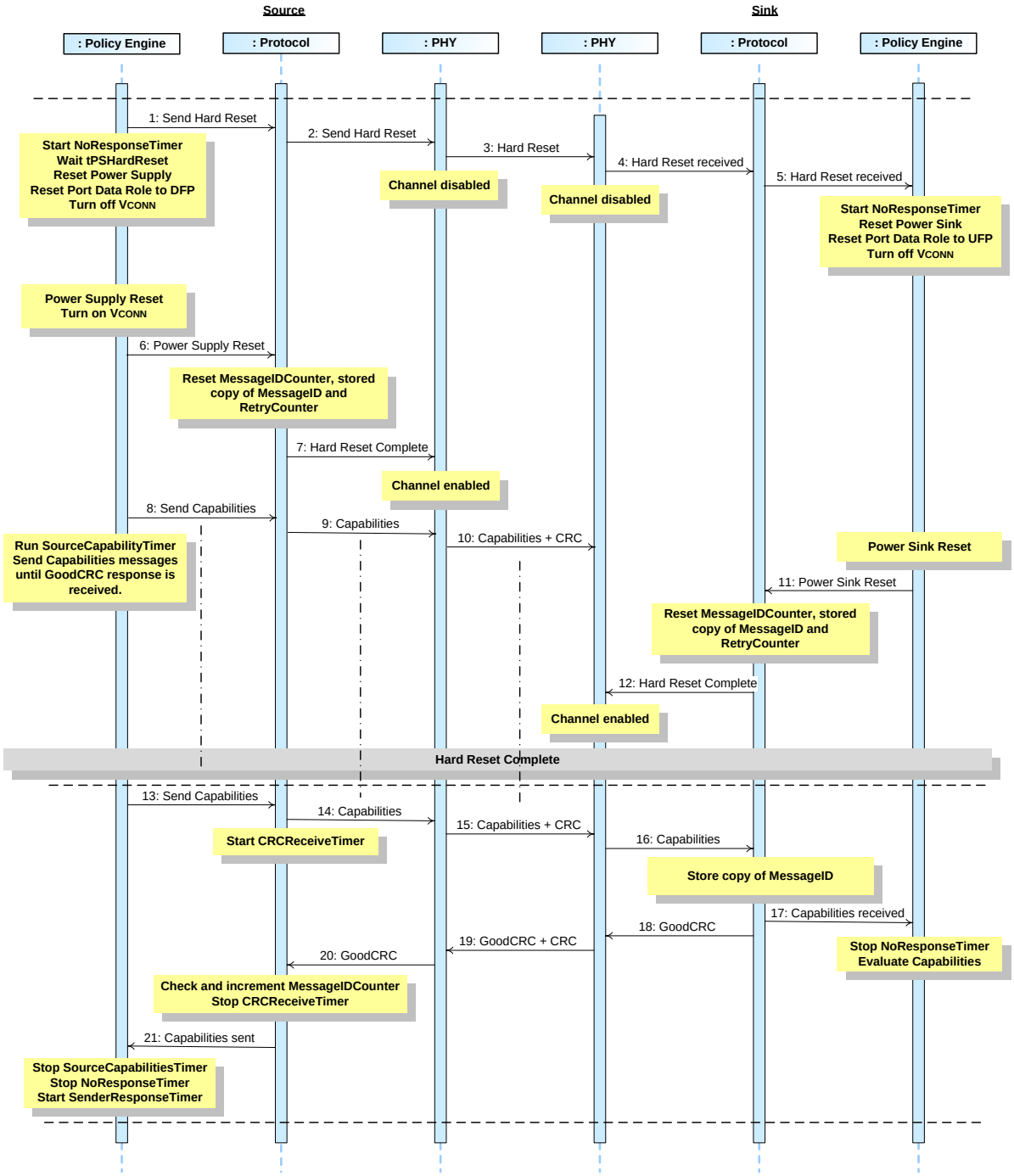
Step	Source	Sink
14		Protocol Layer stores the ***MessageID*** of the incoming Message. The Protocol Layer forwards the received ***Source_Capabilities*** Message information to the Policy Engine that consumes it. The Policy Engine stops the ***NoResponseTimer***.
15		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
16	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
17	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
18	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Source_Capabilities*** Message was successfully sent. Policy Engine stops the ***NoResponseTimer*** and starts the ***SenderResponseTimer***.	
	USB Power Delivery communication is re-established.	

Microchip Ex. 1020, p. 273
Microchip v. Aptiv
IPR2024-00496

8.3.2.5.3 Source Initiated Hard Reset – Sink Long Reset

This is an example of a Hard Reset operation when initiated by a Source. In this example the Sink is slow responding to the reset causing the Source to send multiple ***Source_Capabilities*** Messages before it receives a ***GoodCRC*** Message response. Figure 8-10 shows the Messages as they flow across the bus and within the devices to accomplish the Hard Reset.

Figure 8-10 Source initiated reset - Sink long reset



Microchip Ex. 1020, p. 274
Microchip v. Aptiv
IPR2024-00496

Table 8-10 Steps for Source initiated Hard Reset – Sink long reset

Step	Source	Sink
1	The Policy Engine directs the Protocol Layer to generate ***Hard Reset*** Signaling. The Policy Engine starts the ***NoResponseTimer*** and requests the Device Policy Manager to reset the power supply to USB Default Operation. The Policy Engine requests the Device Policy Manager to reset the Port Data Role to DFP and to turn off VCONN if this is on.	
2	Protocol Layer resets ***MessageIDCounter***, stored copy of ***MessageID*** and ***RetryCounter***. Protocol Layer requests the Physical Layer send ***Hard Reset*** Signaling.	
3	Physical Layer sends the ***Hard Reset*** Signaling and then disables the PHY Layer communications channel for transmission and reception.	Physical Layer receives the ***Hard Reset*** Signaling and disables the PHY Layer communications channel for transmission and reception.
4		Physical Layer informs the Protocol Layer of the Hard Reset. Protocol Layer resets ***MessageIDCounter***, stored copy of ***MessageID*** and ***RetryCounter***.
5		The Protocol Layer Informs the Policy Engine of the Hard Reset. The Policy Engine starts the ***NoResponseTimer*** and requests the Device Policy Manager to reset the Power Sink to default operation. The Policy Engine requests the Device Policy Manager to reset the Port Data Role to UFP and to turn off VCONN if this is on.
6	The power supply is reset to USB Default Operation and VCONN is turned on. The Policy Engine informs the Protocol Layer that the power supply has been reset.	
7	The Protocol Layer informs the PHY Layer that the Hard Reset is complete. The PHY Layer enables the PHY Layer communications channel for transmission and reception.	
	The reset is complete and protocol communication can restart.	
8	Policy Engine directs the Protocol Layer to send a ***Source_Capabilities*** Message that represents the power supply's present capabilities. Policy Engine starts the ***SourceCapabilityTimer***. The ***SourceCapabilityTimer*** times out one or more times until a ***GoodCRC*** Message response is received.	
9	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
10	Physical Layer appends CRC and sends the ***Source_Capabilities*** Message.	Note: ***Source_Capabilities*** Message not received since channel is disabled.
11		The Power Sink returns to USB Default Operation. The Policy Engine informs the Protocol Layer that the Power Sink has been reset.
12		The Protocol Layer informs the PHY Layer that the Hard Reset is complete. The PHY Layer enables the PHY Layer communications channel for transmission and reception.

Microchip Ex. 1020, p. 275
Microchip v. Aptiv
IPR2024-00496

Step	Source	Sink
	The reset is complete and protocol communication can restart.	
13	Policy Engine directs the Protocol Layer to send a ***Source_Capabilities*** Message that represents the power supply's present capabilities. Starts the ***SourceCapabilityTimer***.	
14	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
15	Physical Layer appends CRC and sends the ***Source_Capabilities*** Message.	Physical Layer receives the ***Source_Capabilities*** Message and checks the CRC to verify the Message.
16		Physical Layer removes the CRC and forwards the ***Source_Capabilities*** Message to the Protocol Layer.
17		Protocol Layer stores the ***MessageID*** of the incoming Message. The Protocol Layer forwards the received ***Source_Capabilities*** Message information to the Policy Engine that consumes it. The Policy Engine stops the ***NoResponseTimer*****.**
18		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
19	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
20	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
21	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Source_Capabilities*** Message was successfully sent. Policy Engine stops the ***SourceCapabilityTimer***, stops the ***NoResponseTimer*** and starts the ***SenderResponseTimer***.	
	USB Power Delivery communication is re-established.	

Microchip Ex. 1020, p. 276
Microchip v. Aptiv
IPR2024-00496

8.3.2.6 Power Role Swap

8.3.2.6.1 Source Initiated Power Role Swap without subsequent Power Negotiation

This is an example of a successful Power Role Swap operation initiated by a Port which initially, at the start of this Message sequence, is acting as a Source and therefore has Rp pulled up on its CC wire. It does not include any subsequent Power Negotiation which is required in order to establish an Explicit Contract (see Section 8.3.2.2).

There are four distinct phases to the Power Role Swap negotiation:

1. A ***PR_Swap*** Message is sent.
2. An ***Accept*** Message in response to the ***PR_Swap*** Message.
3. The new Sink sets its power output to ***vSafe0V***, then asserts Rd and sends a ***PS_RDY*** Message when this process is complete.
4. The new Source asserts Rp, then sets its power output to ***vSafe5V*** and sends a ***PS_RDY*** Message when it is ready to supply power.

Figure 8-11 shows the Messages as they flow across the bus and within the devices to accomplish the Power Role Swap sequence.

Microchip Ex. 1020, p. 277
Microchip v. Aptiv
IPR2024-00496

Figure 8-11 Successful Power Role Swap Sequence Initiated by the Source

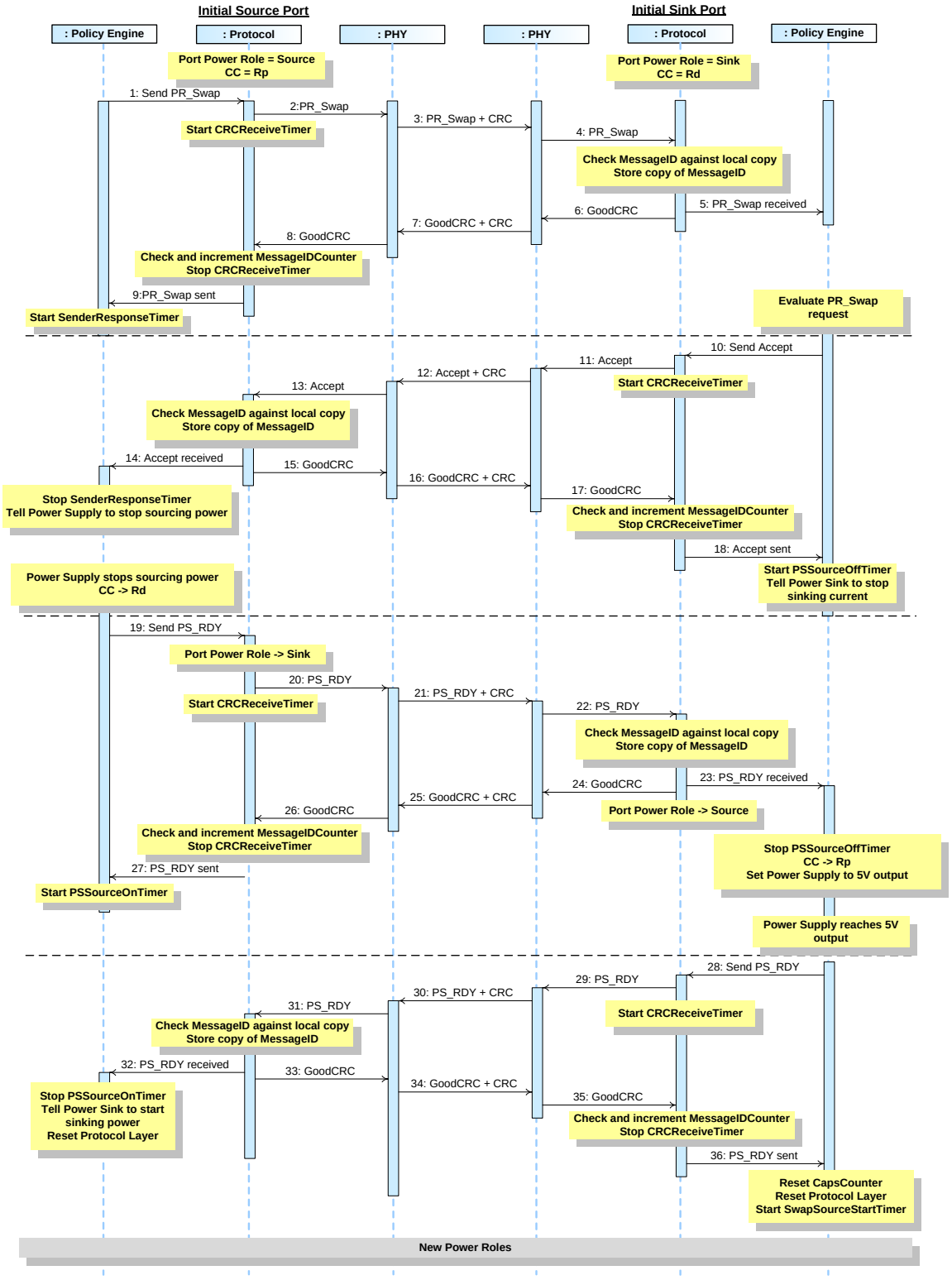


Table 8-11below provides a detailed explanation of what happens at each labeled step in Figure 8-11 above.

Microchip Ex. 1020, p. 278
Microchip v. Aptiv
IPR2024-00496

Table 8-11 Steps for a Successful Source Initiated Power Role Swap Sequence

Step	Initial Source Port	Initially Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***PR_Swap*** Message.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***PR_Swap*** Message.	Physical Layer receives the ***PR_Swap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***PR_Swap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PR_Swap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PR_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine evaluates the ***PR_Swap*** Message sent by the Source and decides that it is able and willing to do the Power Role Swap. It tells the Protocol Layer to form an ***Accept*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Accept*** Message.	Physical Layer appends a CRC and sends the ***Accept*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PR_Swap*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine requests its power supply to stop supplying power and stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 279
Microchip v. Aptiv
IPR2024-00496

Step	Initial Source Port	Initially Sink Port
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent. The Policy Engine starts the ***PSSourceOffTimer*** and tells the power supply to stop sinking current.
19	The Policy Engine determines its power supply is no longer supplying V_{BUS}. The Policy Engine requests the Device Policy Manager to assert the Rd pull down on the CC wire. The Policy Engine then directs the Protocol Layer to generate a ***PS_RDY*** Message, with the ***Port Power Role*** bit in the Message Header set to "Sink", to tell its Port Partner that it can begin to Source V_{BUS}.	
20	Protocol Layer sets the ***Port Power Role*** bit in the Message Header set to "Sink", creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
21	Physical Layer appends CRC and sends the ***PS_RDY*** Message.	Physical Layer receives the ***PS_RDY*** Message and checks the CRC to verify the Message.
22		Physical Layer removes the CRC and forwards the ***PS_RDY*** Message to the Protocol Layer.
23		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PS_RDY*** Message information to the Policy Engine that consumes it. The Policy Engine stops the ***PSSourceOffTimer***, directs the Device Policy Manager to apply the Rp pull up and then starts switching the power supply to ***vSafe5V*** Source operation.
24		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
25	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
26	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
27	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent. Policy Engine starts ***PSSourceOnTimer***.	
28		Policy Engine, when its power supply is ready to supply power, tells the Protocol Layer to form a ***PS_RDY*** Message. The ***Port Power Role*** bit used in this and subsequent Message Headers is now set to "Source".
29		Protocol Layer creates the ***PS_RDY*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
30	Physical Layer receives the ***PS_RDY*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***PS_RDY*** Message.
31	Physical Layer removes the CRC and forwards the ***PS_RDY*** Message to the Protocol Layer.	

Microchip Ex. 1020, p. 280
Microchip v. Aptiv
IPR2024-00496

Step	Initial Source Port	Initially Sink Port
32	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PS_RDY*** Message information to the Policy Engine that consumes it.	
33	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
34	Physical Layer appends a CRC and sends the ***GoodCRC*** Message. The Policy Engine stops the ***PSSourceOnTimer***, informs the power supply it can now Sink power and resets the Protocol Layer.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
35		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
36		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent. The Policy Engine resets the ***CapsCounter***, resets the Protocol Layer and starts the ***SwapSourceStartTimer*** which must timeout before sending any ***Source_Capabilities*** Messages.
	The Power Role Swap is complete, the roles have been reversed and the Port Partners are free to negotiate for more power.	

Microchip Ex. 1020, p. 281
Microchip v. Aptiv
IPR2024-00496

8.3.2.6.2 Sink Initiated Power Role Swap without subsequent Power Negotiation

This is an example of a successful Power Role Swap operation initiated by a Port which initially, at the start of this Message sequence, is acting as a Sink and therefore has Rd pulled down on its CC wire. It does not include any subsequent Power Negotiation which is required in order to establish an Explicit Contract (see Section 8.3.2.2).

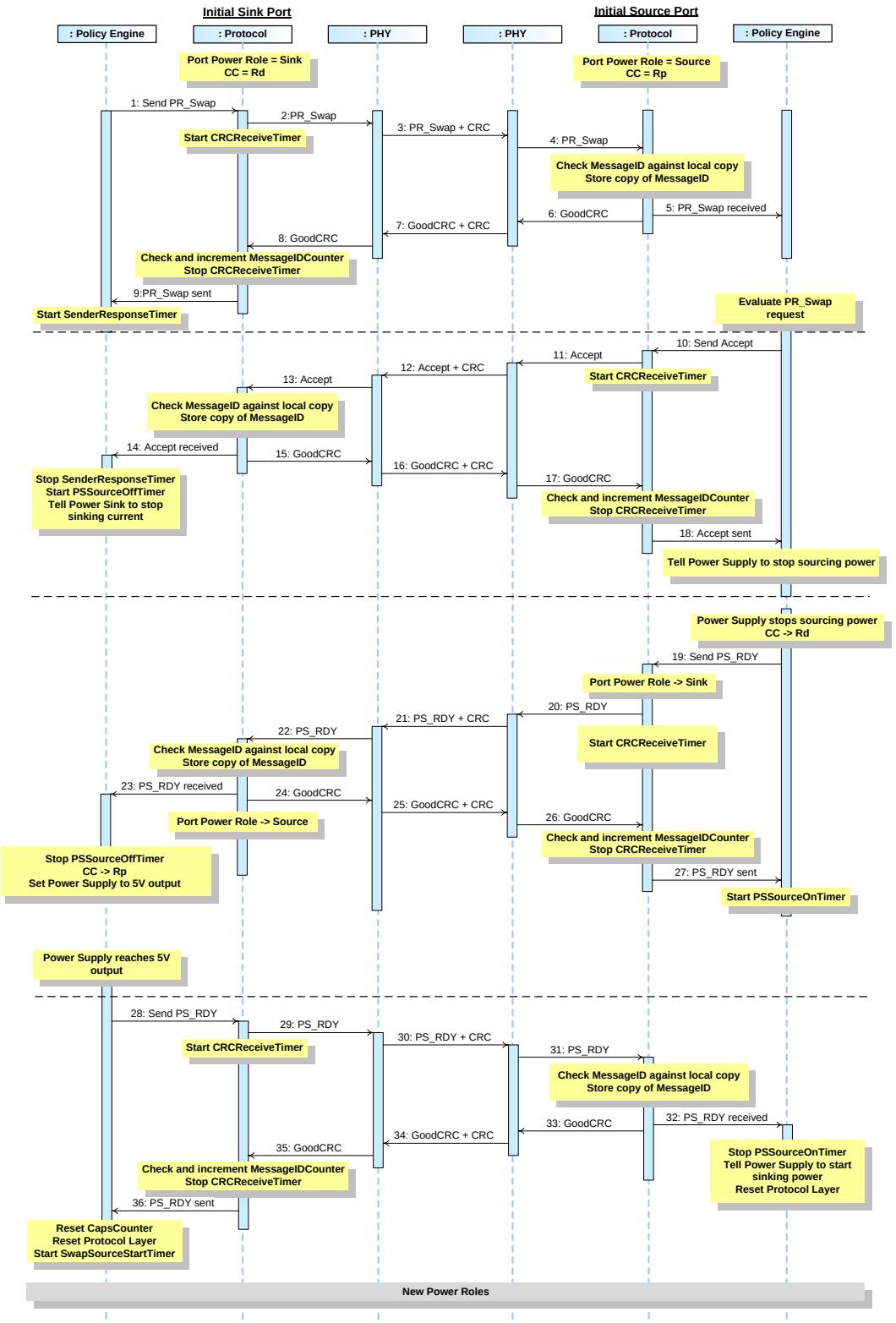
There are four distinct phases to the Power Role Swap negotiation:

1) A ***PR_Swap*** Message is sent.
2) An ***Accept*** Message in response to the ***PR_Swap*** Message.
3) The new Sink sets its power output to ***vSafe0V***, then asserts Rd and sends a ***PS_RDY*** Message when this process is complete.
4) The new Source asserts Rp, then sets its power output to ***vSafe5V*** and sends a ***PS_RDY*** Message when it is ready to supply power.

Figure 8-12 shows the Messages as they flow across the bus and within the devices to accomplish the Power Role Swap.

Microchip Ex. 1020, p. 282
Microchip v. Aptiv
IPR2024-00496

Figure 8-12 Successful Power Role Swap Sequence Initiated by the Sink



Microchip Ex. 1020, p. 283
Microchip v. Aptiv
IPR2024-00496

Table 8-12 Steps for a Successful Sink Initiated Power Role Swap Sequence

Step	Initial Sink Port	Initial Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***PR_Swap*** Message.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***PR_Swap*** Message.	Physical Layer receives the ***PR_Swap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***PR_Swap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PR_Swap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PR_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine evaluates the ***PR_Swap*** Message sent by the Sink and decides that it is able and willing to do the Power Role Swap. It tells the Protocol Layer to form an ***Accept*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Accept*** Message.	Physical Layer appends a CRC and sends the ***Accept*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PR_Swap*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***, starts the ***PSSourceOffTimer*** and tells the power supply to stop sinking current.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 284
Microchip v. Aptiv
IPR2024-00496

Step	Initial Sink Port	Initial Source Port
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent. The Policy Engine tells the power supply to stop supplying power.
19		The Policy Engine determines its power supply is no longer supplying V_{BUS}. The Policy Engine requests the Device Policy Manager to assert the Rd pull down on the CC wire. The Policy Engine then directs the Protocol Layer to generate a ***PS_RDY*** Message, with the ***Port Power Role*** bit in the Message Header set to "Sink", to tell its Port Partner that it can begin to Source V_{BUS}.
20		Protocol Layer sets the ***Port Power Role*** bit in the Message Header set to "Sink", creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
21	Physical Layer receives the ***PS_RDY*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***PS_RDY*** Message.
22	Physical Layer removes the CRC and forwards the ***PS_RDY*** Message to the Protocol Layer.	
23	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PS_RDY*** Message information to the Policy Engine that consumes it. The Policy Engine stops the ***PSSourceOffTimer***, directs the Device Policy Manager to apply the Rp pull up and then starts switching the power supply to ***vSafe5V*** Source operation.	
24	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
25	Physical Layer appends CRC and sends the ***GoodCRC*** Message.	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.
26		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
27		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent. Policy Engine starts ***PSSourceOnTimer***.
28	Policy Engine, when its power supply is ready to supply power, tells the Protocol Layer to form a ***PS_RDY*** Message. The ***Port Power Role*** bit used in this and subsequent Message Headers is now set to "Source".	
29	Protocol Layer creates the ***PS_RDY*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
30	Physical Layer appends a CRC and sends the ***PS_RDY*** Message.	Physical Layer receives the ***PS_RDY*** Message and compares the CRC it calculated with the one sent to verify the Message.
31		Physical Layer removes the CRC and forwards the ***PS_RDY*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 285
Microchip v. Aptiv
IPR2024-00496

Step	Initial Sink Port	Initial Source Port
32		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PS_RDY*** Message information to the Policy Engine that consumes it. The Policy Engine stops the ***PSSourceOnTimer***, informs the power supply that it can start consuming power.
33		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
34	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***GoodCRC*** Message. The Policy Engine stops the ***PSSourceOnTimer***, informs the power supply it can now Sink power and resets the Protocol Layer.
35	Physical Layer removes the CRC and forwards the ***GoodCRC*** to the Protocol Layer.	
36	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent. The Policy Engine resets the ***CapsCounter***, resets the Protocol Layer and starts the ***SwapSourceStartTimer*** which must timeout before sending any ***Source_Capabilities*** Messages.	
	The Power Role Swap is complete, the roles have been reversed and the Port Partners are free to negotiate for more power.	

Microchip Ex. 1020, p. 286
Microchip v. Aptiv
IPR2024-00496

8.3.2.7 Fast Role Swap

This is an example of a successful Fast Role Swap operation initiated by a Port that is initially a Source and therefore has Rp pulled up on its CC Wire and which has lost power and needs to get ***vSafe5V*** quickly. It does not include any subsequent Power Negotiation which is required in order to establish an Explicit Contract (see Section 8.3.2.2).

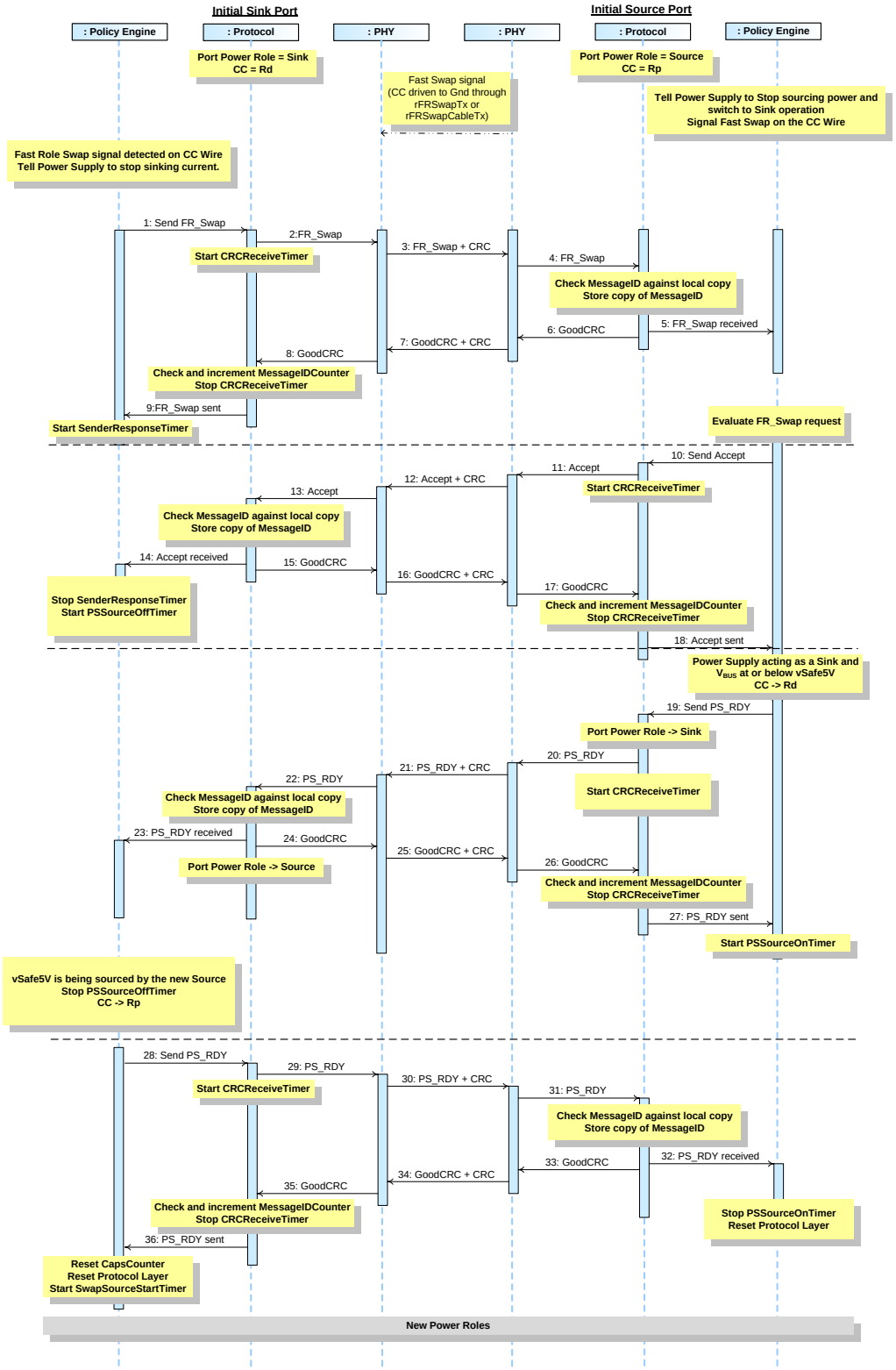
There are several distinct phases to the Fast Role Swap negotiation:

1) The initial Source stops driving its power output which starts transitioning to ***vSafe0V*** and signals Fast Role Swap on the CC Wire.
2) The initial Sink stops Sinking power. At this point the new Source still has Rd asserted and the new Sink still has Rp asserted.
3) An ***FR_Swap*** Message is sent by the new Source within ***tFRSwapInit*** of detecting the Fast Swap signal.
4) An ***Accept*** Message is sent by the new Sink in response to the ***FR_Swap*** Message.
5) The new Sink asserts Rd and sends a ***PS_RDY*** Message indicating that the voltage on V_{BUS} is at or below ***vSafe5V***.
6) The new Source asserts Rp and sends a ***PS_RDY*** Message indicating that it is acting as a Source and is supplying ***vSafe5V***. Note: that the new Source can start applying at any point V_{BUS} is at or below ***vSafe5V*** but will start driving V_{BUS} to ***vSafe5V*** no later than ***tSrcFRSwap*** after detecting that V_{BUS} has dropped below ***vSafe5V***. This can happen at any point after the Fast Role Swap signal is detected.

Figure 8-13 shows the Messages as they flow across the bus and within the devices to accomplish the Fast Role Swap.

Microchip Ex. 1020, p. 287
Microchip v. Aptiv
IPR2024-00496

Figure 8-13 Successful Fast Role Swap Sequence



Microchip Ex. 1020, p. 288
Microchip v. Aptiv
IPR2024-00496

Table 8-13 Steps for a Successful Fast Role Swap Sequence

Step	Initial Sink Port	Initial Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. The Device Policy Manager detects Fast Swap on the CC Wire and tells the power supply to stop sinking current.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. The Device Policy Manager tells the Power Supply to stop sourcing power and switch to Sink operation. The Device Policy Manager signals Fast Swap on the CC Wire by driving CC to ground with a resistance of less than ***rFRSwapTx*** for at least ***tFRSwapTx***.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***FR_Swap*** Message.	Physical Layer receives the ***FR_Swap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***PR_Swap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***FR_Swap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***FR_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine evaluates the ***PR_Swap*** Message sent by the Sink and decides that it is able and willing to do the Power Role Swap. It tells the Protocol Layer to form an ***Accept*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Accept*** Message.	Physical Layer appends a CRC and sends the ***Accept*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PR_Swap*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***, starts the ***PSSourceOffTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.

Microchip Ex. 1020, p. 289
Microchip v. Aptiv
IPR2024-00496

Step	Initial Sink Port	Initial Source Port
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent.
19		The Policy Engine determines its power supply is no longer supplying V_{BUS} and is acting as a Sink. The Policy Engine requests the Device Policy Manager to assert the Rd pull down on the CC wire. The Policy Engine then directs the Protocol Layer to generate a ***PS_RDY*** Message, with the ***Port Power Role*** bit in the Message Header set to "Sink", to tell its Port Partner that it can begin to Source V_{BUS}.
20		Protocol Layer sets the ***Port Power Role*** bit in the Message Header set to "Sink", creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
21	Physical Layer receives the ***PS_RDY*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***PS_RDY*** Message.
22	Physical Layer removes the CRC and forwards the ***PS_RDY*** Message to the Protocol Layer.	
23	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PS_RDY*** Message information to the Policy Engine that consumes it. The Policy Engine stops the ***PSSourceOffTimer***.	
24	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
25	Physical Layer appends CRC and sends the ***GoodCRC*** Message.	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.
26		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
27		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent. Policy Engine starts ***PSSourceOnTimer***.
28	The Policy Engine directs the Device Policy Manager to apply the Rp pull up. Note: at some point (either before or after receiving the ***PS_RDY*** Message) the new Source has applied ***vSafe5V*** no later ***tSrcFRSwap*** after detecting that V_{BUS} has dropped below ***vSafe5V***. When the Source output reaches ***vSafe5V*** the Policy Engine directs the Protocol Layer to send an ***FR_Swap*** Message within ***tFRSwapInit*** of detecting the Fast Swap signal. Policy Engine, when its power supply is ready to supply power, tells the Protocol Layer to form a ***PS_RDY*** Message. The ***Port Power Role*** bit used in this and subsequent Message Headers is now set to "Source".	

Microchip Ex. 1020, p. 290
Microchip v. Aptiv
IPR2024-00496

Step	Initial Sink Port	Initial Source Port
29	Protocol Layer creates the ***PS_RDY*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
30	Physical Layer appends a CRC and sends the ***PS_RDY*** Message.	Physical Layer receives the ***PS_RDY*** Message and compares the CRC it calculated with the one sent to verify the Message.
31		Physical Layer removes the CRC and forwards the ***PS_RDY*** Message to the Protocol Layer.
32		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PS_RDY*** Message information to the Policy Engine that consumes it. The Policy Engine stops the ***PSSourceOnTimer***.
33		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
34	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***GoodCRC*** Message. The Policy Engine resets the Protocol Layer.
35	Physical Layer removes the CRC and forwards the ***GoodCRC*** to the Protocol Layer.	
36	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent. The Policy Engine resets the ***CapsCounter***, resets the Protocol Layer and starts the ***SwapSourceStartTimer*** which must timeout before sending any ***Source_Capabilities*** Messages.	
	The Fast Role Swap is complete, the roles have been reversed and the Port Partners are free to negotiate for more power.	

Microchip Ex. 1020, p. 291
Microchip v. Aptiv
IPR2024-00496

8.3.2.8 Data Role Swap

8.3.2.8.1 Data Role Swap, Initiated by UFP Operating as Sink

Figure 8-14 shows an example sequence between a Port, which is initially a UFP (Device) and a Sink (Rd asserted), and a Port which is initially a DFP (Host) and a Source (Rp asserted). A Data Role Swap is initiated by the UFP. During the process the Port Partners maintain their operation as either a Source or a Sink (power and Rp/Rd remain constant) but exchange data roles between DFP (Host) and UFP (Device).

Figure 8-14 Data Role Swap, UFP operating as Sink initiates

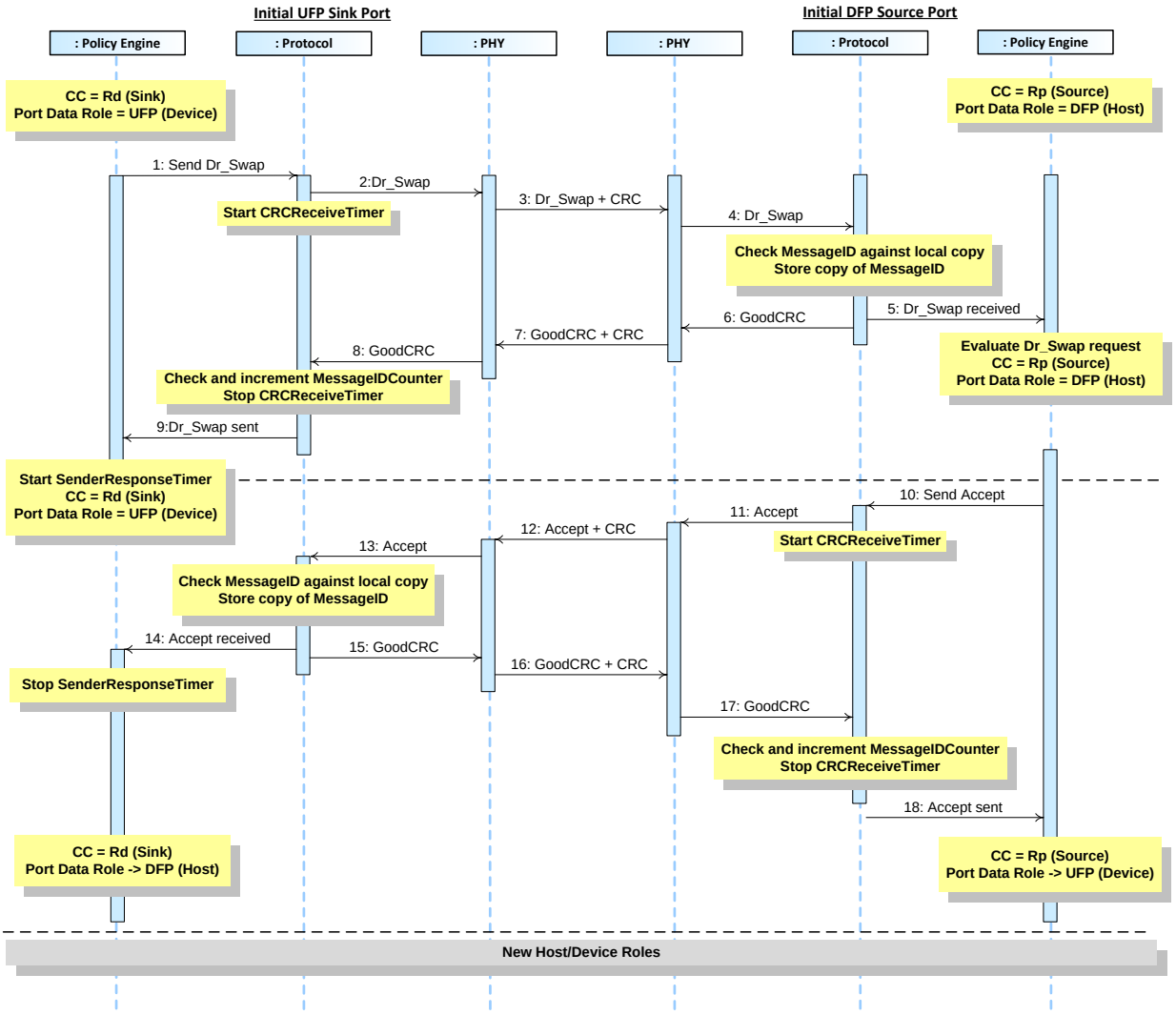


Table 8-14 below provides a detailed explanation of what happens at each labeled step in Figure 8-14 above.

Microchip Ex. 1020, p. 292
Microchip v. Aptiv
IPR2024-00496

Table 8-14 Steps for Data Role Swap, UFP operating as Sink initiates

Step	Initial UFP Sink Port	Initial DFP Source Port
1	Port starts as a UFP (Device) operating as a Sink with Rd asserted and ***Port Data Role*** set to UFP. The Policy Engine directs the Protocol Layer to send a ***DR_Swap*** Message.	Port starts as a DFP (Host) operating as Source with Rp asserted and ***Port Data Role*** set to DFP.
2	Protocol Layer creates the ***DR_Swap*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***DR_Swap*** Message.	Physical Layer receives the ***DR_Swap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***DR_Swap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***DR_Swap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***DR_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine evaluates the ***DR_Swap*** Message and decides that it is able and willing to do the Data Role Swap. It tells the Protocol Layer to form an ***Accept*** Message.
11		Protocol Layer creates the ***Accept*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Accept*** Message and checks the CRC to verify the Message.	Physical Layer appends a CRC and sends the ***Accept*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Accept*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 293
Microchip v. Aptiv
IPR2024-00496

Step	Initial UFP Sink Port	Initial DFP Source Port
18	The Policy Engine requests that Data Role is changed from UFP (Device) to DFP (Host). The Power Delivery role is now a DFP (Host), with ***Port Data Role*** set to DFP, still operating as a Sink (Rd asserted).	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent. The Policy Engine requests that the Data Role is changed to UFP (Device), with ***Port Data Role*** set to UFP and continues supplying power as a Source (Rp asserted).
	The Data Role Swap is complete; the data roles have been reversed while maintaining the direction of power flow.	

Microchip Ex. 1020, p. 294
Microchip v. Aptiv
IPR2024-00496

8.3.2.8.2 Data Role Swap, Initiated by UFP Operating as Source

Figure 8-15 shows an example sequence between a Port, which is initially a UFP (Device) and a Source (Rp asserted), and a Port which is initially a DFP (Host) and a Sink (Rd asserted). A Data Role Swap is initiated by the UFP. During the process the Port Partners maintain their operation as either a Source or a Sink (power and Rp/Rd remain constant) but exchange data roles between DFP (Host) and UFP (Device).

Figure 8-15 Data Role Swap, UFP operating as Source initiates

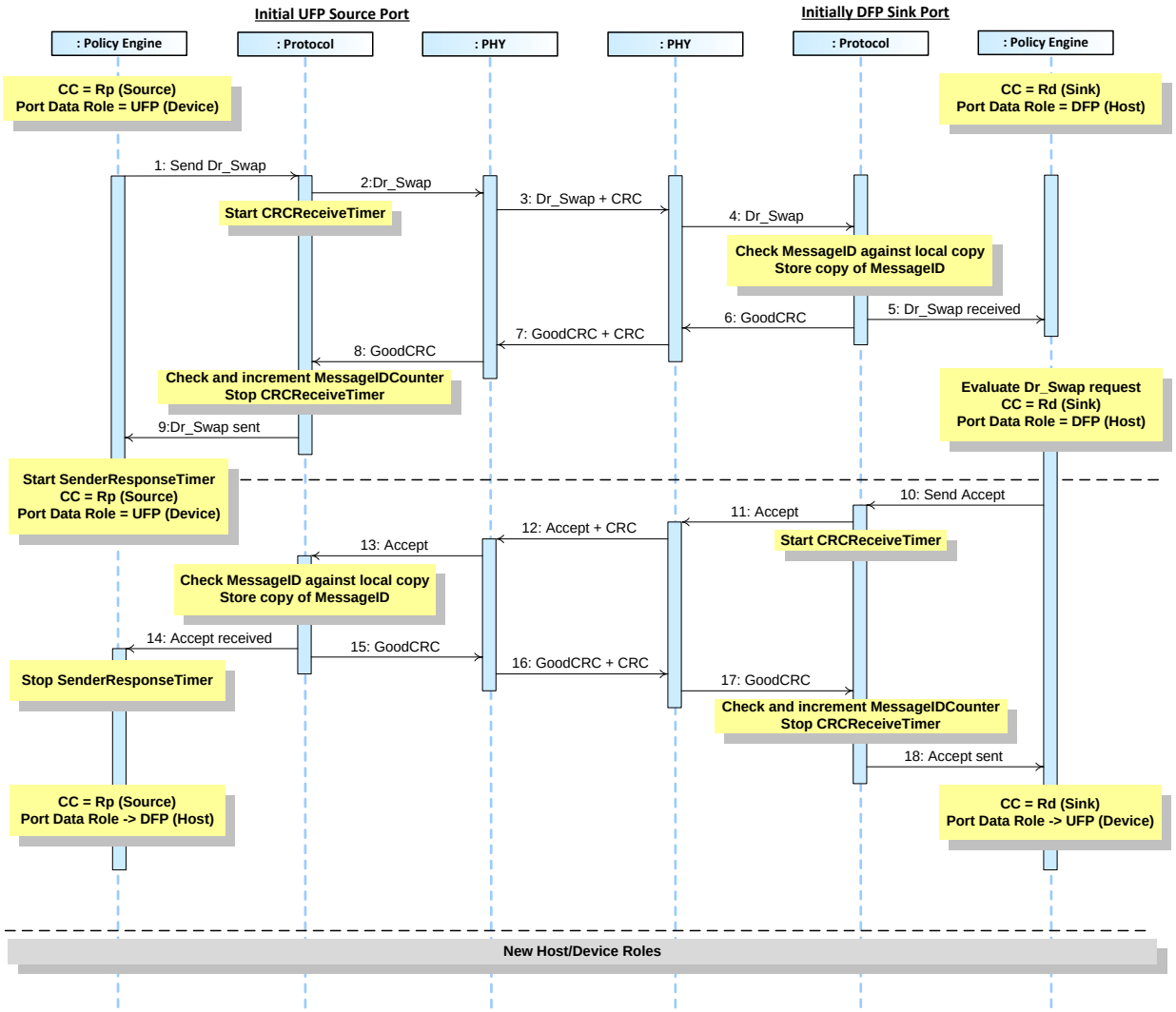


Table 8-15 below provides a detailed explanation of what happens at each labeled step in Figure 8-15 above.

Table 8-15 Steps for Data Role Swap, UFP operating as Source initiates

Step	Initial UFP Source Port	Initial DFP Sink Port
1	Port starts as a UFP (Device) operating as Source with Rp asserted and ***Port Data Role*** set to UFP. The Policy Engine directs the Protocol Layer to send a ***DR_Swap*** Message.	Port starts as a DFP (Host) operating as a Sink with Rd asserted and ***Port Data Role*** set to DFP.
2	Protocol Layer creates the ***DR_Swap*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	

Microchip Ex. 1020, p. 295
Microchip v. Aptiv
IPR2024-00496

Step	Initial UFP Source Port	Initial DFP Sink Port
3	Physical Layer appends CRC and sends the ***DR_Swap*** Message.	Physical Layer receives the ***DR_Swap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***DR_Swap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***DR_Swap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***DR_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine evaluates the ***DR_Swap*** Message and decides that it is able and willing to do the Data Role Swap. It tells the Protocol Layer to form an ***Accept*** Message.
11		Protocol Layer creates the ***Accept*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Accept*** Message and checks the CRC to verify the Message.	Physical Layer appends a CRC and sends the ***Accept*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Accept*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18	The Policy Engine requests that Data Role is changed from UFP (Device) to DFP (Host). The Power Delivery role is now a DFP (Host), and ***Port Data Role*** set to DFP, and continues supplying power as a Source (Rp asserted).	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent. The Policy Engine requests that the Data Role is changed to UFP (Device), with ***Port Data Role*** set to UFP and still operating as a Sink (Rp asserted).
	The Data Role Swap is complete; the data roles have been reversed while maintaining the direction of power flow.	

Microchip Ex. 1020, p. 296
Microchip v. Aptiv
IPR2024-00496

8.3.2.8.3 Data Role Swap, Initiated by DFP Operating as Source

Figure 8-16 shows an example sequence between a Port, which is initially a UFP (Device) and a Sink (Rd asserted), and a Port which is initially a DFP and a Source (Rp asserted). A Data Role Swap is initiated by the DFP. During the process the Port Partners maintain their operation as either a Source or a Sink (power and Rp/Rd remain constant) but exchange data roles between DFP (Host) and UFP (Device).

Figure 8-16 Data Role Swap, DFP operating as Source initiates

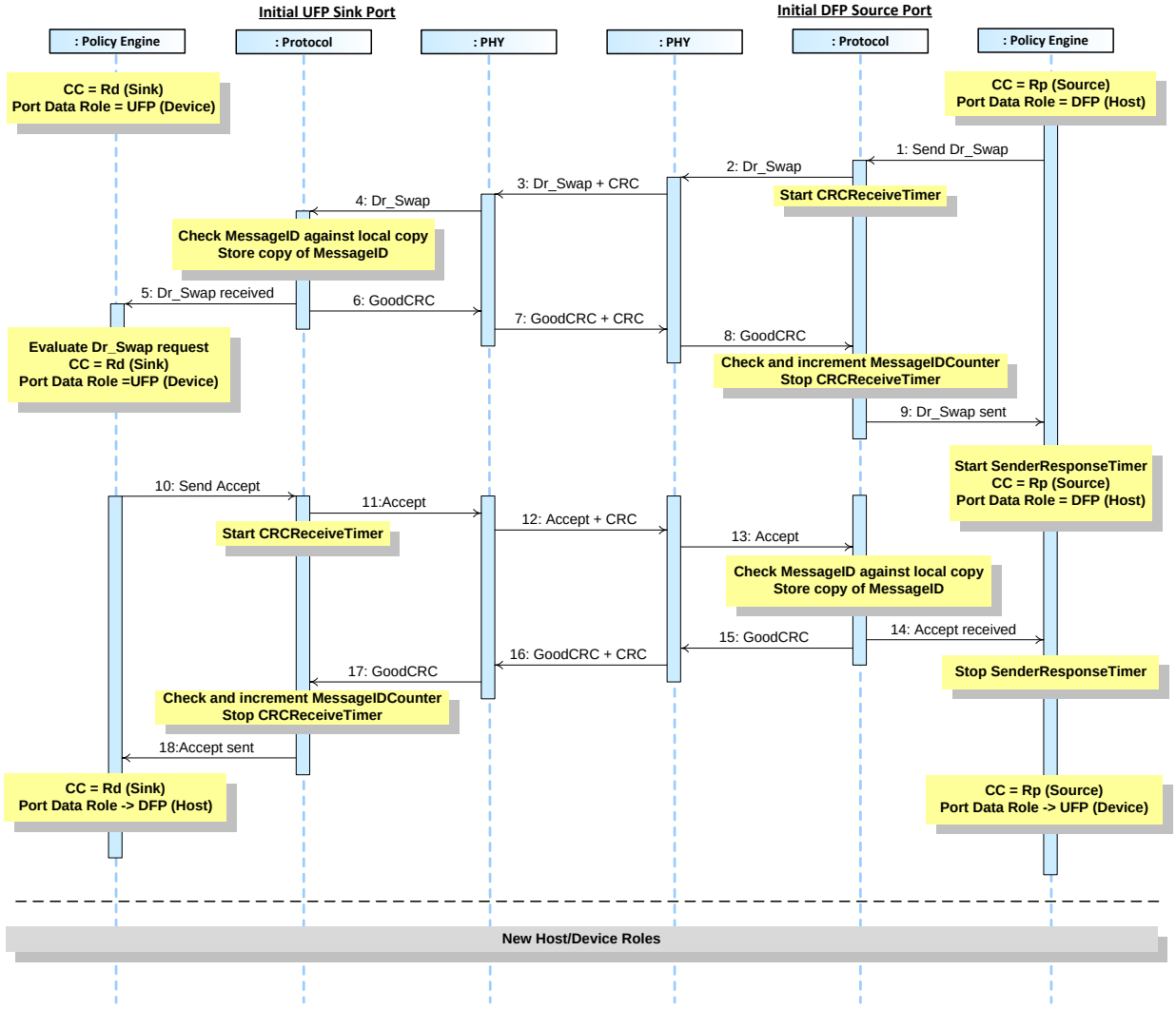


Table 8-16 below provides a detailed explanation of what happens at each labeled step in Figure 8-16 above.

Table 8-16 Steps for Data Role Swap, DFP operating as Source initiates

Step	Initial UFP Sink Port	Initial DFP Source Port
1	Port starts as a UFP (Device) operating as a Sink with Rd asserted and ***Port Data Role*** set to UFP.	Port starts as a DFP (Host) operating as Source with Rp asserted and ***Port Data Role*** set to DFP. The Policy Engine directs the Protocol Layer to send a ***DR_Swap*** Message.
2		Protocol Layer creates the ***DR_Swap*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.

Microchip Ex. 1020, p. 297
Microchip v. Aptiv
IPR2024-00496

Step	Initial UFP Sink Port	Initial DFP Source Port
3	Physical Layer receives the ***DR_Swap*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***DR_Swap*** Message.
4	Physical Layer removes the CRC and forwards the ***DR_Swap*** Message to the Protocol Layer.	
5	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***DR_Swap*** Message information to the Policy Engine that consumes it.	
6	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
7	Physical Layer appends CRC and sends the ***GoodCRC*** Message.	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.
8		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
9		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***DR_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.
10	Policy Engine evaluates the ***DR_Swap*** Message and decides that it is able and willing to do the Data Role Swap. It tells the Protocol Layer to form an ***Accept*** Message.	
11	Protocol Layer creates the ***Accept*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
12	Physical Layer appends a CRC and sends the ***Accept*** Message.	Physical Layer receives the ***Accept*** Message and checks the CRC to verify the Message.
13		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Accept*** Message information to the Policy Engine that consumes it.
14		The Policy Engine stops the ***SenderResponseTimer***.
15		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
16	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.
17	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
18	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent. . The Policy Engine requests that the Data Role is changed to DFP (Host), with ***Port Data Role*** set to DFP, still operating as a Sink (Rd asserted).	The Policy Engine requests that Data Role is changed from DFP (Host) to UFP (Device). The Power Delivery role is now a UFP (Device), with ***Port Data Role*** set to UFP, and continues supplying power as a Source (Rp asserted).
	The Data Role Swap is complete; the data roles have been reversed while maintaining the direction of power flow.	

Microchip Ex. 1020, p. 298
Microchip v. Aptiv
IPR2024-00496

8.3.2.8.4 Data Role Swap, Initiated by DFP Operating as Sink

Figure 8-17 shows an example sequence between a Port, which is initially a UFP (Device) and a Source (Rp asserted), and a Port which is initially a DFP (Host) and a Sink (Rd asserted). A Data Role Swap is initiated by the DFP. During the process the Port Partners maintain their operation as either a Source or a Sink (power and Rp/Rd remain constant) but exchange data roles between DFP (Host) and UFP (Device).

Figure 8-17 Data Role Swap, DFP operating as Sink initiates

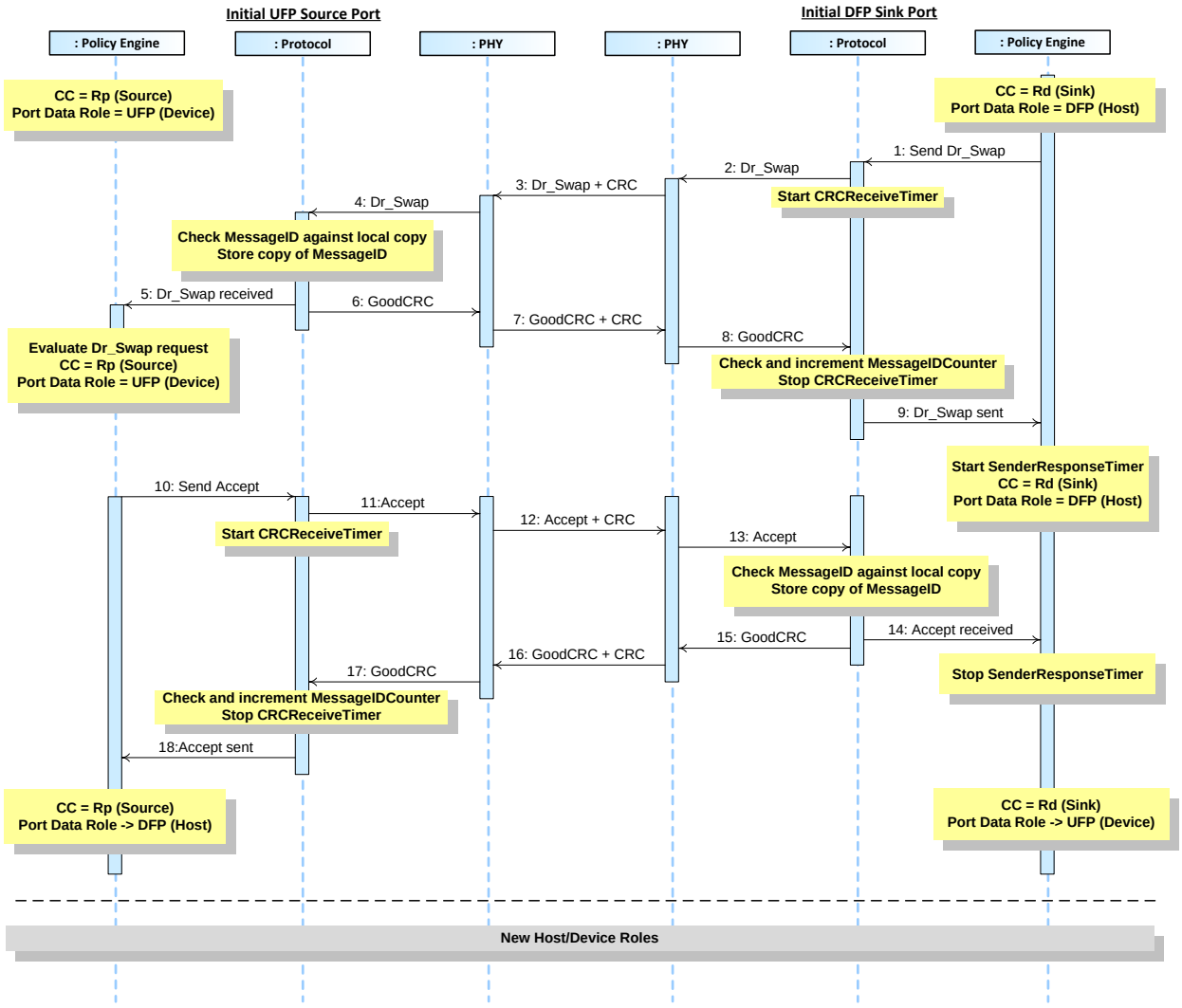


Table 8-17 below provides a detailed explanation of what happens at each labeled step in Figure 8-17 above.

Table 8-17 Steps for Data Role Swap, DFP operating as Sink initiates

Step	Initial UFP Source Port	Initial DFP Sink Port
1	Port starts as a UFP (Device) operating as Source with Rp asserted and ***Port Data Role*** set to UFP.	Port starts as a DFP (Host) operating as a Sink with Rd asserted and ***Port Data Role*** set to DFP. The Policy Engine directs the Protocol Layer to send a ***DR_Swap*** Message.
2		Protocol Layer creates the ***DR_Swap*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.

Microchip Ex. 1020, p. 299
Microchip v. Aptiv
IPR2024-00496

Step	Initial UFP Source Port	Initial DFP Sink Port
3	Physical Layer receives the ***DR_Swap*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***DR_Swap*** Message.
4	Physical Layer removes the CRC and forwards the ***DR_Swap*** Message to the Protocol Layer.	
5	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***DR_Swap*** Message information to the Policy Engine that consumes it.	
6	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
7	Physical Layer appends CRC and sends the ***GoodCRC*** Message.	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.
8		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
9		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***DR_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.
10	Policy Engine evaluates the ***DR_Swap*** Message and decides that it is able and willing to do the Data Role Swap. It tells the Protocol Layer to form an ***Accept*** Message.	
11	Protocol Layer creates the ***Accept*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
12	Physical Layer appends a CRC and sends the ***Accept*** Message.	Physical Layer receives the ***Accept*** Message and checks the CRC to verify the Message.
13		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Accept*** Message information to the Policy Engine that consumes it.
14		The Policy Engine stops the ***SenderResponseTimer***.
15		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
16	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.
17	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
18	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent. The Policy Engine requests that the Data Role is changed to DFP (Host), with ***Port Data Role*** set to DFP, and continues supplying power as a Source (Rp asserted).	The Policy Engine requests that Data Role is changed from DFP (Host) to UFP (Device). The Power Delivery role is now a UFP (Device), with ***Port Data Role*** set to UFP, still operating as a Sink (Rd asserted).
	The Data Role Swap is complete; the data roles have been reversed while maintaining the direction of power flow.	

Microchip Ex. 1020, p. 300
Microchip v. Aptiv
IPR2024-00496

8.3.2.9 VCONN Swap

8.3.2.9.1 Source to Sink VCONN Source Swap

Figure 8-18 shows an example sequence between a Source and Sink, where the Source is initially supplying VCONN and then tells the Sink to supply VCONN. During the process the Port Partners, keep their role as Source or Sink, maintain their operation as either a Source or a Sink (power remains constant) but exchange the VCONN Source from the Source to the Sink.

Figure 8-18 Source to Sink VCONN Source Swap

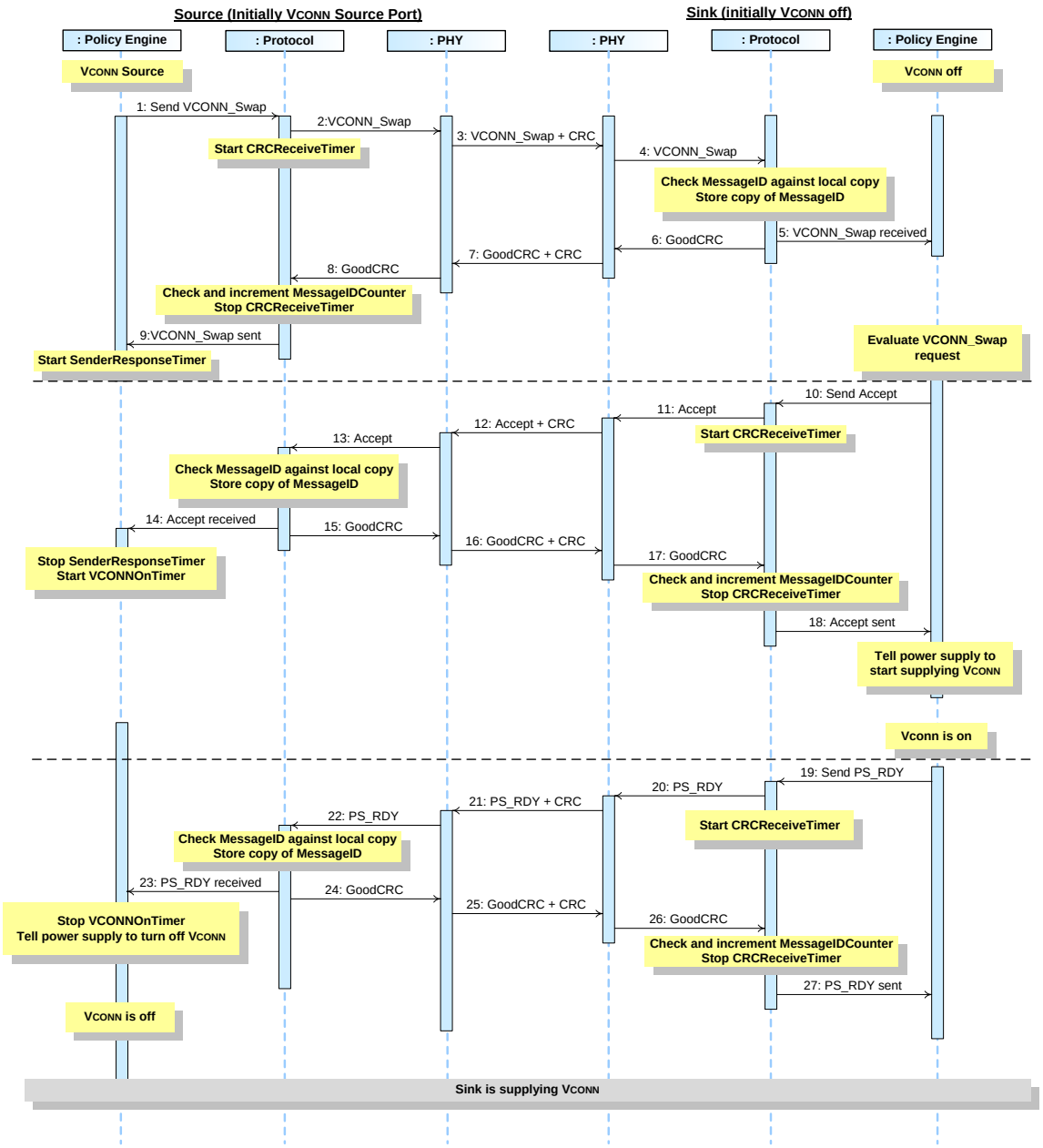


Table 8-18 below provides a detailed explanation of what happens at each labeled step in Figure 8-18 above.

Microchip Ex. 1020, p. 301
Microchip v. Aptiv
IPR2024-00496

Table 8-18 Steps for Source to Sink VCONN Source Swap

Step	Source (initially VCONN Source)	Sink (Initially VCONN off)
1	The Source starts as the VCONN Source. The Policy Engine directs the Protocol Layer to send a ***VCONN_Swap*** Message.	The Sink starts with VCONN off.
2	Protocol Layer creates the ***VCONN_Swap*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***VCONN_Swap*** Message.	Physical Layer receives the ***VCONN_Swap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***VCONN_Swap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***VCONN_Swap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***VCONN_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine evaluates the ***VCONN_Swap*** Message sent by the Source and decides that it is able and willing to do the VCONN Swap. It tells the Protocol Layer to form an ***Accept*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Accept*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Accept*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Accept*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer*** and starts the ***VCONNOnTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 302
Microchip v. Aptiv
IPR2024-00496

Step	Source (initially VCONN Source)	Sink (Initially VCONN off)
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent. The Policy Engine asks the Device Policy Manager to turn on VCONN.
19		The Device Policy Manager informs the Policy Engine that its power supply is supplying VCONN. The Policy Engine directs the Protocol Layer to generate a ***PS_RDY*** Message to tell the Source it can turn off VCONN.
20		Protocol Layer creates the ***PS_RDY*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
21	Physical Layer receives the ***PS_RDY*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***PS_RDY*** Message.
22	Physical Layer removes the CRC and forwards the ***PS_RDY*** Message to the Protocol Layer.	
23	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PS_RDY*** Message information to the Policy Engine that consumes it.	
24	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
25	Physical Layer appends a CRC and sends the ***GoodCRC*** Message. The Policy Engine stops the ***VCONNOnTimer***, and tells the power supply to stop sourcing VCONN.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
26		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
27	VCONN is off.	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent.
	The Sink is now the VCONN Source and the Source has VCONN turned off.	

Microchip Ex. 1020, p. 303
Microchip v. Aptiv
IPR2024-00496

8.3.2.9.2 Sink to Source VCONN Source Swap

Figure 8-19 shows an example sequence between a Source and Sink, where the Sink is initially supplying VCONN and then the Source tells the Sink that it will become the VCONN Source. During the process the Port Partners, keep their role as Source or Sink, maintain their operation as either a Source or a Sink (power remains constant) but exchange the VCONN Source from the Sink to the Source.

Figure 8-19 Sink to Source VCONN Source Swap

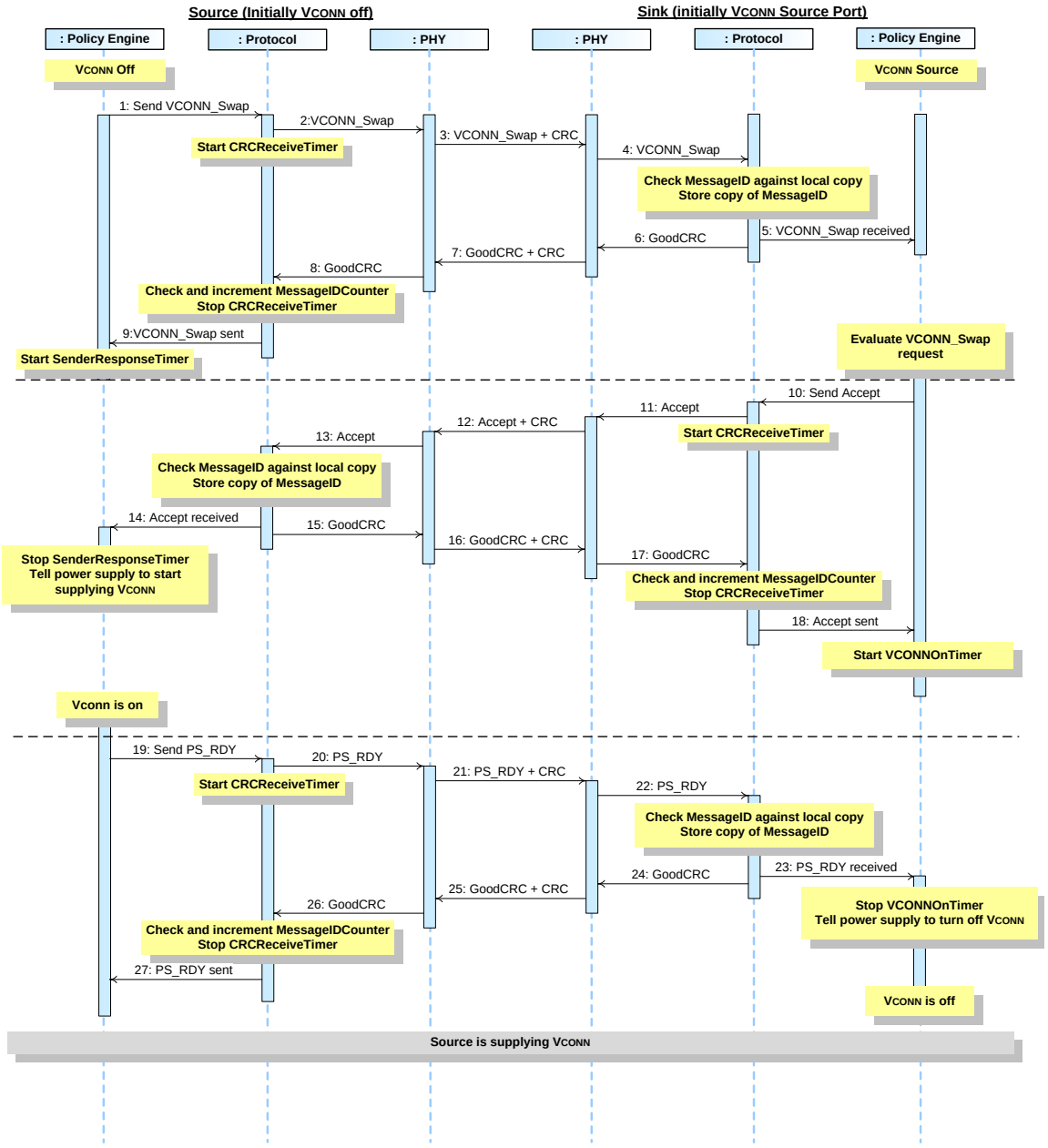


Table 8-19 below provides a detailed explanation of what happens at each labeled step in Figure 8-19 above.

Microchip Ex. 1020, p. 304
Microchip v. Aptiv
IPR2024-00496

Table 8-19 Steps for Sink to Source VCONN Source Swap

Step	Source	Sink
1	The Source starts with VCONN off. The Policy Engine directs the Protocol Layer to send a ***VCONN_Swap*** Message.	The Sink starts as the VCONN Source.
2	Protocol Layer creates the ***VCONN_Swap*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***VCONN_Swap*** Message.	Physical Layer receives the ***VCONN_Swap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***VCONN_Swap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***VCONN_Swap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***VCONN_Swap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine evaluates the ***VCONN_Swap*** Message sent by the Source and decides that it is able and willing to do the VCONN Swap. It tells the Protocol Layer to form an ***Accept*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Accept*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Accept*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Accept*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***. The Policy Engine tells the Device Policy Manger to turn on VCONN.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 305
Microchip v. Aptiv
IPR2024-00496

Step	Source	Sink
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent. The Policy Engine starts the ***VCONNOnTimer***.
19	The Device Policy Manager tells the Policy Engine that its power supply is supplying VCONN. The Policy Engine directs the Protocol Layer to generate a ***PS_RDY*** Message to tell the Sink it can turn off VCONN.	
20	Protocol Layer creates the ***PS_RDY*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
21	Physical Layer appends a CRC and sends the ***PS_RDY*** Message.	Physical Layer receives the ***PS_RDY*** Message and compares the CRC it calculated with the one sent to verify the Message.
22		Physical Layer removes the CRC and forwards the ***PS_RDY*** Message to the Protocol Layer.
23		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***PS_RDY*** Message information to the Policy Engine that consumes it.
24		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
25	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***GoodCRC*** Message. The Policy Engine stops the ***VCONNOnTimer***, and tells the power supply to stop sourcing VCONN.
26	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
27	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***PS_RDY*** Message was successfully sent.	VCONN is off.
	The Source is now the VCONN Source and the Sink has VCONN turned off.	

Microchip Ex. 1020, p. 306
Microchip v. Aptiv
IPR2024-00496

8.3.2.10 Additional Capabilities and Status

8.3.2.10.1 Alert

8.3.2.10.1.1 Source sends Alert to a Sink

Figure 8-20 shows an example sequence between a Source and a Sink where the Source alerts the Sink that there has been a status change. This AMS will be followed by getting the Source status to determine further details of the alert (see Section 8.3.2.10.2).

Figure 8-20 Source Alert to Sink

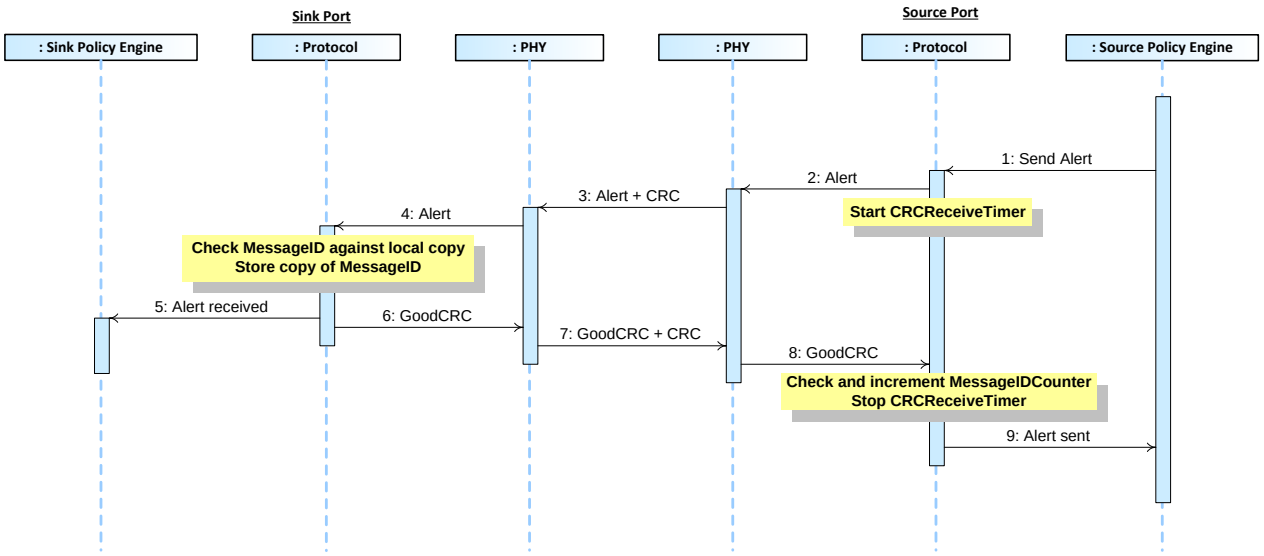


Table 8-50 below provides a detailed explanation of what happens at each labeled step in Figure 8-20 above.

Table 8-20 Steps for Source Alert to Sink

Step	Sink	Source
1		The Device Policy Manager indicates a Source alert condition. The Policy Engine tells the Protocol Layer to form an ***Alert*** Message.
2		Protocol Layer creates the ***Alert*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
3	Physical Layer receives the ***Alert*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Alert*** Message.
4	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Alert*** Message to the Policy Engine that consumes it.	
5	The Policy Engine informs the Device Policy Manager.	
6	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	

Microchip Ex. 1020, p. 307
Microchip v. Aptiv
IPR2024-00496

Step	Sink	Source
7	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
8		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
9		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Alert*** Message was successfully sent.

Microchip Ex. 1020, p. 308
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.1.1 Sink sends Alert to a Source

Figure 8-20 shows an example sequence between a Source and a Sink where the Sink alerts the Source that there has been a status change. This AMS will be followed by getting the Sink status to determine further details of the alert (see Section 8.3.2.10.2).

Figure 8-21 Sink Alert to Source

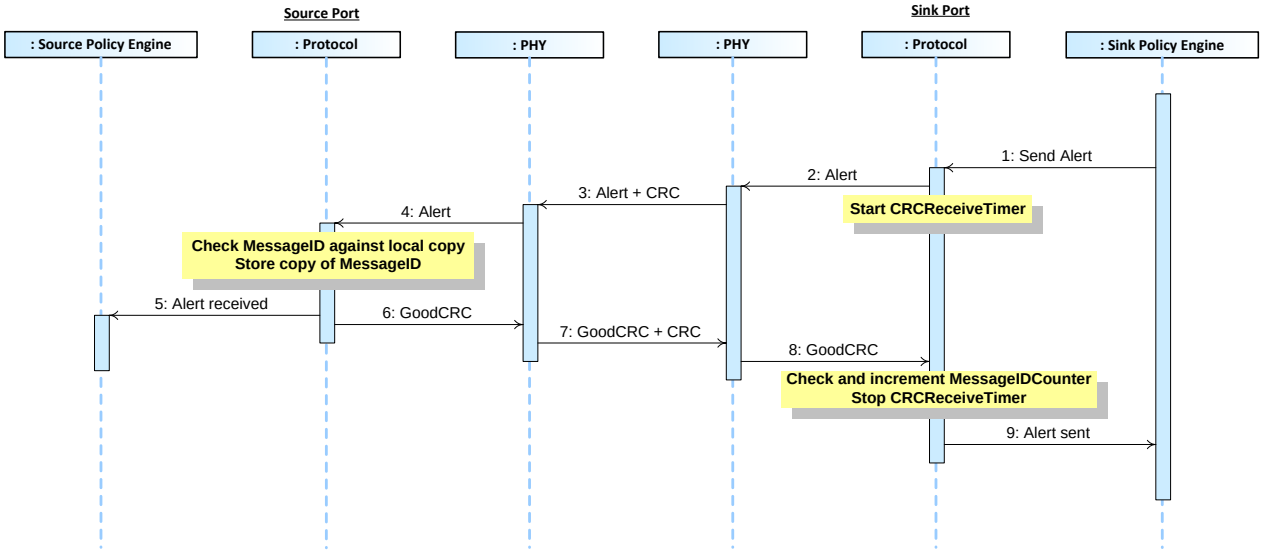


Table 8-50 below provides a detailed explanation of what happens at each labeled step in Figure 8-20 above.

Table 8-21 Steps for Sink Alert to Source

Step	Source	Sink
1		The Device Policy Manager indicates a Sink alert condition. The Policy Engine tells the Protocol Layer to form an ***Alert*** Message.
2		Protocol Layer creates the ***Alert*** Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
3	Physical Layer receives the ***Alert*** Message and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Alert*** Message.
4	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Alert*** Message to the Policy Engine that consumes it.	
5	The Policy Engine informs the Device Policy Manager.	
6	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
7	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
8		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 309
Microchip v. Aptiv
IPR2024-00496

Step	Source	Sink
9		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Alert*** Message was successfully sent.

Microchip Ex. 1020, p. 310
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.2 Status

8.3.2.10.2.1 Sink Gets Source Status

Figure 8-22 shows an example sequence between a Source and a Sink where, after the Sink has received an alert (see Section 8.3.2.10.1) that there has been a status change, the Sink gets more details on the change.

Figure 8-22 Sink Gets Source Status

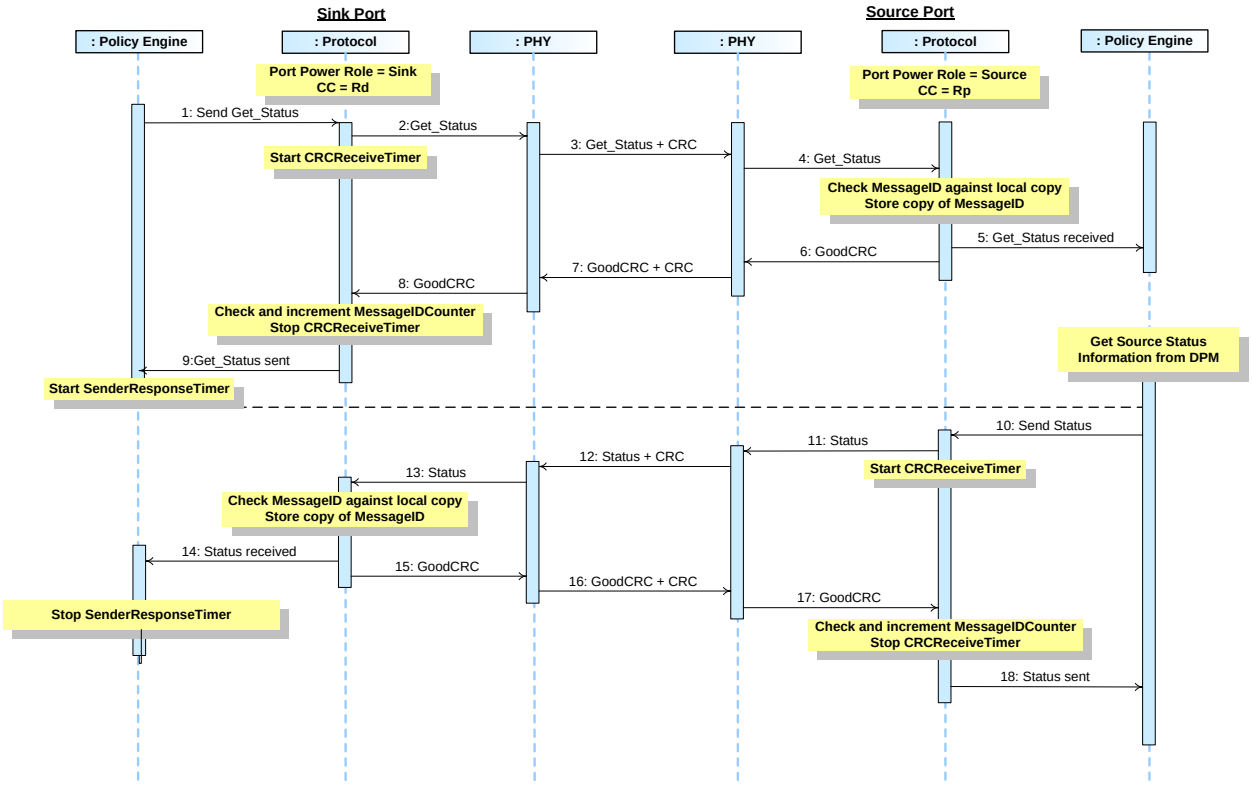


Table 8-22below provides a detailed explanation of what happens at each labeled step in Figure 8-22 above.

Table 8-22 Steps for a Sink getting Source status Sequence

Step	Sink Port	Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Status*** Message.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Status*** Message.	Physical Layer receives the ***Get_Status*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Status*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 311
Microchip v. Aptiv
IPR2024-00496

Step	Sink Port	Source Port
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Status*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Status*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present Source status which is provided. The Policy Engine tells the Protocol Layer to form a ***Status*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Status*** Message.	Physical Layer appends a CRC and sends the ***Status*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Status*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Status*** Message was successfully sent.
	The Source has informed the Sink of its present status.	

Microchip Ex. 1020, p. 312
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.2.1 Source Gets Sink Status

Figure 8-22 shows an example sequence between a Source and a Sink where, after the Source has received an alert (see Section 8.3.2.10.1) that there has been a status change, the Source gets more details on the change.

Figure 8-23 Source Gets Sink Status

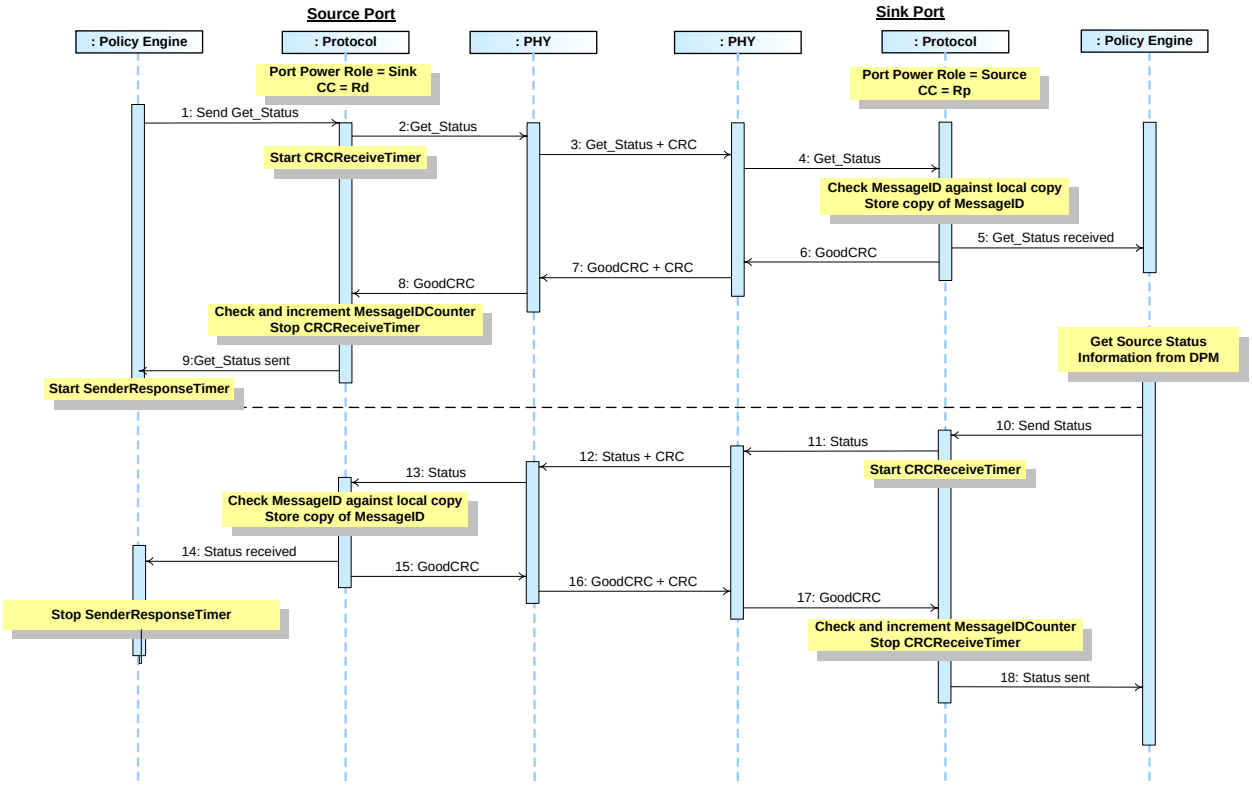


Table 8-22below provides a detailed explanation of what happens at each labeled step in Figure 8-22 above.

Table 8-23 Steps for a Source getting Sink status Sequence

Step	Source Port	Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Status*** Message.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Status*** Message.	Physical Layer receives the ***Get_Status*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Status*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Status*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 313
Microchip v. Aptiv
IPR2024-00496

Step	Source Port	Sink Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Status*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present Source status which is provided. The Policy Engine tells the Protocol Layer to form a ***Status*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Status*** Message.	Physical Layer appends a CRC and sends the ***Status*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Status*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Status*** Message was successfully sent.
	The Sink has informed the Source of its present status.	

Microchip Ex. 1020, p. 314
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.3 Sink Gets Source Capabilities

Figure 8-24 shows an example sequence between a Source and a Sink when the Sink gets the Source's capabilities.

Figure 8-24 Sink Gets Source's Capabilities

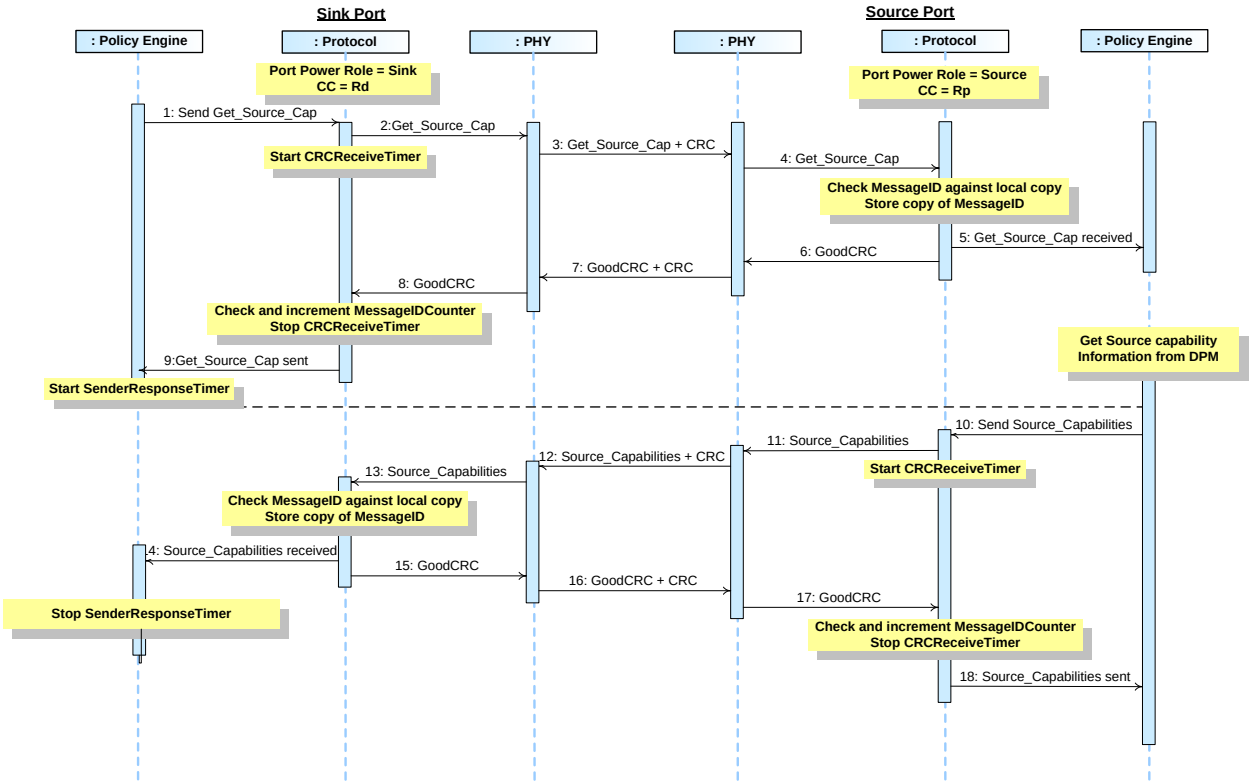


Table 8-24 below provides a detailed explanation of what happens at each labeled step in Figure 8-24 above.

Table 8-24 Steps for a Sink getting Source capabilities Sequence

Step	Sink Port	Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Source_Cap*** Message.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Source_Cap*** Message.	Physical Layer receives the ***Get_Source_Cap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Source_Cap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Source_Cap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.

Microchip Ex. 1020, p. 315
Microchip v. Aptiv
IPR2024-00496

Step	Sink Port	Source Port
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Source_Cap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present Source capabilities which are provided. The Policy Engine tells the Protocol Layer to form a ***Source_Capabilities*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Source_Capabilities*** Message.	Physical Layer appends a CRC and sends the ***Source_Capabilities*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Source_Capabilities*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Source_Capabilities*** Message was successfully sent.
	The Source has informed the Sink of its capabilities.	

Microchip Ex. 1020, p. 316
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.4 Dual-Role Source Gets Source Capabilities from a Dual-Role Sink

Figure 8-25 shows an example sequence between a Dual-Role Source and a Dual-Role Sink when the Source gets the Sink's capabilities as a Source.

Figure 8-25 Dual-Role Source Gets Dual-Role Sink's Capabilities as a Source

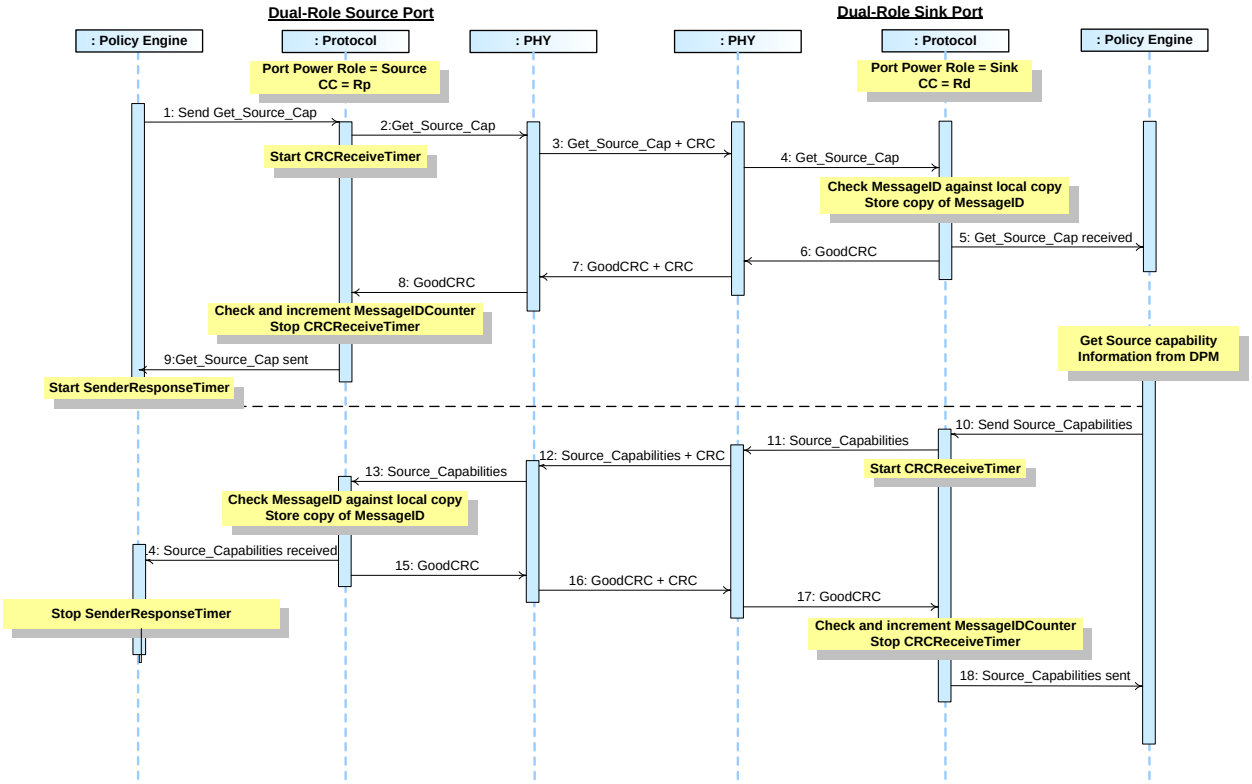


Table 8-25 below provides a detailed explanation of what happens at each labeled step in Figure 8-25 above.

Table 8-25 Steps for a Dual-Role Source getting Dual-Role Sink's capabilities as a Source Sequence

Step	Dual-Role Source Port	Dual-Role Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Source_Cap*** Message.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Source_Cap*** Message.	Physical Layer receives the ***Get_Source_Cap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Source_Cap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Source_Cap*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 317
Microchip v. Aptiv
IPR2024-00496

Step	Dual-Role Source Port	Dual-Role Sink Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Source_Cap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present Source capabilities which are provided. The Policy Engine tells the Protocol Layer to form a ***Source_Capabilities*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Source_Capabilities*** Message.	Physical Layer appends a CRC and sends the ***Source_Capabilities*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Source_Capabilities*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Source_Capabilities*** Message was successfully sent.
	The Dual-Role Sink has informed the Dual-Role Source of its capabilities.	

Microchip Ex. 1020, p. 318
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.5 Source Gets Sink Capabilities

Figure 8-26 shows an example sequence between a Source and a Sink when the Source gets the Sink's capabilities.

Figure 8-26 Source Gets Sink's Capabilities

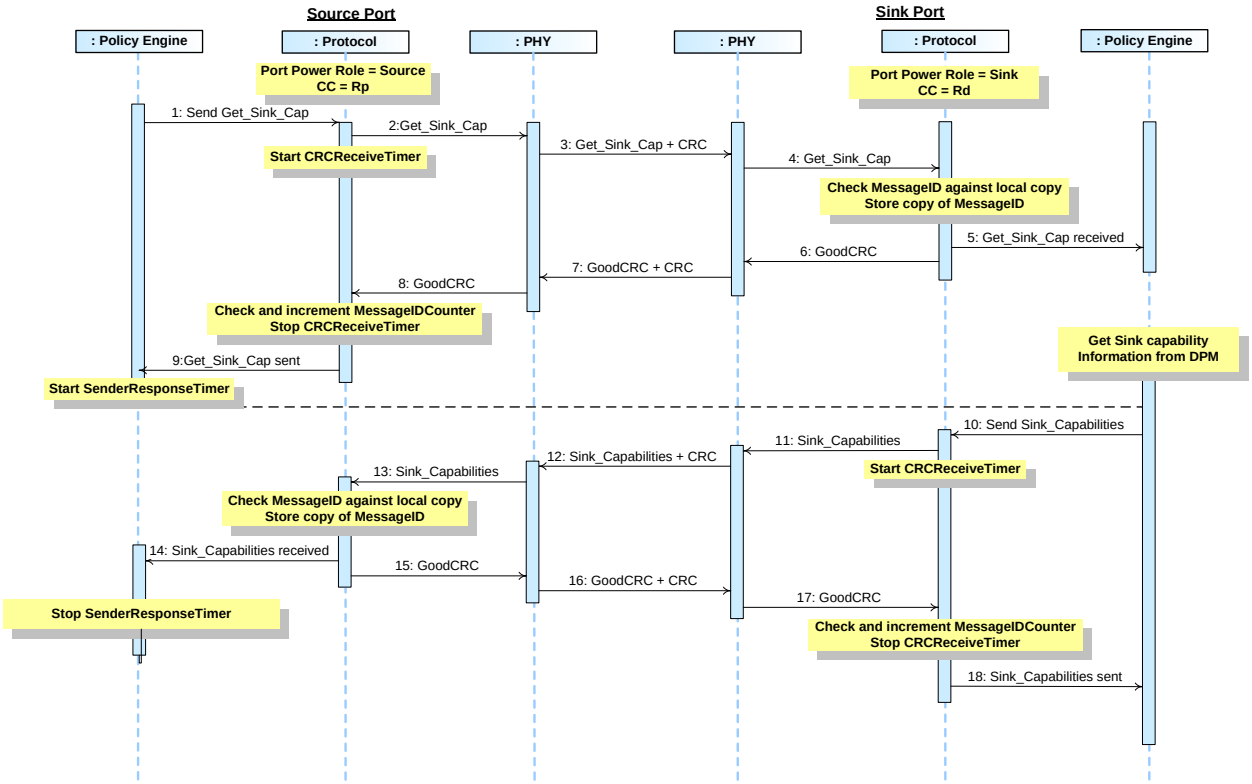


Table 8-26 below provides a detailed explanation of what happens at each labeled step in Figure 8-26 above.

Table 8-26 Steps for a Source getting Sink capabilities Sequence

Step	Source Port	Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Sink_Cap*** Message.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Sink_Cap*** Message.	Physical Layer receives the ***Get_Sink_Cap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Sink_Cap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Sink_Cap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.

Microchip Ex. 1020, p. 319
Microchip v. Aptiv
IPR2024-00496

Step	Source Port	Sink Port
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Sink_Cap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present Sink capabilities which are provided. The Policy Engine tells the Protocol Layer to form a ***Sink_Capabilities*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Sink_Capabilities*** Message.	Physical Layer appends a CRC and sends the ***Sink_Capabilities*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Sink_Capabilities*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Sink_Capabilities*** Message was successfully sent.
	The Sink has informed the Source of its capabilities.	

Microchip Ex. 1020, p. 320
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.6 Dual-Role Sink Get Sink Capabilities from a Dual-Role Source

Figure 8-27 shows an example sequence between a Dual-Role Source and a Dual-Role Sink when the Dual-Role Sink gets the Dual-Role Source's capabilities as a Sink.

Figure 8-27 Dual-Role Sink Gets Dual-Role Source's Capabilities as a Sink

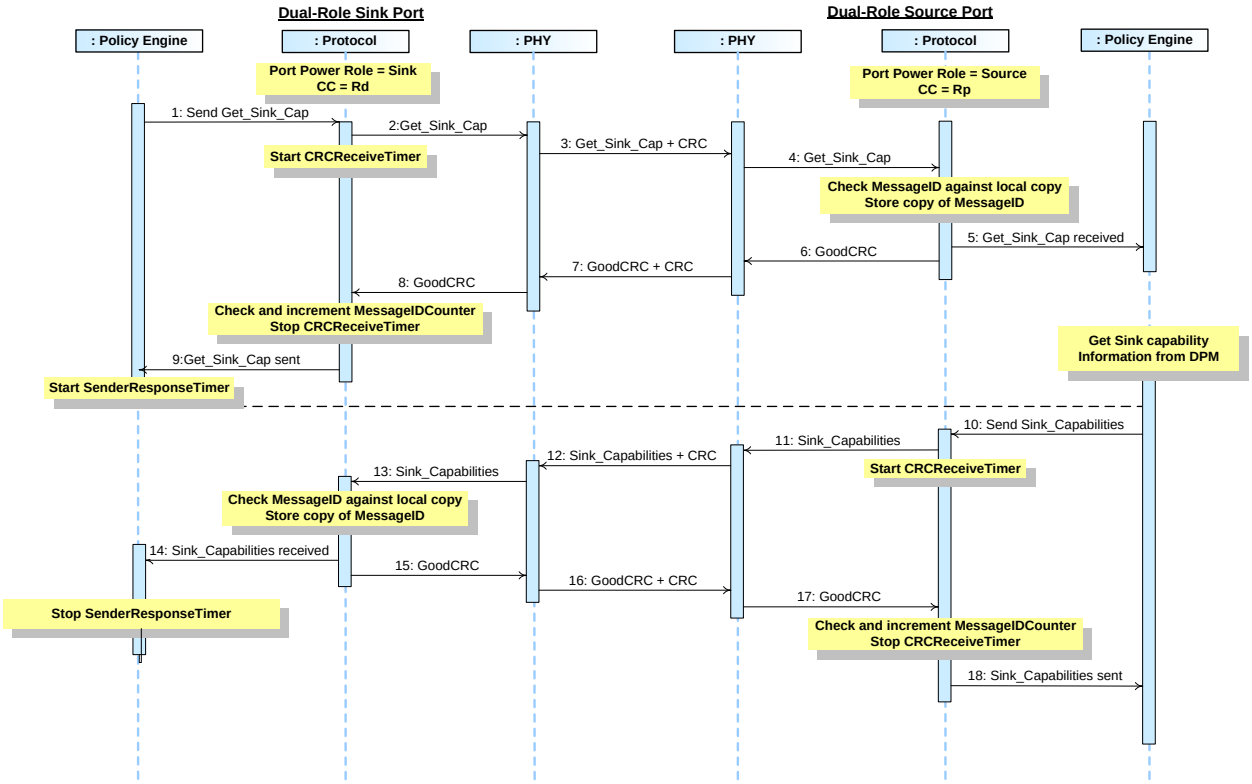


Table 8-27 below provides a detailed explanation of what happens at each labeled step in Figure 8-27 above.

Table 8-27 Steps for a Dual-Role Sink getting Dual-Role Source capabilities as a Sink Sequence

Step	Dual-Role Sink Port	Dual-Role Source Port
1	The Port has ***Port Power Role*** set to Dual-Role Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Sink_Cap*** Message.	The Port has ***Port Power Role*** set to Dual-Role Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Sink_Cap*** Message.	Physical Layer receives the ***Get_Sink_Cap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Sink_Cap*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Sink_Cap*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 321
Microchip v. Aptiv
IPR2024-00496

Step	Dual-Role Sink Port	Dual-Role Source Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Sink_Cap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present Dual-Role Source capabilities which are provided. The Policy Engine tells the Protocol Layer to form a ***Sink_Capabilities*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Sink_Capabilities*** Message.	Physical Layer appends a CRC and sends the ***Sink_Capabilities*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Sink_Capabilities*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Sink_Capabilities*** Message was successfully sent.
	The Dual-Role Source has informed the Dual-Role Sink of its capabilities as a Sink.	

Microchip Ex. 1020, p. 322
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.7 Sink Gets Source Extended Capabilities

Figure 8-28 shows an example sequence between a Source and a Sink when the Sink gets the Source's extended capabilities.

Figure 8-28 Sink Gets Source's Extended Capabilities

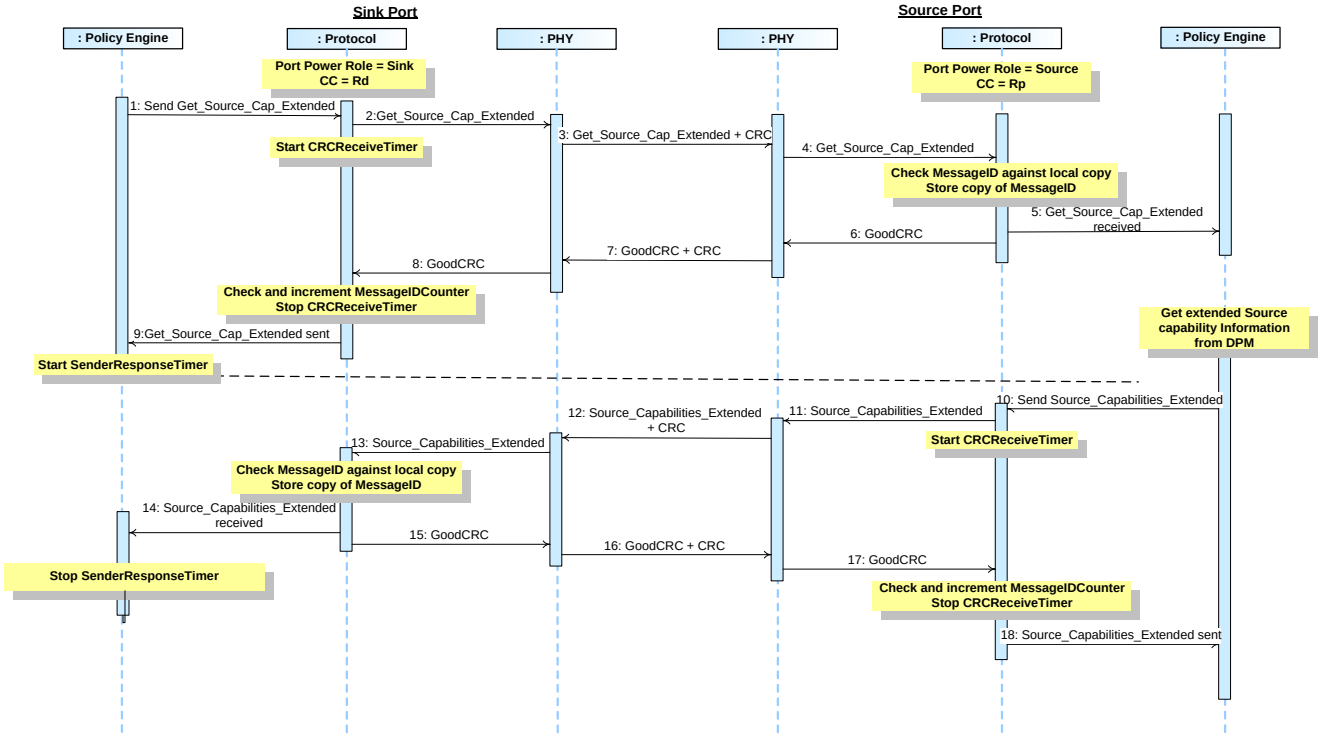


Table 8-28 below provides a detailed explanation of what happens at each labeled step in Figure 8-28 above.

Table 8-28 Steps for a Sink getting Source extended capabilities Sequence

Step	Sink Port	Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Source_Cap_Extended*** Message.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Source_Cap_Extended*** Message.	Physical Layer receives the ***Get_Source_Cap_Extended*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Source_Cap_Extended*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Source_Cap_Extended*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 323
Microchip v. Aptiv
IPR2024-00496

Step	Sink Port	Source Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Source_Cap_Extended*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present extended Source capabilities which are provided. The Policy Engine tells the Protocol Layer to form a ***Source_Capabilities_Extended*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Source_Capabilities_Extended*** Message.	Physical Layer appends a CRC and sends the ***Source_Capabilities_Extended*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Source_Capabilities_Extended*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Source_Capabilities_Extended*** Message was successfully sent.
	The Source has informed the Sink of its extended capabilities.	

Microchip Ex. 1020, p. 324
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.8 Dual-Role Source Gets Source Capabilities Extended from a Dual-Role Sink

Figure 8-28 shows an example sequence between a Source and a Sink when the Dual-Role Source gets the Dual-Role Sink's extended capabilities as a Source.

Figure 8-29 Dual-Role Source Gets Dual-Role Sink's Extended Capabilities

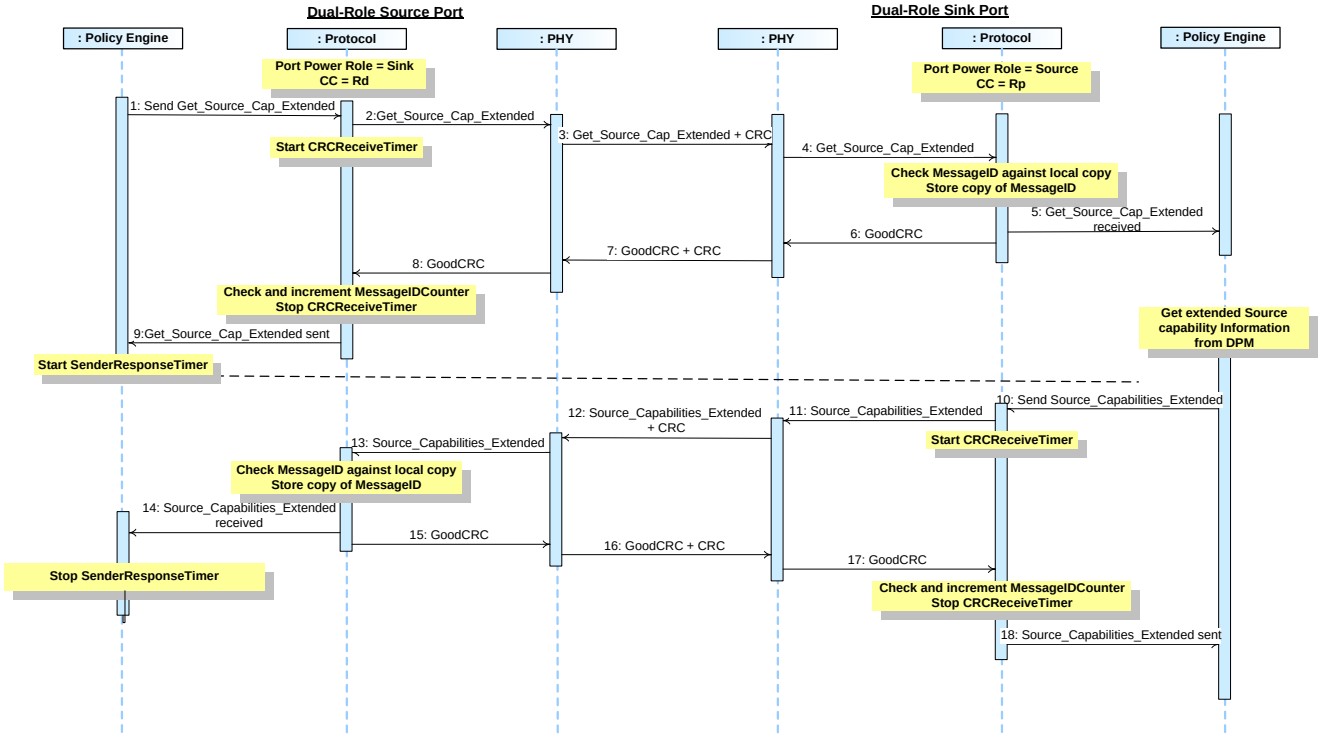


Table 8-28 below provides a detailed explanation of what happens at each labeled step in Figure 8-28 above.

Table 8-29 Steps for a Dual-Role Source getting Dual-Role Sink extended capabilities Sequence

Step	Dual-Role Source Port	Dual-Role Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Source_Cap_Extended*** Message.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Source_Cap_Extended*** Message.	Physical Layer receives the ***Get_Source_Cap_Extended*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Source_Cap_Extended*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Source_Cap_Extended*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 325
Microchip v. Aptiv
IPR2024-00496

Step	Dual-Role Source Port	Dual-Role Sink Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Source_Cap_Extended*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present extended Source capabilities which are provided. The Policy Engine tells the Protocol Layer to form a ***Source_Capabilities_Extended*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Source_Capabilities_Extended*** Message.	Physical Layer appends a CRC and sends the ***Source_Capabilities_Extended*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Source_Capabilities_Extended*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Source_Capabilities_Extended*** Message was successfully sent.
	The Dual-Role Sink has informed the Dual-Role Source of its extended capabilities as a Source.	

Microchip Ex. 1020, p. 326
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.9 Sink Gets Battery Capabilities

Figure 8-30 shows an example sequence between a Source and a Sink when the Sink gets the Source's Battery capabilities for a given Battery.

Figure 8-30 Sink Gets Source's Battery Capabilities

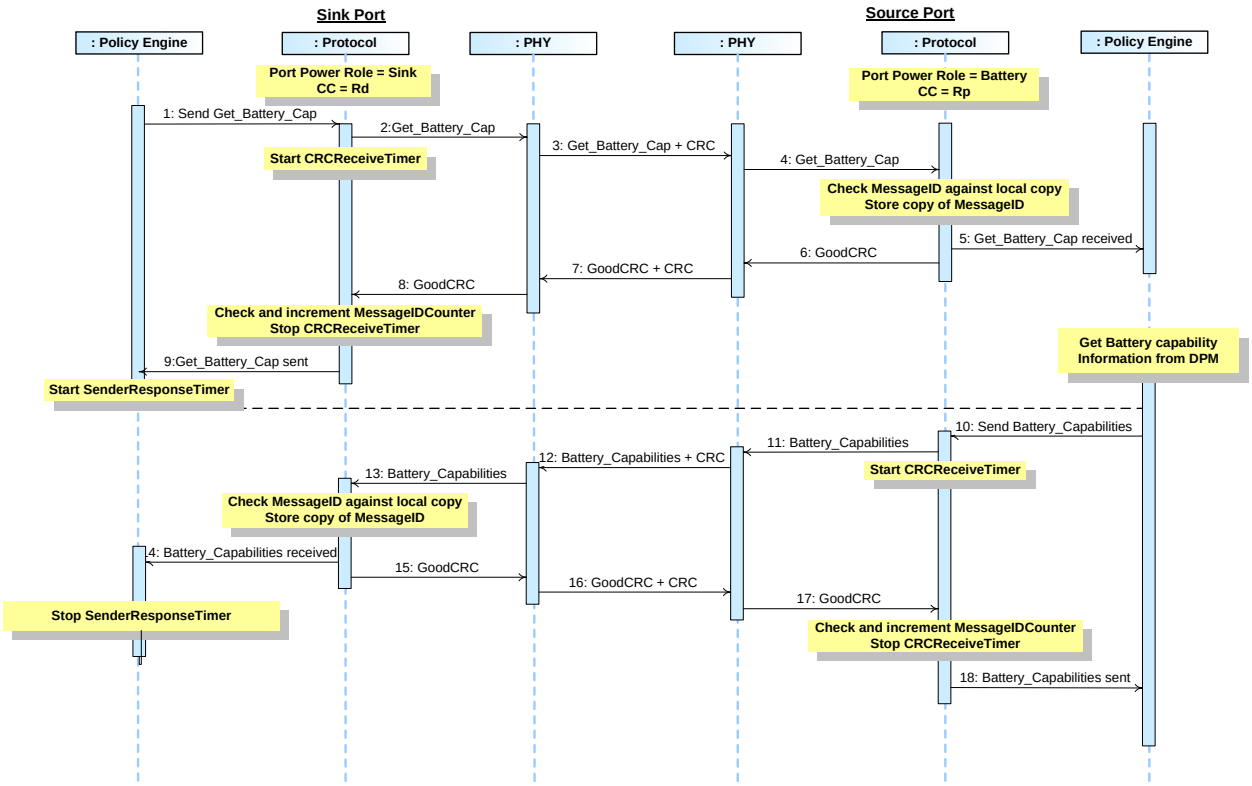


Table 8-30 below provides a detailed explanation of what happens at each labeled step in Figure 8-30 above.

Table 8-30 Steps for a Sink getting Source Battery capabilities Sequence

Step	Sink Port	Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Battery_Cap*** Message containing the number of the Battery for which capabilities are being requested.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Battery_Cap*** Message.	Physical Layer receives the ***Get_Battery_Cap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Battery_Cap*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 327
Microchip v. Aptiv
IPR2024-00496

Step	Sink Port	Source Port
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Battery_Cap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Battery_Cap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present Source Battery capabilities, for the requested Battery number, which are provided. The Policy Engine tells the Protocol Layer to form a ***Battery_Capabilities*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Battery_Capabilities*** Message.	Physical Layer appends a CRC and sends the ***Battery_Capabilities*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Battery_Capabilities*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Battery_Capabilities*** Message was successfully sent.
	The Source has informed the Sink of the Battery capabilities for the requested Battery.	

Microchip Ex. 1020, p. 328
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.10 Source Gets Battery Capabilities

Figure 8-31 shows an example sequence between a Source and a Sink when the Source gets the Sink's Battery capabilities for a given Battery.

Figure 8-31 Source Gets Sink's Battery Capabilities

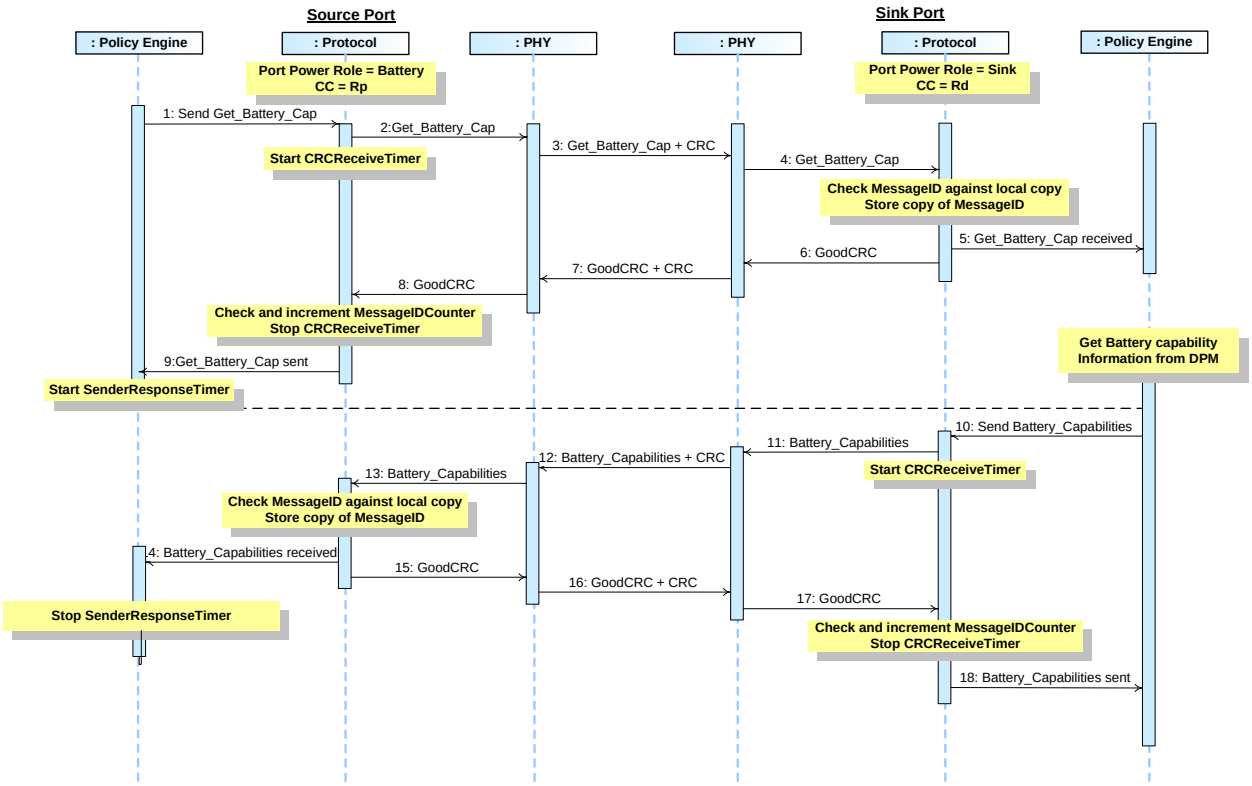


Table 8-31 below provides a detailed explanation of what happens at each labeled step in Figure 8-31 above.

Table 8-31 Steps for a Source getting Sink Battery capabilities Sequence

Step	Source Port	Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Battery_Cap*** Message containing the number of the Battery for which capabilities are being requested.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Battery_Cap*** Message.	Physical Layer receives the ***Get_Battery_Cap*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Battery_Cap*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 329
Microchip v. Aptiv
IPR2024-00496

Step	Source Port	Sink Port
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Battery_Cap*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Battery_Cap*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the present Source Battery capabilities, for the requested Battery number, which are provided. The Policy Engine tells the Protocol Layer to form a ***Battery_Capabilities*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Battery_Capabilities*** Message.	Physical Layer appends a CRC and sends the ***Battery_Capabilities*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Battery_Capabilities*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Battery_Capabilities*** Message was successfully sent.
	The Sink has informed the Source of the Battery capabilities for the requested Battery.	

Microchip Ex. 1020, p. 330
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.11 Source Gets Port Manufacturer Information from a Sink

Figure 8-32 shows an example sequence between a Source and a Sink when the Source gets the Sink's Manufacturer information for the Port.

Figure 8-32 Source Gets Sink's Port Manufacturer Information

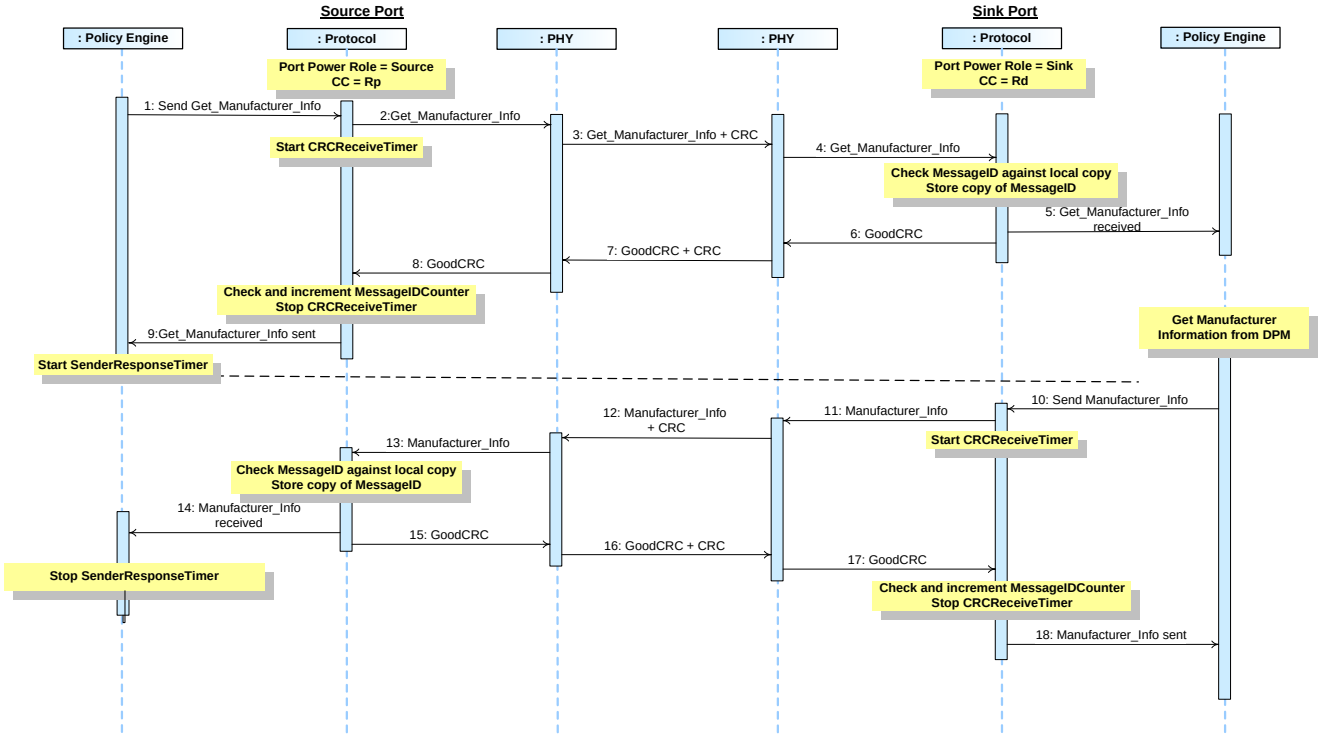


Table 8-31 below provides a detailed explanation of what happens at each labeled step in Figure 8-32 above.

Table 8-32 Steps for a Source getting Sink's Port Manufacturer information Sequence

Step	Source Port	Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Manufacturer_Info*** Message with a request for Port information.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Manufacturer_Info*** Message.	Physical Layer receives the ***Get_Manufacturer_Info*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Manufacturer_Info*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Manufacturer_Info*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 331
Microchip v. Aptiv
IPR2024-00496

Step	Source Port	Sink Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Manufacturer_Info*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the Port's manufacturer information which is provided. The Policy Engine tells the Protocol Layer to form a ***Manufacturer_Info*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Manufacturer_Info*** Message.	Physical Layer appends a CRC and sends the ***Manufacturer_Info*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Manufacturer_Info*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Manufacturer_Info*** Message was successfully sent.
	The Sink has informed the Source of the manufacturer information for the Port.	

Microchip Ex. 1020, p. 332
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.12 Sink Gets Port Manufacturer Information from a Source

Figure 8-33 shows an example sequence between a Source and a Sink when the Source gets the Sink's Manufacturer information for the Port.

Figure 8-33 Sink Gets Source's Port Manufacturer Information

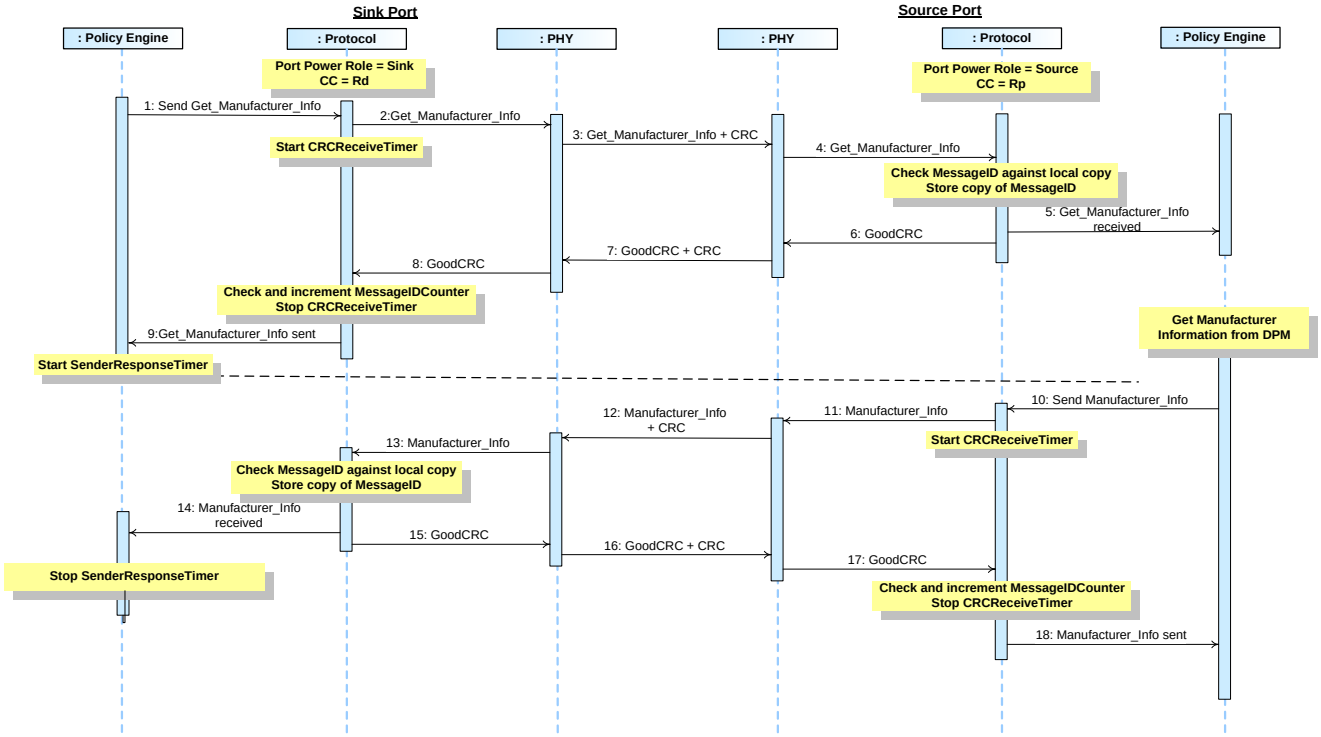


Table 8-33 below provides a detailed explanation of what happens at each labeled step in Figure 8-33 above.

Table 8-33 Steps for a Source getting Sink's Port Manufacturer information Sequence

Step	Sink Port	Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Manufacturer_Info*** Message with a request for Port information.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Manufacturer_Info*** Message.	Physical Layer receives the ***Get_Manufacturer_Info*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Manufacturer_Info*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Manufacturer_Info*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 333
Microchip v. Aptiv
IPR2024-00496

Step	Sink Port	Source Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Manufacturer_Info*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the Port's manufacturer information which is provided. The Policy Engine tells the Protocol Layer to form a ***Manufacturer_Info*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Manufacturer_Info*** Message.	Physical Layer appends a CRC and sends the ***Manufacturer_Info*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Manufacturer_Info*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Manufacturer_Info*** Message was successfully sent.
	The Sink has informed the Source of the manufacturer information for the Port.	

Microchip Ex. 1020, p. 334
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.13 Source Gets Battery Manufacturer Information from a Sink

Figure 8-34 shows an example sequence between a Source and a Sink when the Source gets the Sink's Manufacturer information for one of its Batteries.

Figure 8-34 Source Gets Sink's Battery Manufacturer Information

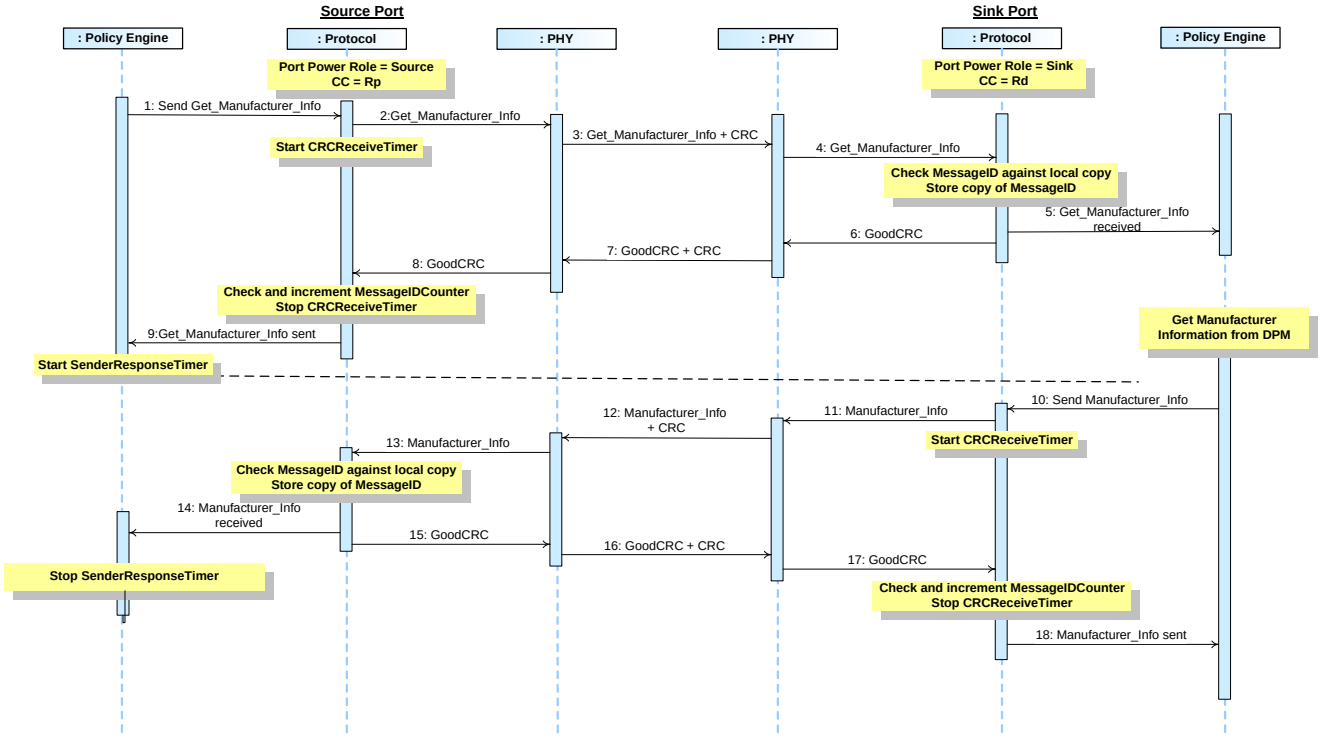


Table 8-34 below provides a detailed explanation of what happens at each labeled step in Figure 8-34 above.

Table 8-34 Steps for a Source getting Sink's Battery Manufacturer information Sequence

Step	Source Port	Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Manufacturer_Info*** Message with a request for Battery information for a given Battery.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Manufacturer_Info*** Message.	Physical Layer receives the ***Get_Manufacturer_Info*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Manufacturer_Info*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Manufacturer_Info*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 335
Microchip v. Aptiv
IPR2024-00496

Step	Source Port	Sink Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Manufacturer_Info*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the Battery's manufacturer information for a given Battery which is provided. The Policy Engine tells the Protocol Layer to form a ***Manufacturer_Info*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Manufacturer_Info*** Message.	Physical Layer appends a CRC and sends the ***Manufacturer_Info*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Manufacturer_Info*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Manufacturer_Info*** Message was successfully sent.
	The Sink has informed the Source of the manufacturer information for the requested Battery.	

Microchip Ex. 1020, p. 336
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.14 Sink Gets Battery Manufacturer Information from a Source

Figure 8-35 shows an example sequence between a Source and a Sink when the Source gets the Sink's Manufacturer information for the Port.

Figure 8-35 Sink Gets Source's Battery Manufacturer Information

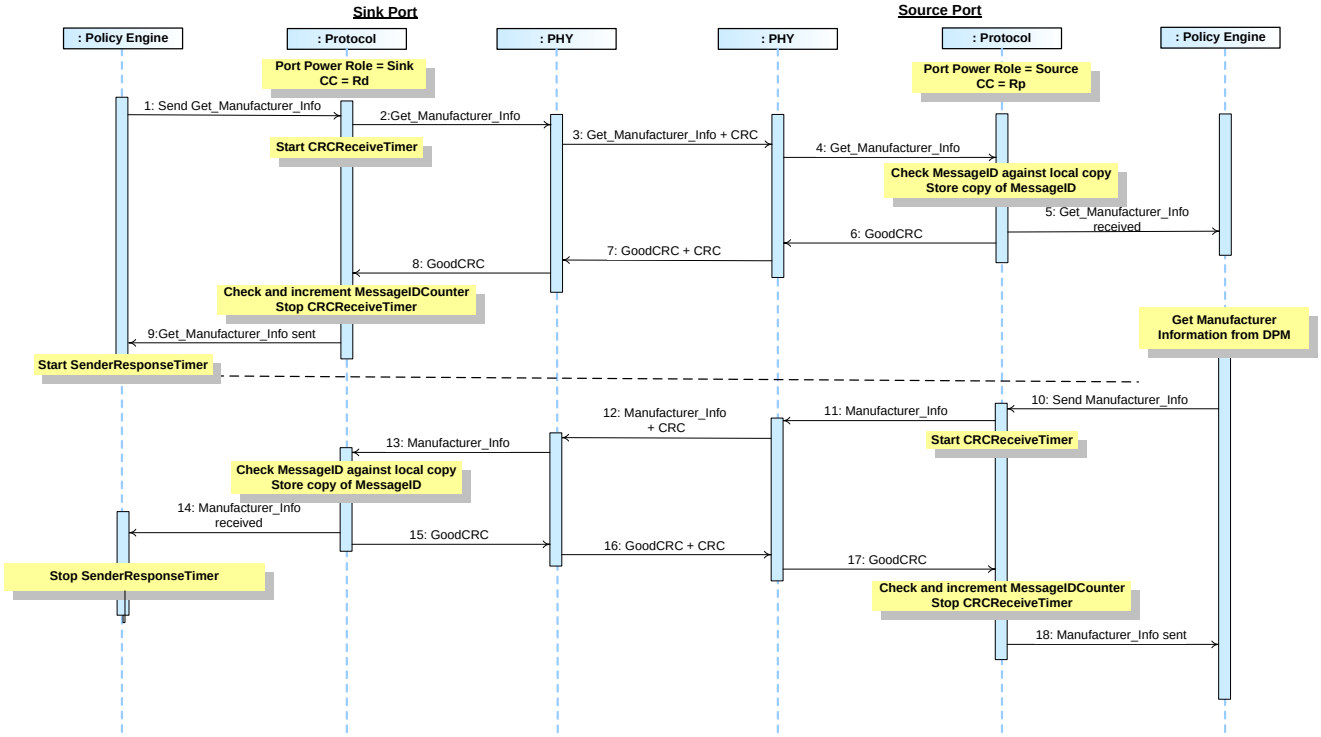


Table 8-35 below provides a detailed explanation of what happens at each labeled step in Figure 8-35 above.

Table 8-35 Steps for a Source getting Sink's Battery Manufacturer information Sequence

Step	Sink Port	Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Get_Manufacturer_Info*** Message with a request for Battery information for a given Battery.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Manufacturer_Info*** Message.	Physical Layer receives the ***Get_Manufacturer_Info*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Manufacturer_Info*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Manufacturer_Info*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 337
Microchip v. Aptiv
IPR2024-00496

Step	Sink Port	Source Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Manufacturer_Info*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the Battery's manufacturer information for a given Battery which is provided. The Policy Engine tells the Protocol Layer to form a ***Manufacturer_Info*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Manufacturer_Info*** Message.	Physical Layer appends a CRC and sends the ***Manufacturer_Info*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Manufacturer_Info*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Manufacturer_Info*** Message was successfully sent.
	The Sink has informed the Source of the manufacturer information for the requested Battery.	

Microchip Ex. 1020, p. 338
Microchip v. Aptiv
IPR2024-00496

8.3.2.10.15 VCONN Source Gets Manufacturer Information from a Cable Plug

Figure 8-36 shows an example sequence between a VCONN Source (Source or Sink) and a Cable Plug when the VCONN Source gets the Cable Plug's Manufacturer information.

Figure 8-36 VCONN Source Gets Cable Plug's Manufacturer Information

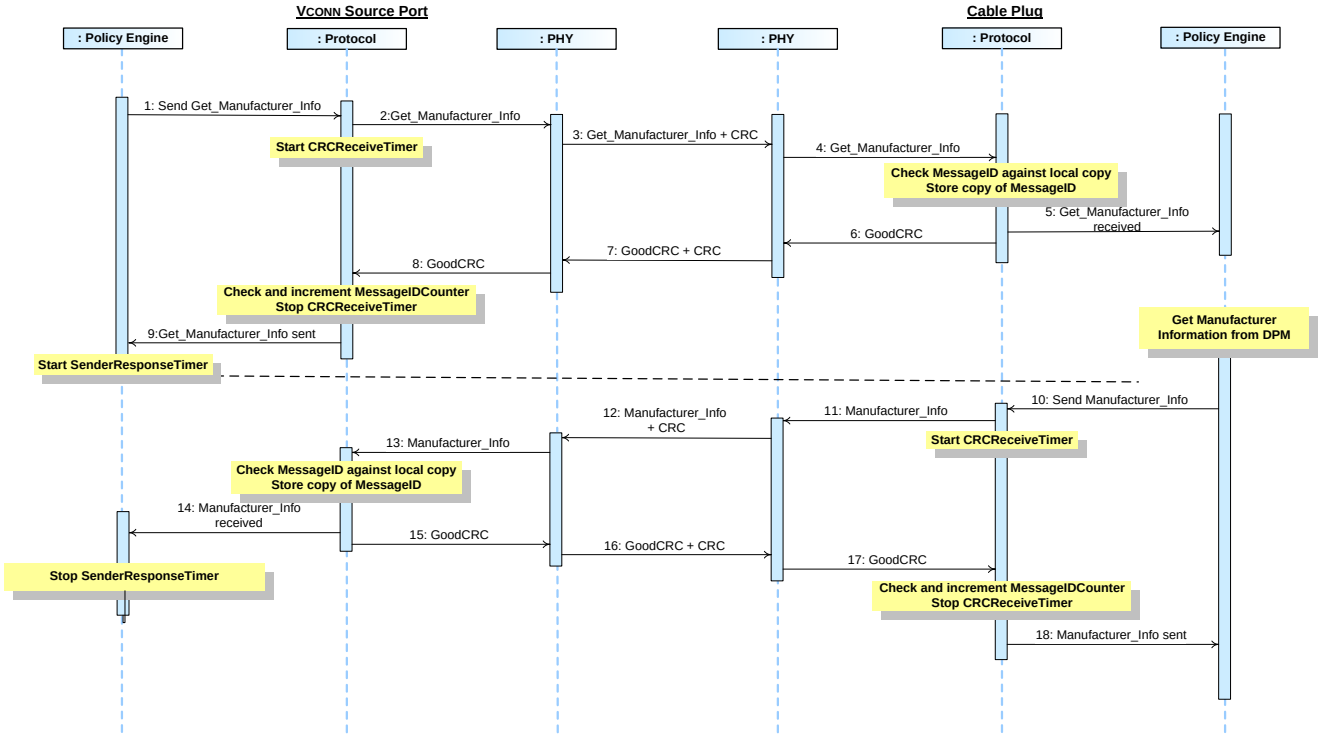


Table 8-36 below provides a detailed explanation of what happens at each labeled step in Figure 8-36 above.

Table 8-36 Steps for a Source getting Sink's Port Manufacturer information Sequence

Step	VCONN Source	Cable Plug
1	The Port is currently acting as the VCONN Source. Policy Engine directs the Protocol Layer to send a ***Get_Manufacturer_Info*** Message with a request for Port information.	
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Get_Manufacturer_Info*** Message.	Physical Layer receives the ***Get_Manufacturer_Info*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Get_Manufacturer_Info*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Get_Manufacturer_Info*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 339
Microchip v. Aptiv
IPR2024-00496

Step	VCONN Source	Cable Plug
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Get_Manufacturer_Info*** Message was successfully sent. Policy Engine starts ***SenderResponseTimer***.	
10		Policy Engine requests the DPM for the Cable Plug's manufacturer information which is provided. The Policy Engine tells the Protocol Layer to form a ***Manufacturer_Info*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Manufacturer_Info*** Message.	Physical Layer appends a CRC and sends the ***Manufacturer_Info*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Manufacturer_Info*** Message information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***SenderResponseTimer***.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Manufacturer_Info*** Message was successfully sent.
	The Cable Plug has informed the Source of its manufacturer information.	

Microchip Ex. 1020, p. 340
Microchip v. Aptiv
IPR2024-00496

8.3.2.11 Security

8.3.2.11.1 Source requests security exchange with Sink

Figure 8-37 shows an example sequence for a security exchange between a Source and a Sink.

Figure 8-37 Source requests security exchange with Sink

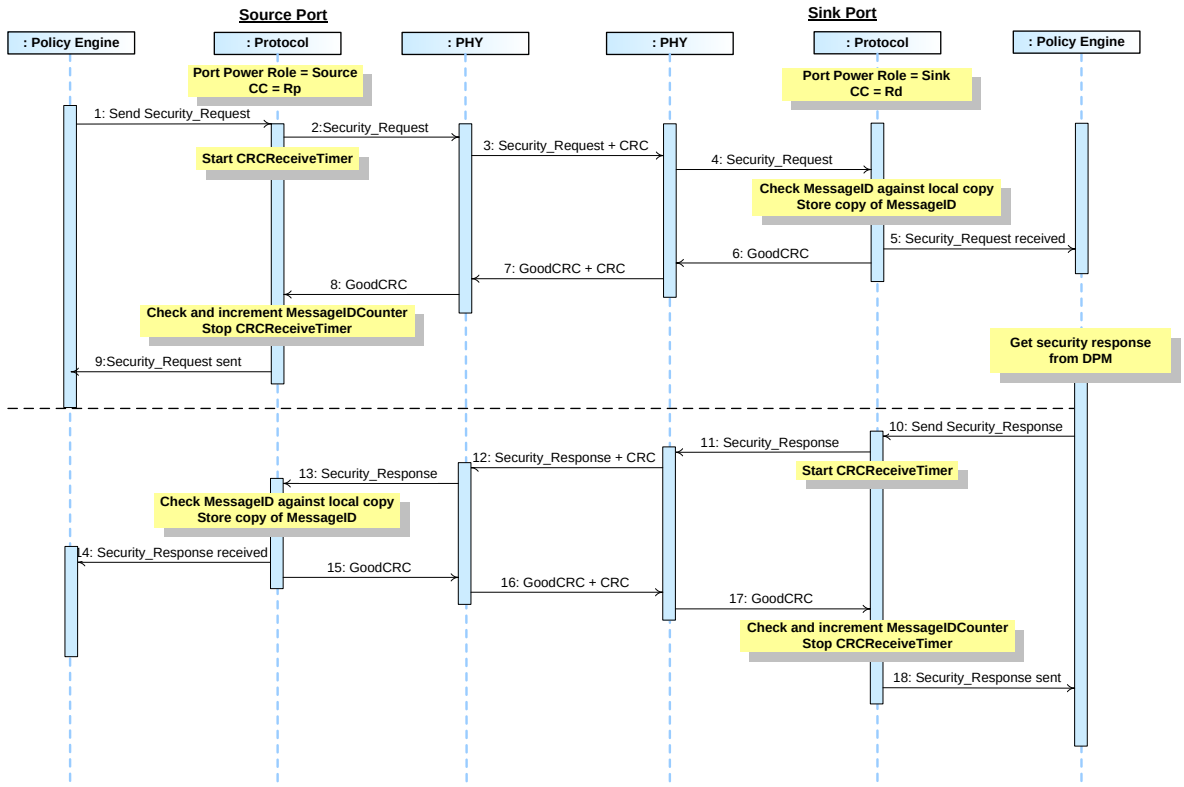


Table 8-37 below provides a detailed explanation of what happens at each labeled step in Figure 8-37 above.

Table 8-37 Steps for a Source requesting a security exchange with a Sink Sequence

Step	Source Port	Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Security_Request*** Message using a payload supplied by the DPM.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Security_Request*** Message.	Physical Layer receives the ***Security_Request*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Security_Request*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 341
Microchip v. Aptiv
IPR2024-00496

Step	Source Port	Sink Port
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Security_Request*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Security_Request*** Message was successfully sent.	
10		Policy Engine requests the DPM for the response to the security request which is provided. The Policy Engine tells the Protocol Layer to form a ***Security_Response*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Security_Response*** Message.	Physical Layer appends a CRC and sends the ***Security_Response*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Security_Response*** Message information to the Policy Engine that consumes it.	
14	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
15	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
16		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
17		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Security_Response*** Message was successfully sent.
	The security exchange is complete.	

Microchip Ex. 1020, p. 342
Microchip v. Aptiv
IPR2024-00496

8.3.2.11.2 Sink requests security exchange with Source

Figure 8-38 shows an example sequence for a security exchange between a Sink and a Source.

Figure 8-38 Sink requests security exchange with Source

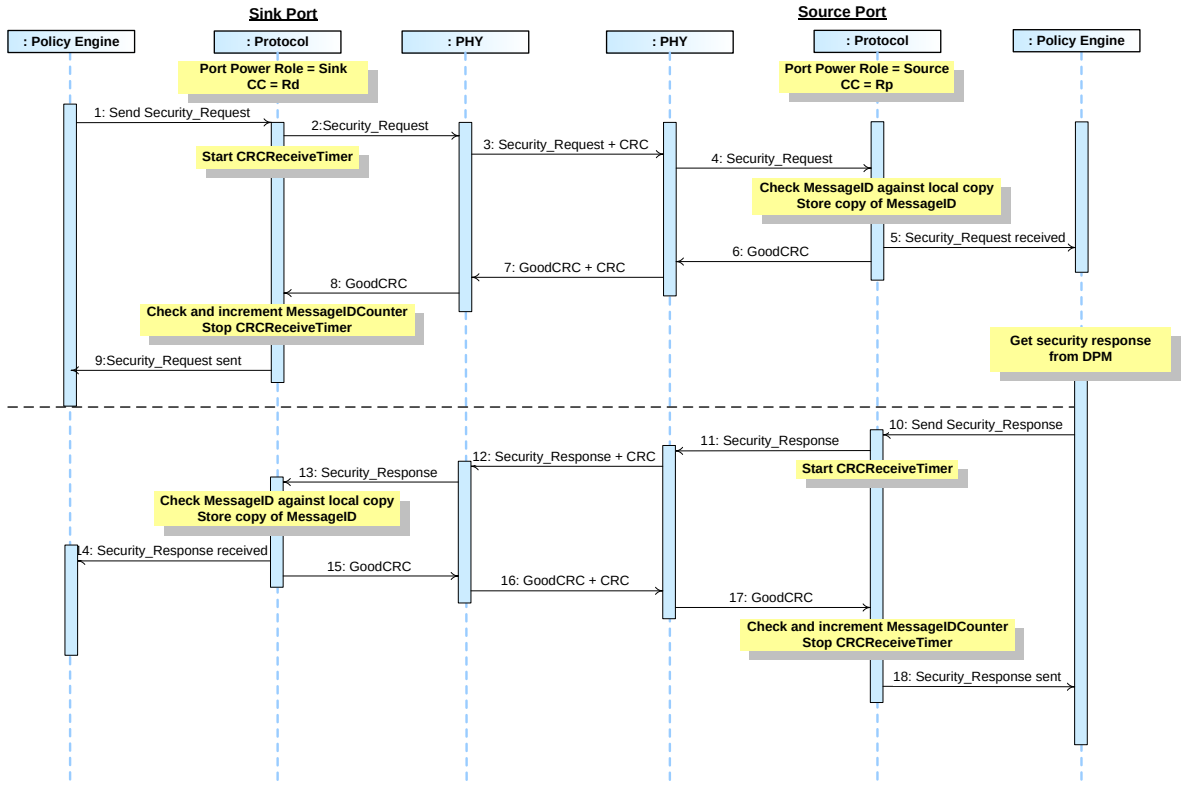


Table 8-38 below provides a detailed explanation of what happens at each labeled step in Figure 8-38 above.

Table 8-38 Steps for a Sink requesting a security exchange with a Source Sequence

Step	Sink Port	Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Security_Request*** Message using a payload supplied by the DPM.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Security_Request*** Message.	Physical Layer receives the ***Security_Request*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Security_Request*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Security_Request*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 343
Microchip v. Aptiv
IPR2024-00496

Step	Sink Port	Source Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Security_Request*** Message was successfully sent.	
10		Policy Engine requests the DPM for the response to the security request which is provided. The Policy Engine tells the Protocol Layer to form a ***Security_Response*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Security_Response*** Message.	Physical Layer appends a CRC and sends the ***Security_Response*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Security_Response*** Message information to the Policy Engine that consumes it.	
14	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
15	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
16		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
17		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Security_Response*** Message was successfully sent.
	The security exchange is complete.	

Microchip Ex. 1020, p. 344
Microchip v. Aptiv
IPR2024-00496

8.3.2.11.3 Vconn Source requests security exchange with Cable Plug

Figure 8-39 shows an example sequence for a security exchange between a Vconn Source and a Cable Plug.

Figure 8-39 Vconn Source requests security exchange with Cable Plug

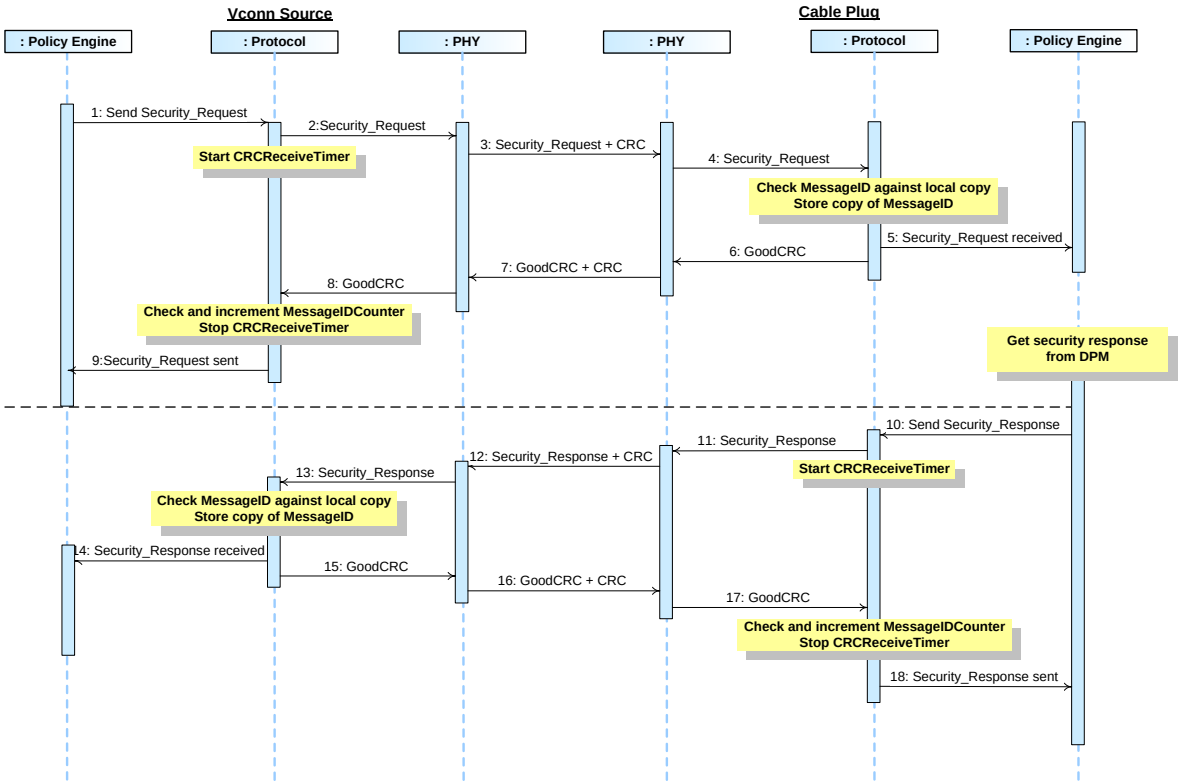


Table 8-39 below provides a detailed explanation of what happens at each labeled step in Figure 8-39 above.

Table 8-39 Steps for a Vconn Source requesting a security exchange with a Cable Plug Sequence

Step	Vconn Source	Cable Plug
1	Policy Engine directs the Protocol Layer to send a ***Security_Request*** Message using a payload supplied by the DPM.	
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Security_Request*** Message.	Physical Layer receives the ***Security_Request*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Security_Request*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Security_Request*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.

Microchip Ex. 1020, p. 345
Microchip v. Aptiv
IPR2024-00496

Step	Vconn Source	Cable Plug
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Security_Request*** Message was successfully sent.	
10		Policy Engine requests the DPM for the response to the security request which is provided. The Policy Engine tells the Protocol Layer to form a ***Security_Response*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Security_Response*** Message.	Physical Layer appends a CRC and sends the ***Security_Response*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Security_Response*** Message information to the Policy Engine that consumes it.	
14	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
15	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
16		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
17		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Security_Response*** Message was successfully sent.
	The security exchange is complete.	

Microchip Ex. 1020, p. 346
Microchip v. Aptiv
IPR2024-00496

8.3.2.12 Firmware Update

8.3.2.12.1 Source requests firmware update exchange with Sink

Figure 8-40 shows an example sequence for a firmware update exchange between a Source and a Sink.

Figure 8-40 Source requests firmware update exchange with Sink

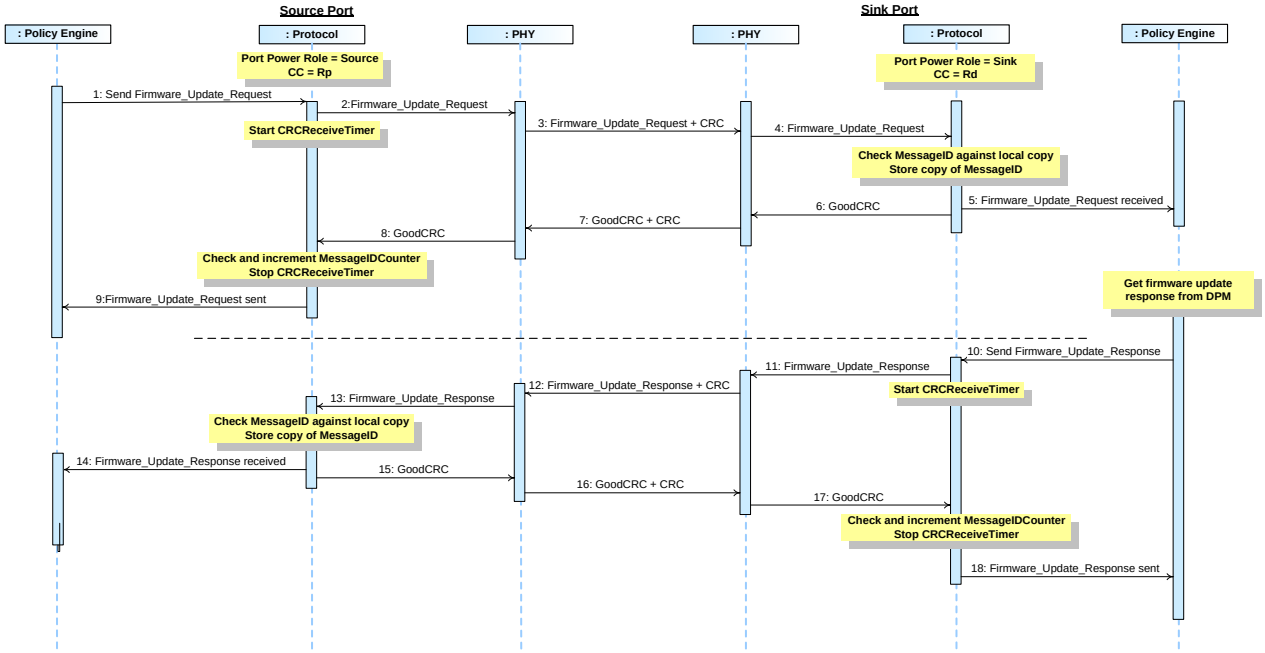


Table 8-40 below provides a detailed explanation of what happens at each labeled step in Figure 8-40 above.

Table 8-40 Steps for a Source requesting a firmware update exchange with a Sink Sequence

Step	Source Port	Sink Port
1	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire. Policy Engine directs the Protocol Layer to send a ***Firmware_Update_Request*** Message using a payload supplied by the DPM.	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Firmware_Update_Request*** Message.	Physical Layer receives the ***Firmware_Update_Request*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Firmware_Update_Request*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Firmware_Update_Request*** Message information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 347
Microchip v. Aptiv
IPR2024-00496

Step	Source Port	Sink Port
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Firmware_Update_Request*** Message was successfully sent.	
10		Policy Engine requests the DPM for the response to the firmware update request which is provided. The Policy Engine tells the Protocol Layer to form a ***Firmware_Update_Response*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Firmware_Update_Response*** Message.	Physical Layer appends a CRC and sends the ***Firmware_Update_Response*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Firmware_Update_Response*** Message information to the Policy Engine that consumes it.	
14	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
15	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
16		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
17		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Firmware_Update_Response*** Message was successfully sent.
	The firmware update exchange is complete.	

Microchip Ex. 1020, p. 348
Microchip v. Aptiv
IPR2024-00496

8.3.2.12.2 Sink requests firmware update exchange with Source

Figure 8-41 shows an example sequence for a firmware update exchange between a Sink and a Source.

Figure 8-41 Sink requests firmware update exchange with Source

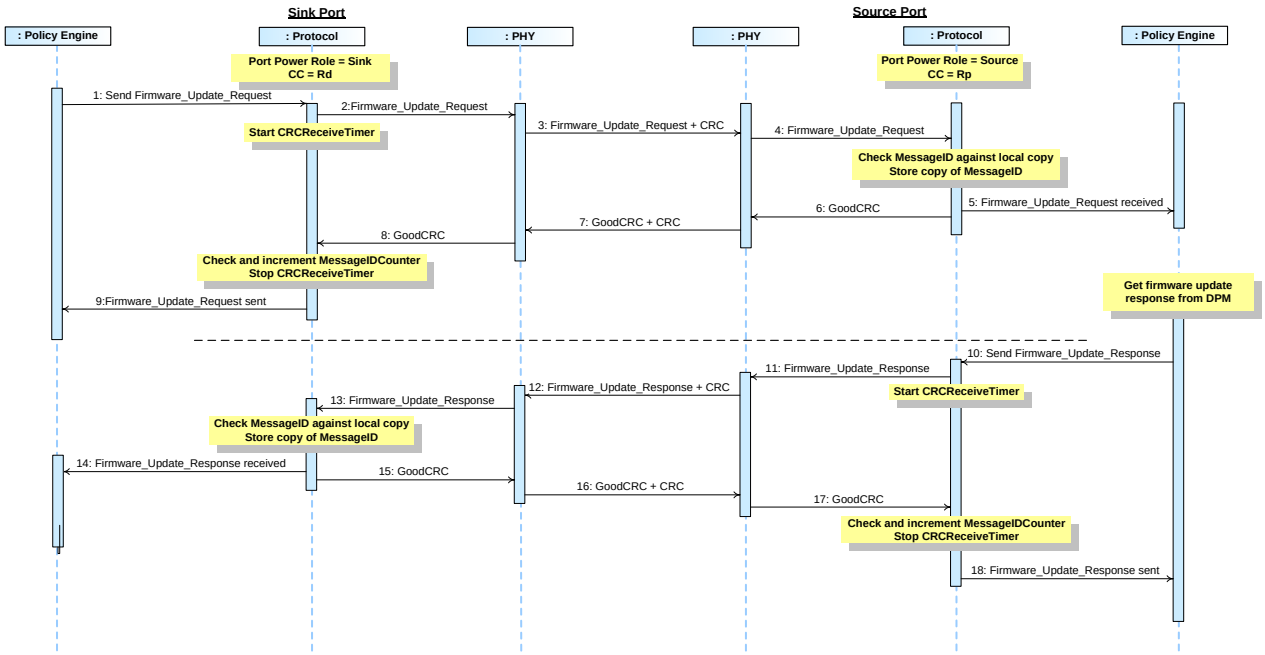


Table 8-41 below provides a detailed explanation of what happens at each labeled step in Figure 8-41 above.

Table 8-41 Steps for a Sink requesting a firmware update exchange with a Source Sequence

Step	Sink Port	Source Port
1	The Port has ***Port Power Role*** set to Sink with the Rd pull down on its CC wire. Policy Engine directs the Protocol Layer to send a ***Firmware_Update_Request*** Message using a payload supplied by the DPM.	The Port has ***Port Power Role*** set to Source and the Rp pull up on its CC wire.
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Firmware_Update_Request*** Message.	Physical Layer receives the ***Firmware_Update_Request*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Firmware_Update_Request*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Firmware_Update_Request*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.

Microchip Ex. 1020, p. 349
Microchip v. Aptiv
IPR2024-00496

Step	Sink Port	Source Port
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Firmware_Update_Request*** Message was successfully sent.	
10		Policy Engine requests the DPM for the response to the firmware update request which is provided. The Policy Engine tells the Protocol Layer to form a ***Firmware_Update_Response*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Firmware_Update_Response*** Message.	Physical Layer appends a CRC and sends the ***Firmware_Update_Response*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Firmware_Update_Response*** Message information to the Policy Engine that consumes it.	
14	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
15	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
16		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
17		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Firmware_Update_Response*** Message was successfully sent.
	The firmware update exchange is complete.	

Microchip Ex. 1020, p. 350
Microchip v. Aptiv
IPR2024-00496

8.3.2.12.3 Vconn Source requests firmware update exchange with Cable Plug

Figure 8-42 shows an example sequence for a firmware update exchange between a Vconn Source and a Cable Plug.

Figure 8-42 Vconn Source requests firmware update exchange with Cable Plug

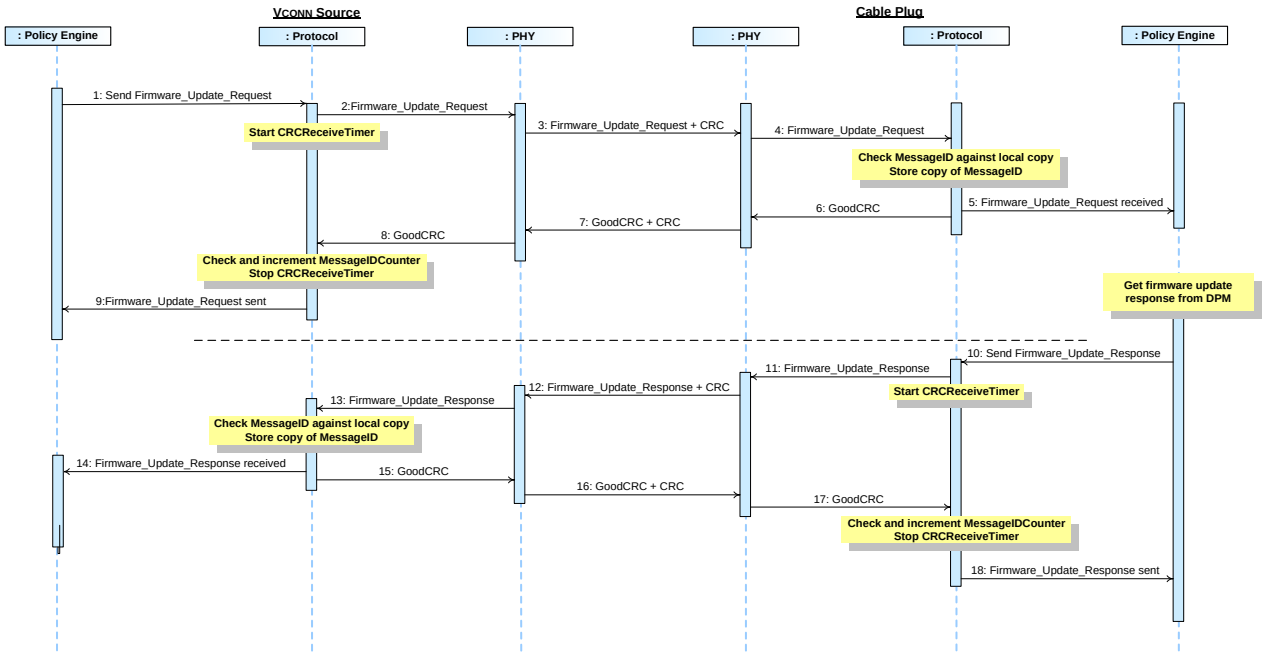


Table 8-42 below provides a detailed explanation of what happens at each labeled step in Figure 8-42 above.

Table 8-42 Steps for a Vconn Source requesting a firmware update exchange with a Cable Plug Sequence

Step	Vconn Source	Cable Plug
1	Policy Engine directs the Protocol Layer to send a ***Firmware_Update_Request*** Message using a payload supplied by the DPM.	
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Firmware_Update_Request*** Message.	Physical Layer receives the ***Firmware_Update_Request*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Firmware_Update_Request*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Firmware_Update_Request*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.

Microchip Ex. 1020, p. 351
Microchip v. Aptiv
IPR2024-00496

Step	Vconn Source	Cable Plug
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Firmware_Update_Request*** Message was successfully sent.	
10		Policy Engine requests the DPM for the response to the firmware update request which is provided. The Policy Engine tells the Protocol Layer to form a ***Firmware_Update_Response*** Message.
11		Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the Message and compares the CRC it calculated with the one sent to verify the ***Firmware_Update_Response*** Message.	Physical Layer appends a CRC and sends the ***Firmware_Update_Response*** Message.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Firmware_Update_Response*** Message information to the Policy Engine that consumes it.	
14	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
15	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
16		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
17		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Firmware_Update_Response*** Message was successfully sent.
	The firmware update exchange is complete.	

8.3.2.13 Structured VDM

8.3.2.13.1 DFP to UFP Discover Identity

Figure 8-43 shows an example sequence between a DFP and UFP, where both Port Partners are in an Explicit Contract and the DFP attempts to discover identity information from the UFP.

Microchip Ex. 1020, p. 352
Microchip v. Aptiv
IPR2024-00496

Figure 8-43 DFP to UFP Discover Identity

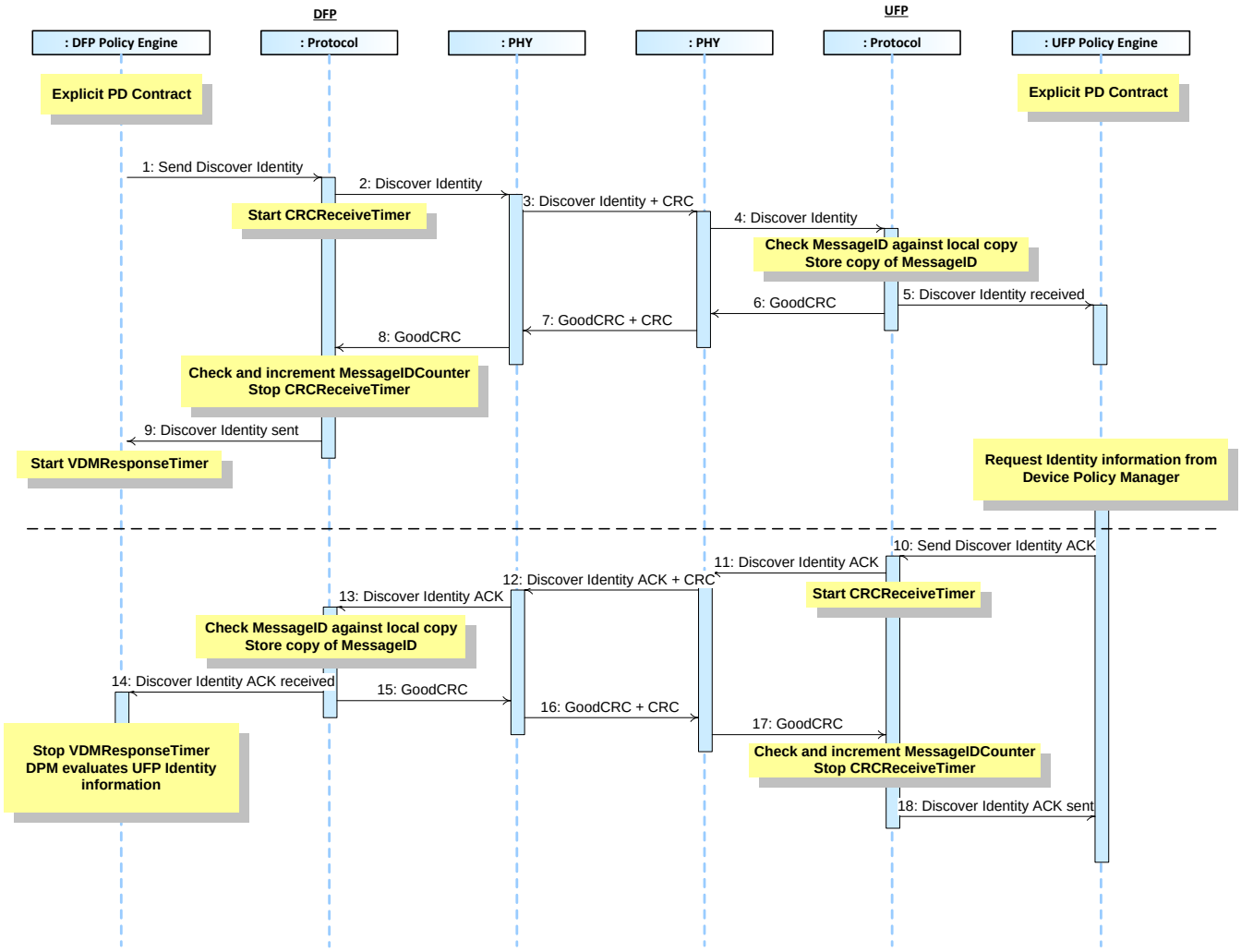


Table 8-43 below provides a detailed explanation of what happens at each labeled step in Figure 8-43 above.

Table 8-43 Steps for DFP to UFP Discover Identity

Step	DFP	UFP
1	The DFP has an Explicit Contract. The Policy Engine directs the Protocol Layer to send a ***Discover Identity*** Command request.	The UFP has an Explicit Contract.
2	Protocol Layer creates the ***Discover Identity*** Command request and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Discover Identity*** Command request.	Physical Layer receives the ***Discover Identity*** Command request and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Discover Identity*** Command request to the Protocol Layer.

Microchip Ex. 1020, p. 353
Microchip v. Aptiv
IPR2024-00496

Step	DFP	UFP
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Discover Identity*** Command request information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Discover Identity*** Command request was successfully sent. Policy Engine starts the ***VDMResponseTimer***.	
10		Policy Engine requests the identity information from the Device Policy Manager. The Policy Engine tells the Protocol Layer to form a ***Discover Identity*** Command ACK response.
11		Protocol Layer creates the ***Discover Identity*** Command ACK response and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Discover Identity*** Command ACK response and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Discover Identity*** Command ACK response.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Discover Identity*** Command ACK response information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***VDMResponseTimer*** and passed the Identity information to the Device Policy Manager for evaluation.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Discover Identity*** Command ACK response was successfully sent.

8.3.2.13.2 Source Port to Cable Plug Discover Identity

Figure 8-44 shows an example sequence between Source and a Cable Plug, where the Source attempts to discover identity information from the Cable Plug prior to establishing an Explicit Contract with its Port Partner.

Microchip Ex. 1020, p. 354
Microchip v. Aptiv
IPR2024-00496

Figure 8-44 Source Port to Cable Plug Discover Identity

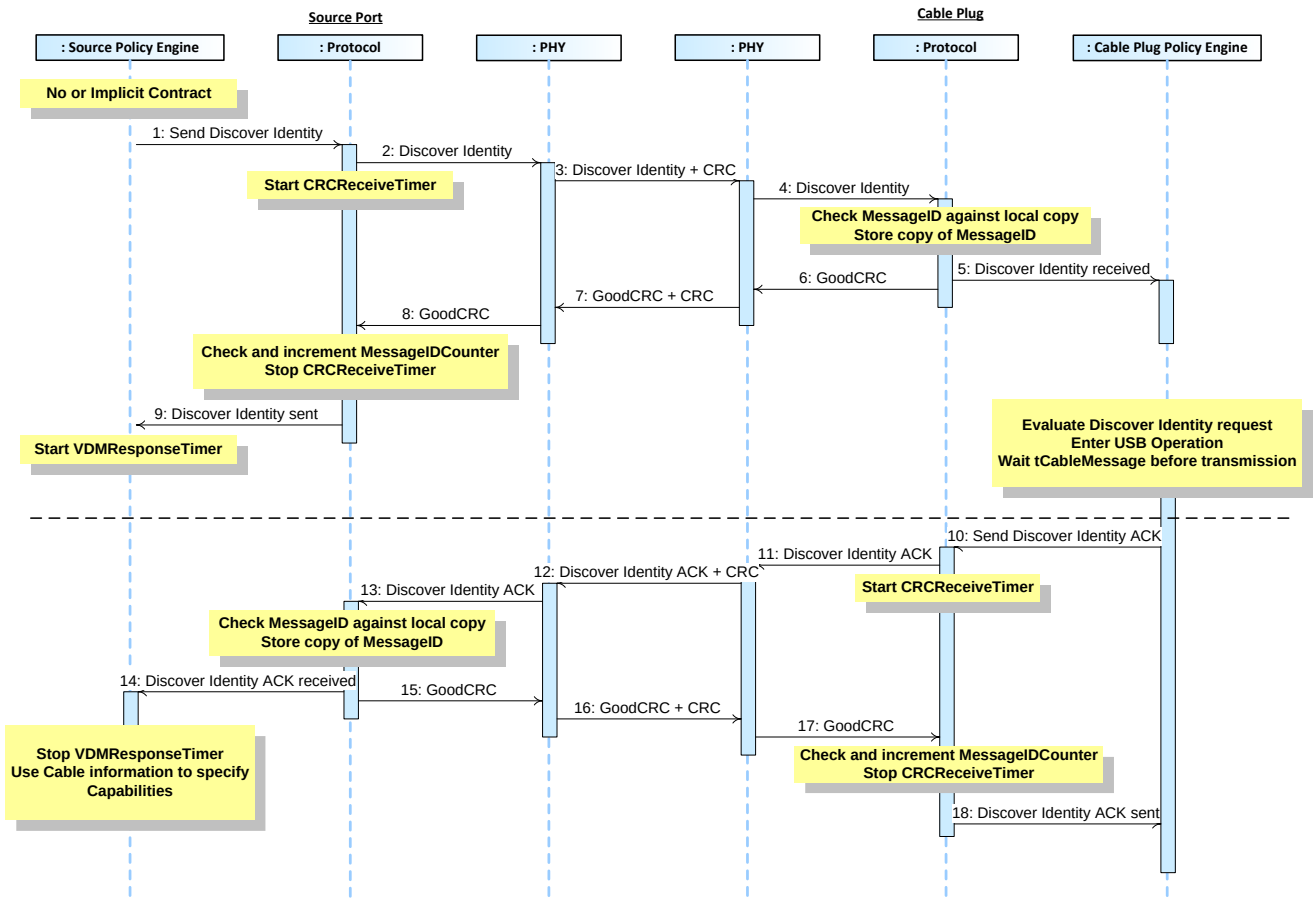


Table 8-44 below provides a detailed explanation of what happens at each labeled step in Figure 8-44 above.

Table 8-44 Steps for Source Port to Cable Plug Discover Identity

Step	Source Port	Cable Plug
1	The Source has no Contract or an Implicit Contract with its Port Partner. The Policy Engine directs the Protocol Layer to send a ***Discover Identity*** Command request.	
2	Protocol Layer creates the ***Discover Identity*** Command request and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Discover Identity*** Command request.	Physical Layer receives the ***Discover Identity*** Command request and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Discover Identity*** Command request to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Discover Identity*** Command request information to the Policy Engine that consumes it.

Microchip Ex. 1020, p. 355
Microchip v. Aptiv
IPR2024-00496

Step	Source Port	Cable Plug
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Discover Identity*** Command request was successfully sent. Policy Engine starts the ***VDMResponseTimer***.	
10		Policy Engine requests the identity information from the Device Policy Manager. . ***tCableMessage*** after the ***GoodCRC*** Message was sent the Policy Engine tells the Protocol Layer to form a ***Discover Identity*** Command ACK response.
11		Protocol Layer creates the ***Discover Identity*** Command ACK response and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Discover Identity*** Command ACK response and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Discover Identity*** Command ACK response.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Discover Identity*** Command ACK response information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***VDMResponseTimer*** and passes the identity information to the Device Policy Manager for evaluation.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18	The Source uses the Cable Plug information as input to its offered capabilities.	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent.

8.3.2.13.3 DFP to Cable Plug Discover Identity

Figure 8-45 shows an example sequence between a DFP and a Cable Plug, where the DFP attempts to discover identity information from the Cable Plug.

Microchip Ex. 1020, p. 356
Microchip v. Aptiv
IPR2024-00496

Figure 8-45 DFP to Cable Plug Discover Identity

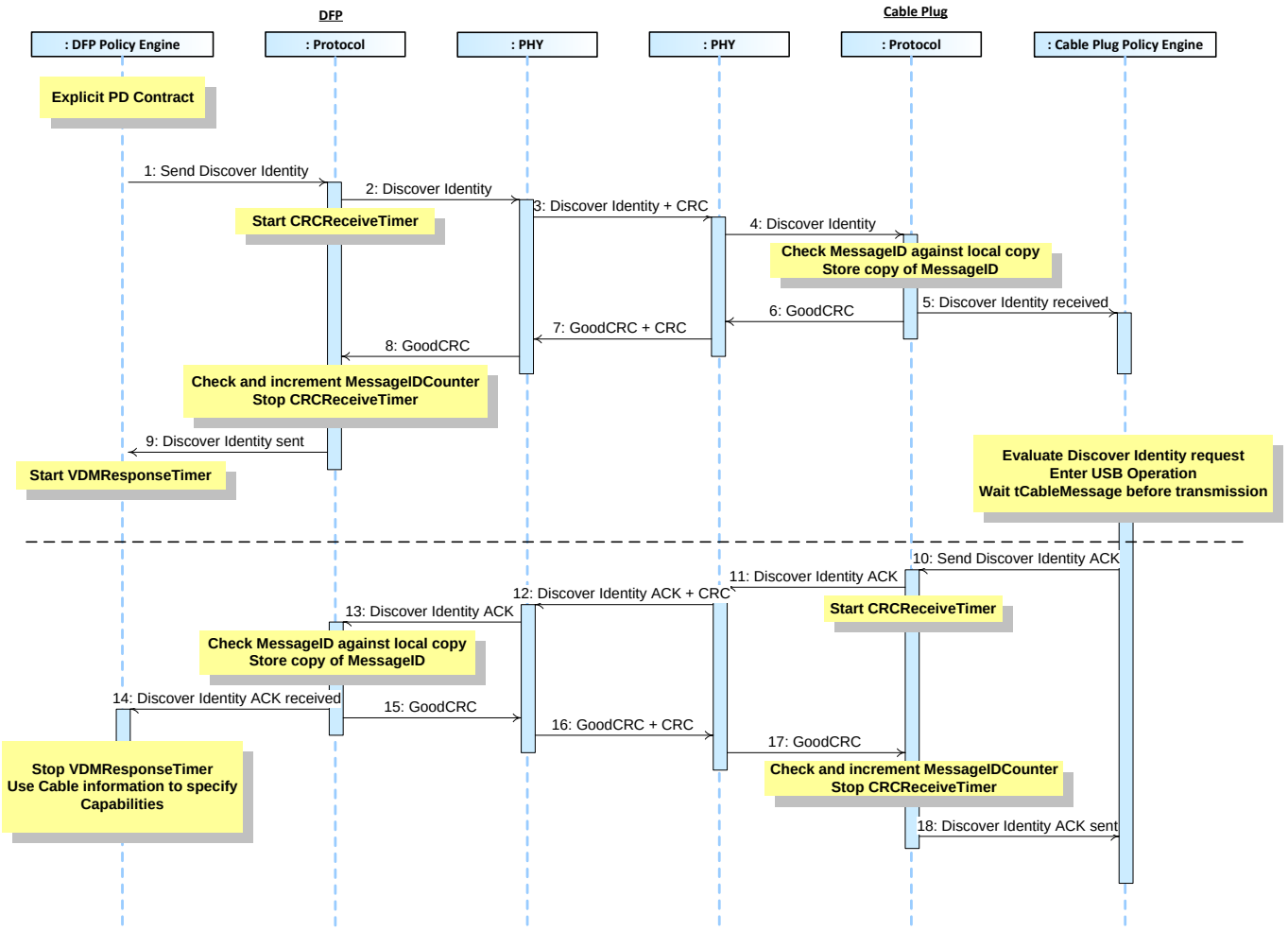


Table 8-45 below provides a detailed explanation of what happens at each labeled step in Figure 8-45 above.

Table 8-45 Steps for DFP to Cable Plug Discover Identity

Step	DFP	Cable Plug
1	The DFP has an Explicit Contract with its Port Partner. The Policy Engine directs the Protocol Layer to send a ***Discover Identity*** Command request.	
2	Protocol Layer creates the ***Discover Identity*** Command request and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Discover Identity*** Command request.	Physical Layer receives the ***Discover Identity*** Command request and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Discover Identity*** Command request to the Protocol Layer.

Microchip Ex. 1020, p. 357
Microchip v. Aptiv
IPR2024-00496

Step	DFP	Cable Plug
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Discover Identity*** Command request information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Discover Identity*** Command request was successfully sent. Policy Engine starts the ***VDMResponseTimer***.	
10		Policy Engine requests the identity information from the Device Policy Manager. ***tCableMessage*** after the ***GoodCRC*** Message was sent the Policy Engine tells the Protocol Layer to form a ***Discover Identity*** Command ACK response.
11		Protocol Layer creates the ***Discover Identity*** Command ACK response and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Discover Identity*** Command ACK response and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Discover Identity*** Command ACK response.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Discover Identity*** Command ACK response information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***Discover Identity*** Command ACK response and passes the identity information to the Device Policy Manager for evaluation.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18	The DFP when acting as a Source uses the Cable Plug information as input to its offered capabilities.	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Accept*** Message was successfully sent.

Microchip Ex. 1020, p. 358
Microchip v. Aptiv
IPR2024-00496

8.3.2.13.4 DFP to UFP Enter Mode

Figure 8-46 shows an example sequence between a DFP and a UFP that occurs after the DFP has discovered supported SVIDs and Modes at which point it selects and enters a Mode.

Figure 8-46 DFP to UFP Enter Mode

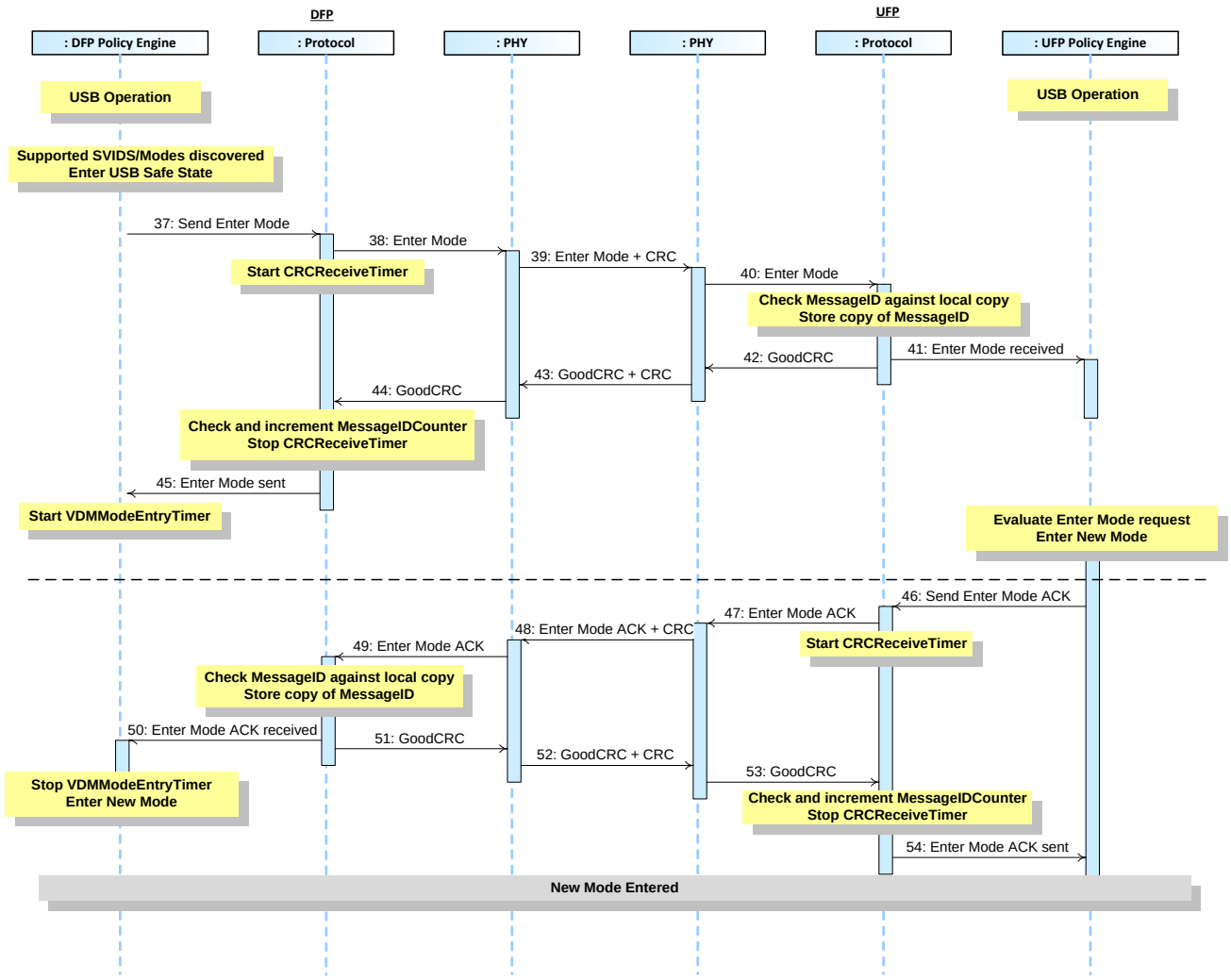


Table 8-46 below provides a detailed explanation of what happens at each labeled step in Figure 8-46 above.

Microchip Ex. 1020, p. 359
Microchip v. Aptiv
IPR2024-00496

Table 8-46 Steps for DFP to UFP Enter Mode

Step	DFP	UFP
1	The DFP has an Explicit Contract The DFP has discovered the supported SVIDS using the ***Discover SVIDs*** Command request and the supported Modes using the ***Discover Modes*** Command request The DFP goes to USB Safe State. The Device Policy Manager requests the Policy Engine to enter a Mode. The Policy Engine directs the Protocol Layer to send an ***Enter Mode*** Command request.	The UFP has an Explicit Contract.
2	Protocol Layer creates the ***Enter Mode*** Command request and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Enter Mode*** Command request.	Physical Layer receives the ***Enter Mode*** Command request and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Enter Mode*** Command request to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Enter Mode*** Command request information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Enter Mode*** Command request was successfully sent. Policy Engine starts the ***VDMModeEntryTimer***.	
10		Policy Engine requests the Device Policy Manager to enter the new Mode. The Policy Engine tells the Protocol Layer to form an ***Enter Mode*** Command ACK response.
11		Protocol Layer creates the ***Enter Mode*** Command ACK response and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Enter Mode*** Command ACK response and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Enter Mode*** Command ACK response.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Enter Mode*** Command ACK response information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***VDMModeEntryTimer*** and requests the Device Policy Manager to enter the new Mode.	

Microchip Ex. 1020, p. 360
Microchip v. Aptiv
IPR2024-00496

Step	DFP	UFP
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Enter Mode*** Command ACK response was successfully sent.
	DFP and UFP are operating in the new Mode	

8.3.2.13.5 DFP to UFP Exit Mode

Figure 8-47 shows an example sequence between a DFP and a UFP, where the DFP commands the UFP to exit the only Active Mode.

Figure 8-47 DFP to UFP Exit Mode

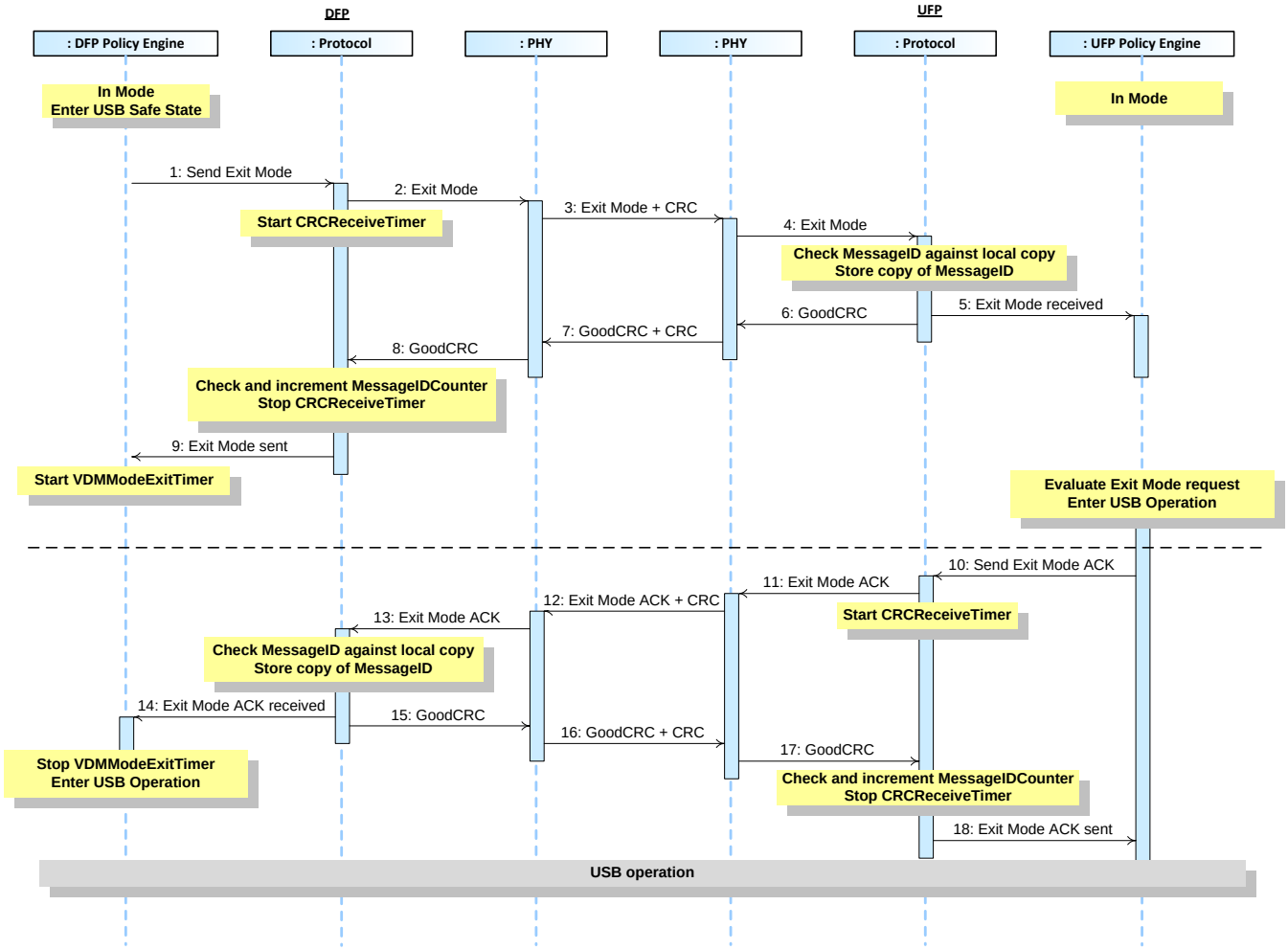


Table 8-47 below provides a detailed explanation of what happens at each labeled step in Figure 8-47 above.

Microchip Ex. 1020, p. 361
Microchip v. Aptiv
IPR2024-00496

Table 8-47 Steps for DFP to UFP Exit Mode

Step	DFP	UFP
1	The DFP is in a Mode and then enters USB Safe State. The Policy Engine directs the Protocol Layer to send an ***Exit Mode*** Command request.	The UFP is in a Mode.
2	Protocol Layer creates the ***Exit Mode*** Command request and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Exit Mode*** Command request.	Physical Layer receives the ***Exit Mode*** Command request and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Exit Mode*** Command request to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Exit Mode*** Command request information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Exit Mode*** Command request was successfully sent. Policy Engine starts the ***VDMModeExitTimer***.	
10		Policy Engine requests the Device Policy Manager to enter USB operation. The Policy Engine tells the Protocol Layer to form an ***Exit Mode*** Command ACK response.
11		Protocol Layer creates the ***Exit Mode*** Command ACK response and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Exit Mode*** Command ACK response and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Exit Mode*** Command ACK response.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Exit Mode*** Command ACK response information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***VDMModeExitTimer*** and requests the Device Policy Manager to enter USB Operation.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 362
Microchip v. Aptiv
IPR2024-00496

Step	DFP	UFP
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Exit Mode*** Command ACK response was successfully sent.
	Both DFP and UFP are in USB Operation	

Microchip Ex. 1020, p. 363
Microchip v. Aptiv
IPR2024-00496

8.3.2.13.6 DFP to Cable Plug Enter Mode

Figure 8-48 shows an example sequence between a DFP and a Cable Plug that occurs after the DFP has discovered supported SVIDs and Modes at which point it selects and enters a Mode.

Figure 8-48 DFP to Cable Plug Enter Mode

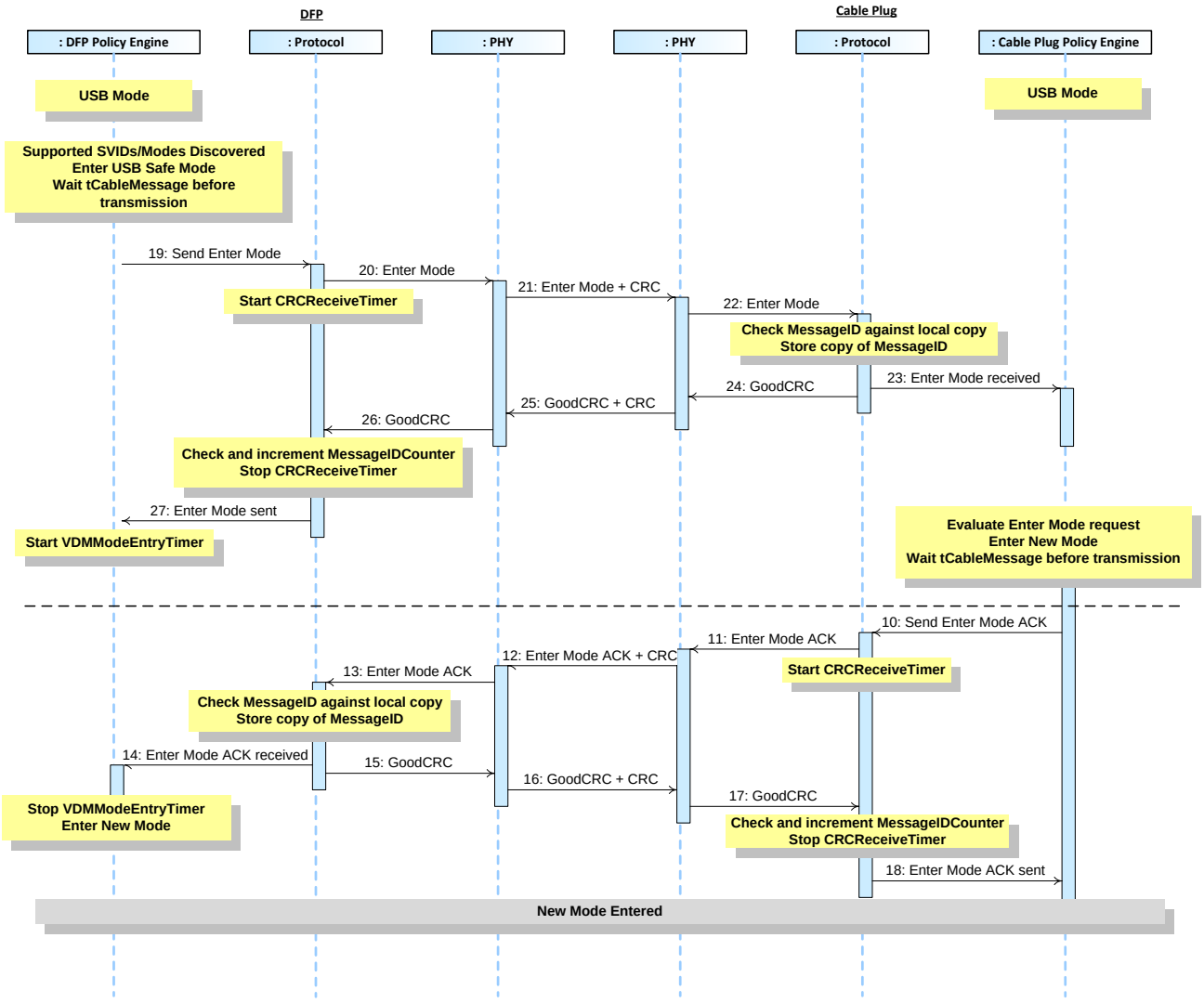


Table 8-48 below provides a detailed explanation of what happens at each labeled step in Figure 8-48 above.

Microchip Ex. 1020, p. 364
Microchip v. Aptiv
IPR2024-00496

Table 8-48 Steps for DFP to Cable Plug Enter Mode

Step	DFP	Cable Plug
1	The DFP has an Explicit Contract The DFP has discovered the supported SVIDS using the ***Discover SVIDs*** Command request and the supported Modes using the ***Discover Modes*** Command request The DFP goes to USB Safe State. The Device Policy Manager requests the Policy Engine to enter a Mode. ***tCableMessage*** after the last ***GoodCRC*** Message was sent the Policy Engine directs the Protocol Layer to send an ***Enter Mode*** Command request.	
2	Protocol Layer creates the ***Enter Mode*** Command request and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Enter Mode*** Command request.	Physical Layer receives the ***Enter Mode*** Command request and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Enter Mode*** Command request to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Enter Mode*** Command request information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Enter Mode*** Command request was successfully sent. Policy Engine starts the ***VDMModeEntryTimer***.	
10		Policy Engine requests the Device Policy Manager to enter the new Mode. ***tCableMessage*** after the ***GoodCRC*** Message was sent the Policy Engine tells the Protocol Layer to form an ***Enter Mode*** Command ACK response.
11		Protocol Layer creates the ***Enter Mode*** Command ACK response and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Enter Mode*** Command ACK response and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Enter Mode*** Command ACK response.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Enter Mode*** Command ACK response information to the Policy Engine that consumes it.	

Microchip Ex. 1020, p. 365
Microchip v. Aptiv
IPR2024-00496

Step	DFP	Cable Plug
14	The Policy Engine stops the ***VDMModeEntryTimer*** and requests the Device Policy Manager to enter the new Mode.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Enter Mode*** Command ACK response was successfully sent.
	DFP and Cable Plug are operating in the new Mode	

Microchip Ex. 1020, p. 366
Microchip v. Aptiv
IPR2024-00496

8.3.2.13.7 DFP to Cable Plug Exit Mode

Figure 8-49 shows an example sequence between a USB Type-C DFP and a Cable Plug, where the DFP commands the Cable Plug to exit an Active Mode.

Figure 8-49 DFP to Cable Plug Exit Mode

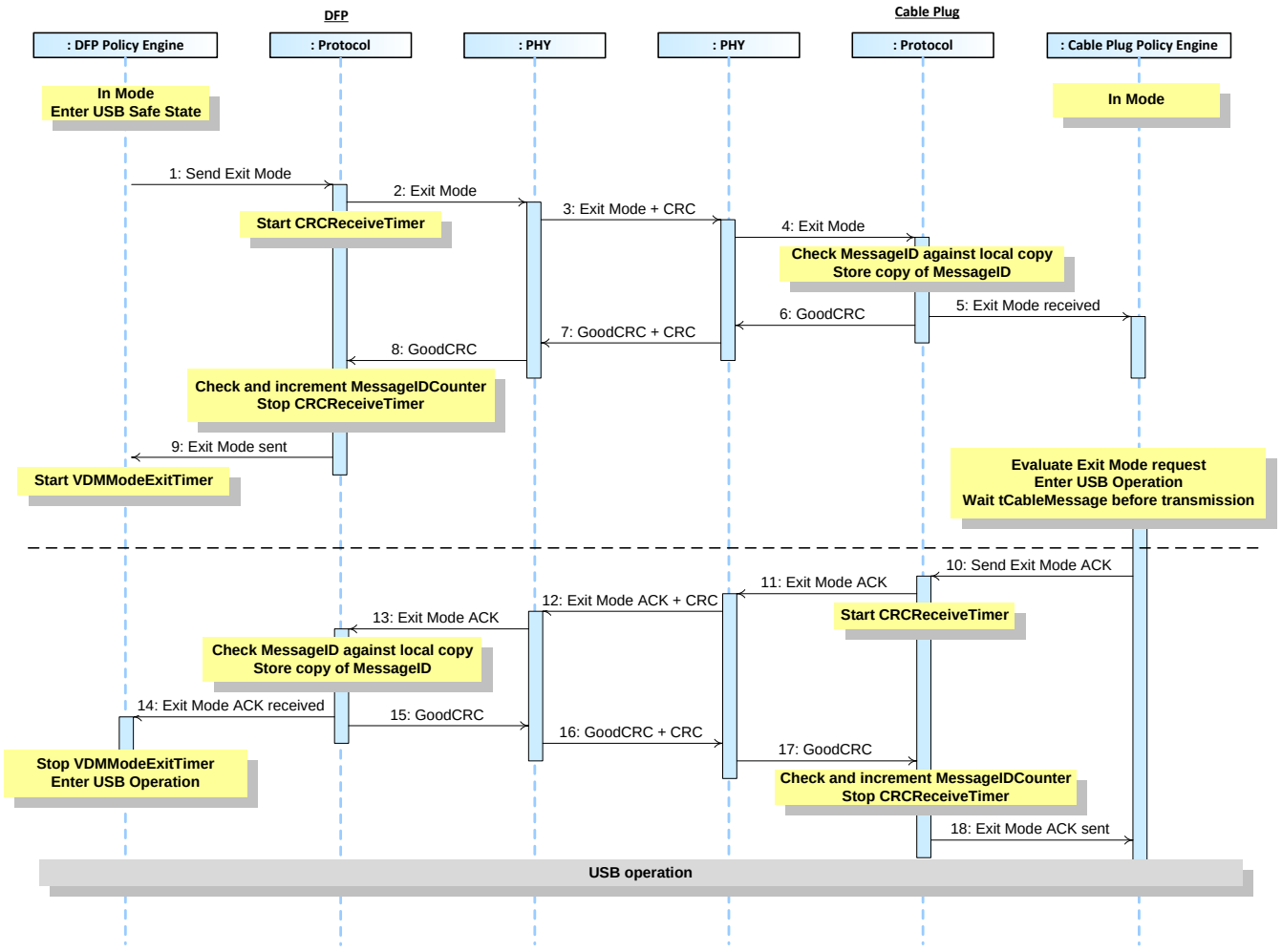


Table 8-49 below provides a detailed explanation of what happens at each labeled step in Figure 8-49 above.

Table 8-49 Steps for DFP to Cable Plug Exit Mode

Step	DFP	Cable Plug
1	The DFP is in a Mode and then enters USB Safe State. The Policy Engine directs the Protocol Layer to send an ***Exit Mode*** Command request.	The Cable Plug is in a Mode.
2	Protocol Layer creates the ***Exit Mode*** Command request and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***Exit Mode*** Command request.	Physical Layer receives the ***Exit Mode*** Command request and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***Exit Mode*** Command request to the Protocol Layer.

Microchip Ex. 1020, p. 367
Microchip v. Aptiv
IPR2024-00496

Step	DFP	Cable Plug
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Exit Mode*** Command request information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** Message and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Exit Mode*** Command request was successfully sent. Policy Engine starts the ***VDMModeExitTimer***.	
10		Policy Engine requests the Device Policy Manager to enter USB operation. ***tCableMessage*** after the ***GoodCRC*** Message was sent the Policy Engine tells the Protocol Layer to form an ***Exit Mode*** Command ACK response.
11		Protocol Layer creates the ***Exit Mode*** Command ACK response and passes to Physical Layer. Starts ***CRCReceiveTimer***.
12	Physical Layer receives the ***Exit Mode*** Command ACK response and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Exit Mode*** Command ACK response.
13	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Exit Mode*** Command ACK response information to the Policy Engine that consumes it.	
14	The Policy Engine stops the ***VDMModeExitTimer*** and requests the Device Policy Manager to enter USB Operation.	
15	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
16	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
17		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.
18		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Exit Mode*** Command ACK response was successfully sent.
	Both DFP and Cable Plug are in USB Operation	

Microchip Ex. 1020, p. 368
Microchip v. Aptiv
IPR2024-00496

8.3.2.13.8 UFP to DFP Attention

Figure 8-50 shows an example sequence between a USB Type-C DFP and a UFP, where the UFP requests attention from the DFP.

Figure 8-50 UFP to DFP Attention

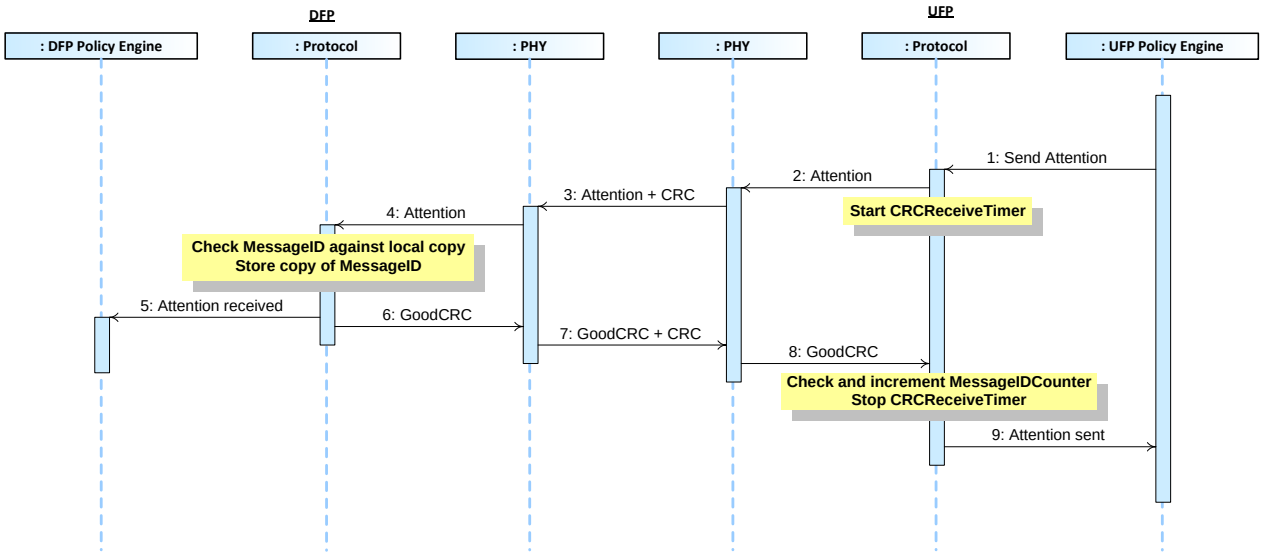


Table 8-50 below provides a detailed explanation of what happens at each labeled step in Figure 8-50 above.

Table 8-50 Steps for UFP to DFP Attention

Step	DFP	UFP
1		The Device Policy Manager requests attention. The Policy Engine tells the Protocol Layer to form an ***Attention*** Command request.
2		Protocol Layer creates the ***Attention*** Command request and passes to Physical Layer. Starts ***CRCReceiveTimer***.
3	Physical Layer receives the ***Attention*** Command request and compares the CRC it calculated with the one sent to verify the Message.	Physical Layer appends a CRC and sends the ***Attention*** Command request.
4	Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***Attention*** Command request information to the Policy Engine that consumes it.	
5	The Policy Engine informs the Device Policy Manager	
6	Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.	
7	Physical Layer appends a CRC and sends the ***GoodCRC*** Message.	Physical Layer receives ***GoodCRC*** Message and compares the CRC it calculated with the one sent to verify the Message.
8		Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.

Microchip Ex. 1020, p. 369
Microchip v. Aptiv
IPR2024-00496

Step	DFP	UFP
9		Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***Attention*** Command request was successfully sent.

Microchip Ex. 1020, p. 370
Microchip v. Aptiv
IPR2024-00496

8.3.2.14 Built in Self-Test (BIST)

8.3.2.14.1 BIST Carrier Mode

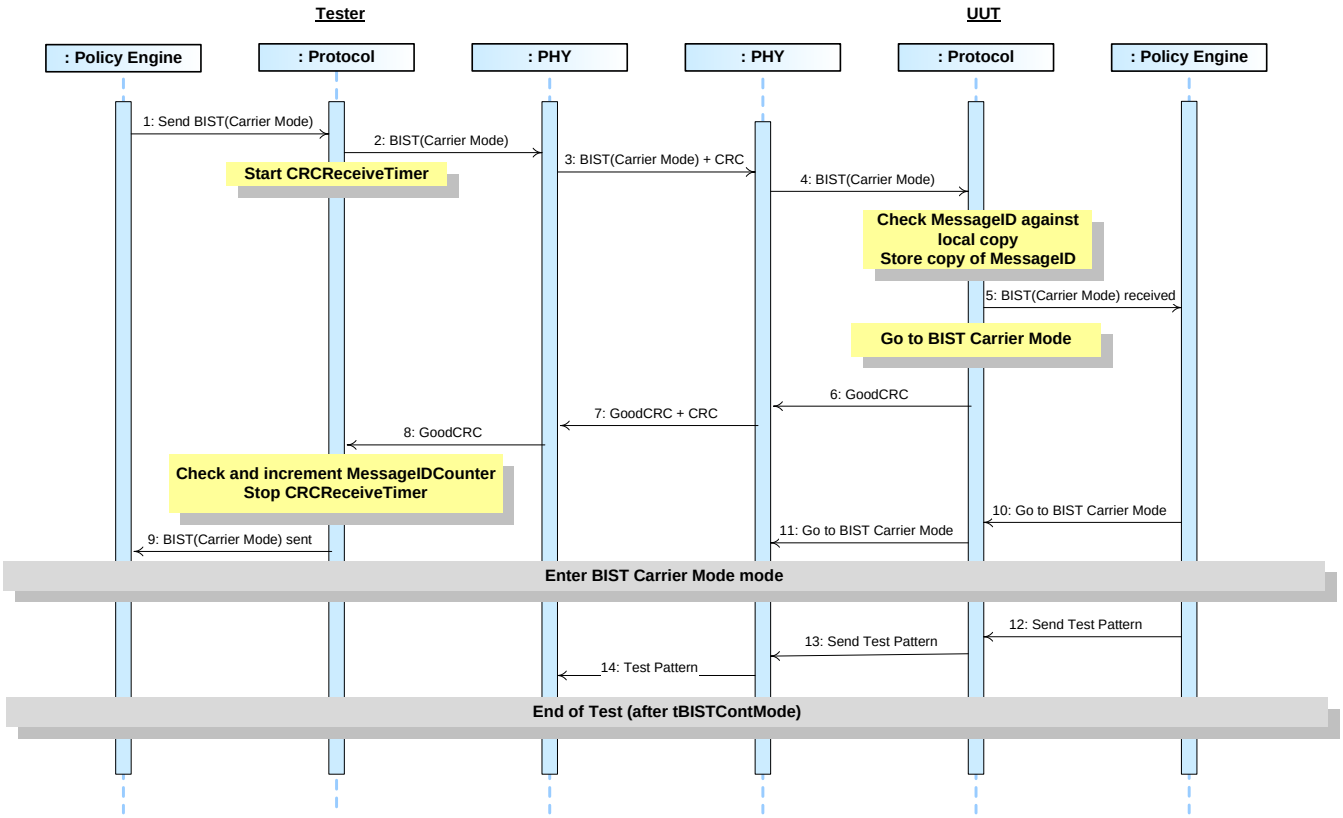
The following is an example of a ***BIST Carrier Mode*** test between a Tester and a UUT. When the UUT is connected to the Tester the sequence below is executed.

Figure 8-51 shows the Messages as they flow across the bus and within the devices. This test enables the measurement of power supply noise and frequency drift.

1) Connection is established and stable.
2) Tester sends a ***BIST*** Message with a ***BIST Carrier Mode*** BIST Data Object.
3) UUT answers with a ***GoodCRC*** Message.
4) UUT starts sending the Test Pattern.
5) Operator does the measurements.
6) The test ends after ***tBISTContMode***.

See also Section 5.9.1 and Section 6.4.3

Figure 8-51 BIST Carrier Mode Test



Microchip Ex. 1020, p. 371
Microchip v. Aptiv
IPR2024-00496

Table 8-51 Steps for BIST Eye Pattern Test

Step	Tester	UUT
1	The Policy Engine directs the Protocol Layer to generate a ***BIST*** Message, with a BIST Data Object of ***BIST Carrier Mode***, to put the UUT into BIST Carrier Mode test mode.	
2	Protocol Layer creates the Message and passes to Physical Layer. Starts ***CRCReceiveTimer***.	
3	Physical Layer appends CRC and sends the ***BIST*** Message.	Physical Layer receives the ***BIST*** Message and checks the CRC to verify the Message.
4		Physical Layer removes the CRC and forwards the ***BIST*** Message to the Protocol Layer.
5		Protocol Layer checks the ***MessageID*** in the incoming Message is different from the previously stored value and then stores a copy of the new value. The Protocol Layer forwards the received ***BIST*** Message information to the Policy Engine that consumes it.
6		Protocol Layer generates a ***GoodCRC*** Message and passes it Physical Layer.
7	Physical Layer receives the ***GoodCRC*** and checks the CRC to verify the Message.	Physical Layer appends CRC and sends the ***GoodCRC*** Message.
8	Physical Layer removes the CRC and forwards the ***GoodCRC*** Message to the Protocol Layer.	
9	Protocol Layer verifies and increments the ***MessageIDCounter*** and stops ***CRCReceiveTimer***. Protocol Layer informs the Policy Engine that the ***BIST*** Message was successfully sent.	
10		Policy Engine tells Protocol Layer to go into ***BIST Carrier Mode***. The Policy Engine goes to ***BIST Carrier Mode***.
11		Protocol Layer tells Physical Layer to go into ***BIST Carrier Mode***.
	UUT enters ***BIST Carrier Mode***	
12		The Policy Engine directs the Protocol Layer to start generation of the Test Pattern.
13		Protocol Layer directs the PHY Layer to generate the Test Pattern.
14	Physical Layer receives the Test Pattern stream.	Physical Layer generates a continuous Test Pattern stream.
	The UUT exits ***BIST Carrier Mode*** after ***tBISTContMode.***	

Microchip Ex. 1020, p. 372
Microchip v. Aptiv
IPR2024-00496

8.3.3 State Diagrams

8.3.3.1 Introduction to state diagrams used in Chapter 8

The state diagrams defined in Section 8.3.3 are ***Normative*** and shall define the operation of the Power Delivery Policy Engine. Note that these state diagrams are not intended to replace a well written and robust design.

Figure 8-52 Outline of States

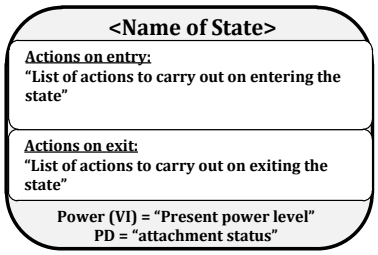


Figure 8-52 shows an outline of the states defined in the following sections. At the top there is the name of the state. This is followed by "Actions on entry" a list of actions carried out on entering the state. If there are also "Actions on exit" a list of actions carried out on exiting the state then 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 8-53 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 pre-conditions (see example in Figure 8-54). It is also possible that the entry and return states are the same. Figure 8-55 indicates a state reference where each referenced state corresponds to either the entry state or the exit state.

Figure 8-53 References to states

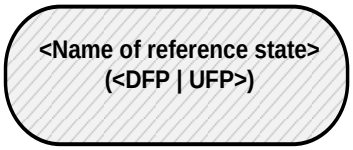
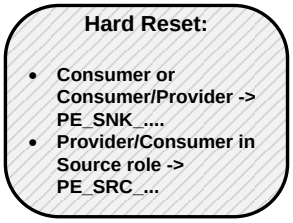
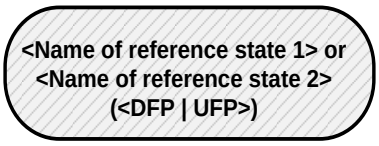


Figure 8-54 Example of state reference with conditions



Microchip Ex. 1020, p. 373
Microchip v. Aptiv
IPR2024-00496

Figure 8-55 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.

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-C Port Control.

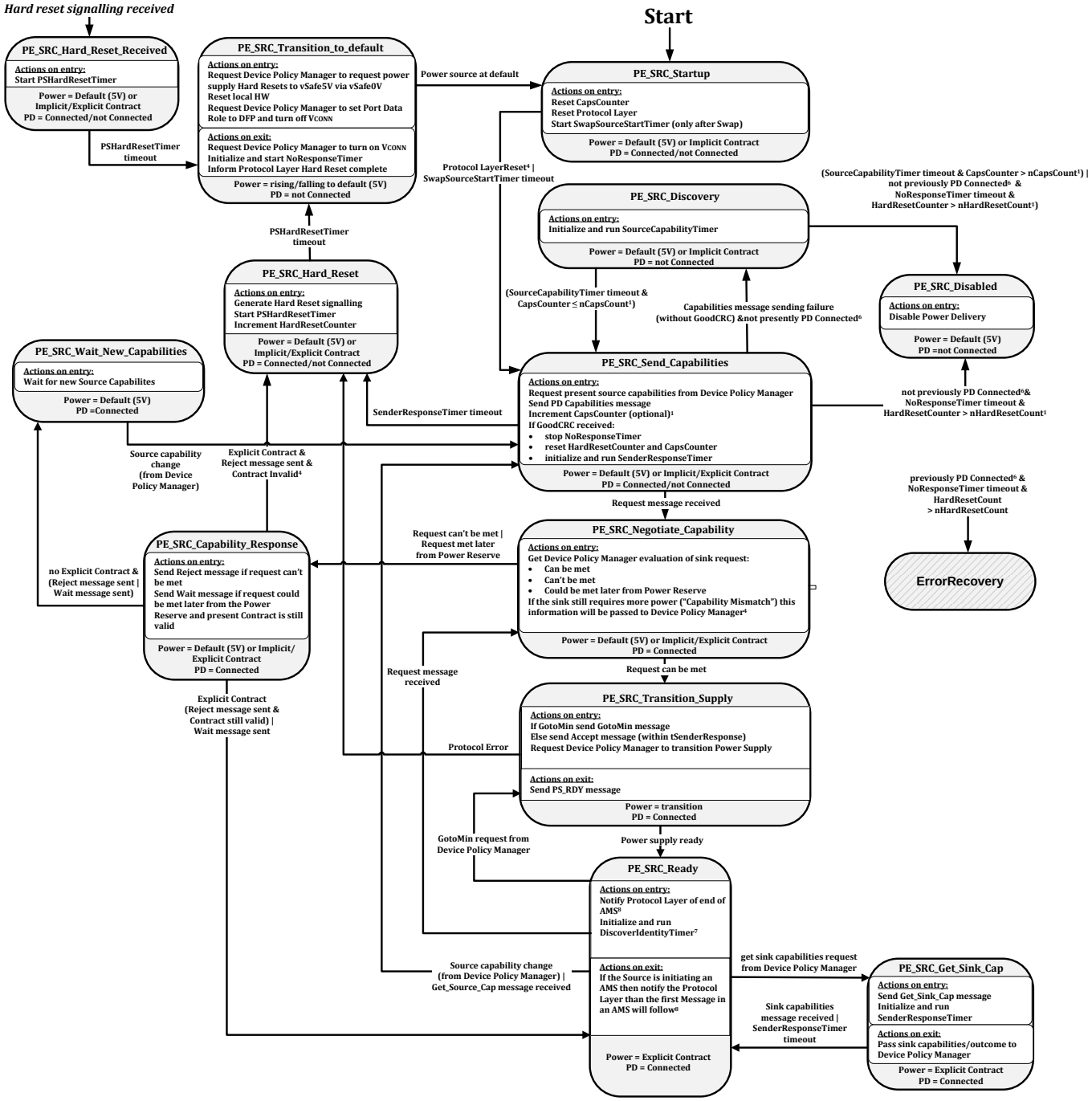
Note: The following state diagrams are not intended to cover all possible corner cases that may be encountered. For example where an outgoing Message is ***Discarded***, due to an incoming Message by the Protocol Layer (see Section 6.11.2.2) it will be necessary for the higher layers of the system to handle a retry of the Message sequence that was being initiated, after first handling the incoming Message.

8.3.3.2 Policy Engine Source Port State Diagram

Figure 8-56 below shows the state diagram for the Policy Engine in a Source Port. The following sections describe operation in each of the states.

Microchip Ex. 1020, p. 374
Microchip v. Aptiv
IPR2024-00496

Figure 8-56 Source Port Policy Engine 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.

Microchip Ex. 1020, p. 375
Microchip v. Aptiv
IPR2024-00496

[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] If transition into the ***PE_SRC_Ready*** state will result in an immediate transition out of the ***PE_SRC_Ready*** state 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.

8.3.3.2.1 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 that resetting the Protocol Layer will also reset the ***MessageIDCounter*** and stored ***MessageID*** (see Section 6.11.2.3).

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.

8.3.3.2.2 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*** ≤ ***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.

Microchip Ex. 1020, p. 376
Microchip v. Aptiv
IPR2024-00496

8.3.3.2.3 PE_SRC_Send_Capabilities state

Note: this state may 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 ***Source_Capabilities*** Message containing these capabilities and 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 that 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.

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***

8.3.3.2.4 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:

Microchip Ex. 1020, p. 377
Microchip v. Aptiv
IPR2024-00496

- 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.

8.3.3.2.5 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 either send a ***GotoMin*** Message (if this was requested by the Device Policy Manager) or otherwise an ***Accept*** Message and inform the Device Policy Manager that it shall transition the power supply to the Requested power level. Note: that 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.

8.3.3.2.6 PE_SRC_Ready state

In the ***PE_SRC_Ready*** state the PD Source shall operating at a stable power with no ongoing negotiation. It shall respond to requests from the Sink, events from the Device Policy Manager and shall send out ***Ping*** Messages to maintain the PD link if required (see Section 6.3.5).

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 8.3.3.4.1) 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 DFP which needs to establish communication with a Cable Plug, the DFP shall initialize and run the ***DiscoverIdentityTimer*** (no ***GoodCRC*** Message response yet received to ***Discover Identity*** Message).

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.

The Policy Engine shall transition to the ***PE_SRC_Transition_Supply*** state when:

- A GotoMin request is received from the Device Policy Manager for the Attached Device to go to minimum power.

The Policy Engine shall transition to the ***PE_SRC_Get_Sink_Cap*** state when:

- The Device Policy Manager asks for the Sink's capabilities.

8.3.3.2.7 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.

Microchip Ex. 1020, p. 378
Microchip v. Aptiv
IPR2024-00496

8.3.3.2.8 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 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_Capability_Response*** 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 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 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 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 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.

8.3.3.2.9 PE_SRC_Hard_Reset state

On entry to the ***PE_SRC_Hard_Reset*** state the Policy Engine shall request the generation of ***Hard Reset*** Signaling by the PHY Layer, initialize and run the ***PSHardResetTimer*** and increment the ***HardResetCounter***. Note that 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.

8.3.3.2.10 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***.

The Policy Engine shall transition to the ***PE_SRC_Transition_to_default*** state when:

- The ***PSHardResetTimer*** times out.

8.3.3.2.11 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 7.1.6)
- 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:

Microchip Ex. 1020, p. 379
Microchip v. Aptiv
IPR2024-00496

- request the Device Policy Manager to turn on VCONN
- initialize and run the ***NoResponseTimer*****.** Note that the ***NoResponseTimer*** shall continue to run in every state until it is stopped or times out.
- 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.

8.3.3.2.12 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's 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:

- A ***Sink_Capabilities*** Message is received.
- Or ***SenderResponseTimer*** times out.

8.3.3.2.13 PE_SRC_Wait_New_Capabilities

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:

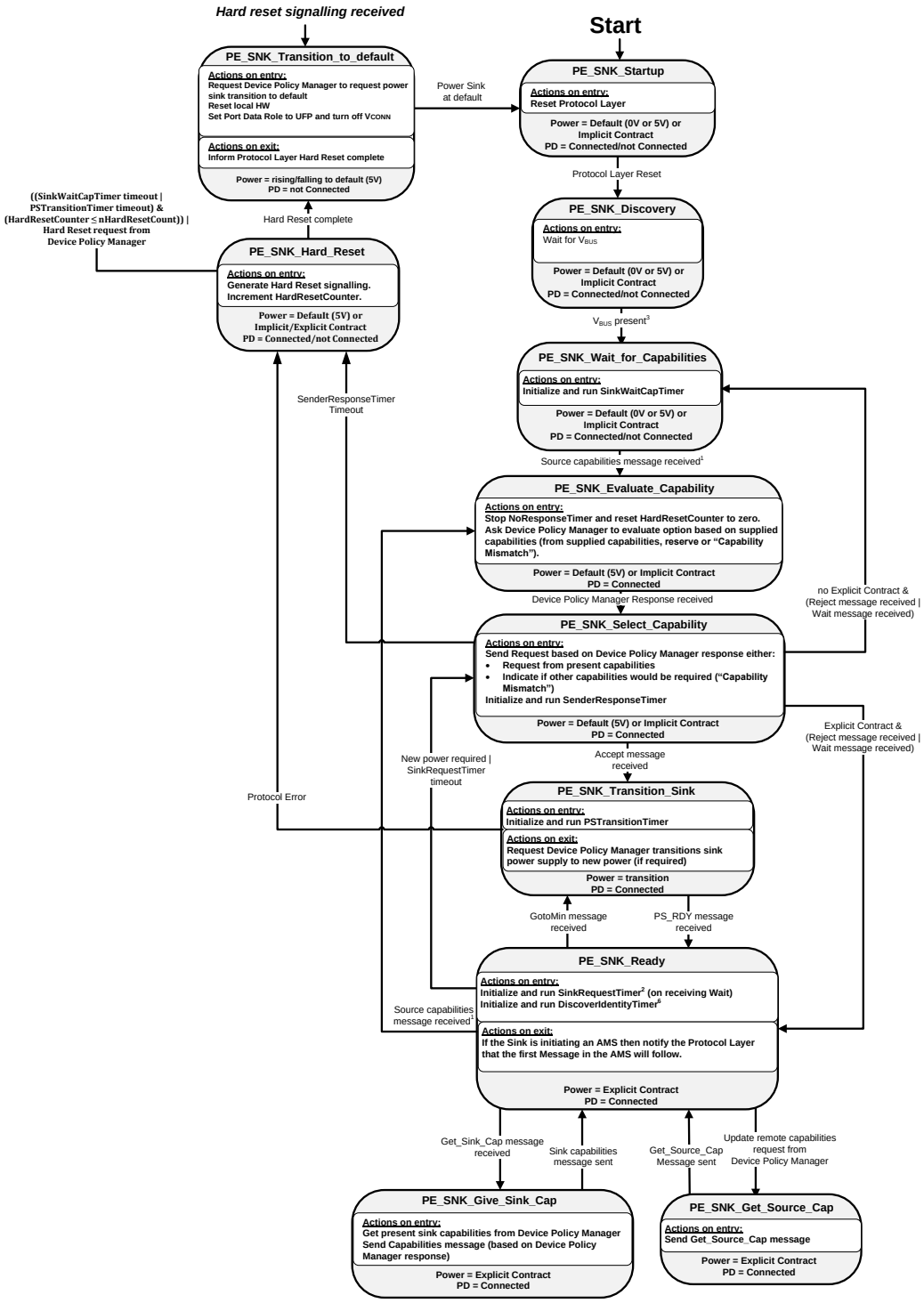
- The Device Policy Manager indicates that Source Capabilities have changed.

8.3.3.3 Policy Engine Sink Port State Diagram

Figure 8-57 below shows the state diagram for the Policy Engine in a Sink Port. The following sections describe operation in each of the states.

Microchip Ex. 1020, p. 380
Microchip v. Aptiv
IPR2024-00496

Figure 8-57 Sink Port state diagram



[1] Source capabilities messages received in states other than ***PE_SNK_Wait_for_Capabilities*** and ***PE_SNK_Ready*** constitute a Protocol Error.

[2] The ***SinkRequestTimer*** should not be stopped if a ***Ping*** 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*** Message.

Microchip Ex. 1020, p. 381
Microchip v. Aptiv
IPR2024-00496

[3] Error Recovery steps can be taken at this point which are defined in the ***[USB Type-C 1.2]*** specification and are outside the scope of this specification.

[4] During a Hard Reset the Source voltage will transition to ***vSafe0V*** and then transition to ***vSafe5V***. Sinks need to ensure that V_{BUS} present is not indicated until after the Source has completed the Hard Reset process by detecting both of these transitions.

[5] PD Connected is defined as a situation when the Port Partners have exchanged a Message and ***GoodCRC*** Message response. The Port Partners remain PD Connected, even after a Swap, until the Port Partners are Detached.

[6] The ***DiscoverIdentityTimer*** is run when this is a V_{CONN} 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.

8.3.3.3.1 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 that resetting the Protocol Layer will also reset the ***MessageIDCounter*** and stored ***MessageID*** (see Section 6.11.2.3).

Once the reset process completes, the Policy Engine shall transition to the ***PE_SNK_Discovery*** state.

8.3.3.3.2 PE_SNK_Discovery state

In the ***PE_SNK_Discovery*** state the Sink Policy Engine waits for V_{BUS} to be present.

The Policy Engine shall transition to the ***PE_SNK_Wait_for_Capabilities*** state when:

- The Device Policy Manager indicates that V_{BUS} has been detected.

The Policy Engine shall transition to the *ErrorRecovery* state when:

- The Port Partners have previously been PD Connected (the Port remains Attached to a Port it has had a PD Connection with during this Attachment) and
- There has been a ***NoResponseTimer*** timeout and
- The ***HardResetCounter*** > ***nHardResetCount***

8.3.3.3.3 PE_SNK_Wait_for_Capabilities state

On entry to the ***PE_SNK_Wait_for_Capabilities*** state the Policy Engine shall initialize and start the ***SinkWaitCapTimer***.

The Policy Engine shall transition to the ***PE_SNK_Evaluate_Capability*** state when:

- A ***Source_Capabilities*** Message is received.

When the ***SinkWaitCapTimer*** times out, the Policy Engine will perform a Hard Reset.

8.3.3.3.4 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 which new power level is required:

- A selection from the present offered capabilities is to be made.
- Capability mismatch; 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.

Microchip Ex. 1020, p. 382
Microchip v. Aptiv
IPR2024-00496

8.3.3.3.5 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).

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 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_Hard_Reset*** state when:

- A ***SenderResponseTimer*** timeout occurs.

8.3.3.3.6 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 8.3.3.3.8 and shall then request the Device Policy Manager to transition the Sink's power supply to the new power level. Note that 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.

8.3.3.3.7 PE_SNK_Ready state

In the ***PE_SNK_Ready*** state the PD Sink shall be operating at a stable power level with no ongoing negotiation. It shall respond to requests from the Source, events from the Device Policy Manager and may monitor for ***Ping*** Messages to maintain the PD link.

On entry to the ***PE_SNK_Ready*** state as the result of a wait the Policy Engine should do the following:

- Initialize and run the ***SinkRequestTimer***.

On entry to the ***PE_SNK_Ready*** state if this is a DFP which needs to establish communication with a Cable Plug, then the Policy Engine shall do the following:

- Initialize and run the ***DiscoverIdentityTimer*** (no ***GoodCRC*** Message response yet received to ***Discover Identity*** Message).

Microchip Ex. 1020, p. 383
Microchip v. Aptiv
IPR2024-00496

On exit from the ***PE_SNK_Ready*** state, if the transition is as a result of a DPM request to start a new Atomic Message Sequence (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_SNK_Evaluate_Capability*** state when:

- A ***Source_Capabilities*** Message is received.

The Policy Engine shall transition to the ***PE_SNK_Select_Capability*** state when:

- A new power level is requested by the Device Policy Manager.
- A ***SinkRequestTimer*** timeout occurs.

The Policy Engine shall transition to the ***PE_SNK_Transition_Sink*** state when:

- A ***GotoMin*** Message is received.

The Policy Engine shall transition back to the ***PE_SNK_Ready*** state when:

- A ***Ping*** Message is received. Note this should not cause the ***SinkRequestTimer*** to be reinitialized.

The Policy Engine shall transition to the ***PE_SNK_Give_Sink_Cap*** state when:

- A ***Get_Sink_Cap*** Message is received from the Protocol Layer.

The Policy Engine shall transition to the ***PE_SNK_Get_Source_Cap*** state when:

- The Device Policy Manager requests an update of the remote Source's capabilities.

8.3.3.3.8 PE_SNK_Hard_Reset state

The Policy Engine shall transition to the ***PE_SNK_Hard_Reset*** state from any state when:

- ((***SinkWaitCapTimer*** timeout |
- ***PSTransitionTimer*** timeout |
- ***NoResponseTimer*** timeout) &
- (***HardResetCounter*** ≤ ***nHardResetCount***)) |
- Hard Reset request from Device Policy Manager

Note: if the ***NoResponseTimer*** 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.

8.3.3.3.9 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 Manger that the Port Data Role is set to UFP.

Microchip Ex. 1020, p. 384
Microchip v. Aptiv
IPR2024-00496

On exit from the ***PE_SNK_Transition_to_default*** state the Policy Engine shall initialize and run the ***NoResponseTimer*** and inform the Protocol Layer that the Hard Reset is complete**.** Note that the ***NoResponseTimer*** shall continue to run in every state until it is stopped or times out.

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.

8.3.3.3.10 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 transition to the ***PE_SNK_Ready*** state when:

- The ***Sink_Capabilities*** Message has been successfully sent.

8.3.3.3.11 PE_SNK_Get_Source_Cap state

In the ***PE_SNK_Get_Source_Cap*** state the Policy Engine, due to a request from the Device Policy Manager, shall request the capabilities from the Attached Source.

On entry to the ***PE_SNK_Get_Source_Cap*** state the Policy Engine shall request the Protocol Layer to send a ***Get_Source_Cap*** Message in order to retrieve the Source's capabilities.

The Policy Engine shall transition to the ***PE_SNK_Ready*** state when:

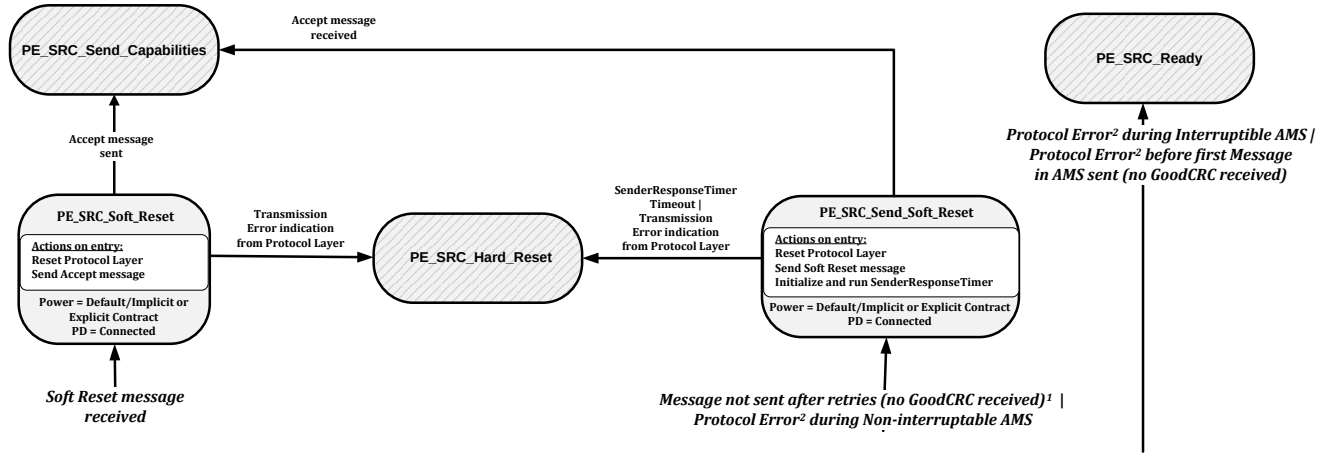
- The ***Get_Source_Cap*** Message is sent.

8.3.3.4 Soft Reset and Protocol Error State Diagrams

8.3.3.4.1 Source Port Soft Reset and Protocol Error State Diagram

Figure 8-58 below shows the state diagram for the Policy Engine in a Source Port when performing a Soft Reset of its Port Partner. The following sections describe operation in each of the states.

Figure 8-58 Source Port Soft Reset and Protocol Error State Diagram



[1] Excludes the ***Soft_Reset*** Message itself.

Microchip Ex. 1020, p. 385
Microchip v. Aptiv
IPR2024-00496

[2] An unrecognized or unsupported Message will result in a ***Not_Supported*** Message response being generated (see Section 6.3.14).

8.3.3.4.1.1 PE_SRC_Send_Soft_Reset state

The ***PE_SRC_Send_Soft_Reset*** state shall be entered from any state when a Protocol Error is detected by the Protocol Layer during a Non-interruptible AMS (see Section 6.8.1) or when a Message has not been sent after retries to the Sink. 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 (without GoodCRC) and the source is not presently Attached (as indicated in Figure 8-56). In this case, the ***PE_SRC_Discovery*** state is entered (see Section 8.3.3.2.3).
- When the voltage is in transition due to a new Explicit Contract being negotiated (see Section 8.3.3.2). In this case a Hard Reset will be generated.
- During a Power Role Swap when the power supply is in transition (see Section 8.3.3.15.3 and Section 8.3.3.15.4). 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 Date Role field (see Section 6.2.1.1.6). In this case USB Type-C Error Recovery will be triggered directly.

Note that 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:

- Protocol Errors occurring during an Interruptible AMS.
- Protocol Errors occurring during any AMS where the first Message in the sequence 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 Protocol Layer to perform a Soft Reset, then shall send a ***Soft_Reset*** Message to the Sink, 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.

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.

8.3.3.4.1.2 PE_SRC_Soft_Reset state

The ***PE_SRC_Soft_Reset*** state shall be entered from any state when a ***Soft_Reset*** Message is received from the Protocol Layer.

On entry to the ***PE_SRC_Soft_Reset*** state the Policy Engine shall reset the Protocol Layer and shall then request the Protocol Layer to send an ***Accept*** Message.

The Policy Engine shall transition to the ***PE_SRC_Send_Capabilities*** state (see Section 8.3.3.2.3) when:

- The ***Accept*** Message has been sent.

The Policy Engine shall transition to the ***PE_SRC_Hard_Reset*** state when:

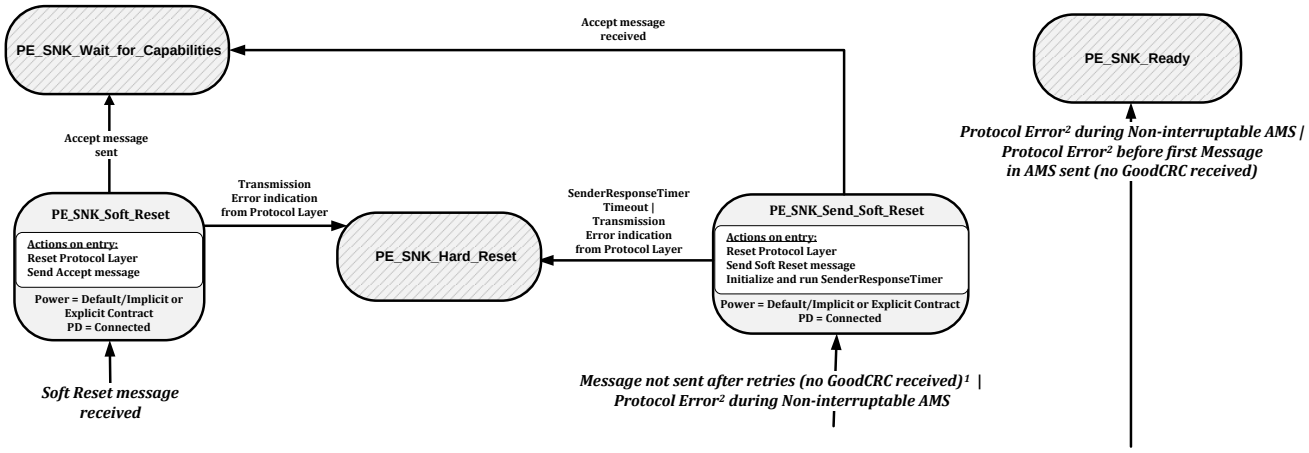
- The Protocol Layer indicates that a transmission error has occurred.

8.3.3.4.2 Sink Port Soft Reset and Protocol Error State Diagram

Figure 8-59 below shows the state diagram for the Policy Engine in a Sink Port when performing a Soft Reset of its Port Partner. The following sections describe operation in each of the states.

Microchip Ex. 1020, p. 386
Microchip v. Aptiv
IPR2024-00496

Figure 8-59 Sink Port Soft Reset and Protocol Error Diagram



[1] Excludes the ***Soft_Reset*** Message itself.

[2] An unrecognized or unsupported Message will result in a ***Not_Supported*** Message response being generated (see Section 6.3.14).

8.3.3.4.2.1 PE_SNK_Send_Soft_Reset state

The ***PE_SNK_Send_Soft_Reset*** state shall be entered from any state when a Protocol Error is detected by the Protocol Layer during a Non-interruptible AMS (see Section 6.8.1) or when a Message has not been sent after retries to the Source. The main exceptions to this rule are when:

- When the voltage is in transition due to a new Explicit Contract being negotiated (see Section 8.3.3.3). In this case a Hard Reset will be generated.
- During a Power Role Swap when the power supply is in transition (see Section 8.3.3.15.3 and Section 8.3.3.15.4). In this case a hard reset will be triggered directly.
- During a Data Role Swap when the DFP/UFP roles are changing. In this case USB Type-C Error Recovery will be triggered directly.

Note that 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:

- Protocol Errors occurring during an Interruptible AMS.
- Protocol Errors occurring during any AMS where the first Message in the sequence 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 Protocol Layer to perform a Soft Reset, then shall send a ***Soft_Reset*** Message 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.

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.

8.3.3.4.2.2 PE_SNK_Soft_Reset state

The ***PE_SNK_Soft_Reset*** state shall be entered from any state when a ***Soft_Reset*** Message is received from the Protocol Layer.

Microchip Ex. 1020, p. 387
Microchip v. Aptiv
IPR2024-00496

On entry to the ***PE_SNK_Soft_Reset*** state the Policy Engine shall reset the Protocol Layer and shall then request the Protocol Layer to send an ***Accept*** Message.

The Policy Engine shall transition to the ***PE_SNK_Wait_for_Capabilities*** state when:

- The ***Accept*** Message has been sent.

The Policy Engine shall transition to the ***PE_SNK_Hard_Reset*** state when:

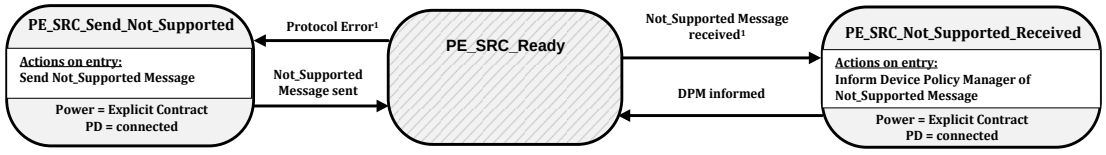
- The Protocol Layer indicates that a transmission error has occurred.

8.3.3.5 Not Supported Message State Diagrams

8.3.3.5.1 Source Port Not Supported Message State Diagram

Figure 8-60 shows the state diagram for a ***Not_Supported*** Message sent or received by a Source Port.

Figure 8-60 Source Port Not Supported Message State Diagram



[1] Transition can either be the result of a Protocol Error during an interruptible AMS or as a result of an unsupported Message being received in the ***PE_SRC_Ready*** state directly (see also Section 8.3.3.4.1).

8.3.3.5.1.1 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 (see also Section 8.3.3.4.1).

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 8-60) when:

- The ***Not_Supported*** Message has been successfully sent.

8.3.3.5.1.2 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 8-60) when:

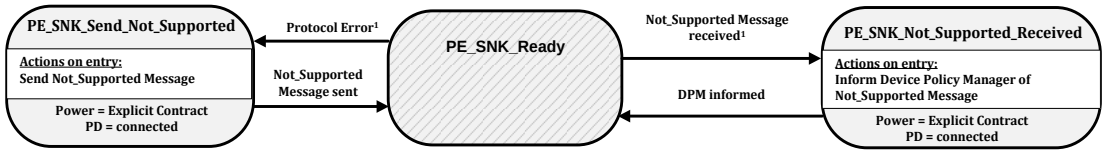
- The Device Policy Manager has been informed.

8.3.3.5.2 Sink Port Not Supported Message State Diagram

Figure 8-61 shows the state diagram for a ***Not_Supported*** Message sent or received by a Sink Port.

Microchip Ex. 1020, p. 388
Microchip v. Aptiv
IPR2024-00496

Figure 8-61 Sink Port Not Supported Message State Diagram



[1] Transition can either be the result of a Protocol Error during an interruptible AMS or as a result of an unsupported Message being received in the ***PE_SNK_Ready*** state directly (see also Section 8.3.3.4.2).

8.3.3.5.2.1 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 (see also Section 8.3.3.4.2).

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 8-61) when:

- The ***Not_Supported*** Message has been successfully sent.

8.3.3.5.2.2 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.

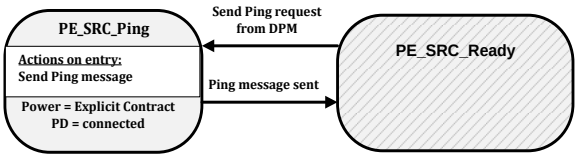
The Policy Engine shall transition back to the previous state (***PE_SNK_Ready*** see Figure 8-61) when:

- The Device Policy Manager has been informed.

8.3.3.6 Source Port Ping State Diagram

Figure 8-60 shows the state diagram for a ***Ping*** Message from a Source Port.

Figure 8-62 Source Port Ping State Diagram



8.3.3.6.1 PE_SRC_Ping state

On entry to the ***PE_SRC_Ping*** state (from the ***PE_SRC_Ready*** state) the Policy Engine shall request the Protocol Layer to send a ***Ping*** Message.

The Policy Engine shall transition back to the previous state (***PE_SRC_Ready)*** (see Figure 8-56) when:

- The ***Ping*** Message has been successfully sent.

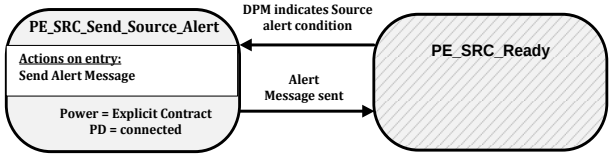
Microchip Ex. 1020, p. 389
Microchip v. Aptiv
IPR2024-00496

8.3.3.7 Source Alert State Diagrams

8.3.3.7.1 Source Port Source Alert State Diagram

Figure 8-63 shows the state diagram for an Alert Message sent by a Source Port.

Figure 8-63 Source Port Source Alert State Diagram



8.3.3.7.1.1 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.

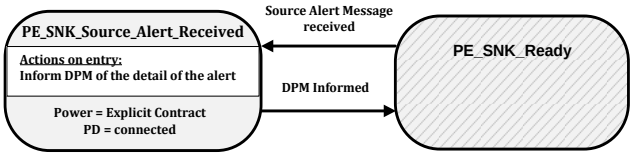
The Policy Engine shall transition back to ***PE_SRC_Ready*** (see Figure 8-56) when:

- The Alert Message has been successfully sent.

8.3.3.7.2 Sink Port Source Alert State Diagram

Figure 8-64 shows the state diagram for an Alert Message received by a Sink Port.

Figure 8-64 Sink Port Source Alert State Diagram



8.3.3.7.2.1 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 back to ***PE_SNK_Ready*** (see Figure 8-57) when:

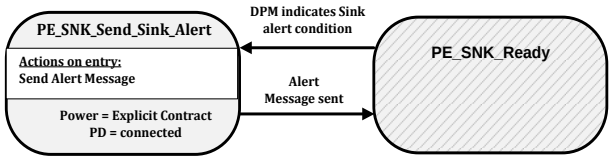
- The DPM has been informed.

8.3.3.7.3 Sink Port Sink Alert State Diagram

Figure 8-65 shows the state diagram for an Alert Message sent by a Sink Port.

Microchip Ex. 1020, p. 390
Microchip v. Aptiv
IPR2024-00496

Figure 8-65 Sink Port Sink Alert State Diagram



8.3.3.7.3.1 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.

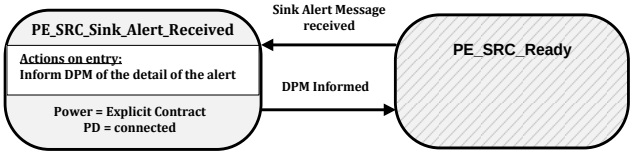
The Policy Engine shall transition back to ***PE_SNK_Ready*** (see Figure 8-57) when:

- The Alert Message has been successfully sent.

8.3.3.7.4 Source Port Sink Alert State Diagram

Figure 8-66 shows the state diagram for an Alert Message received by a Source Port.

Figure 8-66 Source Port Sink Alert State Diagram



8.3.3.7.4.1 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 back to ***PE_SRC_Ready*** (see Figure 8-56) when:

- The DPM has been informed.

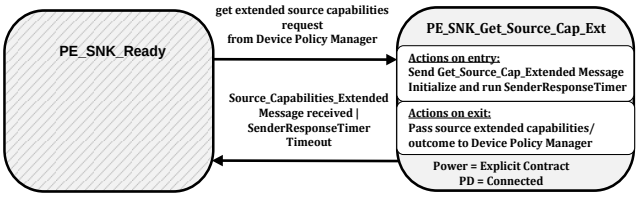
8.3.3.8 Source Capabilities Extended State Diagrams

8.3.3.8.1 Sink Port Get Source Capabilities Extended State Diagram

Figure 8-67 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 6.5.1.

Microchip Ex. 1020, p. 391
Microchip v. Aptiv
IPR2024-00496

Figure 8-67 Sink Port Get Source Capabilities Extended state diagram



8.3.3.8.1.1 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).

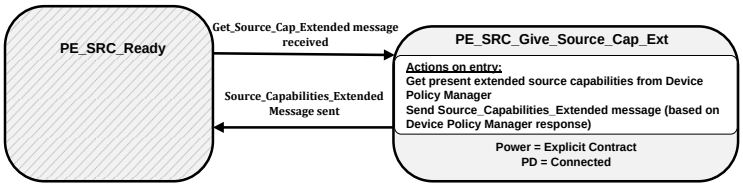
The Policy Engine shall transition back to the ***PE_SNK_Ready*** state (see Figure 8-57) when:

- A ***Source_Capabilities_Extended*** Message is received
- Or ***SenderResponseTimer*** times out

8.3.3.8.2 Source Give Source Capabilities Extended State Diagram

Figure 8-68 shows the state diagram for a Source on receiving a ***Get_Source_Cap_Extended*** Message. See also Section 6.5.1.

Figure 8-68 Source Give Source Capabilities Extended state diagram



8.3.3.8.2.1 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 8-56) when:

- The ***Source_Capabilities_Extended*** Message has been successfully sent.

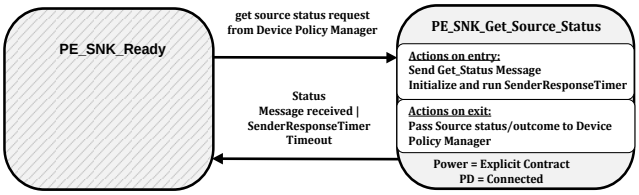
8.3.3.9 Status State Diagrams

8.3.3.9.1 Sink Port Get Source Status State Diagram

Figure 8-69 shows the state diagram for a Sink on receiving a request from the Device Policy Manager to get the Port Partner's Source status. See also Section 6.5.2.

Microchip Ex. 1020, p. 392
Microchip v. Aptiv
IPR2024-00496

Figure 8-69 Sink Port Get Source Status state diagram



8.3.3.9.1.1 PE_SNK_Get_Source_Status state

The Policy Engine shall transition to the ***PE_SNK_Get_Source_Status*** state, from the ***PE_SNK_Ready*** state, due to a request to get the remote source status from the Device Policy Manager.

On entry to the ***PE_SNK_Get_Source_Status*** state the Policy Engine shall send a ***Get_Status*** Message and initialize and run the ***SenderResponseTimer***.

On exit from the ***PE_SNK_Get_Source_Status*** state the Policy Engine shall inform the Device Policy Manager of the outcome (status or response timeout).

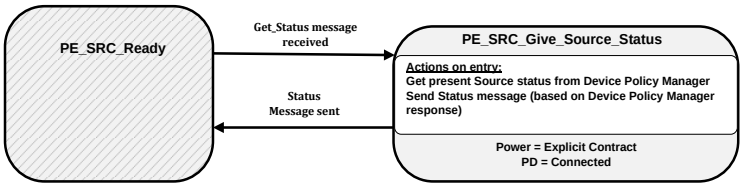
The Policy Engine shall transition back to the ***PE_SNK_Ready*** state (see Figure 8-57) when:

- A ***Status*** Message is received
- Or ***SenderResponseTimer*** times out

8.3.3.9.2 Source Give Source Status State Diagram

Figure 8-70 Source Give Source Status state diagram shows the state diagram for a Source on receiving a ***Get_Status*** Message. See also Section 6.5.1.

Figure 8-70 Source Give Source Status state diagram



8.3.3.9.2.1 PE_SRC_Give_Source_Status state

The Policy Engine shall transition to the ***PE_SRC_Give_Source_Status*** state, from the ***PE_SRC_Ready*** state, when a ***Get_Status*** Message is received.

On entry to the ***PE_SRC_Give_Source_Status*** state the Policy Engine shall request the present extended Source capabilities 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*** state (see Figure 8-56) when:

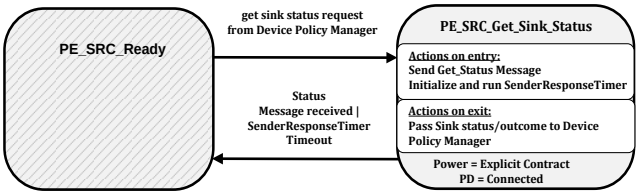
- The ***Status*** Message has been successfully sent.

8.3.3.9.3 Source Port Get Sink Status State Diagram

Figure 8-69 shows the state diagram for a Source on receiving a request from the Device Policy Manager to get the Port Partner's Sink status. See also Section 6.5.2.

Microchip Ex. 1020, p. 393
Microchip v. Aptiv
IPR2024-00496

Figure 8-71 Source Port Get Sink Status state diagram



8.3.3.9.3.1 PE_SRC_Get_Sink_Status state

The Policy Engine shall transition to the ***PE_SRC_Get_Sink_Status*** state, from the ***PE_SRC_Ready*** state, due to a request to get the remote source status from the Device Policy Manager.

On entry to the ***PE_SRC_Get_Sink_Status*** state the Policy Engine shall send a ***Status*** Message and initialize and run the ***SenderResponseTimer***.

On exit from the ***PE_SRC_Get_Sink_Status*** state the Policy Engine shall inform the Device Policy Manager of the outcome (status or response timeout).

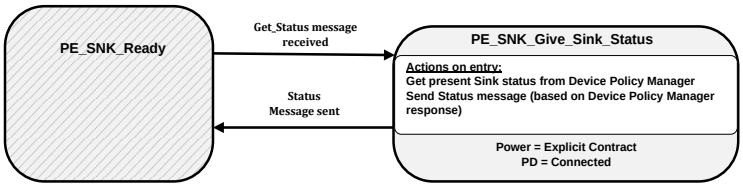
The Policy Engine shall transition back to the ***PE_SRC_Ready*** state (see Figure 8-56) when:

- A ***Status*** Message is received
- Or ***SenderResponseTimer*** times out

8.3.3.9.4 Sink Give Sink Status State Diagram

Figure 8-70 Source Give Source Status state diagram shows the state diagram for a Source on receiving a ***Get_Status*** Message. See also Section 6.5.1.

Figure 8-72 Sink Give Sink Status state diagram



8.3.3.9.4.1 PE_SNK_Give_Sink_Status state

The Policy Engine shall transition to the ***PE_SNK_Give_Sink_Status*** state, from the ***PE_SNK_Ready*** state, when a ***Get_Status*** Message is received.

On entry to the ***PE_SNK_Give_Sink_Status*** state the Policy Engine shall request the present extended Source capabilities from the Device Policy Manager and then send a ***Status*** Message based on these capabilities.

The Policy Engine shall transition back to the ***PE_SNK_Ready*** state (see Figure 8-57) when:

- The ***Status*** Message has been successfully sent.

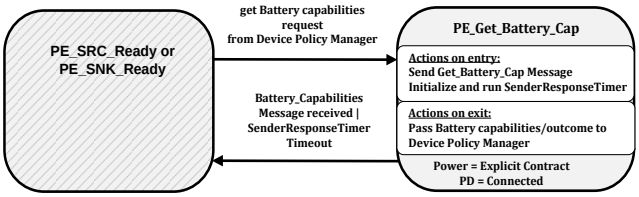
8.3.3.10 Battery Capabilities State Diagrams

8.3.3.10.1 Get Battery Capabilities State Diagram

Figure 8-73 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 6.5.5.

Microchip Ex. 1020, p. 394
Microchip v. Aptiv
IPR2024-00496

Figure 8-73 Get Battery Capabilities state diagram



8.3.3.10.1.1 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).

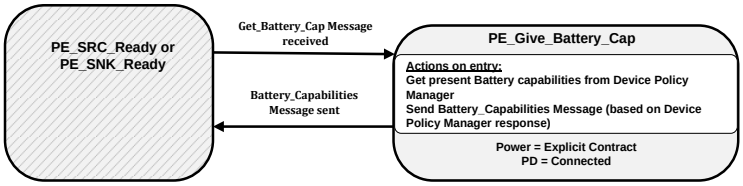
The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state as appropriate (see Figure 8-56 and Figure 8-57) when:

- A ***Battery_Capabilities*** Message is received
- Or ***SenderResponseTimer*** times out

8.3.3.10.2 Give Battery Capabilities State Diagram

Figure 8-74 shows the state diagram for a Source or Sink on receiving a ***Get_Battery_Cap*** Message. See also Section 6.5.5.

Figure 8-74 Give Battery Capabilities state diagram



8.3.3.10.2.1 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 ***Source_Capabilities_Extended*** 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 8-56 and Figure 8-57) when:

- The ***Battery_Capabilities*** Message has been successfully sent.

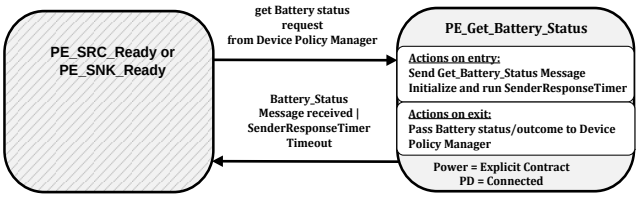
Microchip Ex. 1020, p. 395
Microchip v. Aptiv
IPR2024-00496

8.3.3.11 Battery Status State Diagrams

8.3.3.11.1 Get Battery Status State Diagram

Figure 8-75 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 6.5.4.

Figure 8-75 Get Battery Status state diagram



8.3.3.11.1.1 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).

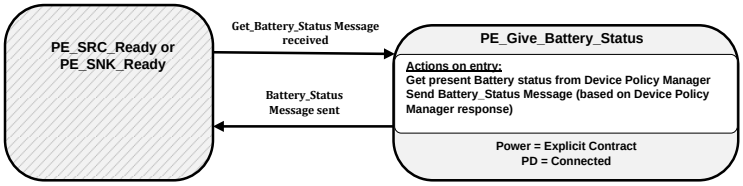
The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state as appropriate (see Figure 8-56 and Figure 8-57) when:

- A ***Battery_Status*** Message is received
- Or ***SenderResponseTimer*** times out

8.3.3.11.2 Give Battery Status State Diagram

Figure 8-76 shows the state diagram for a Source or Sink on receiving a ***Get_Battery_Status*** Message. See also Section 6.5.4.

Figure 8-76 Give Battery Status state diagram



8.3.3.11.2.1 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.

Microchip Ex. 1020, p. 396
Microchip v. Aptiv
IPR2024-00496

The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state as appropriate (see Figure 8-56 and Figure 8-57) when:

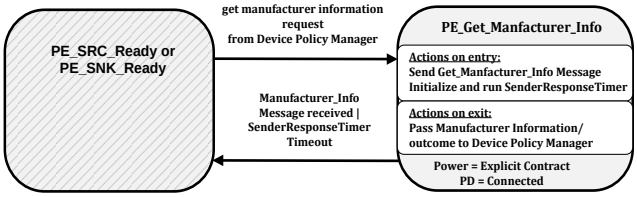
- The ***Battery_Status*** Message has been successfully sent.

8.3.3.12 Manufacturer Information State Diagrams

8.3.3.12.1 Get Manufacturer Information State Diagram

Figure 8-77 shows the state diagram for a Source or Sink on receiving a request from the Device Policy Manager to get the Port Partner's Manufacturer Information. See also Section 6.5.6.

Figure 8-77 Get Manufacturer Information state diagram



8.3.3.12.1.1 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 (status or response timeout).

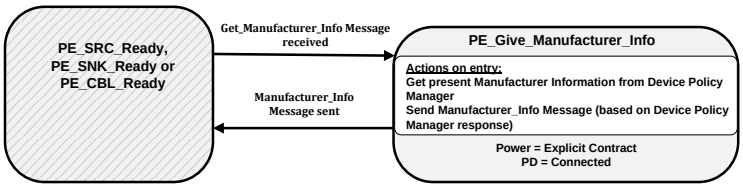
The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state as appropriate (see Figure 8-56 and Figure 8-57) when:

- A ***Manufacturer_Info*** Message is received
- Or ***SenderResponseTimer*** times out

8.3.3.12.2 Give Manufacturer Information State Diagram

Figure 8-78 shows the state diagram for a Source, Sink or Cable Plug on receiving a ***Get_Manufacturer_Info*** Message. See also Section 6.5.6.

Figure 8-78 Give Manufacturer Information state diagram



8.3.3.12.2.1 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.

Microchip Ex. 1020, p. 397
Microchip v. Aptiv
IPR2024-00496

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 8-56, Figure 8-57and Figure 8-110) when:

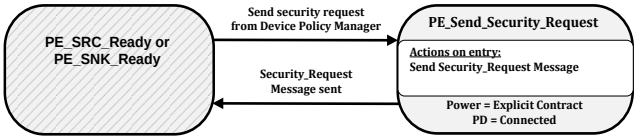
- The ***Manufacturer_Info*** Message has been successfully sent.

8.3.3.13 Security State Diagrams

8.3.3.13.1 Send Security Request State Diagram

Figure 8-79 shows the state diagram for a Source or Sink on receiving a request from the Device Policy Manager to send a security request. See also Section 6.5.8.

Figure 8-79 Send security request state diagram



8.3.3.13.1.1 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.

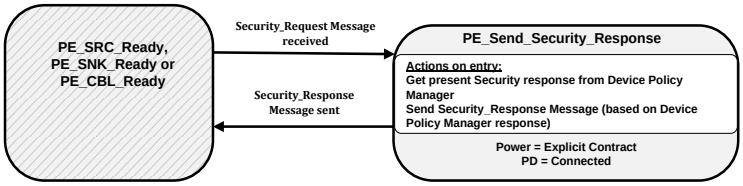
The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state as appropriate (see Figure 8-56 and Figure 8-57) when:

- The ***Security_Request*** Message has been sent

8.3.3.13.2 Send Security Response State Diagram

Figure 8-80 shows the state diagram for a Source, Sink or Cable Plug on receiving a ***Security_Request*** Message. See also Section 6.5.8.

Figure 8-80 Send security response state diagram



8.3.3.13.2.1 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.

Microchip Ex. 1020, p. 398
Microchip v. Aptiv
IPR2024-00496

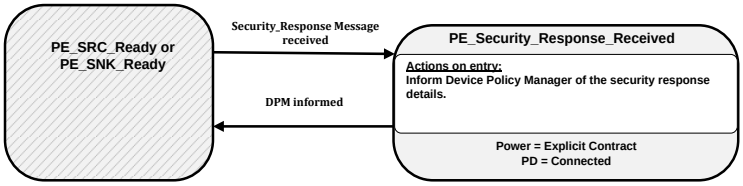
The Policy Engine shall transition back to either the ***PE_SRC_Ready, PE_SNK_Ready*** or ***PE_CBL_Ready*** state as appropriate (see Figure 8-56, Figure 8-57and Figure 8-110) when:

- The ***Security_Response*** Message has been successfully sent.

8.3.3.13.3 Security Response Received State Diagram

Figure 8-81 shows the state diagram for a Source or Sink on receiving a ***Security_Response*** Message. See also Section 6.5.8.

Figure 8-81 Security response received state diagram



8.3.3.13.3.1 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 8-56, Figure 8-57 and Figure 8-110) when:

- The Device Policy Manager has been informed.

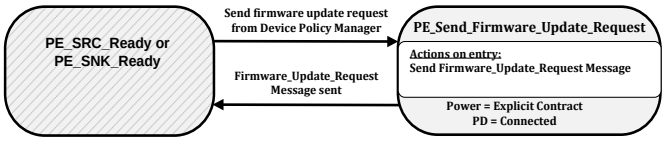
Microchip Ex. 1020, p. 399
Microchip v. Aptiv
IPR2024-00496

8.3.3.14 Firmware Update State Diagrams

8.3.3.14.1 Send Firmware Update Request State Diagram

Figure 8-82 shows the state diagram for a Source or Sink on receiving a request from the Device Policy Manager to send a firmware update request. See also Section 6.5.9.

Figure 8-82 Send firmware update request state diagram



8.3.3.14.1.1 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.

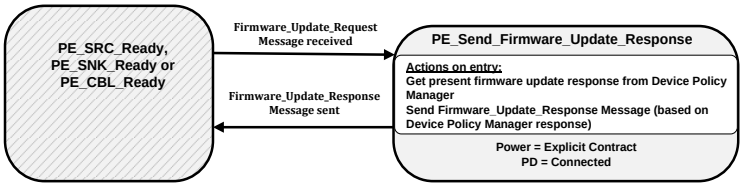
The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state as appropriate (see Figure 8-56 and Figure 8-57) when:

- The ***Firmware_Update_Request*** Message has been sent

8.3.3.14.2 Send Firmware Update Response State Diagram

Figure 8-83 shows the state diagram for a Source, Sink or Cable Plug on receiving a ***Firmware_Update_Request*** Message. See also Section 6.5.9.

Figure 8-83 Send firmware update response state diagram



8.3.3.14.2.1 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 8-56, Figure 8-57and Figure 8-110) when:

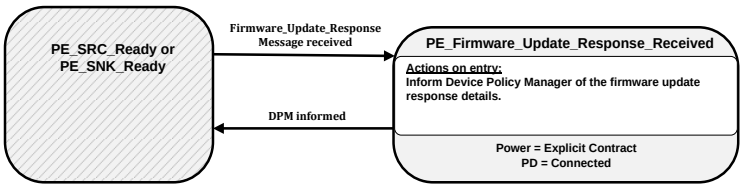
- The ***Firmware_Update_Response*** Message has been successfully sent.

Microchip Ex. 1020, p. 400
Microchip v. Aptiv
IPR2024-00496

8.3.3.14.3 Firmware Update Response Received State Diagram

Figure 8-84 shows the state diagram for a Source or Sink on receiving a ***Firmware_Update_Response*** Message. See also Section 6.5.9.

Figure 8-84 Firmware update response received state diagram



8.3.3.14.3.1 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 8-56, Figure 8-57 and Figure 8-110) when:

- The Device Policy Manager has been informed.

8.3.3.15 Dual-Role Port State Diagrams

Dual-Role Ports that combine Source and Sink capabilities 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.

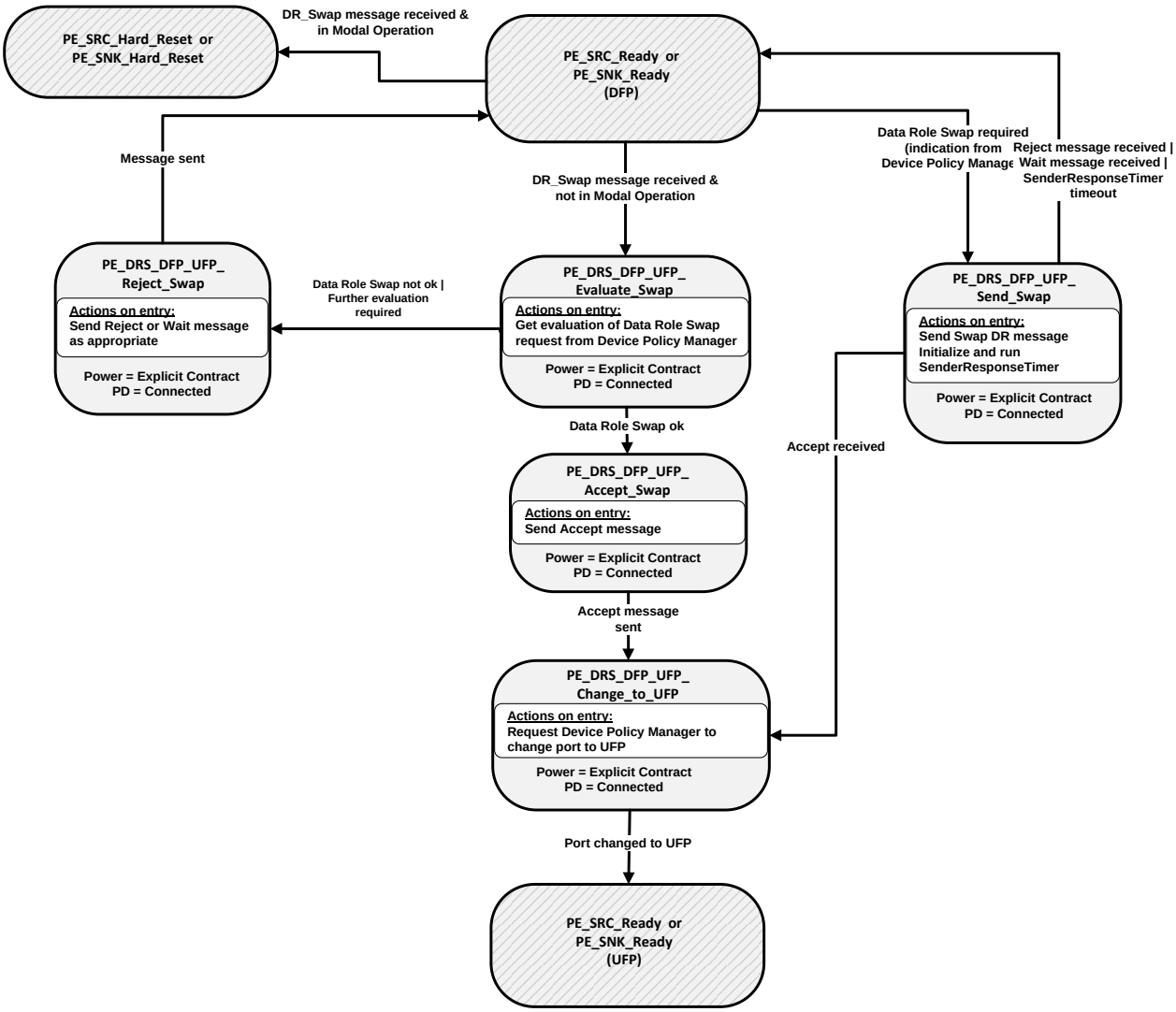
The State Diagrams in this section shall apply to every ***[USB Type-C 1.2]*** DRP.

Microchip Ex. 1020, p. 401
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.1 DFP to UFP Data Role Swap State Diagram

Figure 8-85 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 8-85: DFP to UFP Data Role Swap State Diagram



8.3.3.15.1.1 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

Microchip Ex. 1020, p. 402
Microchip v. Aptiv
IPR2024-00496

- 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.

8.3.3.15.1.2 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.

8.3.3.15.1.3 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.

8.3.3.15.1.4 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.

8.3.3.15.1.5 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.

8.3.3.15.1.6 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 6.3.12.3).

Microchip Ex. 1020, p. 403
Microchip v. Aptiv
IPR2024-00496

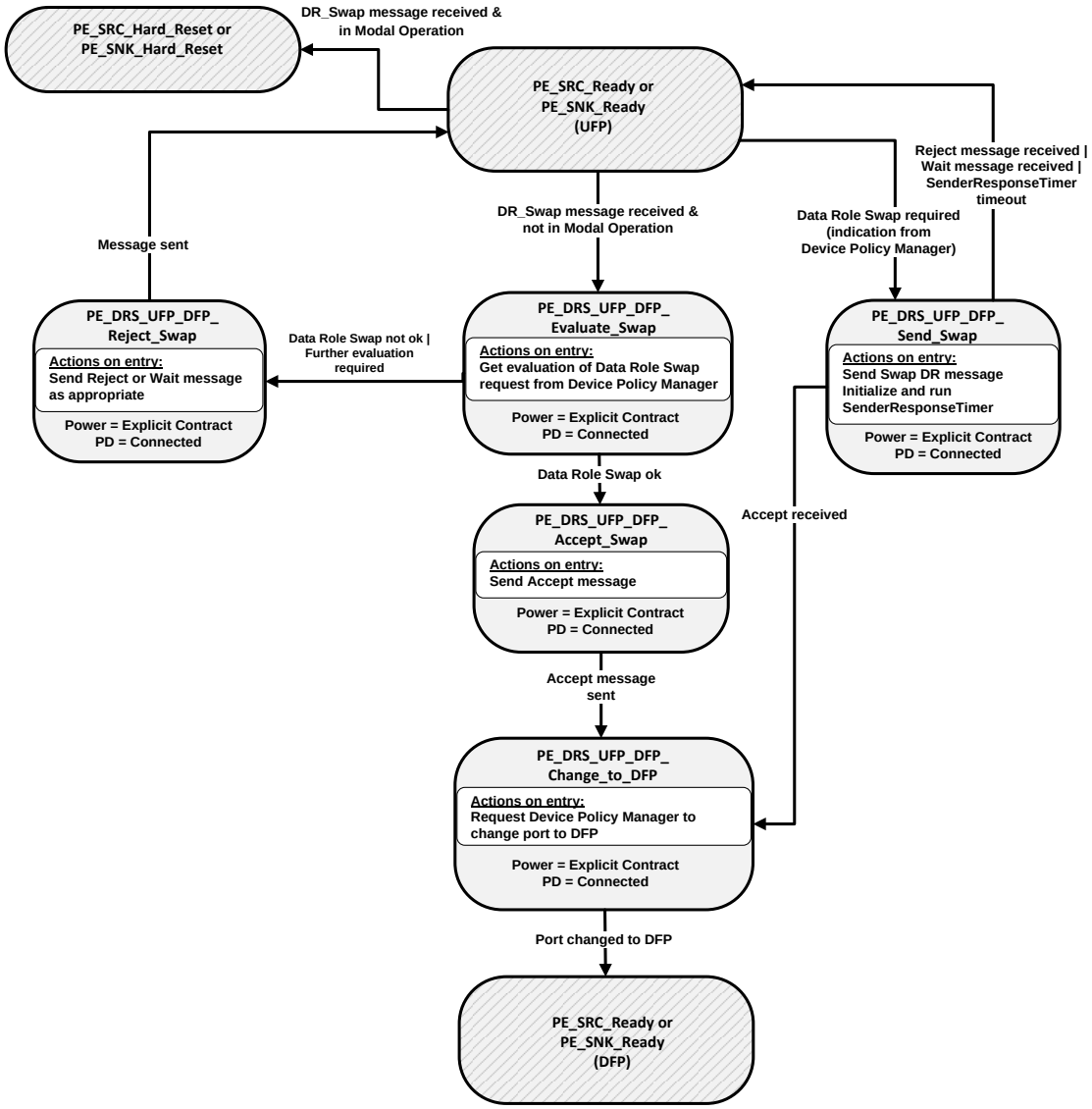
The Policy Engine shall continue as a DFP and shall transition to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state when:

- The ***Reject*** or ***Wait*** Message has been sent.

8.3.3.15.2 UFP to DFP Data Role Swap State Diagram

Figure 8-86 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 8-86: UFP to DFP Data Role Swap State Diagram



8.3.3.15.2.1 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:

Microchip Ex. 1020, p. 404
Microchip v. Aptiv
IPR2024-00496

- 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.

8.3.3.15.2.2 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.

8.3.3.15.2.3 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.

8.3.3.15.2.4 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.

8.3.3.15.2.5 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.

Microchip Ex. 1020, p. 405
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.2.6 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 6.3.12.3).

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.

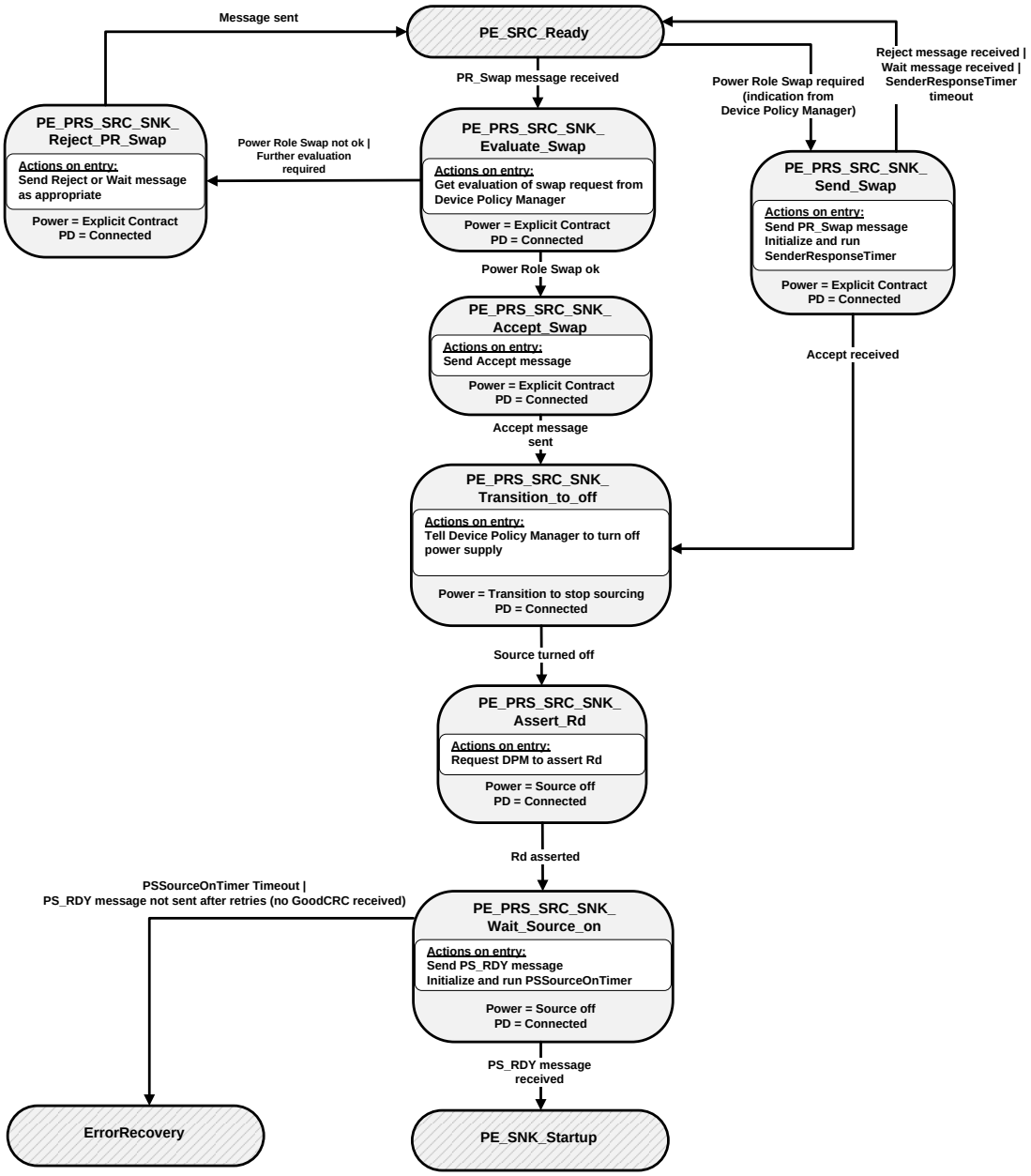
Microchip Ex. 1020, p. 406
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.3 Policy Engine in Source to Sink Power Role Swap State Diagram

Dual-Role Ports that combine Source and Sink capabilities 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 8-87 shows the additional state diagram required to perform a Power Role Swap from Source to Sink roles and the changes that shall be followed for error handling.

Figure 8-87: Dual-Role Port in Source to Sink Power Role Swap State Diagram



8.3.3.15.3.1 PE_SRC_Ready state

The Power Role Swap process shall start only from the ***PE_SRC_Ready*** state where power is stable.

Microchip Ex. 1020, p. 407
Microchip v. Aptiv
IPR2024-00496

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.

8.3.3.15.3.2 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.

8.3.3.15.3.3 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.

8.3.3.15.3.4 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.

8.3.3.15.3.5 PE_PRS_SRC_SNK_Assert_Rd

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.

8.3.3.15.3.6 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.

Microchip Ex. 1020, p. 408
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.3.7 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.

8.3.3.15.3.8 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 6.3.12.2).

The Policy Engine shall transition to the ***PE_SRC_Ready*** when:

- The ***Reject*** or ***Wait*** Message has been sent.

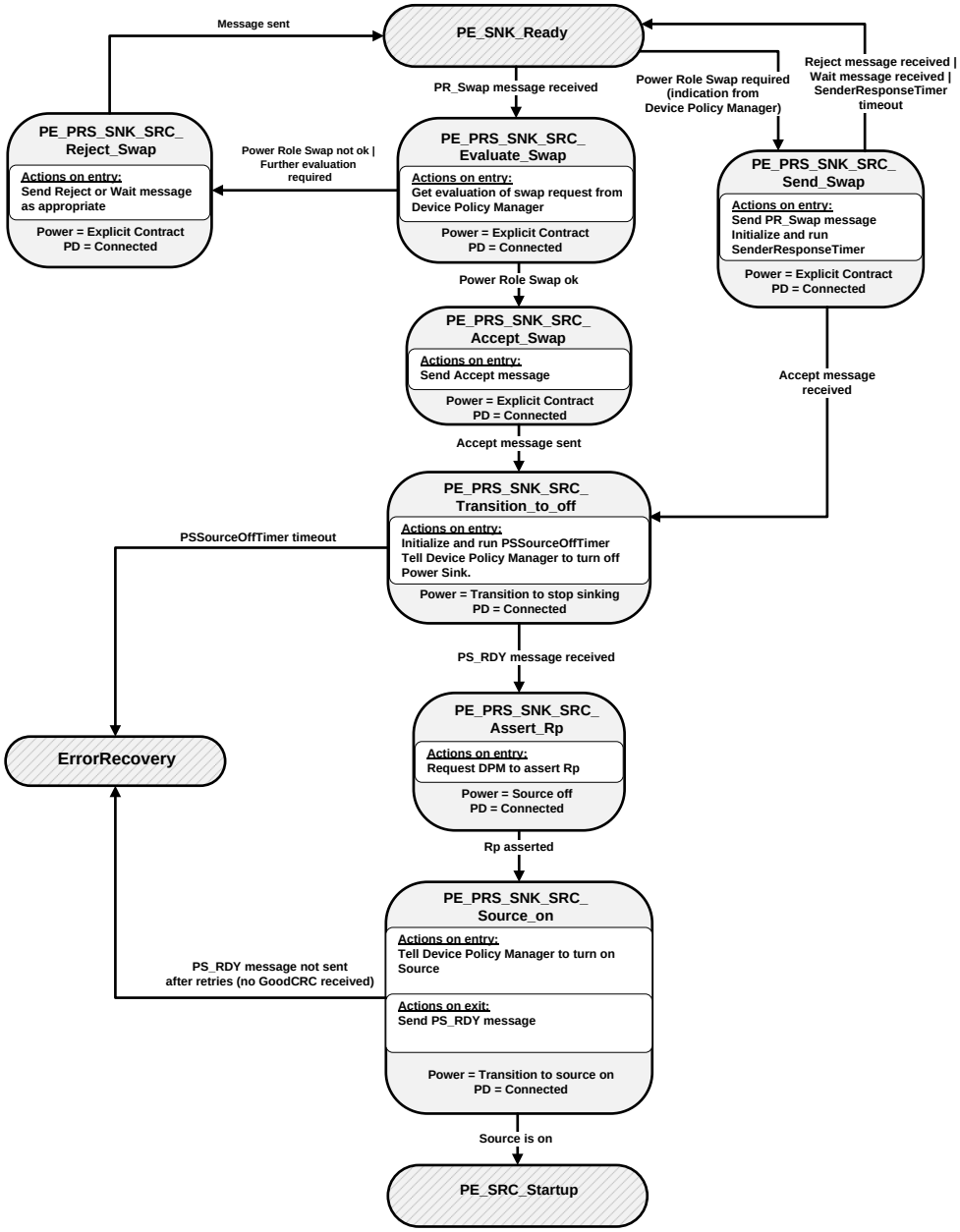
8.3.3.15.4 Policy Engine in Sink to Source Power Role Swap State Diagram

Dual-Role Ports that combine Sink and Source capabilities 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 8-88 shows the additional state diagram required to perform a Power Role Swap from Sink to Source roles and the changes that shall be followed for error handling.

Microchip Ex. 1020, p. 409
Microchip v. Aptiv
IPR2024-00496

Figure 8-88: Dual-role Port in Sink to Source Power Role Swap State Diagram



8.3.3.15.4.1 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.

Microchip Ex. 1020, p. 410
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.4.2 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.

8.3.3.15.4.3 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.

The Policy Engine shall transition to the ***PE_PRS_SNK_SRC_Transition_to_off*** state when:

- The ***Accept*** Message has been sent.

8.3.3.15.4.4 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_Source_on*** state when:

- A ***PS_RDY*** Message is received.

The Policy Engine shall transition to the ***PE_PRS_SNK_SRC_Assert_Rp*** state when:

- A ***PS_RDY*** Message is received.

8.3.3.15.4.5 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.

8.3.3.15.4.6 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 has been turned on.

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.

Microchip Ex. 1020, p. 411
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.4.7 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.

8.3.3.15.4.8 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 6.3.12.2).

The Policy Engine shall transition to the ***PE_SNK_Ready*** state when:

- The ***Reject*** or ***Wait*** Message has been sent.

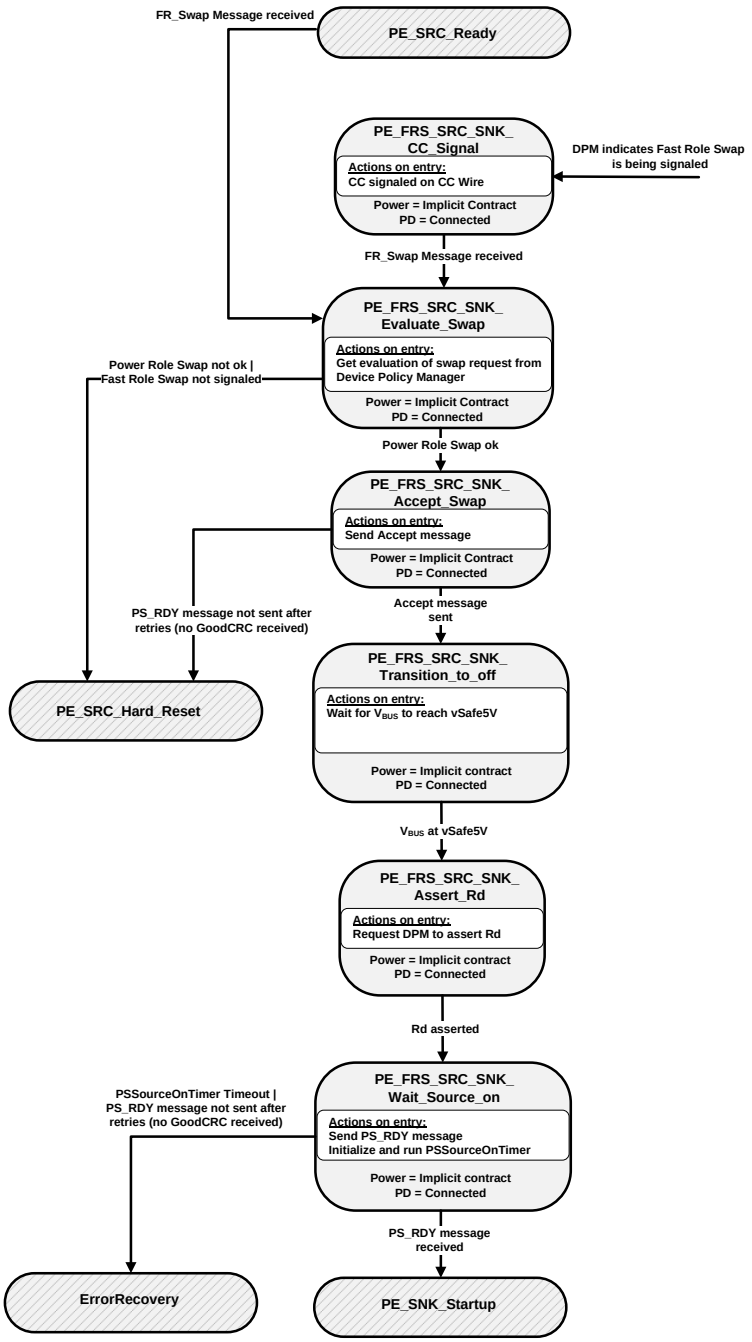
Microchip Ex. 1020, p. 412
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.5 Policy Engine in Source to Sink Fast Role Swap State Diagram

Dual-Role Ports that combine Source and Sink capabilities shall comprise Source and Sink Policy Engine state machines. In addition they should have the capability to do a Fast Role Swap from the ***PE_SRC_Ready*** state and shall return to USB Default Operation on a Hard Reset.

Figure 8-87 shows the additional state diagram required to perform a Fast Role Swap from Source to Sink roles and the changes that shall be followed for error handling.

Figure 8-89: Dual-Role Port in Source to Sink Fast Role Swap State Diagram



Microchip Ex. 1020, p. 413
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.5.1 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.

8.3.3.15.5.2 PE_FRS_SRC_SNK_CC_Signal state

The Policy Engine shall transition to the ***PE_FRS_SRC_SNK_CC_Signal*** state from any other state provided there is an Explicit Contract in place when:

- The Device Policy Manager indicates that Fast Role Swap signaling is being applied to the CC Wire.

In the ***PE_FRS_SRC_SNK_CC_Signal*** state the Policy Engine waits for an ***FR_Swap*** Message from the new Source.

The Policy Engine shall transition to the ***PE_FRS_SRC_SNK_Evaluate_Swap*** state when:

- An ***FR_Swap*** Message is received.

8.3.3.15.5.3 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 a Fast Role Swap can be made.

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 is ok.

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 ok or
- The Device Policy Manager indicates that a Fast Role Swap is not being signaled.

8.3.3.15.5.4 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_SNK_SRC_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.

8.3.3.15.5.5 PE_FRS_SRC_SNK_Transition_to_off state

On entry to the ***PE_FRS_SNK_SRC_Transition_to_off*** state the Policy Engine shall until V_{BUS} has discharged to ***vSafe5V***.

The Policy Engine shall transition to the ***PE_PRS_SRC_SNK_Assert_Rd*** state when:

- The Device Policy Manager indicates that V_{BUS} has discharged to ***vSafe5V***.

8.3.3.15.5.6 PE_FRS_SRC_SNK_Assert_Rd

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.

Microchip Ex. 1020, p. 414
Microchip v. Aptiv
IPR2024-00496

8.3.3.15.5.7 PE_FRS_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 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.

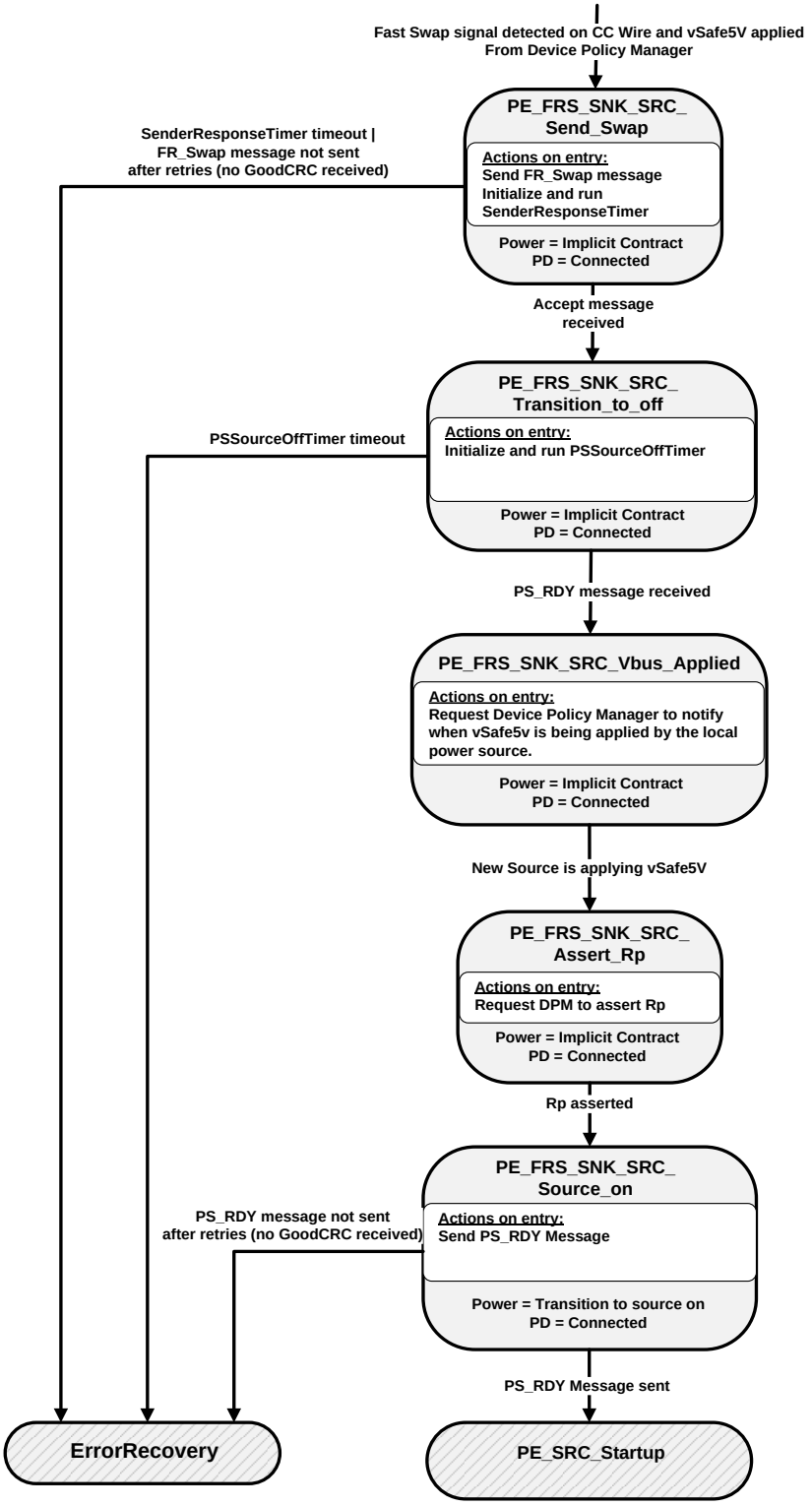
8.3.3.15.6 Policy Engine in Sink to Source Fast Role Swap State Diagram

Dual-Role Ports that combine Sink and Source capabilities 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 8-88 shows the additional state diagram required to perform a Fast Role Swap from Sink to Source roles and the changes that shall be followed for error handling.

Microchip Ex. 1020, p. 415
Microchip v. Aptiv
IPR2024-00496

Figure 8-90: Dual-role Port in Sink to Source Fast Role Swap State Diagram



8.3.3.15.6.1 PE_FRS_SNK_SRC_Send_Swap state

Microchip Ex. 1020, p. 416
Microchip v. Aptiv
IPR2024-00496

The Policy Engine shall transition to the ***PE_FRS_SNK_SRC_Send_Swap*** state from any other state provided there is an Explicit Contract in place when:

- The Device Policy Manager indicates that a Fast Role Swap signal has been detected on the CC Wire and ***vSafe5V*** has been applied to V_{BUS}.

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.

8.3.3.15.6.2 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.

8.3.3.15.6.3 PE_FRS_SNK_SRC_Vbus_Applied

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 V_{BUS} (see Section5.8.6.3). 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.

8.3.3.15.6.4 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.

8.3.3.15.6.5 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 (except if the exit is to send a ***Ping*** Message) 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:

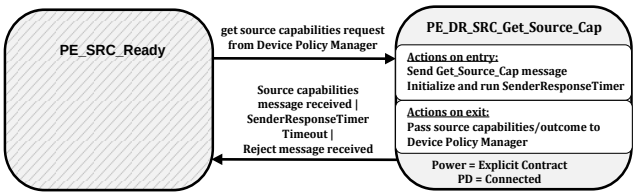
Microchip Ex. 1020, p. 417
Microchip v. Aptiv
IPR2024-00496

- 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.

8.3.3.15.7 Dual-Role (Source Port) Get Source Capabilities State Diagram

Figure 8-91 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 6.4.1.1.3.

Figure 8-91 Dual-Role (Source) Get Source Capabilities diagram



8.3.3.15.7.1 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 send a ***Get_Source_Cap*** Message and initialize and run 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).

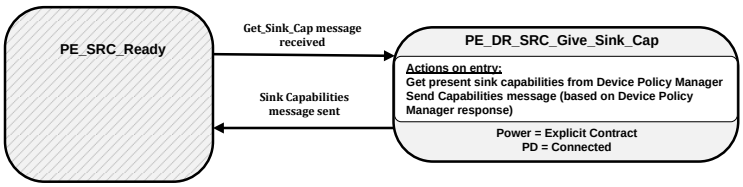
The Policy Engine shall transition back to the ***PE_SRC_Ready*** state (see Figure 8-56) when:

- A ***Source_Capabilities*** Message is received
- Or ***SenderResponseTimer*** times out
- Or a ***Reject*** Message is received

8.3.3.15.8 Dual-Role (Source Port) Give Sink Capabilities State Diagram

Figure 8-92 shows the state diagram for a Dual-Role device, presently operating as a Source, on receiving a ***Get_Sink_Cap*** Message. See also Section 6.4.1.1.3.

Figure 8-92 Dual-Role (Source) Give Sink Capabilities diagram



8.3.3.15.8.1 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 is received.

On entry to the ***PE_DR_SRC_Give_Sink_Cap*** state the Policy Engine shall request the present capabilities from the Device Policy Manager and then send a ***Sink_Capabilities*** Message based on these capabilities.

The Policy Engine shall transition back to the ***PE_SRC_Ready*** state (see Figure 8-56) when:

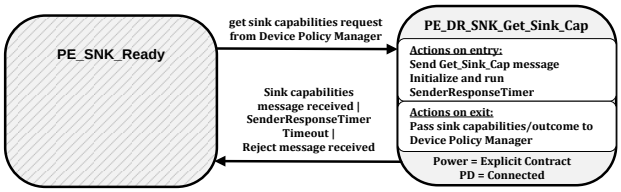
Microchip Ex. 1020, p. 418
Microchip v. Aptiv
IPR2024-00496

- The ***Sink_Capabilities*** Message has been successfully sent.

8.3.3.15.9 Dual-Role (Sink Port) Get Sink Capabilities State Diagram

Figure 8-93 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 6.4.1.1.3.

Figure 8-93 Dual-Role (Sink) Get Sink Capabilities State Diagram



8.3.3.15.9.1 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 send a ***Get_Sink_Cap*** Message and initialize and run the ***SenderResponseTimer***.

On exit from the ***PE_DR_SNK_Get_Sink_Cap*** state the Policy Engine shall inform the Device Policy Manager of the outcome (capabilities or response timeout).

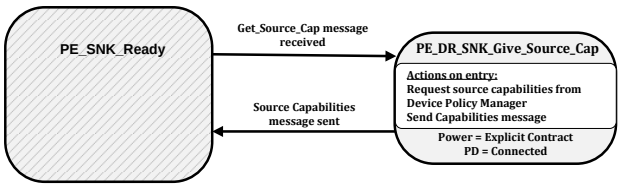
The Policy Engine shall transition back to the ***PE_SNK_Ready*** state (see Figure 8-57) when:

- A ***Source_Capabilities*** Message is received
- Or ***SenderResponseTimer*** times out
- Or a ***Reject*** Message is received

8.3.3.15.10 Dual-Role (Sink Port) Give Source Capabilities State Diagram

Figure 8-94 shows the state diagram for a Dual-Role device, presently operating as a Sink, on receiving a ***Get_Source_Cap*** Message. See also Section 6.4.1.1.3.

Figure 8-94 Dual-Role (Sink) Give Source Capabilities State Diagram



8.3.3.15.10.1 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 present capabilities from the Device Policy Manager and then send a ***Source_Capabilities*** Message based on these capabilities.

The Policy Engine shall transition back to the ***PE_SNK_Ready*** state (see Figure 8-57) when:

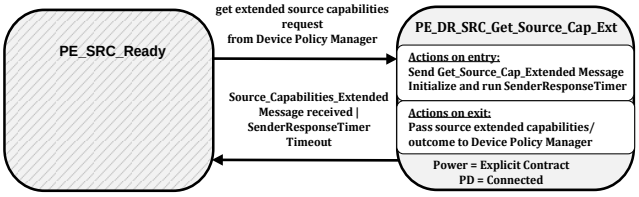
Microchip Ex. 1020, p. 419
Microchip v. Aptiv
IPR2024-00496

- The ***Source_Capabilities*** Message has been successfully sent.

8.3.3.15.11 Dual-Role (Source Port) Get Source Capabilities Extended State Diagram

Figure 8-95 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 extended Source capabilities. See also Section 6.5.1.

Figure 8-95 Dual-Role (Source) Get Source Capabilities Extended state diagram



8.3.3.15.11.1 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).

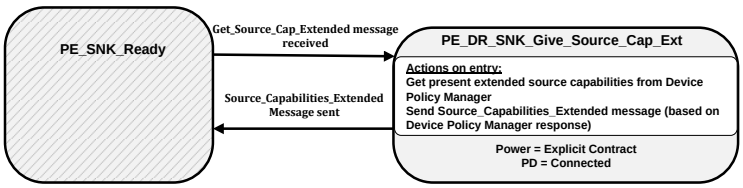
The Policy Engine shall transition back to the ***PE_SRC_Ready*** state (see Figure 8-56) when:

- A ***Source_Capabilities_Extended*** Message is received
- Or ***SenderResponseTimer*** times out

8.3.3.15.12 Dual-Role (Sink Port) Give Source Capabilities Extended State Diagram

Figure 8-96 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 6.5.1.

Figure 8-96 Dual-Role (Source) Give Sink Capabilities diagram



8.3.3.15.12.1 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 8-57) when:

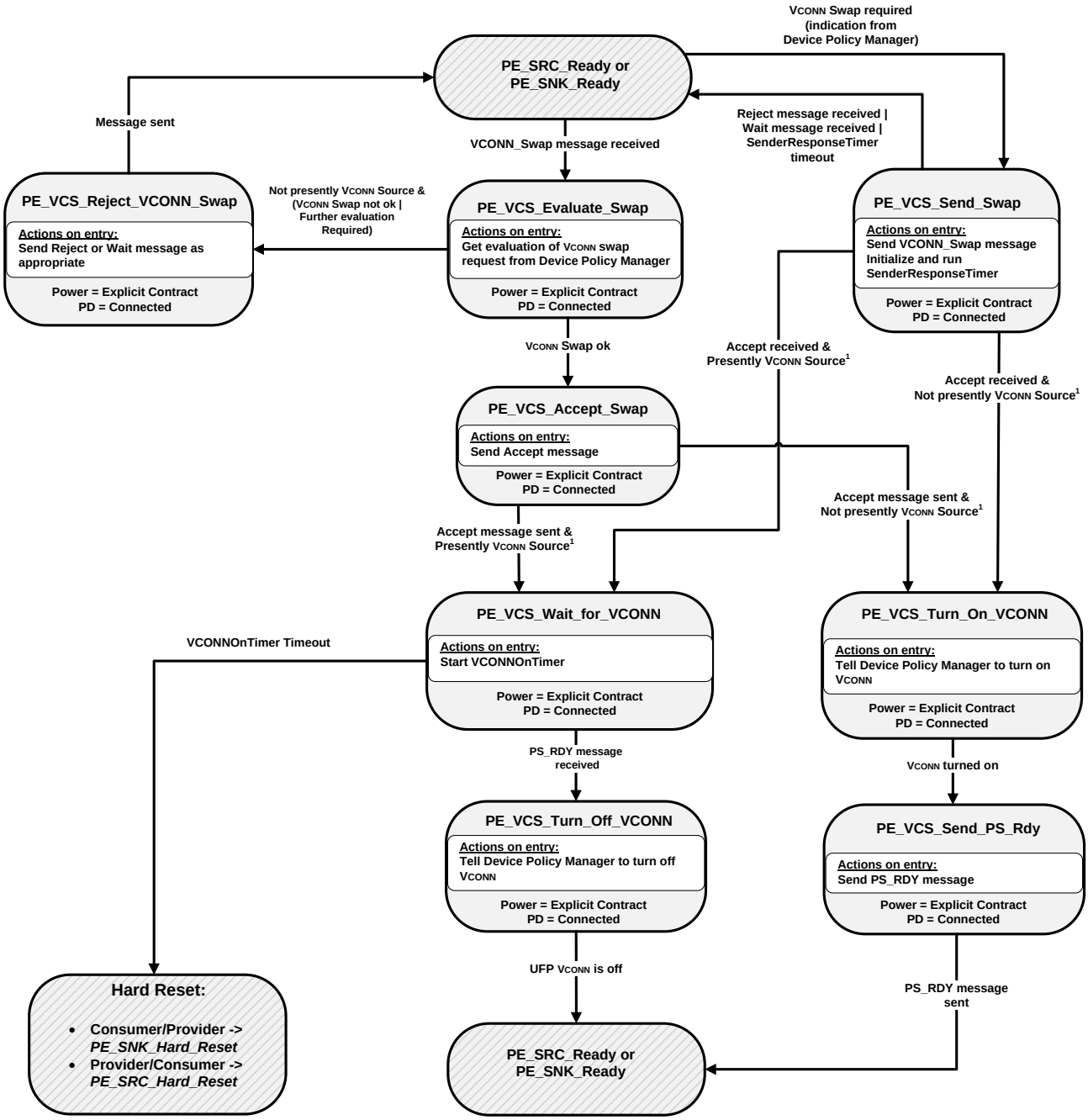
- The ***Source_Capabilities_Extended*** Message has been successfully sent.

Microchip Ex. 1020, p. 420
Microchip v. Aptiv
IPR2024-00496

8.3.3.16 VCONN Swap State Diagram

The State Diagram in this section shall apply to Ports that supply VCONN. Figure 8-97 shows the state operation for a Port on sending or receiving a VCONN Swap request.

Figure 8-97 VCONN Swap State Diagram



Microchip Ex. 1020, p. 421
Microchip v. Aptiv
IPR2024-00496

[1] A Port is presently the VCONN Source if it has the responsibility for supplying VCONN even if VCONN has been turned off.

8.3.3.16.1.1 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
- DFP current has VCONN turned on.

The Policy Engine shall transition to the ***PE_VCS_Turn_On_VCONN*** state when:

- An ***Accept*** Message is received and
- DFP current has VCONN turned off.

The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state for a DFP when:

- A ***Reject*** Message is received or
- A ***Wait*** Message is received or
- The ***SenderResponseTimer*** times out.

8.3.3.16.1.2 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. Note: Ports that are presently the VCONN Source must always accept a VCONN swap request (see Section 6.3.11).

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.

8.3.3.16.1.3 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 UFP'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 UFP's VCONN is off.

8.3.3.16.1.4 PE_VCS_Reject_Swap state

On entry to the ***PE_VCS_Reject_Swap*** state the Policy Engine shall request the Protocol Layer to send:

Microchip Ex. 1020, p. 422
Microchip v. Aptiv
IPR2024-00496

- 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 DFP will send a ***VCONN_Swap*** Message at a later time.

The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state when:

- The ***Reject*** or ***Wait*** Message has been sent.

8.3.3.16.1.5 PE_VCS_UFP_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.

8.3.3.16.1.6 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*** or ***PE_SNK_Ready*** state for a UFP when:

- The UFP's VCONN is off.

8.3.3.16.1.7 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 UFP's VCONN is on.

8.3.3.16.1.8 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*** or ***PE_SNK_Ready*** state for a UFP when:

- The ***PS_RDY*** Message has been sent.

8.3.3.17 Initiator Structured VDM State Diagrams

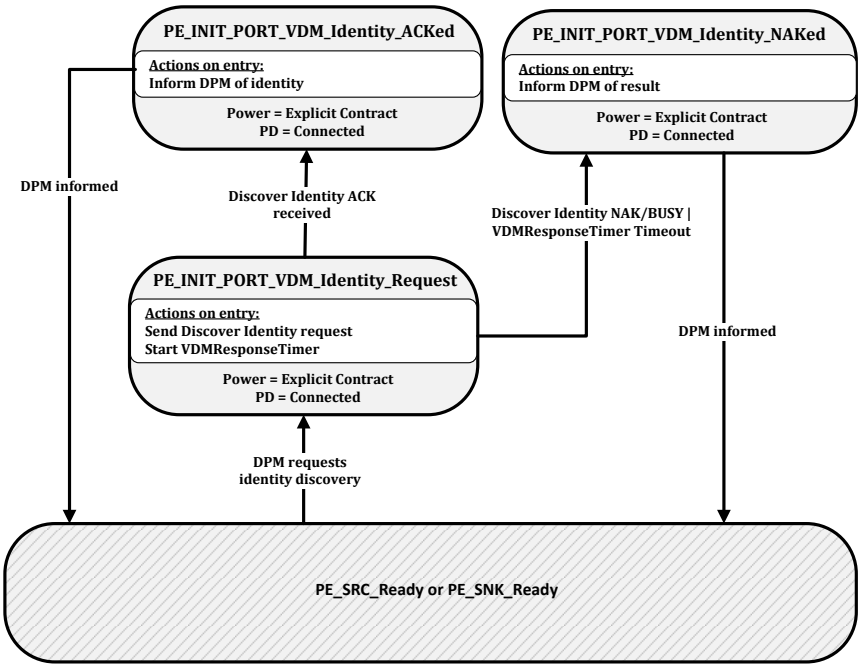
The State Diagrams in this section shall apply to all Initiators.

8.3.3.17.1 Initiator Structured VDM Discover Identity State Diagram

Figure 8-98 shows the state diagram for an Initiator when discovering the identity of its Port Partner or Cable Plug.

Microchip Ex. 1020, p. 423
Microchip v. Aptiv
IPR2024-00496

Figure 8-98 Initiator to Port VDM Discover Identity State Diagram



8.3.3.17.1.1 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.

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

8.3.3.17.1.2 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*** or ***PE_SNK_Ready*** state when:

- The Device Policy Manager has been informed.

8.3.3.17.1.3 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*** or ***PE_SNK_Ready*** state when:

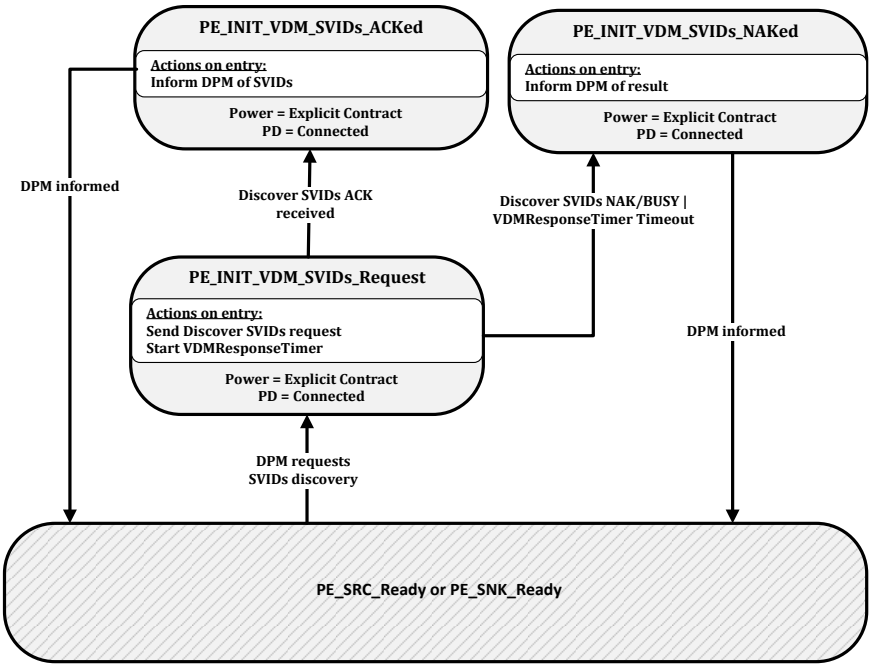
- The Device Policy Manager has been informed.

Microchip Ex. 1020, p. 424
Microchip v. Aptiv
IPR2024-00496

8.3.3.17.2 Initiator Structured VDM Discover SVIDs State Diagram

Figure 8-99 shows the state diagram for an Initiator when discovering SVIDs of its Port Partner or Cable Plug.

Figure 8-99 Initiator VDM Discover SVIDs State Diagram



8.3.3.17.2.1 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.

8.3.3.17.2.2 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.

8.3.3.17.2.3 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).

Microchip Ex. 1020, p. 425
Microchip v. Aptiv
IPR2024-00496

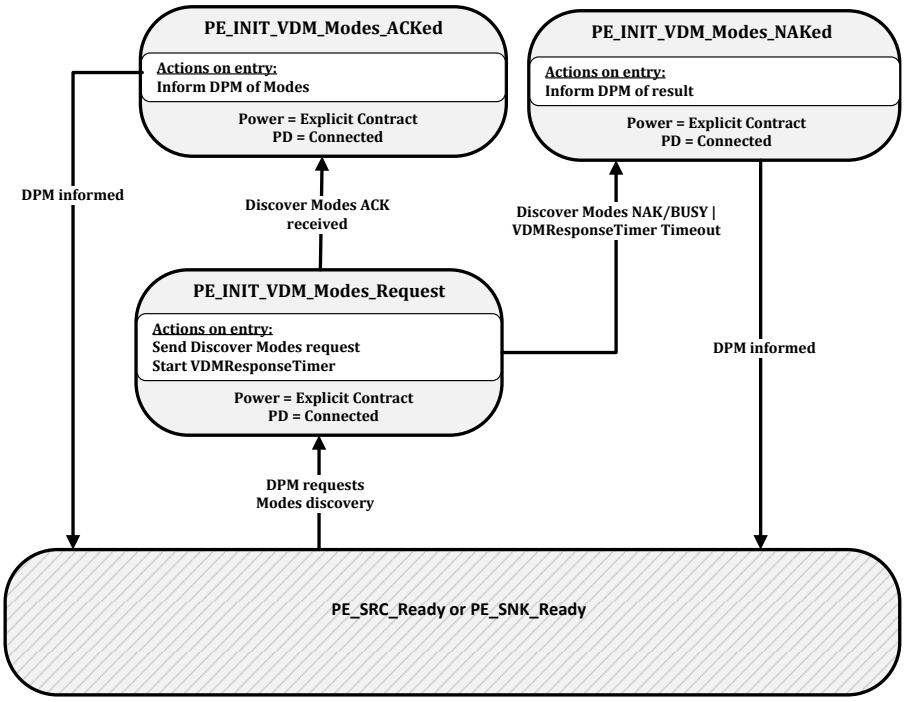
The Policy Engine shall transition to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state when:

- The Device Policy Manager has been informed.

8.3.3.17.3 Initiator Structured VDM Discover Modes State Diagram

Figure 8-100 shows the state diagram for an Initiator when discovering Modes of its Port Partner or Cable Plug.

Figure 8-100 Initiator VDM Discover Modes State Diagram



8.3.3.17.3.1 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.

8.3.3.17.3.2 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.

Microchip Ex. 1020, p. 426
Microchip v. Aptiv
IPR2024-00496

8.3.3.17.3.3 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).

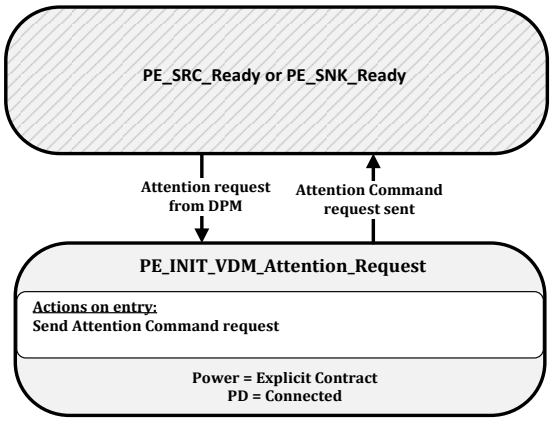
The Policy Engine shall transition to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state for a DFP when:

- The Device Policy Manager has been informed.

8.3.3.17.4 Initiator Structured VDM Attention State Diagram

Figure 8-105 shows the state diagram for an Initiator when sending an ***Attention*** Command request.

Figure 8-101 Initiator VDM Attention State Diagram



8.3.3.17.4.1 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.

8.3.3.18 Responder Structured VDM State Diagrams

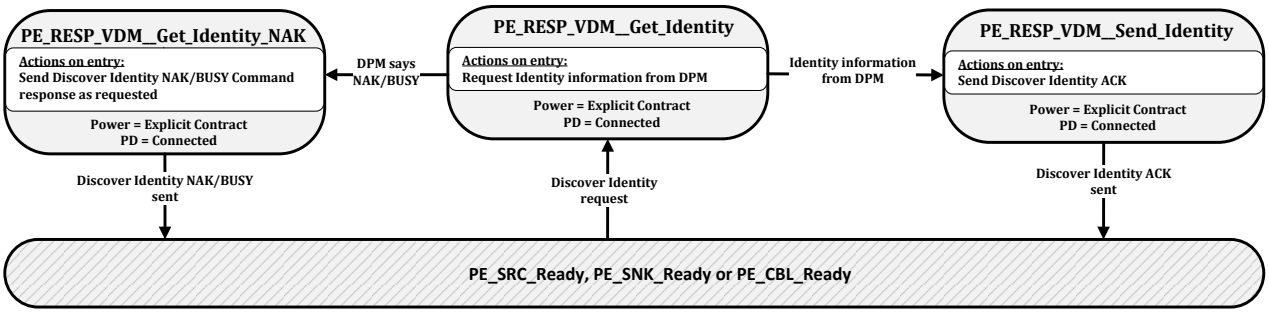
The State Diagrams in this section shall apply to all Responders that are not Cable Plugs.

8.3.3.18.1 Responder Structured VDM Discover Identity State Diagram

Figure 8-102 shows the state diagram for a Responder receiving a ***Discover Identity*** Command request.

Microchip Ex. 1020, p. 427
Microchip v. Aptiv
IPR2024-00496

Figure 8-102 Responder Structured VDM Discover Identity State Diagram



8.3.3.18.1.1 PE_RESP_VDM_Get_Identity state

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.

8.3.3.18.1.2 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.

8.3.3.18.1.3 PE_RESP_VDM_Get_Identity_NAK

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:

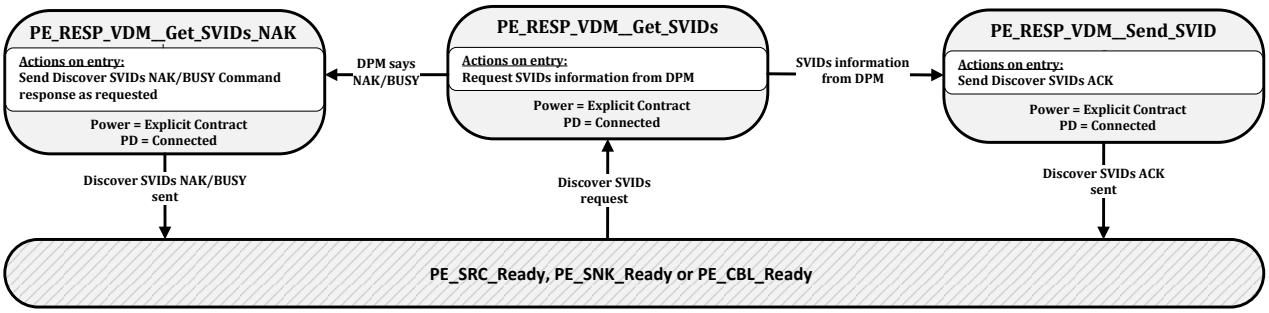
- The Structured VDM ***Discover Identity*** NAK or BUSY Command response has been sent.

8.3.3.18.2 Responder Structured VDM Discover SVIDs State Diagram

Figure 8-103 shows the state diagram for a Responder when receiving a ***Discover SVIDs*** Command.

Microchip Ex. 1020, p. 428
Microchip v. Aptiv
IPR2024-00496

Figure 8-103 Responder Structured VDM Discover SVIDs State Diagram



8.3.3.18.2.1 PE_RESP_VDM_Get_SVIDs state

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.

8.3.3.18.2.2 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.

8.3.3.18.2.3 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*** or ***PE_SNK_Ready*** state when:

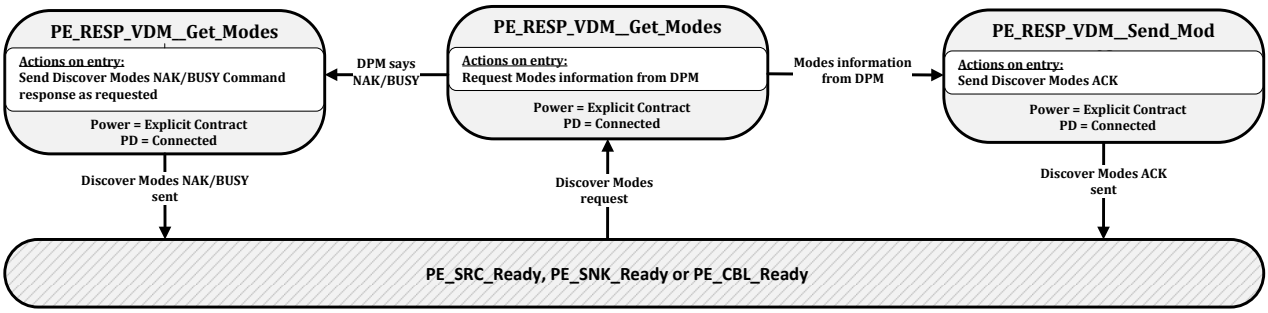
- The Structured VDM ***Discover SVIDs*** NAK or BUSY Command response has been sent.

8.3.3.18.3 Responder Structured VDM Discover Modes State Diagram

Figure 8-104 shows the state diagram for a Responder on receiving a ***Discover Modes*** Command.

Microchip Ex. 1020, p. 429
Microchip v. Aptiv
IPR2024-00496

Figure 8-104 Responder Structured VDM Discover Modes State Diagram



8.3.3.18.3.1 PE_RESP_VDM_Get_Modes state

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.

8.3.3.18.3.2 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.

8.3.3.18.3.3 PE_RESP_VDM_Get_Modes_NAK

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*** or ***PE_SNK_Ready*** state when:

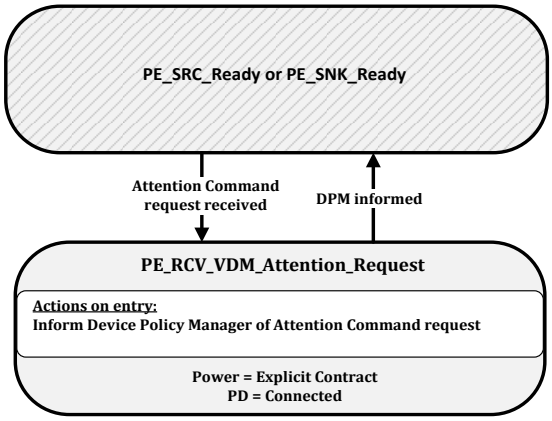
- The Structured VDM ***Discover Modes*** NAK or BUSY Command response has been sent.

8.3.3.18.4 Receiving a Structured VDM Attention State Diagram

Figure 8-105 shows the state diagram when receiving an ***Attention*** Command request.

Microchip Ex. 1020, p. 430
Microchip v. Aptiv
IPR2024-00496

Figure 8-105 Receiving a Structured VDM Attention State Diagram



8.3.3.18.4.1 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.

8.3.3.19 DFP Structured VDM State Diagrams

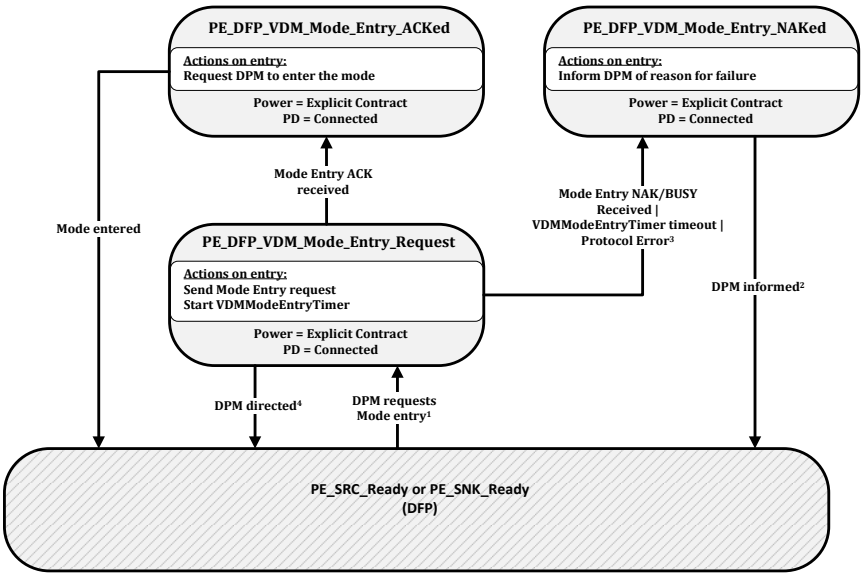
The State Diagrams in this section shall apply to all DFPS that support Structured VDMs.

8.3.3.19.1 DFP Structured VDM Mode Entry State Diagram

Figure 8-106 shows the state operation for a DFP when entering a Mode.

Microchip Ex. 1020, p. 431
Microchip v. Aptiv
IPR2024-00496

Figure 8-106 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.

[4] A DPM directed transition is an Interruption to the Mode Entry AMS by the DPM itself. The Device Policy Manager shall have placed the system into USB Safe State before issuing this request.

8.3.3.19.1.1 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.

The Policy Engine shall transition back to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state for a DFP when:

- Directed by the DPM to start another AMS.

8.3.3.19.1.2 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.

Microchip Ex. 1020, p. 432
Microchip v. Aptiv
IPR2024-00496

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.

8.3.3.19.1.3 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).

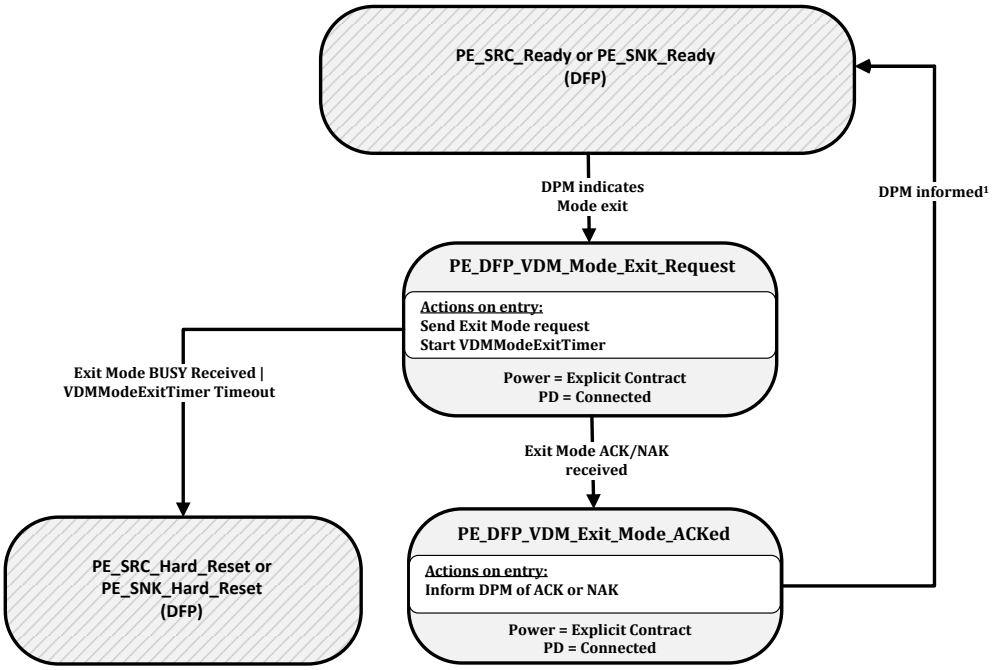
The Policy Engine shall transition to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state for a DFP when:

- The Device Policy Manager has been informed.

8.3.3.19.2 DFP Structured VDM Mode Exit State Diagram

Figure 8-107 shows the state diagram for a DFP when exiting a Mode.

Figure 8-107 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.

8.3.3.19.2.1 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_Exit_ACKed*** state when:

- A Structured VDM ***Exit Mode*** ACK or NAK Command response is received.

Microchip Ex. 1020, p. 433
Microchip v. Aptiv
IPR2024-00496

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.

8.3.3.19.2.2 PE_DFP_VDM_DFP_Mode_Exit_ACKed state

On Exit to the ***PE_DFP_VDM_Mode_Exit_ACKed*** state the Policy Engine shall request the Device Policy Manager to exit the Mode.

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.

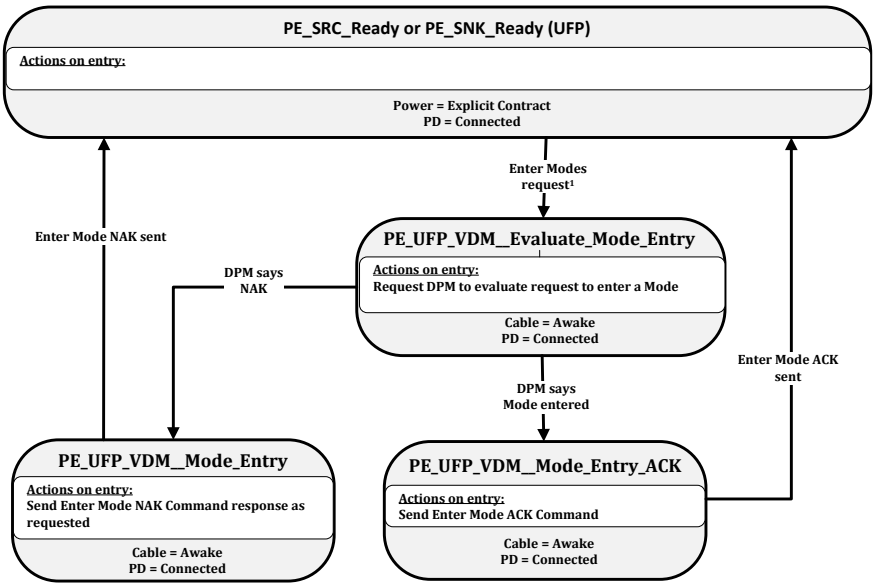
8.3.3.20 UFP Structured VDM State Diagrams

The State Diagrams in this section shall apply to all UFPS that support Structured VDMs.

8.3.3.20.1 UFP Structured VDM Enter Mode State Diagram

Figure 8-108 shows the state diagram for a UFP in response to an ***Enter Mode*** Command.

Figure 8-108 UFP Structured VDM Enter Mode State Diagram



[1] The UFP is required to be in USB operation or USB Safe State at this point.

8.3.3.20.1.1 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:

Microchip Ex. 1020, p. 434
Microchip v. Aptiv
IPR2024-00496

- 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.

8.3.3.20.1.2 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.

8.3.3.20.1.3 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.

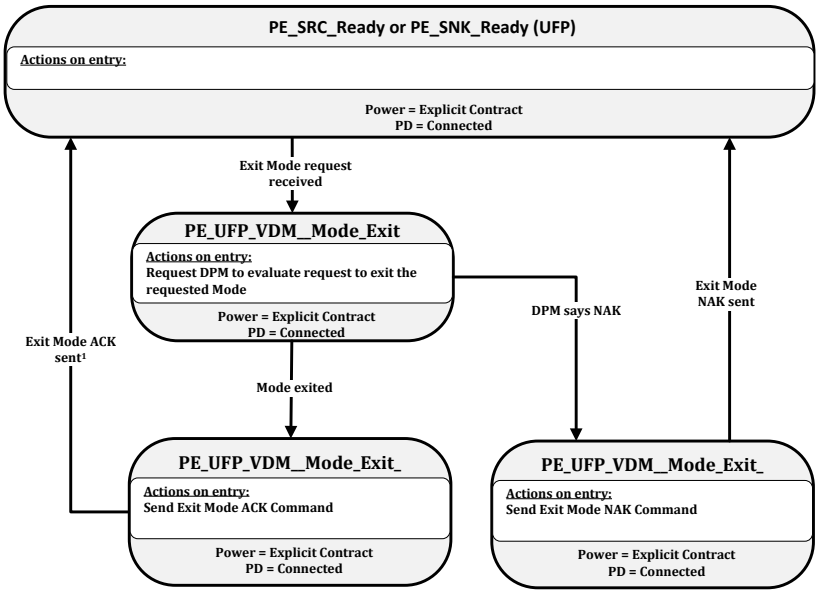
The Policy Engine shall transition to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state for a UFP when:

- The Structured VDM ***Enter Mode*** NAK Command response has been sent.

8.3.3.20.2 UFP Structured VDM Exit Mode State Diagram

Figure 8-109 shows the state diagram for a UFP in response to an ***Exit Mode*** Command.

Figure 8-109 UFP Structured VDM Exit Mode State Diagram



[1] The UFP is required to be in USB operation or USB Safe State at this point.

8.3.3.20.2.1 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.

Microchip Ex. 1020, p. 435
Microchip v. Aptiv
IPR2024-00496

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 Manger 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.

8.3.3.20.2.2 PE_CBL_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.

8.3.3.20.2.3 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.

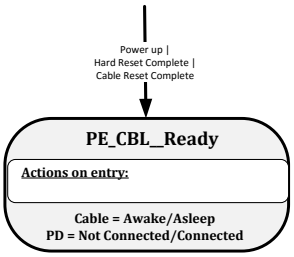
8.3.3.21 Cable Plug Specific State Diagrams

The State Diagrams in this section shall apply to all Cable Plugs that support Structured VDMs.

8.3.3.21.1 Cable Plug Cable Ready State Diagram

Figure 8-110 shows the Cable Ready state diagram for a Cable Plug.

Figure 8-110 Cable Ready VDM State Diagram



8.3.3.21.1.1 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.

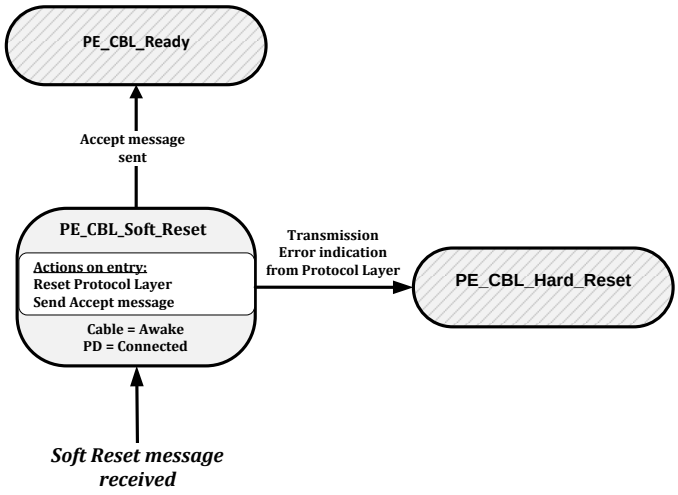
8.3.3.21.2 Soft/Hard/Cable Reset

8.3.3.21.2.1 Cable Plug Soft Reset State Diagram

Figure 8-111 shows the Cable Plug state diagram on reception of a ***Soft_Reset*** Message.

Microchip Ex. 1020, p. 436
Microchip v. Aptiv
IPR2024-00496

Figure 8-111 Cable Plug Soft Reset State Diagram



8.3.3.21.2.1.1 PE_CBL_Soft_Reset state

The ***PE_CBL_Soft_Reset*** state shall be entered from any state when a 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.

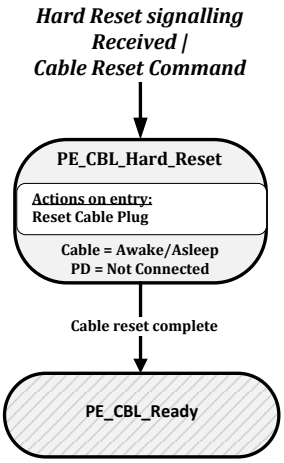
The Policy Engine shall transition to the ***PE_CBL_Hard_Reset*** state when:

- The Protocol Layer indicates that a transmission error has occurred.

8.3.3.21.2 Cable Plug Hard Reset State Diagram

Figure 8-112 shows the Cable Plug state diagram for a Hard Reset or Cable Reset.

Figure 8-112 Cable Plug Hard Reset State Diagram



Microchip Ex. 1020, p. 437
Microchip v. Aptiv
IPR2024-00496

8.3.3.21.2.2.1 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).

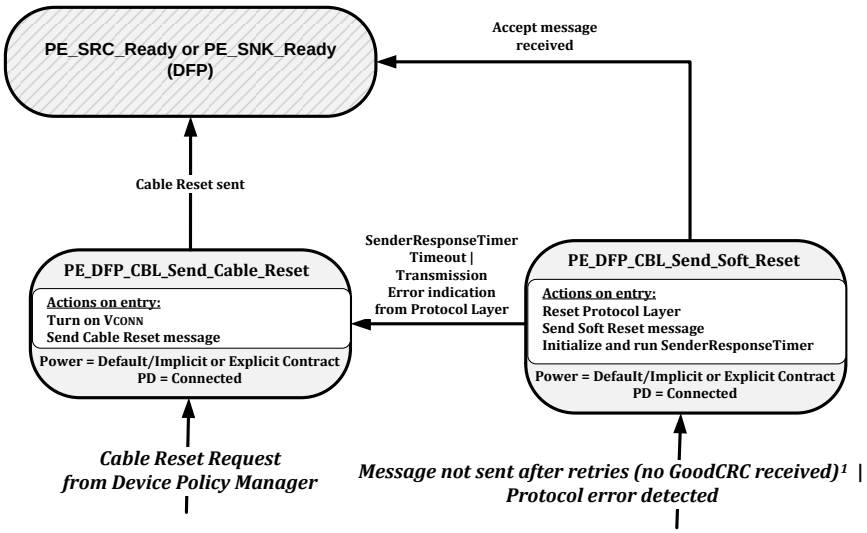
The Policy Engine shall transition to the ***PE_CBL_Ready*** state when:

- The Cable Plug reset is complete.

8.3.3.21.2.3 DFP Soft Reset or Cable Reset of a Cable Plug State Diagram

Figure 8-113 below shows the state diagram for the Policy Engine in a DFP when performing a Soft Reset or Cable Reset of a Cable Plug. The following sections describe operation in each of the states.

Figure 8-113 DFP Soft Reset or Cable Reset of a Cable Plug State Diagram



[1] Excludes the ***Soft_Reset*** Message itself.

8.3.3.21.2.3.1 PE_DFP_CBL_Send_Soft_Reset state

The ***PE_DFP_CBL_Send_Soft_Reset*** state shall be entered from any state when a Protocol Error is detected by the Protocol Layer (see Section 6.8.1) when a Message has not been sent after retries while communicating with a Cable Plug or whenever the Device Policy Manager directs a Soft Reset.

On entry to the ***PE_DFP_CBL_Send_Soft_Reset*** state the Policy Engine shall request the Protocol Layer to perform a Soft Reset, then shall send a ***Soft_Reset*** Message 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 DFP's Power Role, when:

- An ***Accept*** Message has been received.

The Policy Engine shall transition to the ***PE_DFP_CBL_Send_Cable_Reset*** state when:

- A ***SenderResponseTimer*** timeout occurs
- Or the Protocol Layer indicates that a transmission error has occurred.

Microchip Ex. 1020, p. 438
Microchip v. Aptiv
IPR2024-00496

8.3.3.21.2.3.2 PE_DFP_CBL_Send_Cable_Reset state

The ***PE_DFP_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_CBL_Send_Cable_Reset*** state the Policy Engine shall request the Protocol Layer to send ***Cable Reset*** Signaling.

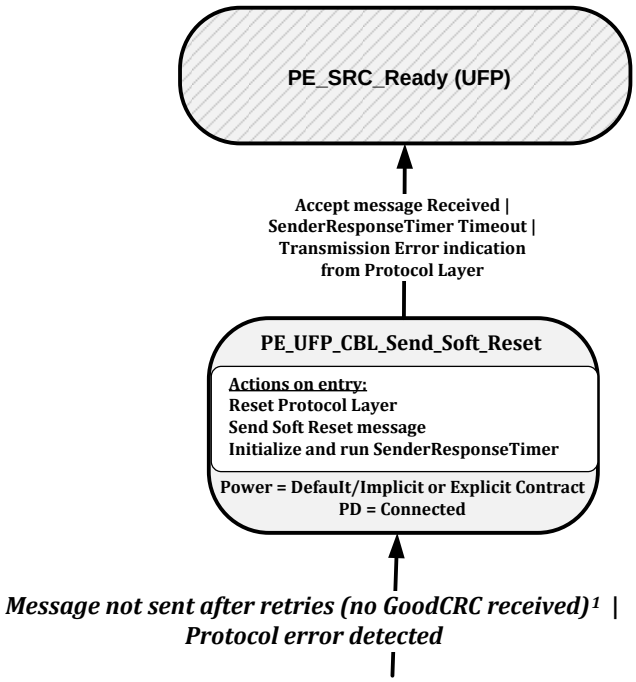
The Policy Engine shall transition to either the ***PE_SRC_Ready*** or ***PE_SNK_Ready*** state, depending on the DFP's Power Role, when:

- ***Cable Reset*** Signaling has been sent.

8.3.3.21.2.4 UFP Source Soft Reset of a Cable Plug State Diagram

Figure 8-114 below shows the state diagram for the Policy Engine in a UFP Source, prior to an Explicit Contract, when performing a Soft Reset of a Cable Plug. The following sections describe operation in each of the states.

Figure 8-114 UFP Source Soft Reset of a Cable Plug State Diagram



[1] Excludes the ***Soft_Reset*** Message itself.

8.3.3.21.2.4.1 PE_UFP_CBL_Send_Soft_Reset state

The ***PE_UFP_CBL_Send_Soft_Reset*** state shall be entered from any state when a Protocol Error is detected by the Protocol Layer, when a Message has not been sent after retries while communicating with a Cable Plug or whenever the Device Policy Manager directs a Soft Reset.

Note that there are corner cases that are not shown in the defined state diagrams that could be handled without generating a Protocol Error.

On entry to the ***PE_UFP_CBL_Send_Soft_Reset*** state the Policy Engine shall request the Protocol Layer to perform a Soft Reset, then shall send a ***Soft_Reset*** Message to the Cable Plug, and initialize and run the ***SenderResponseTimer***.

Microchip Ex. 1020, p. 439
Microchip v. Aptiv
IPR2024-00496

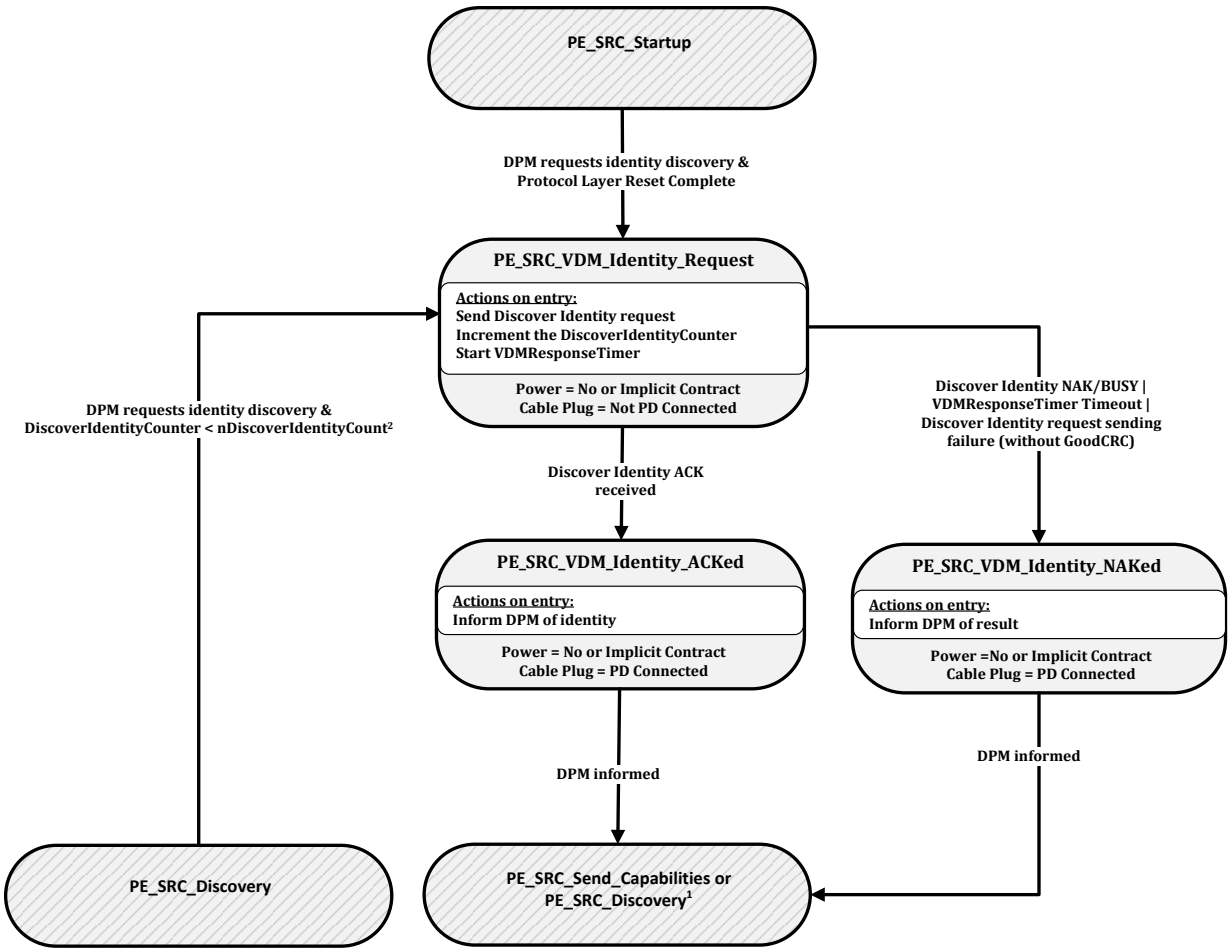
The Policy Engine shall transition to the ***PE_SRC_Ready*** state when:

- An ***Accept*** Message has been received
- Or a ***SenderResponseTimer*** timeout occurs
- Or the Protocol Layer indicates that a transmission error has occurred

8.3.3.21.3 Source Startup Structured VDM Discover Identity of a Cable Plug State Diagram

Figure 8-115 shows the state diagram for Source discovery of identity information from a Cable Plug during the startup sequence.

Figure 8-115 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.

8.3.3.21.3.1 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:

Microchip Ex. 1020, p. 440
Microchip v. Aptiv
IPR2024-00496

- 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 8-115 and shall not be initialized on re-entry to ***PE_SRC_Discovery***.

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).

8.3.3.21.3.2 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.

8.3.3.21.3.3 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.

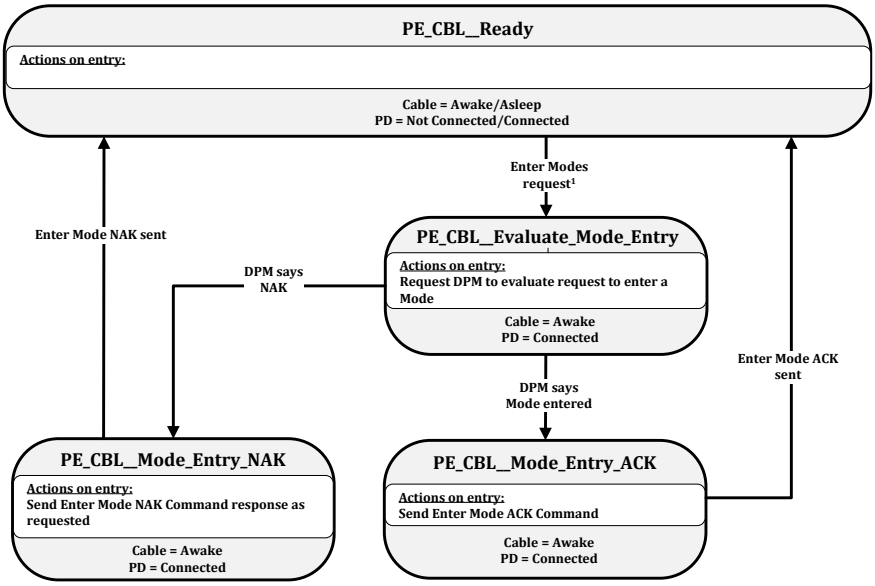
8.3.3.21.4 Cable Plug Mode Entry/Exit

8.3.3.21.4.1 Cable Plug Structured VDM Enter Mode State Diagram

Figure 8-116 shows the state diagram for a Cable Plug in response to an ***Enter Mode*** Command.

Microchip Ex. 1020, p. 441
Microchip v. Aptiv
IPR2024-00496

Figure 8-116 Cable Plug Structured VDM Enter Mode State Diagram



[1] The Cable is required to be in USB operation or USB Safe State at this point.

8.3.3.21.4.1.1 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.

8.3.3.21.4.1.2 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.

8.3.3.21.4.1.3 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:

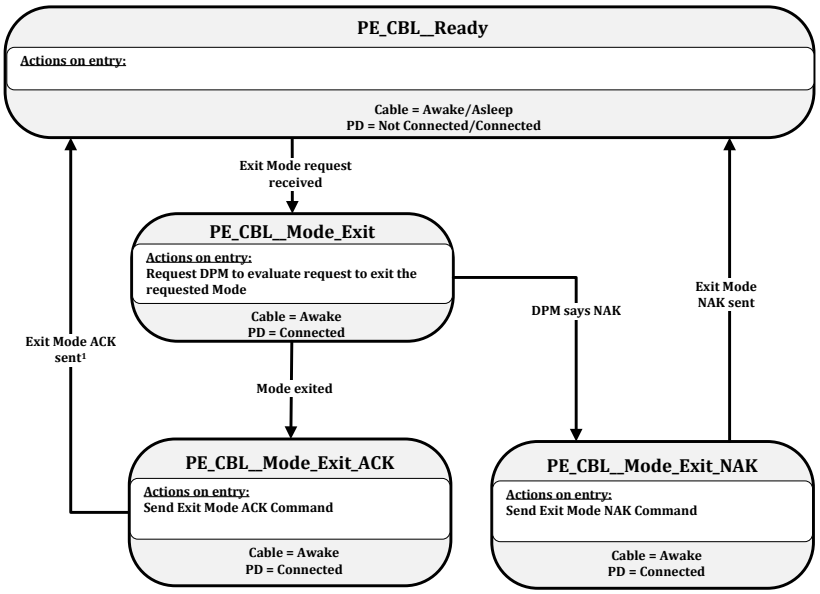
- The Structured VDM ***Enter Mode*** NAK Command response has been sent.

Microchip Ex. 1020, p. 442
Microchip v. Aptiv
IPR2024-00496

8.3.3.21.4.2 Cable Plug Structured VDM Exit Mode State Diagram

Figure 8-117 shows the state diagram for a Cable Plug in response to an ***Exit Mode*** Command.

Figure 8-117 Cable Plug Structured VDM Exit Mode State Diagram



[1] The Cable is required to be in USB operation or USB Safe State at this point.

8.3.3.21.4.2.1 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 Manger 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.

8.3.3.21.4.2.2 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.

8.3.3.21.4.2.3 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:

Microchip Ex. 1020, p. 443
Microchip v. Aptiv
IPR2024-00496

- The Structured VDM ***Exit Mode*** NAK Command response has been sent.

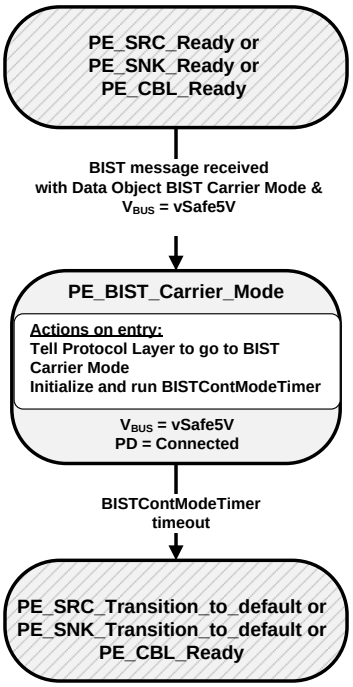
Microchip Ex. 1020, p. 444
Microchip v. Aptiv
IPR2024-00496

8.3.3.22 BIST State diagrams

8.3.3.22.1 BIST Carrier Mode State Diagram

Figure 8-118 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 8-118 BIST Carrier Mode State Diagram



8.3.3.22.1.1 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 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.

Microchip Ex. 1020, p. 445
Microchip v. Aptiv
IPR2024-00496

8.3.3.23 USB Type-C Referenced States

This section contains states cross-referenced from the ***[USB Type-C 1.2]*** specification.

8.3.3.23.1 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 Type-C 1.2]*** specification, including any other state transitions mandated in cases where USB Type-C ErrorRecovery is not supported.

On entry to the ***ErrorRecovery*** state the 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.

Microchip Ex. 1020, p. 446
Microchip v. Aptiv
IPR2024-00496

8.3.3.24 Policy Engine States

Table 8-52 lists the states used by the various state machines.

Table 8-52 Policy Engine States

State name	Reference
Source Port	
PE_SRC_Startup	Section 8.3.3.2.1
PE_SRC_Discovery	Section 8.3.3.2.2
PE_SRC_Send_Capabilities	Section 8.3.3.2.3
PE_SRC_Negotiate_Capability	Section 8.3.3.2.4
PE_SRC_Transition_Supply	Section 8.3.3.2.5
PE_SRC_Ready	Section 8.3.3.2.6
PE_SRC_Disabled	Section 8.3.3.2.7
PE_SRC_Capability_Response	Section 8.3.3.2.8
PE_SRC_Hard_Reset	Section 8.3.3.2.9
PE_SRC_Hard_Reset_Received	Section 8.3.3.2.10
PE_SRC_Transition_to_default	Section 8.3.3.2.11
PE_SRC_Get_Sink_Cap	Section 8.3.3.2.12
PE_SRC_Wait_New_Capabilities	Section 8.3.3.2.13
Sink Port	
PE_SNK_Startup	Section 8.3.3.3.1
PE_SNK_Discovery	Section 8.3.3.3.2
PE_SNK_Wait_for_Capabilities	Section 8.3.3.3.3
PE_SNK_Evaluate_Capability	Section 8.3.3.3.4
PE_SNK_Select_Capability	Section 8.3.3.3.5
PE_SNK_Transition_Sink	Section 8.3.3.3.6
PE_SNK_Ready	Section 8.3.3.3.7
PE_SNK_Hard_Reset	Section 8.3.3.3.8
PE_SNK_Transition_to_default	Section 8.3.3.3.9
PE_SNK_Give_Sink_Cap	Section 8.3.3.3.10
PE_SNK_Get_Source_Cap	Section 8.3.3.3.11
Soft Reset and Protocol Error	
Source Port Soft Reset	
PE_SRC_Send_Soft_Reset	Section 8.3.3.4.1.1
PE_SRC_Soft_Reset	Section 8.3.3.4.1.2
Sink Port Soft Reset	
PE_SNK_Send_Soft_Reset	Section 8.3.3.4.2.1
PE_SNK_Soft_Reset	Section 8.3.3.4.2.2
Not Supported Message	
Source Port Not Supported	
PE_SRC_Send_Not_Supported	Section 8.3.3.5.1.1

Microchip Ex. 1020, p. 447
Microchip v. Aptiv
IPR2024-00496

State name	Reference
PE_SRC_Not_Supported_Received	Section 8.3.3.5.1.2
Sink Port Not Supported	
PE_SNK_Send_Not_Supported	Section 8.3.3.5.2.1
PE_SNK_Not_Supported_Received	Section 8.3.3.5.2.2
Source Port Ping	
PE_SRC_Ping	Section 8.3.3.6.1
Source Alert	
Source Port Source Alert	
PE_SRC_Send_Source_Alert	Section 8.3.3.7.1.1
Sink Port Source Alert	
PE_SNK_Source_Alert_Received	Section 8.3.3.7.2.1
Sink Port Sink Alert	
PE_SNK_Send_Sink_Alert	Section 8.3.3.7.3.1
Source Port Sink Alert	
PE_SRC_Sink_Alert_Received	Section 8.3.3.7.4.1
Source Extended Capabilities	
Sink Port Get Source Capabilities Extended	
PE_SNK_Get_Source_Cap_Ext	Section 8.3.3.8.1.1
Source Port Give Source Capabilities Extended	
PE_SRC_Give_Source_Cap_Ext	Section 8.3.3.8.2.1
Source Status	
Sink Port Get Source Status	
PE_SNK_Get_Source_Status	Section 8.3.3.9.1.1
Source Port Give Source Status	
PE_SRC_Give_Source_Status	Section 8.3.3.9.2.1
Source Port Get Sink Status	
PE_SRC_Get_Sink_Status	Section 8.3.3.9.3.1
Sink Port Give Sink Status	
PE_SNK_Give_Sink_Status	Section 8.3.3.9.4.1
Battery Capabilities	
Get Battery Capabilities	
PE_Get_Battery_Cap	Section 8.3.3.10.1.1
Give Battery Capabilities	
PE_Give_Battery_Cap	Section 8.3.3.10.2.1
Battery Status	
Get Battery Status	
PE_Get_Battery_Status	Section 8.3.3.11.1.1
Give Battery Status	
PE_Give_Battery_Status	Section 8.3.3.11.2.1
Manufacturer Information	
Get Manufacturer Information	
PE_Get_Manufacturer_Info	Section 8.3.3.12.1

Microchip Ex. 1020, p. 448
Microchip v. Aptiv
IPR2024-00496

State name	Reference
Give Manufacturer Information	
PE_Give_Manufacturer_Info	Section 8.3.3.12.2
Security Request/Response	
Send Security Request	
PE_Send_Security_Request	Section 8.3.3.13.1
Send Security Response	
PE_Send_Security_Response	Section 8.3.3.13.2
Security Response Received	
PE_Security_Response_Received	Section 8.3.3.13.3
Firmware Update Request/Response	
Send Firmware Update Request	
PE_Send_Firmware_Update_Request	Section 8.3.3.14.1.1
Send Firmware Update Response	
PE_Send_Firmware_Update_Response	Section 8.3.3.14.2.1
Firmware Update Response Received	
PE_Firmware_Update_Response_Received	Section 8.3.3.14.3.1
Dual-Role Port	
DFP to UFP Data Role Swap	
PE_DRS_DFP_UFP_Evaluate_Swap	Section 8.3.3.15.1.2
PE_DRS_DFP_UFP_Accept_Swap	Section 8.3.3.15.1.3
PE_DRS_DFP_UFP_Change_to_UFP	Section 8.3.3.15.1.4
PE_DRS_DFP_UFP_Send_Swap	Section 8.3.3.15.1.5
PE_DRS_DFP_UFP_Reject_Swap	Section 8.3.3.15.1.6
UFP to DFP Data Role Swap	
PE_DRS_UFP_DFP_Evaluate_Swap	Section 8.3.3.15.2.2
PE_DRS_UFP_DFP_Accept_Swap	Section 8.3.3.15.2.3
PE_DRS_UFP_DFP_Change_to_DFP	Section 8.3.3.15.2.4
PE_DRS_UFP_DFP_Send_Swap	Section 8.3.3.15.2.5
PE_DRS_UFP_DFP_Reject_Swap	Section 8.3.3.15.2.6
Source to Sink Power Role Swap	
PE_PRS_SRC_SNK_Evaluate_Swap	Section 8.3.3.15.3.2
PE_PRS_SRC_SNK_Accept_Swap	Section 8.3.3.15.3.3
PE_PRS_SRC_SNK_Transition_to_off	Section 8.3.3.15.3.4
PE_PRS_SRC_SNK_Assert_Rd	Section 8.3.3.15.3.5
PE_PRS_SRC_SNK_Wait_Source_on	Section 8.3.3.15.3.6
PE_PRS_SRC_SNK_Send_Swap	Section 8.3.3.15.3.7
PE_PRS_SRC_SNK_Reject_Swap	Section 8.3.3.15.3.8
Sink to Source Power Role Swap	
PE_PRS_SNK_SRC_Evaluate_Swap	Section 8.3.3.15.4.2
PE_PRS_SNK_SRC_Accept_Swap	Section 8.3.3.15.4.3
PE_PRS_SNK_SRC_Transition_to_off	Section 8.3.3.15.4.4

Microchip Ex. 1020, p. 449
Microchip v. Aptiv
IPR2024-00496

State name	Reference
PE_PRS_SNK_SRC_Assert_Rp	
PE_PRS_SNK_SRC_Source_on	Section 8.3.3.15.4.5
PE_PRS_SNK_SRC_Send_Swap	Section 8.3.3.15.4.7
PE_PRS_SNK_SRC_Reject_Swap	Section 8.3.3.15.4.8
Source to Sink Fast Role Swap	
PE_FRS_SRC_SNK_CC_Signal	Section 8.3.3.15.5.2
PE_FRS_SRC_SNK_Evaluate_Swap	Section 8.3.3.15.5.3
PE_FRS_SRC_SNK_Accept_Swap	Section 8.3.3.15.5.4
PE_FRS_SRC_SNK_Transition_to_off	Section 8.3.3.15.5.5
PE_FRS_SRC_SNK_Assert_Rd	Section 8.3.3.15.5.6
PE_FRS_SRC_SNK_Wait_Source_on	Section 8.3.3.15.5.7
Sink to Source Fast Role Swap	
PE_FRS_SNK_SRC_Send_Swap	Section 8.3.3.15.6.1
PE_FRS_SNK_SRC_Transition_to_off	Section 8.3.3.15.6.2
PE_FRS_SNK_SRC_Vbus_Applied	Section 8.3.3.15.6.3
PE_FRS_SNK_SRC_Assert_Rp	Section 8.3.3.15.6.4
PE_FRS_SNK_SRC_Source_on	Section 8.3.3.15.6.5
Dual-Role Source Port Get Source Capabilities	
PE_DR_SRC_Get_Source_Cap	Section 8.3.3.15.7.1
Dual-Role Source Port Give Sink Capabilities	
PE_DR_SRC_Give_Sink_Cap	Section 8.3.3.15.8.1
Dual-Role Sink Port Get Sink Capabilities	
PE_DR_SNK_Get_Sink_Cap	Section 8.3.3.15.9.1
Dual-Role Sink Port Give Source Capabilities	
PE_DR_SNK_Give_Source_Cap	Section 8.3.3.15.10.1
Dual-Role Source Port Get Source Capabilities Extended	
PE_DR_SRC_Get_Source_Cap_Ext	Section 8.3.3.15.11.1
Dual-Role Sink Port Give Source Capabilities Extended	
PE_DR_SNK_Give_Source_Cap_Ext	Section 8.3.3.15.12.1
USB Type-C VCONN Swap	
PE_VCS_Send_Swap	Section 8.3.3.16.1.1
PE_VCS_Evaluate_Swap	Section 8.3.3.16.1.2
PE_VCS_Accept_Swap	Section 8.3.3.16.1.3
PE_VCS_Reject_Swap	Section 8.3.3.16.1.4
PE_VCS_Wait_For_VCONN	Section 8.3.3.16.1.5
PE_VCS_Turn_Off_VCONN	Section 8.3.3.16.1.6
PE_VCS_Turn_On_VCONN	Section 8.3.3.16.1.7
PE_VCS_Send_Ps_Rdy	Section 8.3.3.16.1.8
Initiator Structured VDM	
Initiator to Port Structured VDM Discover Identity	
PE_INIT_PORT_VDM_Identity_Request	Section 8.3.3.17.1.1

Microchip Ex. 1020, p. 450
Microchip v. Aptiv
IPR2024-00496

State name	Reference
PE_INIT_PORT_VDM_Identity_ACKed	Section 8.3.3.17.1.2
PE_INIT_PORT_VDM_Identity_NAKed	Section 8.3.3.17.1.3
Initiator Structured VDM Discover SVIDs	
PE_INIT_VDM_SVIDs_Request	Section 8.3.3.17.2.1
PE_INIT_VDM_SVIDs_ACKed	Section 8.3.3.17.2.2
PE_INIT_VDM_SVIDs_NAKed	Section 8.3.3.17.2.3
Initiator Structured VDM Discover Modes	
PE_INIT_VDM_Modes_Request	Section 8.3.3.17.3.1
PE_INIT_VDM_Modes_ACKed	Section 8.3.3.17.3.2
PE_INIT_VDM_Modes_NAKed	Section 8.3.3.17.3.3
Initiator Structured VDM Attention	
PE_INIT_VDM_Attention_Request	Section 8.3.3.17.4.1
Responder Structured VDM	
Responder Structured VDM Discovery Identity	
PE_RESP_VDM_Get_Identity	Section 8.3.3.18.1.1
PE_RESP_VDM_Send_Identity	Section 8.3.3.18.1.2
PE_RESP_VDM_Get_Identity_NAK	Section 8.3.3.18.1.3
Responder Structured VDM Discovery SVIDs	
PE_RESP_VDM_Get_SVIDs	Section 8.3.3.18.2.1
PE_RESP_VDM_Send_SVIDs	Section 8.3.3.18.2.2
PE_RESP_VDM_Get_SVIDs_NAK	Section 8.3.3.18.2.3
Responder Structured VDM Discovery Modes	
PE_RESP_VDM_Get_Modes	Section 8.3.3.18.3.1
PE_RESP_VDM_Send_Modes	Section 8.3.3.18.3.2
PE_RESP_VDM_Get_Modes_NAK	Section 8.3.3.18.3.3
Receiving a Structured VDM Attention	
PE_RCV_VDM_Attention_Request	Section 8.3.3.18.4.1
DFP Structured VDM	
DFP Structured VDM Mode Entry	
PE_DFP_VDM_Mode_Entry_Request	Section 8.3.3.19.1.1
PE_DFP_VDM_Mode_Entry_ACKed	Section 8.3.3.19.1.2
PE_DFP_VDM_Mode_Entry_NAKed	Section 8.3.3.19.1.3
DFP Structured VDM Mode Exit	
PE_DFP_VDM_Mode_Exit_Request	Section 8.3.3.19.2.1
PE_DFP_VDM_Mode_Exit_ACKed	Section 8.3.3.19.2.2
UFP Structure VDM	
UFP Structured VDM Enter Mode	
PE_UFP_VDM_Evaluate_Mode_Entry	Section 8.3.3.20.1.1
PE_UFP_VDM_Mode_Entry_ACK	Section 8.3.3.20.1.2
PE_UFP_VDM_Mode_Entry_NAK	Section 8.3.3.20.1.3

Microchip Ex. 1020, p. 451
Microchip v. Aptiv
IPR2024-00496

State name	Reference
UFP Structured VDM Exit Mode	
PE_UFP_VDM_Mode_Exit	Section 8.3.3.20.2.1
PE_UFP_VDM_Mode_Exit_ACK	Section 8.3.3.20.2.2
PE_UFP_VDM_Mode_Exit_NAK	Section 8.3.3.20.2.3
Cable Plug Specific	
Cable Ready	
PE_CBL_Ready	Section 8.3.3.21.1.1
Mode Entry	
PE_CBL_Evaluate_Mode_Entry	Section 8.3.3.21.4.1.1
PE_CBL_Mode_Entry_ACK	Section 8.3.3.21.4.1.2
PE_CBL_Mode_Entry_NAK	Section 8.3.3.21.4.1.3
Mode Exit	
PE_CBL_Mode_Exit	Section 8.3.3.21.4.2.1
PE_CBL_Mode_Exit_ACK	Section 8.3.3.21.4.2.2
PE_CBL_Mode_Exit_NAK	Section 8.3.3.21.4.2.3
Cable Soft Reset	
PE_CBL_Soft_Reset	Section 8.3.3.21.2.1.1
Cable Hard Reset	
PE_CBL_Hard_Reset	Section 8.3.3.21.2.2.1
DFP Soft Reset or Cable Reset	
PE_DFP_CBL_Send_Soft_Reset	Section 8.3.3.21.2.3.1
PE_DFP_CBL_Send_Cable_Reset	Section 8.3.3.21.2.3.2
UFP Source Soft Reset	
PE_UFP_CBL_Send_Soft_Reset	Section 8.3.3.21.2.4
Source Startup Structured VDM Discover Identity	
PE_SRC_VDM_Identity_Request	Section 8.3.3.21.3.1
PE_SRC_VDM_Identity_ACKed	Section 8.3.3.21.3.2
PE_SRC_VDM_Identity_NAKed	Section 8.3.3.21.3.3
BIST Carrier Mode	
PE_BIST_Carrier_Mode	Section 8.3.3.22.1.1
USB Type-C referenced states	
ErrorRecovery	Section 8.3.3.23.1

Microchip Ex. 1020, p. 452
Microchip v. Aptiv
IPR2024-00496

9. States and Status Reporting

9.1 Overview

This chapter describes the Status reporting mechanisms for devices with data connections (e.g. D+/D- and or SSTx+/- and SSRx+/-). It also describes the corresponding USB state a device that supports USB PD shall transition to as a result of changes to the USB PD state that the device is in.

This chapter does not define the System Policy or the System Policy Manager. That is defined in ***[USBTypeCBridge 1.0]***. In addition the Policies themselves are not described here; these are left to the implementers of the relevant products and systems to define.

All PD Capable USB (PDUSB) Devices shall report themselves as self-powered devices (over USB) when plugged into a PD capable Port even if they are entirely powered from V_{BUS}. However, there are some differences between PD and ***[USB 2.0]*** / ***[USB 3.1]***; for example, the presence of V_{BUS} alone does not mean that the device (Consumer) moves from the USB Attached state to the USB Powered state. Similarly the removal of V_{BUS} alone does not move the device (Consumer) from any of the USB states to the Attached state. See Section 9.1.2 for details.

PDUSB Devices shall follow the PD requirements when it comes to suspend (see Section 6.4.1.2.2.2), configured, and operational power. The PD requirements when the device is configured or operational are defined in this section (see Table 9-4). Note that the power requirements reported in the PD Consumer Port descriptor of the device shall override the power draw reported in the *bMaxPower* field in the configuration descriptor. A PDUSB Device shall report zero in the *bMaxPower* field after successfully negotiating a mutually agreeable Contract and shall disconnect and re-enumerate when it switches operation back to operating in standard ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** When operating in ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** mode it shall report its power draw via the *bMaxPower* field.

As shown in Figure 9-1, each Provider and Consumer will have their own Local Policies which operate between directly connected ports. An example of a typical PD system is shown in Figure 9-1. This example consists of a Provider, Consumer/Providers and Consumers connected together in a tree topology. Between directly connected devices there is both a flow of Power and also Communication consisting of both Status and Control information.

Microchip Ex. 1020, p. 453
Microchip v. Aptiv
IPR2024-00496

Figure 9-1 Example PD Topology

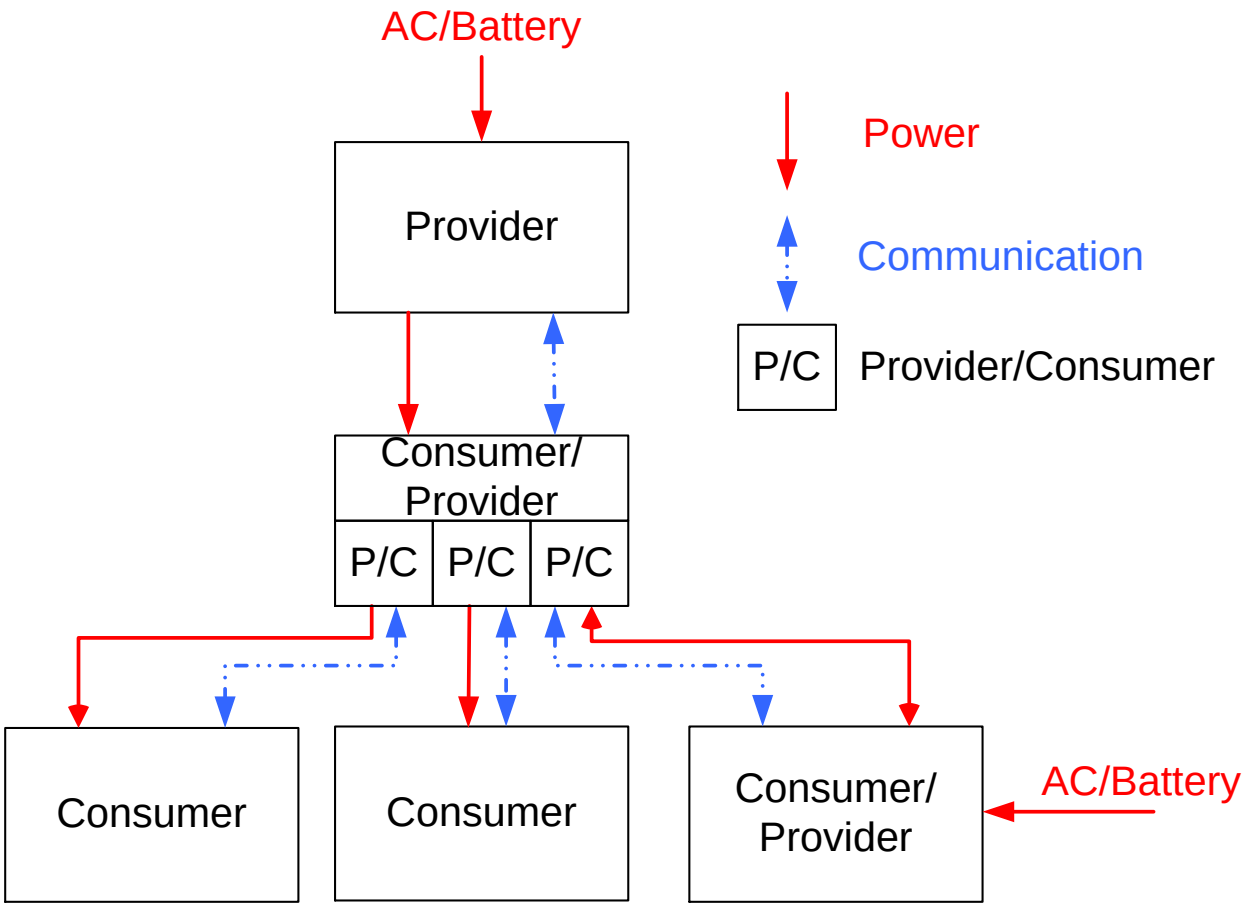


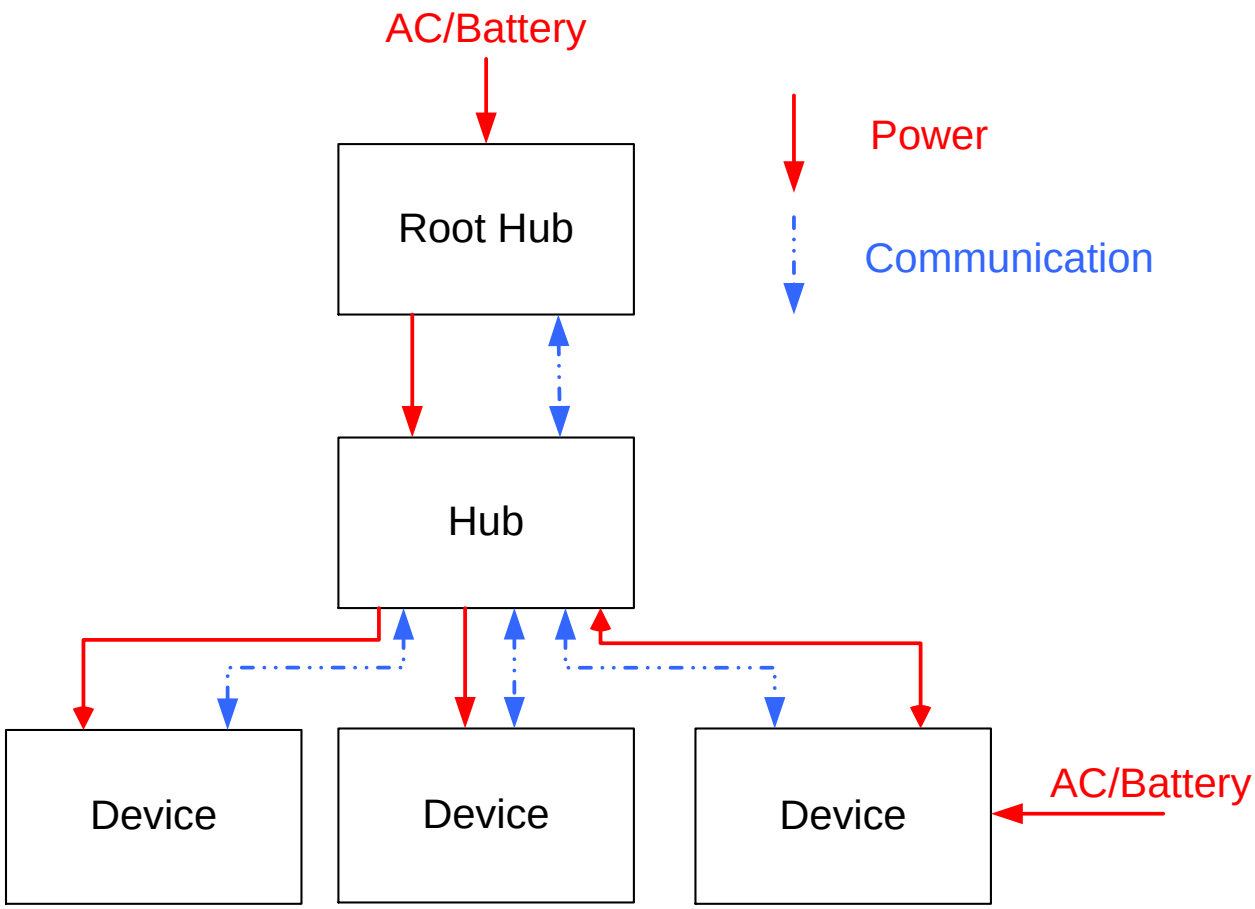
Figure 9-2 shows how this same topology can be mapped to USB. In a USB based system, policy is managed by the host and communication of system level policy information is via standard USB data line communication. This is a separate mechanism to the USB Power Delivery V_{BUS} protocol which is used to manage Local Policy. When USB data line communication is used, status information and control requests are passed directly between the System Policy Manager (SPM) on the host and the Provider or Consumer.

Status information comes from a Provider or Consumer to the SPM so it can better manage the resources on the host and provide feedback to the end user.

Real systems will be a mixture of devices which in terms of power management support may have implemented PD, ***[USB 2.0]***, ***[USB 3.1]***, ***[USB Type-C 1.2]*** or ***[USBBC 1.2]*** or they may even just be non-compliant Power Sucking Devices. The level of communication of system status to the SPM will therefore not necessarily be comprehensive. The aim of the status mechanisms described here is to provide a mechanism whereby each connected entity in the system provides as much information as possible on the status of itself.

Microchip Ex. 1020, p. 454
Microchip v. Aptiv
IPR2024-00496

Figure 9-2 Mapping of PD Topology to USB



Information described in this section that is communicated to the SPM is as follows:

- Versions of USB Type-C Current, PD and BC supported
- Capabilities as a Provider/Consumer
- Current operational state of each Port e.g. Standard, USB Type-C Current, BC, PD and negotiated power level
- Status of AC or Battery Power for each PDUSB Device in the system

The SPM can negotiate with Providers or Consumers in the system in order to request a different Local Policy, or to request the amount of power to be delivered by the Provider to the Consumer. Any change in Local Policy could trigger a renegotiation of the Contract, using USB Power Delivery protocols, between a directly connected Provider and Consumer. A change in how much power is to be delivered will, for example, cause a renegotiation.

9.1.1 PDUSB Device and Hub Requirements

All PDUSB Devices shall return all relevant descriptors mentioned in this chapter. PDUSB Hubs shall also support a PD bridge as defined in ***[USBTypeCBridge 1.0]***.

9.1.2 Mapping to USB Device States

As mentioned in Section 9.1 a PDUSB Device reports itself as a self-powered device. However, the device shall determine whether or not it is in the USB Attached or USB Powered states as described in Figure 9-3, Figure 9-4 and Figure 9-5. All other USB states of the PDUSB Device shall be as described in Chapter 9 of ***[USB 2.0]*** and ***[USB 3.1]***.

Microchip Ex. 1020, p. 455
Microchip v. Aptiv
IPR2024-00496

Figure 9-3 shows how a PDUSB Device determines when to transition from the USB Attached to the USB Powered state. USB Type-C Dead Battery operation does not require special handling since the default state at Attach or after a Hard Reset is that the USB Device is a Sink.

Figure 9-3 USB Attached to USB Powered State Transition

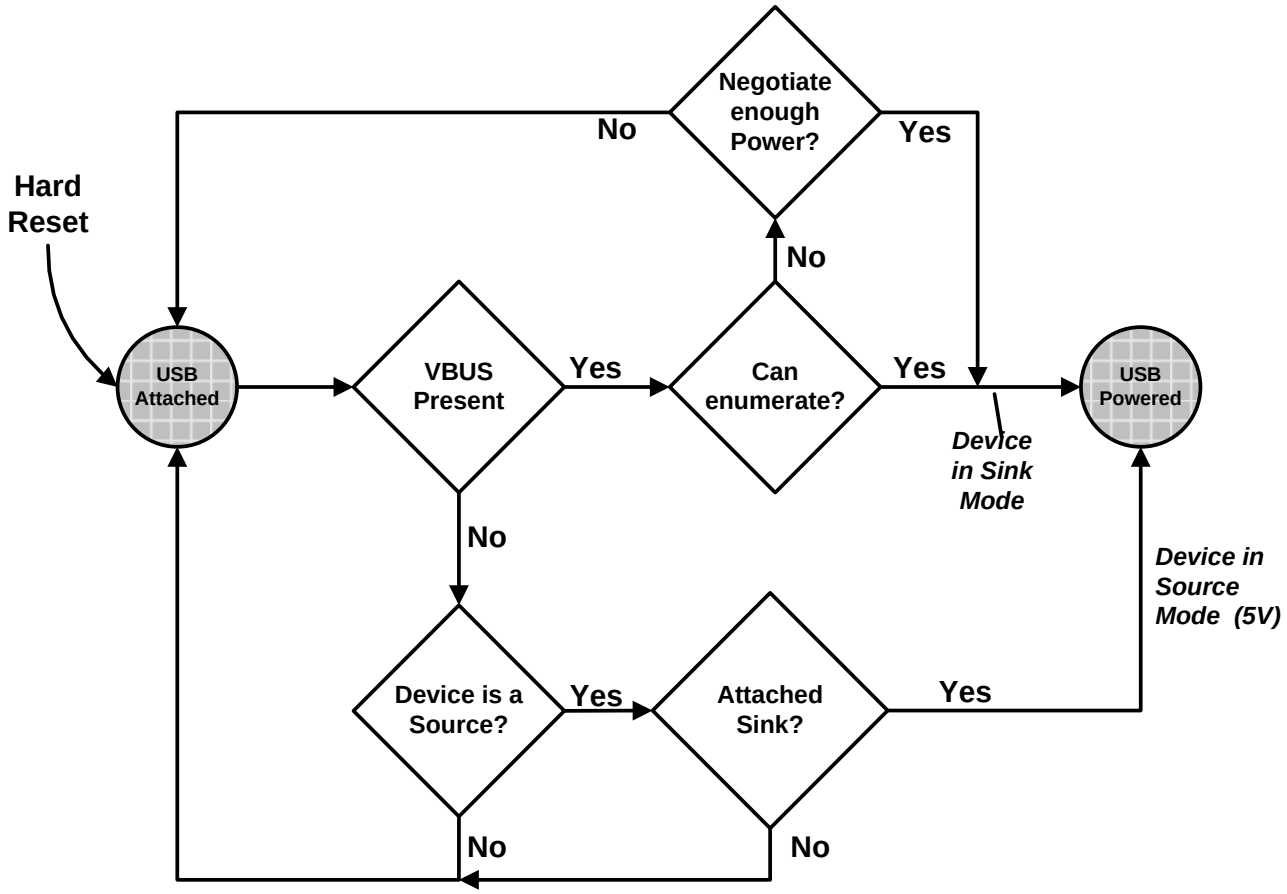


Figure 9-4 shows how a PDUSB Device determines when to transition from the USB Powered state to the USB Attached state when the device is a Consumer. A PDUSB Device determines that it is performing a Power Role Swap as described in Section 8.3.3.15.3 and Section 8.3.3.15.4. See Section 7.1.6 for additional information on device behavior during Hard Resets.

Microchip Ex. 1020, p. 456
Microchip v. Aptiv
IPR2024-00496

Figure 9-4 Any USB State to USB Attached State Transition (When operating as a Consumer)

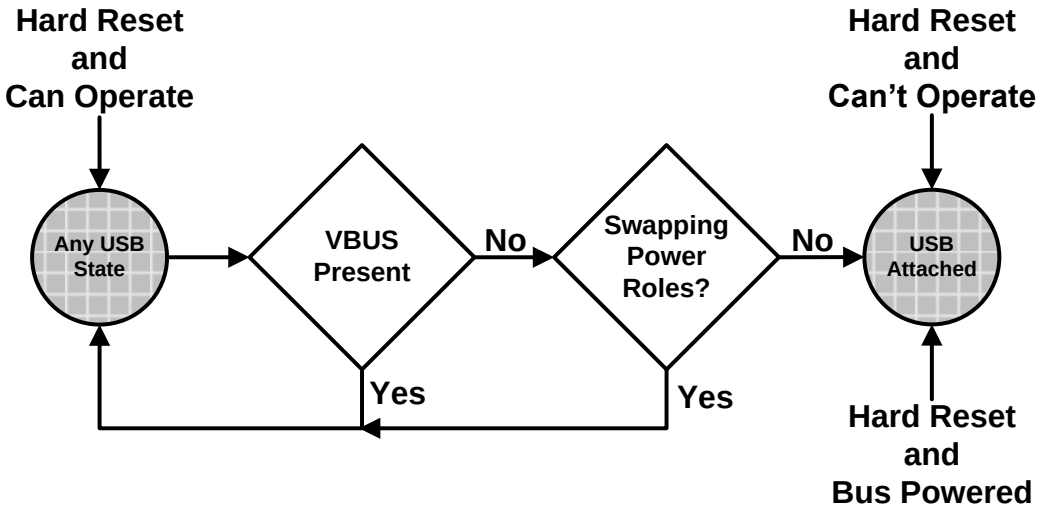


Figure 9-5 shows how a PDUSB Device determines when to transition from the USB Powered state to the USB Attached state when the device is a Provider.

Figure 9-5 Any USB State to USB Attached State Transition (When operating as a Provider)

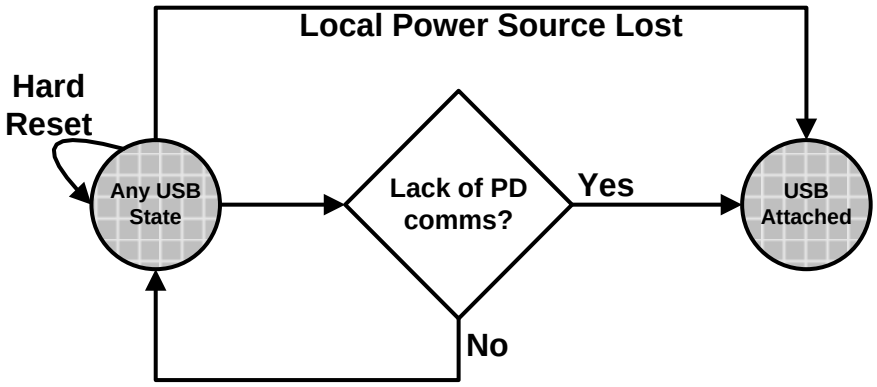
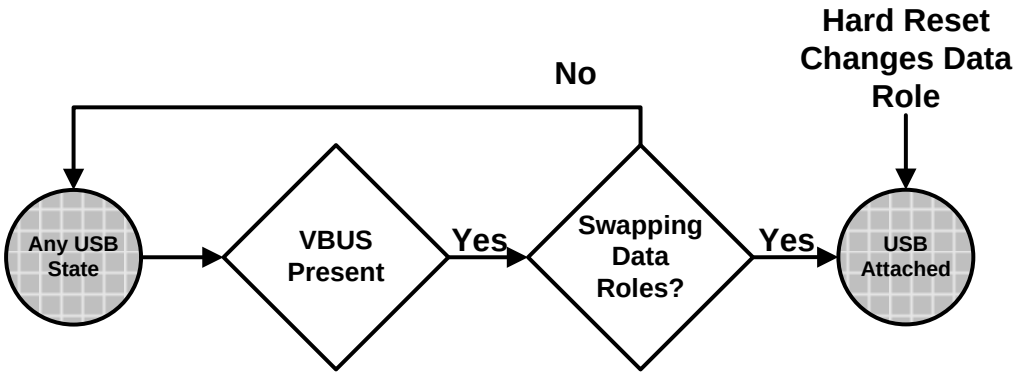


Figure 9-6 shows how a PDUSB Device using the USB Type-C connector determines when to transition from the USB Powered state to the USB Attached state after a Data Role Swap has been performed i.e. it has just changed from operation as a PDUSB Host to operation as a PDUSB Device. The Data Role Swap is described in Section 6.3.9. A Hard Reset will also return a Sink acting as a PDUSB Host to PDUSB Device operation as described in Section 6.8.2. See Section 7.1.6 for additional information on device behavior during Hard Resets.

Microchip Ex. 1020, p. 457
Microchip v. Aptiv
IPR2024-00496

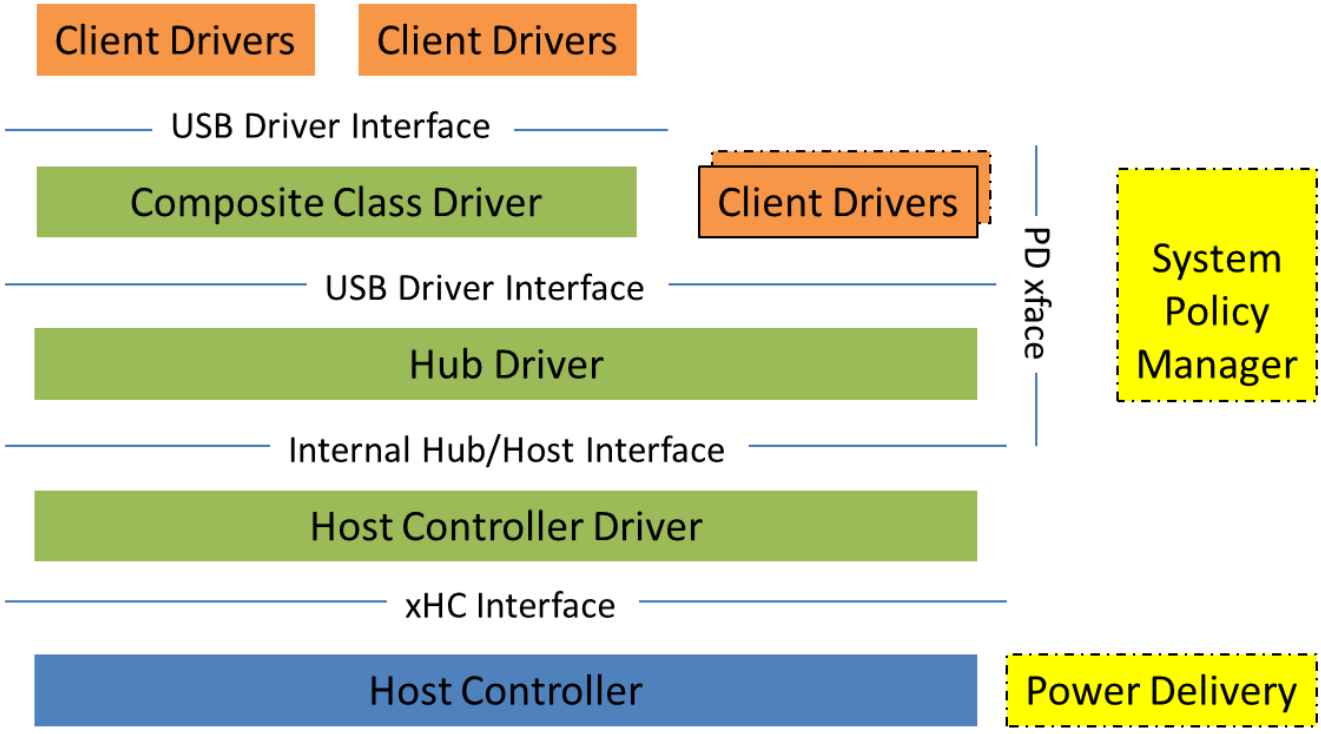
Figure 9-6 Any USB State to USB Attached State Transition (After a USB Type-C Data Role Swap)



9.1.3 PD Software Stack

Figure 9-7 gives an example of the software stack on a PD aware OS. In this stack we are using the example of a system with an xHCI based controller. The USB Power Delivery hardware may or may not be a part of the xHC.

Figure 9-7 Software stack on a PD aware OS



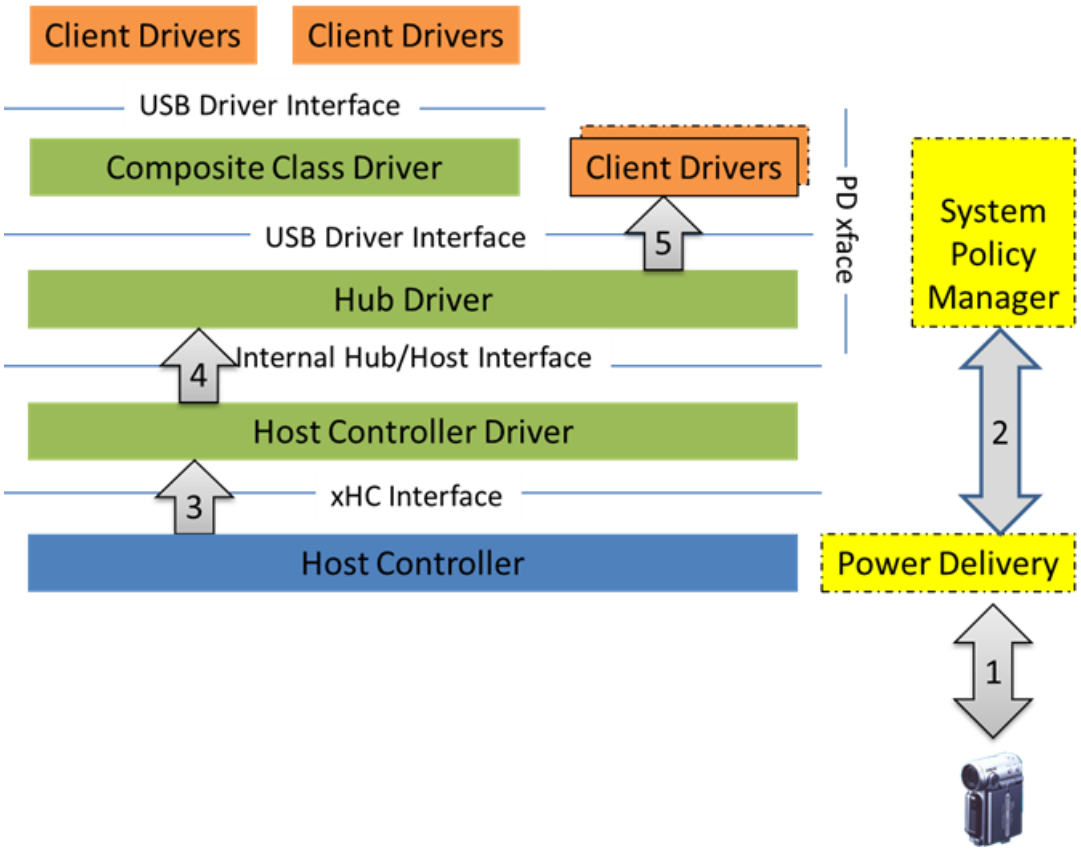
9.1.4 PDUSB Device Enumeration

As described earlier, a PDUSB Device acts as a self-powered device with some caveats with respect to how it transitions from the USB Attached state to USB Powered state. Figure 9-8 gives a high level overview of the enumeration steps involved due to this change. A PDUSB Device will first (Step1) interact with the Power Delivery

Microchip Ex. 1020, p. 458
Microchip v. Aptiv
IPR2024-00496

hardware and the Local Policy manager to determine whether or not it can get sufficient power to enumerate/operate. Note: PD is likely to have established a Contract prior to enumeration. The SPM will be notified (Step 2) of the result of this negotiation between the Power Delivery hardware and the PDUSB Device. After successfully negotiating a mutually agreeable Contract the device will signal a connect to the xHC. The standard USB enumeration process (Steps 3, 4 and 5) is then followed to load the appropriate driver for the function(s) that the PDUSB Device exposes.

Figure 9-8 Enumeration of a PDUSB Device



If a PDUSB Device cannot perform its intended function with the amount of power that it can get from the Port it is connected to then the host system should display a Message (on a PD aware OS) about the failure to provide sufficient power to the device. In addition the device shall follow the requirements listed in Section 8.2.5.2.1.

Microchip Ex. 1020, p. 459
Microchip v. Aptiv
IPR2024-00496

9.2 PD Class Specific Descriptors

A PDUSB Device shall return all relevant descriptors mentioned in this section.

The device shall return its capability descriptors as part of the device's Binary Object Store (BOS) descriptor set. Table 9-1 lists the type of PD device capabilities.

Table 9-1 USB Power Delivery Type Codes

Capability Code	Value	Description
POWER_DELIVERY_CAPABILITY	06H	Defines the various PD Capabilities of this device
BATTERY_INFO_CAPABILITY	07H	Provides information on each Battery supported by the device
PD_CONSUMER_PORT_CAPABILITY	08H	The Consumer characteristics of a Port on the device
PD_PROVIDER_PORT_CAPABILITY	09H	The provider characteristics of a Port on the device

9.2.1 USB Power Delivery Capability Descriptor

Table 9-2 USB Power Delivery Capability Descriptor

Offset	Field	Size	Value	Description
0	*bLength*	1	Number	Size of descriptor
1	*bDescriptorType*	1	Constant	DEVICE CAPABILITY Descriptor type
2	*bDevCapabilityType*	1	Constant	Capability type: ***POWER_DELIVERY_CAPABILITY***
3	*bReserved*	1	***Reserved***	Shall be set to zero.
4	*bmAttributes*	4	Bitmap	Bitmap encoding of supported device level features. A value of one in a bit location indicates a feature is supported; a value of zero indicates it is not supported. Encodings are: Bit — Description 0 — ***Reserved***. Shall be set to zero. 1 — Battery Charging. This bit shall be set to one to indicate this device supports the Battery Charging Specification as per the value reported in the *bcdBCVersion* field. 2 — USB Power Delivery. This bit shall be set to one to indicate this device supports the USB Power Delivery Specification as per the value reported in the *bcdPDVersion* field. 3 — Provider. This bit shall be set to one to indicate this device is capable of providing power. This field is only valid if Bit 2 is set to one. 4 — Consumer. This bit shall be set to one to indicate that this device is a consumer of power. This field is only valid if Bit 2 is set to one.

Microchip Ex. 1020, p. 460
Microchip v. Aptiv
IPR2024-00496

<table>
<tr><th>Offset</th><th>Field</th><th>Size</th><th>Value</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td></td><td>
5 This bit shall be set to 1 to indicate that this device supports the feature CHARGING_POLICY. Note that supporting the CHARGING_POLICY feature does not require a BC or PD mechanism to be implemented.

6 USB Type-C Current. This bit shall be set to one to indicate this device supports power capabilities defined in the USB Type-C Specification as per the value reported in the bcdUSBTypeCVersion field

7 Reserved. Shall be set to zero.

15:8 bmPowerSource. At least one of the following bits 8, 9 and 14 shall be set to indicate which power sources are supported.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>8</td><td>AC Supply</td></tr>
<tr><td>9</td><td>Battery</td></tr>
<tr><td>10</td><td>Other</td></tr>
<tr><td>13:11</td><td>NumBatteries. This field shall only be valid when the Battery field is set to one and shall be used to report the number of batteries in the device.</td></tr>
<tr><td>14</td><td>Uses V_{BUS}</td></tr>
<tr><td>15</td><td>Reserved and shall be set to zero.</td></tr>
</table>
31:16 Reserved and shall be set to zero.
</td></tr>
<tr><td>8</td><td>bcdBCVersion</td><td>2</td><td>BCD</td><td>Battery Charging Specification Release Number in Binary-Coded Decimal (e.g., V1.20 is 120H). This field shall only be valid if the device indicates that it supports BC in the bmAttributes field.</td></tr>
<tr><td>10</td><td>bcdPDVersion</td><td>2</td><td>BCD</td><td>USB Power Delivery Specification Release Number in Binary-Coded Decimal. This field shall only be valid if the device indicates that it supports PD in the bmAttributes field.</td></tr>
<tr><td>12</td><td>bcdUSBTypeCVersion</td><td>2</td><td>BCD</td><td>USB Type-C Specification Release Number in Binary-Coded Decimal. This field shall only be valid if the device indicates that it supports USB Type-C in the bmAttributes field.</td></tr>
</table>

9.2.2 Battery Info Capability Descriptor

A PDUSB Device shall support this capability descriptor if it reported that one of its power sources was a Battery in the *bmPowerSource* field in its Power Deliver Capability Descriptor. It shall return one Battery Info Descriptor per Battery it supports.

Microchip Ex. 1020, p. 461
Microchip v. Aptiv
IPR2024-00496

Table 9-3 Battery Info Capability Descriptor

Offset	Field	Size	Value	Description
0	*bLength*	1	Number	Size of descriptor
1	*bDescriptorType*	1	Constant	DEVICE CAPABILITY Descriptor type
2	*bDevCapabilityType*	1	Constant	Capability type: ***BATTERY_INFO_CAPABILITY***
3	*iBattery*	1	Index	Index of string descriptor shall contain the user friendly name for this Battery.
4	*iSerial*	1	Index	Index of string descriptor shall contain the Serial Number String for this Battery.
5	*iManufacturer*	1	Index	Index of string descriptor shall contain the name of the Manufacturer for this Battery.
6	*bBatteryId*	1	Number	Value shall be used to uniquely identify this Battery in status Messages.
7	*bReserved*	1	Number	***Reserved*** and shall be set to zero.
8	*dwChargedThreshold*	4	mWh	Shall contain the Battery Charge value above which this Battery is considered to be fully charged but not necessarily "topped off."
12	*dwWeakThreshold*	4	mWh	Shall contain the minimum charge level of this Battery such that above this threshold, a device can be assured of being able to power up successfully (see Battery Charging 1.2).
16	*dwBatteryDesignCapacity*	4	mWh	Shall contain the design capacity of the Battery.
20	*dwBatteryLastFullchargeCapacity*	4	mWh	Shall contain the maximum capacity of the Battery when fully charged.

9.2.3 PD Consumer Port Capability Descriptor

A PDUSB Device shall support this capability descriptor if it is a Consumer.

Microchip Ex. 1020, p. 462
Microchip v. Aptiv
IPR2024-00496

Table 9-4 PD Consumer Port Descriptor

Offset	Field	Size	Value	Description
0	*bLength*	1	Number	Size of descriptor
1	*bDescriptorType*	1	Constant	DEVICE CAPABILITY Descriptor type
2	*bDevCapabilityType*	1	Constant	Capability type: ***PD_CONSUMER_PORT_CAPABILITY***
3	*bReserved*	1	Number	***Reserved*** and shall be set to zero.
4	*bmCapabilities*	2	Bitmap	Capability: This field shall indicate the specification the Consumer Port will operate under. **Bit** / **Description** 0 – Battery Charging (BC) 1 – USB Power Delivery (PD) 2 – USB Type-C Current 15:3 – ***Reserved*** and shall be set to zero.
6	*wMinVoltage*	2	Number	Shall contain the minimum voltage in 50mV units that this Consumer is capable of operating at.
8	*wMaxVoltage*	2	Number	Shall contain the maximum voltage in 50mV units that this Consumer is capable of operating at.
10	*wReserved*	2	Number	***Reserved*** and shall be set to zero.
12	*dwMaxOperatingPower*	4	Number	Shall contain the maximum power in 10mW units this Consumer can draw when it is in a steady state operating mode.
16	*dwMaxPeakPower*	4	Number	Shall contain the maximum power in 10mW units this Consumer can draw for a short duration of time (*dwMaxPeakPowerTime*) before it falls back into a steady state.
20	*dwMaxPeakPowerTime*	4	Number	Shall contain the time in 100ms units that this Consumer can draw peak current. A device shall set this field to 0xFFFF if this value is unknown.

9.2.4 PD Provider Port Capability Descriptor

A PDUSB Device shall support this capability descriptor if it is a Provider.

Microchip Ex. 1020, p. 463
Microchip v. Aptiv
IPR2024-00496

Table 9-5 PD Provider Port Descriptor

Offset	Field	Size	Value	Description
0	*bLength*	1	Number	Size of descriptor
1	*bDescriptorType*	1	Constant	DEVICE CAPABILITY Descriptor type
2	*bDevCapabilityType*	1	Constant	Capability type: ***PD_PROVIDER_PORT_CAPABILITY***
3	*bReserved*	1	Number	***Reserved*** and shall be set to zero.
4	*bmCapabilities*	2	Bitmap	This field shall indicate the specification the Provider Port will operation under. **Bit** / **Description** 0 – Battery Charging (BC) 1 – USB Power Delivery (PD) 2 – USB Type-C Current 15:3 – ***Reserved***. Shall be set to zero.
6	*bNumOfPDObjects*	1	Number	Shall indicate the number of Power Data Objects.
7	*bReserved*	1	Number	***Reserved*** and shall be set to zero.
8	*wPowerDataObject1*	4	Bitmap	Shall contain the first Power Data Object supported by this Provider Port. See Section 6.4.1 for details of the Power Data Objects.
…	…	…	…	…
N+4	*wPowerDataObjectN*	4	Bitmap	Shall contain the 2nd and subsequent Power Data Objects supported by this Provider Port. See Section 6.4.1 for details of the Power Data Objects.

Microchip Ex. 1020, p. 464
Microchip v. Aptiv
IPR2024-00496

9.3 PD Class Specific Requests and Events

A PDUSB Device that is compliant to this specification shall support the Battery related requests if it has a battery.

A PDUSB Hub that is compliant to this specification shall support a USB PD Bridge as described in ***[USBTypeCBridge 1.0]*** irrespective of whether the PDUSB Hub is a Provider, a Consumer, or both.

9.3.1 Class-specific Requests

The PD class defines requests to which PDUSB Devices shall respond as outlined in Table 9-6. All valid requests in Table 9-6 shall be implemented by PDUSB Devices.

Table 9-6 PD Class Requests

Request	bmRequestType	bRequest	wValue	wIndex	wLength	Data
GetBatteryStatus	10000000B	***Get_Battery_Status***	Zero	Battery ID	Eight	Battery Status
SetPDFeature	00000000B	SET_FEATURE	Feature Selector	Feature Specific	Zero	None

Table 9-7 gives the bRequest values for commands that are not listed in the hub/device framework chapters of ***[USB 2.0]***, ***[USB 3.1]***.

Table 9-7 PD Class Request Codes

bRequest	Value
GET_BATTERY_STATUS	21

Table 9-8 gives the valid feature selectors for the PD class. Refer to Section 9.4.2.1, and Section 9.4.2.2 for a description of the features.

Table 9-8 PD Class Feature Selectors

Feature Selector	Recipient	Value
BATTERY_WAKE_MASK	Device	40
CHARGING_POLICY	Device	54

Microchip Ex. 1020, p. 465
Microchip v. Aptiv
IPR2024-00496

9.4 PDUSB Hub and PDUSB Peripheral Device Requests

9.4.1 GetBatteryStatus

This request returns the current status of the Battery in a PDUSB Hub/Peripheral.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	***Get_Battery_Status***	Zero	Battery ID	Eight	Battery Status

The PDUSB Hub/Peripheral shall return the Battery Status of the Battery identified by the value of *wIndex* field.

Every PDUSB Device that has a Battery shall return its Battery Status when queried with this request. For Providers or Consumers with multiple batteries, the status of each Battery shall be reported per Battery.

Table 9-9 Battery Status Structure

<table>
<tr><th>Offset</th><th>Field</th><th>Size</th><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>bBatteryAttributes</td><td>1</td><td>Number</td><td>Shall indicate whether a Battery is installed and whether this is charging or discharging.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>There is no Battery</td></tr>
<tr><td>1</td><td>The Battery is charging</td></tr>
<tr><td>2</td><td>The Battery is discharging</td></tr>
<tr><td>3</td><td>The Battery is neither discharging nor charging</td></tr>
<tr><td>255-4</td><td>Reserved and shall not be used</td></tr>
</table></td></tr>
<tr><td>1</td><td>bBatterySOC</td><td>1</td><td>Number</td><td>Shall indicate the Battery State of Charge given as percentage value from Battery Remaining Capacity.</td></tr>
<tr><td>2</td><td>bBatteryStatus</td><td>1</td><td>Number</td><td>If a Battery is present shall indicate the present status of the Battery.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>No error</td></tr>
<tr><td>1</td><td>Battery required and not present</td></tr>
<tr><td>2</td><td>Battery non-chargeable/wrong chemistry</td></tr>
<tr><td>3</td><td>Over-temp shutdown</td></tr>
<tr><td>4</td><td>Over-voltage shutdown</td></tr>
<tr><td>5</td><td>Over-current shutdown</td></tr>
<tr><td>6</td><td>Fatigued Battery</td></tr>
<tr><td>7</td><td>Unspecified error</td></tr>
<tr><td>255-8</td><td>Reserved and shall not be used</td></tr>
</table></td></tr>
</table>

Microchip Ex. 1020, p. 466
Microchip v. Aptiv
IPR2024-00496

<table>
<tr><th>Offset</th><th>Field</th><th>Size</th><th>Value</th><th>Description</th></tr>
<tr><td>3</td><td>bRemoteWakeCapStatus</td><td>1</td><td>Bitmap</td><td>If the device supports remote wake, then the device shall support Battery Remote wake events. The default value for the Remote wake events shall be turned off (set to zero) and can be enable/disabled by the host as required. If set to one the device shall generate a wake event when a change of status occurs. See Section 9.4.2 for more details.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>Battery present event</td></tr>
<tr><td>1</td><td>Charging flow</td></tr>
<tr><td>2</td><td>Battery error</td></tr>
<tr><td>7:3</td><td>Reserved and shall be set to zero</td></tr>
</table></td></tr>
<tr><td>4</td><td>wRemainingOperatingTime</td><td>2</td><td>Number</td><td>Shall contain the operating time (in minutes) until the Weak Battery threshold is reached, based on Present Battery Strength and the device’s present operational power needs. Note: this value shall exclude any additional power received from charging.

A Battery that is not capable of returning this information shall return a value of 0xFFFF.</td></tr>
<tr><td>6</td><td>wRemainingChargeTime</td><td>2</td><td>Number</td><td>Shall contain the remaining time (in minutes) until the Charged Battery threshold is reached based on Present Battery Strength, charging power and the device’s present operational power needs. Value shall only be valid if the Charging Flow is “Charging”.

A Battery that is not capable of returning this information shall return a value of 0xFFFF.</td></tr>
</table>

If *wValue* or *wLength* are not as specified above, then the behavior of the PDUSB Device is not specified.

If *wIndex* refers to a Battery that does not exist, then the PDUSB Device shall respond with a Request Error.

If the PDUSB Device is not configured, the PDUSB Hub's response to this request is undefined.

If the PDUSB Hub is not configured, the PDUSB Hub's response to this request is undefined.

9.4.2 SetPDFeature

This request sets the value requested in the PDUSB Hub/Peripheral.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ FEATURE	Feature Selector	Feature Specific	Zero	None

Setting a feature enables that feature or starts a process associated with that feature; see Table 9-8 for the feature selector definitions. Features that may be set with this request are:

- ***BATTERY_WAKE_MASK***
- ***CHARGING_POLICY***

Microchip Ex. 1020, p. 467
Microchip v. Aptiv
IPR2024-00496

9.4.2.1 BATTERY_WAKE_MASK Feature Selector

When the feature selector is set to ***BATTERY_WAKE_MASK***, then the *wIndex* field is structured as shown in the following table.

Table 9-10 Battery Wake Mask

Bit	Description
0	**Battery Present**: When this bit is set then the PDUSB Device shall generate a wake event if it detects that a Battery has been Attached.
1	**Charging Flow**: When this bit is set then the PDUSB Device shall generate a wake event if it detects that a Battery switched from charging to discharging or vice versa.
2	**Battery Error**: When this bit is set then the PDUSB Device shall generate a wake event if the Battery has detected an error condition.
15:3	***Reserved*** and shall not be used.

The SPM may Enable or Disable the wake events associated with one or more of the above events by using this feature.

If the PDUSB Hub is not configured, the PDUSB Hub's response to this request is undefined.

9.4.2.2 CHARGING_POLICY Feature Selector

When the feature selector is set to ***CHARGING_POLICY***, the wIndex field shall be set to one of the values defined in Table 9-11. If the device is using USB Type-C Current above the default value or is using PD then this feature setting has no effect and the rules for power levels specified in the ***[USB Type-C 1.2]*** or USB PD specifications shall apply.

Table 9-11 Charging Policy Encoding

Value	Description
00H	The device shall follow the default current limits as defined in the USB 2.0 or USB 3.1 specification, or as negotiated through other USB mechanisms such as BC. This is the default value.
01H	The Device may draw additional power during the unconfigured and suspend states for the purposes of charging. For charging the device itself, the device shall limit its current draw to the higher of these two values: • ICCHPF as defined in the USB 2.0 or USB 3.1 specification, regardless of its USB state. • Current limit as negotiated through other USB mechanisms such as BC.
02H	The Device may draw additional power during the unconfigured and suspend states for the purposes of charging. For charging the device itself, the device shall limit its current draw to the higher of these two values: • ICCLPF as defined in the USB 2.0 or USB 3.1 specification, regardless of its USB state. • Current limit as negotiated through other USB mechanisms such as BC.
03H	The device shall not consume any current for charging the device itself regardless of its USB state.
04H-FFFFH	***Reserved*** and shall not be used

Microchip Ex. 1020, p. 468
Microchip v. Aptiv
IPR2024-00496

This is a valid command for the PDUSB Hub/Peripheral in the Address or Configured USB states. Further, it is only valid if the device reports a USB PD capability descriptor in its BOS descriptor and Bit 5 of the bmAttributes in that descriptor is set to 1. The device will go back to the wIndex default value of 0 whenever it is reset.

Microchip Ex. 1020, p. 469
Microchip v. Aptiv
IPR2024-00496

10. Power Rules

10.1 Introduction

The flexibility of power provision on USB Type-C is expected to lead to adapter re-use and the increasingly widespread provision of USB power outlets in domestic and public places and in transport of all kinds. Environmental considerations could result in unbundled adapters. Rules are needed to avoid incompatibility between the Sources and the Sinks they are used to power, in order to avoid user confusion and to meet user expectations. This section specifies a set of rules that Sources and Sinks shall follow. These rules provide a simple and consistent user experience.

10.2 Source Power Rules

10.2.1 Source Power Rule Considerations

The Source power rules are designed to:

- Ensure the PD Power (PDP) of an adapter specified in watts explicitly defines the voltages and currents at each voltage the adapter supports
- Ensure that adapters with a large PDP are always capable of providing the power to devices designed for use with adapters with a smaller PDP
- Enable an ecosystem of adapters that are interoperable with the devices in the ecosystem.

The considerations that lead to the Source power rules are based are summarized in Table 10-1.

Table 10-1 Considerations for Sources

Considerations	Rationale	Consequence
Simple to identify capability	A user going into an electronics retailer should know what they need	Cannot have a complex identification scheme
Higher power Sources are a superset of smaller ones	Bigger is always better in user's eyes – don't want a degradation in performance	Higher power Sources do everything smaller ones do
Unambiguous Source definitions	Sources with the same power rating but different VI combinations might not interoperate	To avoid user confusion, any given power rating has a single definition
A range of power ratings	Users and companies will want freedom to pick appropriate Source ratings	Fixed profiles at specific power levels don't provide adequate flexibility, e.g. profiles as defined in previous versions of PD.
5V@3A USB Type-C Source is defined by *[USB Type-C 1.2]*	5V@3A USB Type-C Source is considered	All > 15W adapters must support 5V@3A or superset consideration is violated
Maximize 3A cable utilization	3A cables will be ubiquitous	Increase to maximum voltage (20V) before increasing current beyond 3A
Optimize voltage rail count	More rails are a higher burden for Sources, particularly in terms of testing	5V is a basic USB requirement. 20V provides the maximum capability.
Some Sources are not able to provide significant power	Some small Battery operated Sources e.g. mobile devices, are able to provide more power directly from their Battery than from a regulated 5V supply	In addition to the minimal 5V advertisement are able to advertise more power from their Battery
Some Sources share power between multiple Ports (Hubs)	Hubs have to be supported	See Section 10.2.4

Microchip Ex. 1020, p. 470
Microchip v. Aptiv
IPR2024-00496

10.2.2 Normative Voltages and Currents

The voltages and currents a Source with a PDP of x Watts shall support are as defined in Table 10-2.

Table 10-2 Normative Voltages and Currents

PDP (W)	Current at 5V (A)	Current at 9V (A)	Current at 15V (A)	Current at 20V (A)
0.5 ≤ x ≤ 15	x ÷ 5			
15 < x ≤ 27	3	x ÷ 9		
27 < x ≤ 45	3	3	x ÷ 15	
45 < x ≤ 60	3	3	3	x ÷ 20
60 < x ≤ 100	3	3	3	x ÷ 20[1]
[1] Requires a 5A cable.				

Figure 10-1 illustrates the maximum current and power rails that a Source shall support at each voltage for a given PDP.

Figure 10-1 Source Power Rule Illustration

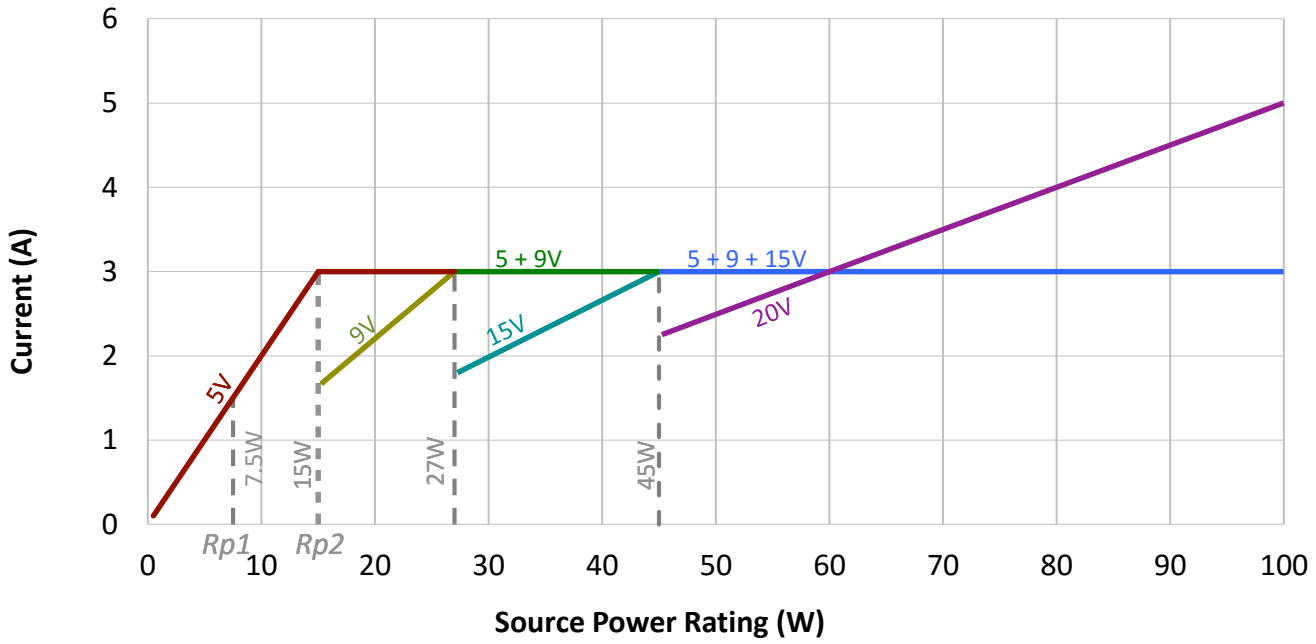


Figure 10-2 shows an example of an adapter with a rating at 50W. The adapter is required to support 20V at 2.5A, 15V at 3A, 9V at 3A and 5V at 3A.

Microchip Ex. 1020, p. 471
Microchip v. Aptiv
IPR2024-00496

Figure 10-2 Source Power Rule Example

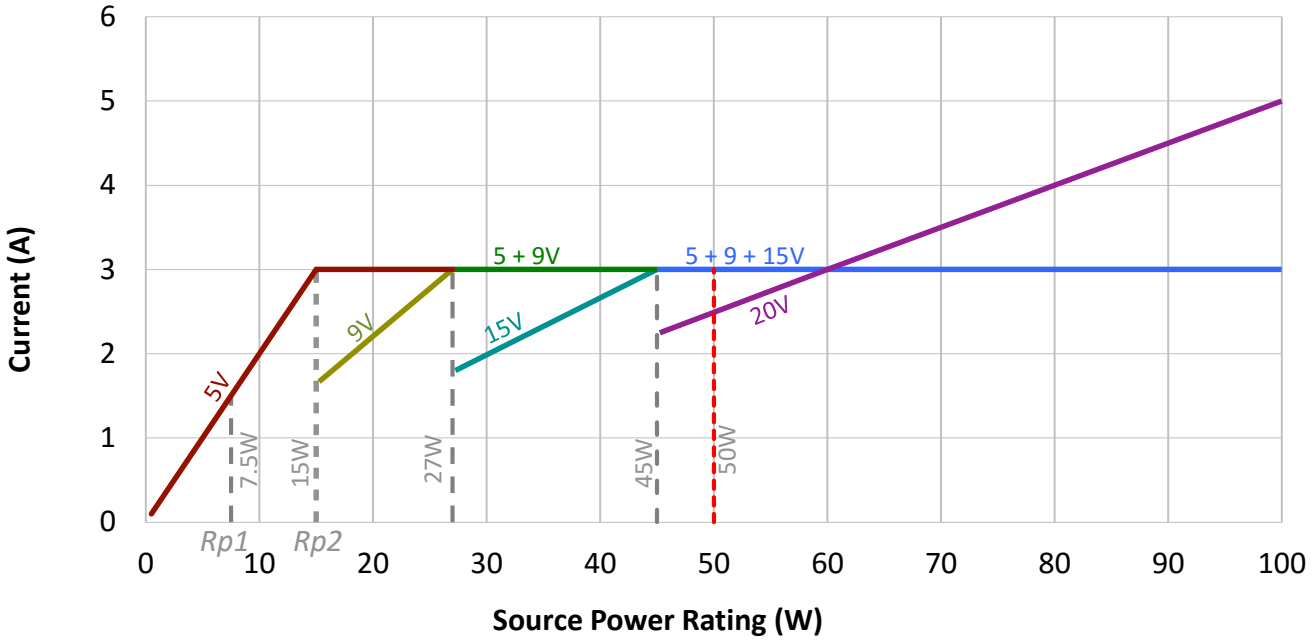


Table 10-3, Table 10-4, Table 10-5 and Table 10-6 show the Fixed Supply PDOs that shall be supported for each of the ***Normative*** voltages defined in Table 10-2.

Table 10-3 Fixed Supply PDO – Source 5V

Bit(s)	Description
B31..30	Fixed supply
B29	Dual-Role Power
B28	USB Suspend Supported
B27	Externally Powered
B26	USB Communications Capable
B25	Dual-Role Data
B24..22	***Reserved*** – shall be set to zero.
B21..20	Peak Current
B19..10	5V
B9..0	

PDP (x)	Current (A)
0.5 ≤ x ≤ 15	x ÷ 5
15 < x ≤ 100	3

Microchip Ex. 1020, p. 472
Microchip v. Aptiv
IPR2024-00496

Table 10-4 Fixed Supply PDO – Source 9V

Bit(s)	Description
B31..30	Fixed Supply
B29..22	***Reserved*** – shall be set to zero.
B21..20	Peak Current
B19..10	9V
B9..0	**PDP (x)** / **Current (A)** 0.5 ≤ x ≤ 15 / PDO not required 15 < x ≤ 27 / x ÷ 9 27 < x ≤ 100 / 3

Table 10-5 Fixed Supply PDO – Source 15V

Bit(s)	Description
B31..30	Fixed Supply
B29..22	***Reserved*** – shall be set to zero.
B21..20	Peak Current
B19..10	15V
B9..0	**PDP (x)** / **Current (A)** 0.5 ≤ x ≤ 27 / PDO not required 27 < x ≤ 45 / x ÷ 15 45 < x ≤ 100 / 3

Table 10-6 Fixed Supply PDO – Source 20V

Bit(s)	Description
B31..30	Fixed Supply
B29..22	***Reserved*** – shall be set to zero.
B21..20	Peak Current
B19..10	20V
B9..0	**PDP (x)** / **Current (A)** 0.5 ≤ x ≤ 45 / PDO not required 45 < x ≤ 100 / x ÷ 20

More current may be offered in the PDOs when ***Optional*** voltages/currents are supported and a 5A cable is being used (see Section 10.2.3).

Microchip Ex. 1020, p. 473
Microchip v. Aptiv
IPR2024-00496

10.2.3 Optional Voltages/Currents

In addition to the voltages and currents specified in Section 10.2.2 a Source that is optimized for use with a specific Sink or a specific class of Sinks may ***Optionally*** supply additional voltages and increased currents. When ***Optional*** voltages and increased currents are provided, the following requirements shall apply:

- The Source shall be able to meet its PDP at the ***Normative*** voltages and currents as specified in Section 10.2.2.
- A Source shall not advertise Fixed Supply PDO ***Optional*** voltages and currents that exceed the PDP.
- A Source shall not advertise Variable Supply PDO ***Optional*** maximum voltages and currents that exceed the PDP.
- A Source shall not advertise a Battery Supply PDO ***Optional*** maximum allowable power that exceeds the PDP.

10.2.4 Power sharing between ports

The Source power rules defined in Section 10.2.2 and Section 10.2.3 shall apply independently to each port on a system with multiple ports.

10.3 Sink Power Rules

10.3.1 Sink Power Rule Considerations

The Sink power rules are designed to ensure the best possible user experience when a given Sink used with a compliant Source of arbitrary Output Power Rating that only supplies the ***Normative*** voltages and currents.

The Sink Power Rules 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 high PDP are used with low power Sinks.
- Preventing Sinks that only function well (or at all) when using ***Optional*** voltages and currents.

10.3.2 Normative Sink Rules

Sinks designed to use Sources with a PDP of x W shall:

- Either operate or charge from Sources that have a PDP ≥ x W
- Either operate, charge or indicate a capability mismatch (see Section 6.4.2.3) from Sources that have a PDP < x W and ≥ 0.5W

A Sink optimized for a Source with ***Optional*** voltages and currents or power as described in Section 10.2.3 with a PDP of x W shall provide a similar user experience when powered from a Source with a PDP of ≥ x W that supplies only the ***Normative*** voltages and currents as specified in Section 10.2.2.

The following requirements shall apply to the advertised Sink Capabilities:

- A Sink shall not advertise Fixed Supply PDO maximum voltages and currents that exceed the PDP they were designed to use.
- A Sink shall not advertise Variable Supply PDO maximum voltages and currents that exceed the PDP they were designed to use.
- A Sink shall not advertise a Battery Supply PDO maximum allowable power that exceeds the PDP they were designed to use.

Microchip Ex. 1020, p. 474
Microchip v. Aptiv
IPR2024-00496

A. CRC calculation

A.1 C code example

```
//
// USB PD CRC Demo Code.
//
#include <stdio.h>

int crc;

//------------------------------------------------------------------------------

void crcBits(int x, int len) {

  const int poly = 0x04C11DB6;  //spec 04C1 1DB7h
  int newbit, newword, rl_crc;

  for(int i=0; i<len; i++) {

    newbit = ((crc>>31) ^ ((x>>i)&1)) & 1;
    if(newbit) newword=poly; else newword=0;
    rl_crc = (crc<<1) | newbit;
    crc = rl_crc ^ newword;
    printf("%2d newbit=%d, x>>i=0x%x, crc=0x%x\n", i, newbit,(x>>i),crc);
  }
}

int crcWrap(int c){

  int ret = 0;
  int j, bit;

  c = ~c;
  printf("~crc=0x%x\n", c);

  for(int i=0;i<32;i++) {
    j = 31-i;

    bit = (c>>i) & 1;
    ret |= bit<<j;
```


Microchip Ex. 1020, p. 475
Microchip v. Aptiv
IPR2024-00496

```
  }
  return ret;

}

//------------------------------------------------------------------------------

int main(){

  int txCrc=0,rxCrc=0,residue=0,data;

  printf("using packet data 0x%x\n", data=0x0101);

  crc = 0xffffffff;
  crcBits(data,16);
  txCrc = crcWrap(crc);

  printf("crc=0x%x, txCrc=0x%x\n", crc, txCrc);



  printf("received packet after decode= 0x%x, 0x%x\n", data, txCrc);

  crc = 0xffffffff;
  crcBits(data,16);
  rxCrc = crcWrap(crc);

  printf("Crc of the received packet data is (of course) =0x%x\n", rxCrc);

  printf("continue by running the transmit crc through the crc\n");
  crcBits(rxCrc,32);

  printf("Now the crc residue is 0x%x\n", crc);

  printf("should be 0xc704dd7b\n");

}
```


Microchip Ex. 1020, p. 476
Microchip v. Aptiv
IPR2024-00496

A.2 Table showing the full calculation over one Message

Function	Nibble	Symbol	Bits	CRC register transmitter	CRC register receiver	bit nr.
Preamble			0	FFFFFFFF	FFFFFFFF	1
			1	FFFFFFFF	FFFFFFFF	2
			0	FFFFFFFF	FFFFFFFF	3
			1	FFFFFFFF	FFFFFFFF	4
			0	FFFFFFFF	FFFFFFFF	5
			1	FFFFFFFF	FFFFFFFF	6
			0	FFFFFFFF	FFFFFFFF	7
			1	FFFFFFFF	FFFFFFFF	8
			0	FFFFFFFF	FFFFFFFF	9
			1	FFFFFFFF	FFFFFFFF	10
			0	FFFFFFFF	FFFFFFFF	11
			1	FFFFFFFF	FFFFFFFF	12
			0	FFFFFFFF	FFFFFFFF	13
			1	FFFFFFFF	FFFFFFFF	14
			0	FFFFFFFF	FFFFFFFF	15
			1	FFFFFFFF	FFFFFFFF	16
			0	FFFFFFFF	FFFFFFFF	17
			1	FFFFFFFF	FFFFFFFF	18
			0	FFFFFFFF	FFFFFFFF	19
			1	FFFFFFFF	FFFFFFFF	20
			0	FFFFFFFF	FFFFFFFF	21
			1	FFFFFFFF	FFFFFFFF	22
			0	FFFFFFFF	FFFFFFFF	23
			1	FFFFFFFF	FFFFFFFF	24
			0	FFFFFFFF	FFFFFFFF	25
			1	FFFFFFFF	FFFFFFFF	26
			0	FFFFFFFF	FFFFFFFF	27
			1	FFFFFFFF	FFFFFFFF	28
			0	FFFFFFFF	FFFFFFFF	29
			1	FFFFFFFF	FFFFFFFF	30
			0	FFFFFFFF	FFFFFFFF	31
			1	FFFFFFFF	FFFFFFFF	32
			0	FFFFFFFF	FFFFFFFF	33
			1	FFFFFFFF	FFFFFFFF	34
			0	FFFFFFFF	FFFFFFFF	35
			1	FFFFFFFF	FFFFFFFF	36
			0	FFFFFFFF	FFFFFFFF	37
			1	FFFFFFFF	FFFFFFFF	38
			0	FFFFFFFF	FFFFFFFF	39
			1	FFFFFFFF	FFFFFFFF	40
			0	FFFFFFFF	FFFFFFFF	41
			1	FFFFFFFF	FFFFFFFF	42
			0	FFFFFFFF	FFFFFFFF	43
			1	FFFFFFFF	FFFFFFFF	44
			0	FFFFFFFF	FFFFFFFF	45
			1	FFFFFFFF	FFFFFFFF	46
			0	FFFFFFFF	FFFFFFFF	47
			1	FFFFFFFF	FFFFFFFF	48
			0	FFFFFFFF	FFFFFFFF	49
			1	FFFFFFFF	FFFFFFFF	50
			0	FFFFFFFF	FFFFFFFF	51
			1	FFFFFFFF	FFFFFFFF	52
			0	FFFFFFFF	FFFFFFFF	53
			1	FFFFFFFF	FFFFFFFF	54
			0	FFFFFFFF	FFFFFFFF	55
			1	FFFFFFFF	FFFFFFFF	56
			0	FFFFFFFF	FFFFFFFF	57
			1	FFFFFFFF	FFFFFFFF	58
			0	FFFFFFFF	FFFFFFFF	59
			1	FFFFFFFF	FFFFFFFF	60
			0	FFFFFFFF	FFFFFFFF	61
			1	FFFFFFFF	FFFFFFFF	62
			0	FFFFFFFF	FFFFFFFF	63
			1	FFFFFFFF	FFFFFFFF	64
SOP		Sync1 (#18)	0	FFFFFFFF	FFFFFFFF	65
			0	FFFFFFFF	FFFFFFFF	66
			0	FFFFFFFF	FFFFFFFF	67
			1	FFFFFFFF	FFFFFFFF	68
			1	FFFFFFFF	FFFFFFFF	69
		Sync1 (#18)	0	FFFFFFFF	FFFFFFFF	70
			0	FFFFFFFF	FFFFFFFF	71
			0	FFFFFFFF	FFFFFFFF	72
			1	FFFFFFFF	FFFFFFFF	73
			1	FFFFFFFF	FFFFFFFF	74
		Sync1 (#18)	0	FFFFFFFF	FFFFFFFF	75
			0	FFFFFFFF	FFFFFFFF	76
			0	FFFFFFFF	FFFFFFFF	77
			1	FFFFFFFF	FFFFFFFF	78
			1	FFFFFFFF	FFFFFFFF	79
		Sync2 (#11)	1	FFFFFFFF	FFFFFFFF	80
			0	FFFFFFFF	FFFFFFFF	81
			0	FFFFFFFF	FFFFFFFF	82
			0	FFFFFFFF	FFFFFFFF	83
			1	FFFFFFFF	FFFFFFFF	84

Function	Nibble	Symbol	Bits	CRC register transmitter	CRC register receiver	bit nr.
GoodCRC Header #0101	#1	#09	1	FFFFFFFF	FFFFFFFF	85
			0	FFFFFFFE	FFFFFFFF	86
			0	FB3EE24B	FFFFFFFF	87
			1	F2BCD921	FFFFFFFF	88
			0	E1B8AFF5	FFFFFFFF	89
	#0	#1E	0	E1B8AFF5	FFFFFFFF	90
			1	C7B0425D	FFFFFFFE	91
			1	8BA1990D	FB3EE24B	92
			1	13822FAD	F2BCD921	93
			1	27045F5A	E1B8AFF5	94
	#1	#09	1	27045F5A	E1B8AFF5	95
			0	4AC9A303	C7B0425D	96
			0	95934606	8BA1990D	97
			1	2FE791BB	13822FAD	98
			0	5FCF2376	27045F5A	99
	#0	#1E	0	5FCF2376	27045F5A	100
			1	BF9E46EC	4AC9A303	101
			1	7BFD906F	95934606	102
			1	F7FB20DE	2FE791BB	103
			1	EB375C0B	5FCF2376	104
CRC-32 = swapped and inverted EB375C0B = 2FC51328	#8	#12	0	EB375C0B	5FCF2376	105
			1	EB375C0B	BF9E46EC	106
			0	EB375C0B	7BFD906F	107
			0	EB375C0B	F7FB20DE	108
			1	EB375C0B	EB375C0B	109
	#2	#14	0	EB375C0B	EB375C0B	110
			0	EB375C0B	D2AFA5A1	111
			1	EB375C0B	A19E56F5	112
			0	EB375C0B	47FDB05D	113
			1	EB375C0B	8B3A7D0D	114
	#3	#15	1	EB375C0B	8B3A7D0D	115
			0	EB375C0B	12B5E7AD	116
			1	EB375C0B	21AAD2ED	117
			0	EB375C0B	4355A5DA	118
			1	EB375C0B	86AB4BB4	119
	#1	#09	1	EB375C0B	86AB4BB4	120
			0	EB375C0B	0D569768	121
			0	EB375C0B	1E6C3367	122
			1	EB375C0B	3CD866CE	123
			0	EB375C0B	79B0CD9C	124
	#5	#0B	1	EB375C0B	79B0CD9C	125
			1	EB375C0B	F7A0868F	126
			0	EB375C0B	EB8010A9	127
			1	EB375C0B	D3C13CE5	128
			0	EB375C0B	A343647D	129
	#C	#1A	0	EB375C0B	A343647D	130
			1	EB375C0B	4686C8FA	131
			0	EB375C0B	8D0D91F4	132
			1	EB375C0B	1A1B23E8	133
			1	EB375C0B	343647D0	134
	#F	#1D	1	EB375C0B	343647D0	135
			0	EB375C0B	686C8FA0	136
			1	EB375C0B	D0D91F40	137
			1	EB375C0B	A1B23E80	138
			1	EB375C0B	43647D00	139
	#2	#14	0	EB375C0B	43647D00	140
			0	EB375C0B	8209E7B7	141
			1	EB375C0B	0413CF6E	142
			0	EB375C0B	0CE6836B	143
			1	EB375C0B	1D0C1B61	144
EOP		#0D	1	EB375C0B	1D0C1B61	145
			0	EB375C0B	3A1836C2	146
			1	EB375C0B	70F17033	147
			1	EB375C0B	E1E2E066	148
			0	EB375C0B	C704DD7B	149

Note: CRC transmitter is calculated over data bytes only, in casu marked nibbles, and calculation results are available one (bit-) clock later

Note: CRC receiver is calculated over data bytes and received CRC bytes, in casu marked nibbles, and calculation results are available five (bit-) clocks later

Fixed residual

Microchip Ex. 1020, p. 477
Microchip v. Aptiv
IPR2024-00496

B. PD Message Sequence Examples

The following examples are intended to show how the Device Policy Manager may operate and the sequence of Power Delivery messaging which will result. The aim of this section is to inform implementer's how some of the mechanisms detailed in this specification may be applied; it does not contain any ***Normative*** requirements.

All ports are assumed to be Enhanced SuperSpeed capable, with a default operating voltage of 5V and a unit load of 150mA. This 0.75W is assumed to be enough power to enable an externally powered device to maintain communication over USB and is enough to allow such a device to enumerate but not operate until more power is negotiated.

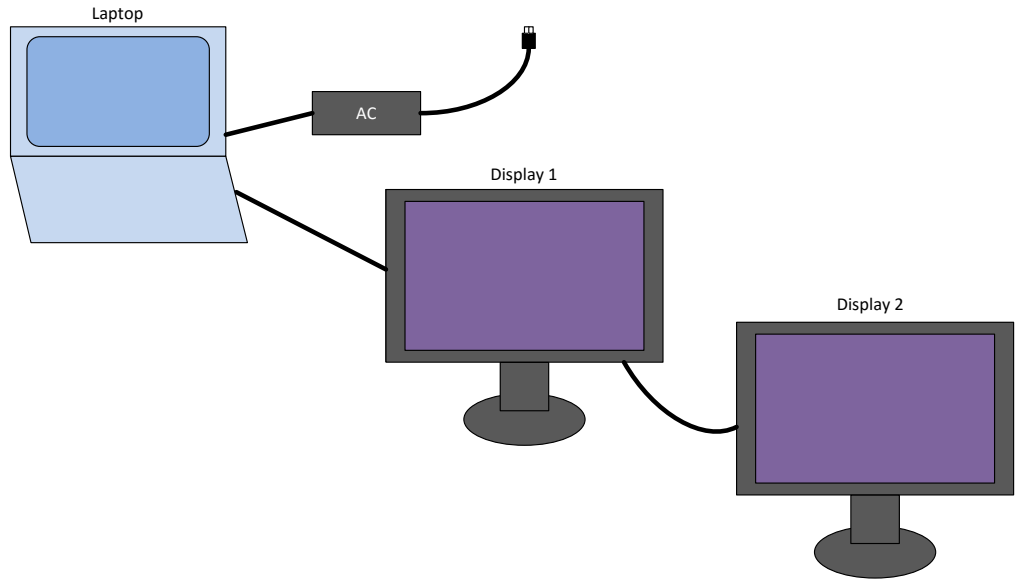
Although the Hubs in these illustrations support Power Delivery on both their UFPs and DFPs this is only one possible Hub implementation.

HDDs are assumed to spin up immediately after they are Attached. This follows the typical operation of current systems.

Ideal power transmission is assumed so that there are no power losses through a device; in practice these would need to be taken into account when requesting power.

B.1 External power is supplied downstream

Figure B-1 External Power supplied downstream



Configuration:

1. Laptop with an AC supply. AC supply provides sufficient power to charge the laptop and, in addition, to provide up to 60W downstream via its Enhanced SuperSpeed Port. According to the Source Power Rules described in Section 10.2 this means that the Port has a PD Power of 60W and so can supply: 5V@3A, 9V@3A, 15V@3A and 20V@3A.
2. Display 1 requires 30W to display and therefore a PD Power of 60W to operate itself plus Display 2 connected downstream. Display 1 initially uses 15V@2A to operate itself, since this also allows operation with a Source of 30W PD Power. On connection of Display 2, Display 1 will move to operation at 20V@3A to allow operation of the additional 30W ganged display. According to the Sink Power Rules described in Section 10.3 this means that

Microchip Ex. 1020, p. 478
Microchip v. Aptiv
IPR2024-00496

Display 1 requires a Source with a PD Power of 60W to fully operate. Display 1 contains a Hub allowing Display 2 to be connected to Display 1.

3. Display 2 requires 30W operate itself and does not support an additional display connected downstream. Display 1 uses 15V@2A to operate itself from a Source of 30W PD Power.
4. In USB suspend Display 1 and Display 2 will power down but can maintain USB connection using the PD power provided.

Table B-1 External power is supplied downstream

Step	Laptop	Display 1	Display 2	Device Policy Manager	Power (W)
Display 1					
1	Connected to wall supply	Detached	Detached		0
2	Display 1 Attached, V_{BUS} powered.	Attached, drawing 5V@150mA.	Detached		0.75
3	Set of Source Capabilities sent including: 5V@3A (15W), 9V@3A (27W), 15V@3A(45W) and 20V@3A (60W). The externally powered and USB suspend bits are set.	Source Capabilities received	Detached	Laptop determines its Source Capabilities based on its needs and the presence of a wall supply.	0.75
4	Request received	Requests 15V@2A (30W) from laptop	Detached	Display 1 knows it needs 20v@1.5A (30W) for its own operation, evaluates the supplied capabilities and determines that this is available.	0.75
5	Sends Accept	Accept received	Detached	Waiting for PS_RDY before drawing additional power.	0.75
6	Sends PS_RDY	PS_RDY received. Starts drawing 15V@2A. Display 1 turns on and starts operating.	Detached	Laptop evaluates the request, finds that it can meet this and so sends an accept.	30
Display 2					
7	Powering Display 1	Detects Attach	Attached, no V_{BUS}		30

Microchip Ex. 1020, p. 479
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Display 1	Display 2	Device Policy Manager	Power (W)
8	Request received	Display 1 requests 20V@1.73A (34.6W) from Laptop.	Attached, no V_{BUS}	Display 1 detects Attach and requests additional 4.5W of power for USB 3.1 Port.	30
9	Sends Accept	Accept received.	Attached, no V_{BUS}		34.6
10	Sends PS_RDY	PS_RDY received	Attached, no V_{BUS}		
11		Powers V_{BUS}	Attached, drawing 5V@150mA.		34.6
12		Sends out Source Capabilities including: 5V@0.9A to Display 2. The externally powered and USB suspend bits are set.	Source Capabilities received	Display 1 has 4.5W to allocate to Display 1. This is offered as a standard USB 3.1 Port.	34.6
13		Request received	Display 2 requests 5V@0.15A but indicates a Capability Mismatch. Display 2 remains off.	Display 2 decides it can manage to run its USB/PD function with 1 unit load but needs more power to function as a display.	34.6
14		Sends Accept	Accept received		34.6
15		Sends PS_RDY	PS_RDY received	Display 2 indicates a capability mismatch to the user.	34.6
16		Get Sink Capabilities sent	Get Sink Capabilities received	Display 1 needs to assess the capability mismatch by first determining what Display 2 actually needs.	34.6
17		Sink Capabilities received	Display 2 returns Sink Capabilities indicating operation at 15V@2A.		34.6

Microchip Ex. 1020, p. 480
Microchip v. Aptiv
IPR2024-00496

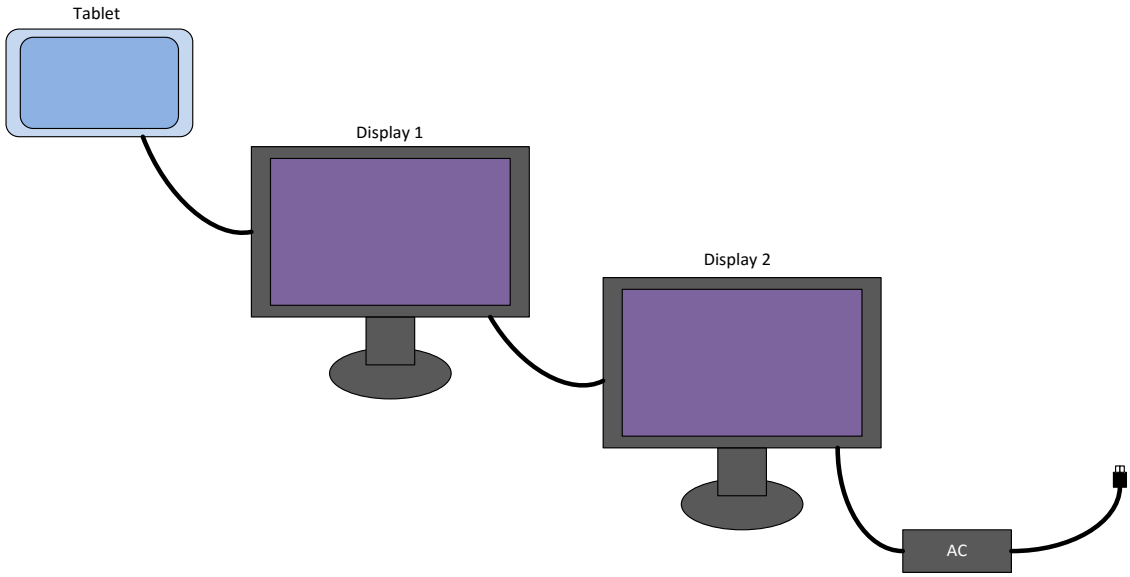
Step	Laptop	Display 1	Display 2	Device Policy Manager	Power (W)
18	Request received	Display 1 requests 20V@3A (60W) from Laptop.		Display1 now knows what Display 2 needs and requests the additional power from the laptop.	34.6
19	Sends Accept	Accept received.			34.6
20	Sends PS_RDY	PS_RDY received		An additional 30W is now available to Display 1 to offer to Display 2.	60
21		Sends out Source Capabilities including: 5V@0.9A and 20V@1.5A to Display 2. The externally powered and USB suspend bits are set.	Source Capabilities received	Now that Display 1 can power Display 2 correctly this power is offered by Display 1 via a new capabilities Message.	60
22		Request received	Display 2 requests 15V@2A.		60
23		Sends Accept	Accept received	Display 1 determines that the request by Display 2 is within the offered capabilities so the request is accepted.	60
24		Sends PS_RDY. Drawing 20V@3A from laptop.	PS_RDY received. Starts drawing 15V@2A, turns on and starts operating.	Display 2 now has the power it needs and can start working.	60
USB Suspend					
25	Laptop OS goes into suspend (S3), V_{BUS} remains on but USB bus is also suspended.	Display 1 turns off but draws 50mW, 25mW to maintain PDUSB Hub functions. The additional 25mW is used to supply the Port used by Display 2.	Display 2 turns off but draws 25mW to maintain USB/PD functions.	No changes in Contract. This is a power reduction purely based on the USB state.	60

Microchip Ex. 1020, p. 481
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Display 1	Display 2	Device Policy Manager	Power (W)
26	Laptop OS wakes up. USB is woken up.	Display 1 turns on and returns to drawing 20V@3A.	Display 2 turns on and returns to drawing 15V@2A.	No changes in PD Contract. This purely relates to USB bus state.	60

B.2 External power is supplied upstream

Figure B-2 External Power supplied upstream



Configuration:

1. Tablet with no AC supply. Tablet is a USB host and can use 5V@0.2A (1W) during normal operation and up to 5V@2.4A (12W) in order to charge.
2. Display 1 requires 30W to operate using their own display and therefore a PD Power of 42W to operate itself and charge the tablet. Display 1 uses 15V@2A to operate itself, since this allows operation with a Source of 30W PD Power and then moves to operation at 20V@2.1A to allow charging of the laptop. According to the Sink Power Rules described in Section 10.3 this means that the Display 1 requires a Source with a PD Power of 42W to fully operate.
3. Display 2 has an AC supply connected. AC supply provides sufficient power to power Display 2 and, in addition, to provide up to 60W PD Power upstream.

Table B-2 External power is supplied upstream

Step	Tablet	Display 1	Display 2	Device Policy Manager	Power (W)
Display 1 - Dead Battery					
1	Detached	Detached	Connected to the wall supply.		0

Microchip Ex. 1020, p. 482
Microchip v. Aptiv
IPR2024-00496

Step	Tablet	Display 1	Display 2	Device Policy Manager	Power (W)
2		Attached to Display 2	Display 1 Attached		0
3		USB Type-C Power drawn 5V@1.5A	USB Type-C Power advertised 5V@1.5A		0
4		Attached to Display 2, drawing 5V@1.5A (7.5W)	Providing 1 unit load to Display 1.		7.5
5		Source Capabilities received	Display2 sends out a set of capabilities including: 5V@3A (15W), 9V@3A (27W), 15V@3A (45W) and 20V@3A (60W). The externally powered and USB suspend bits are set.	Based on the capabilities of the wall supply and its own needs Display 2 calculates what it can offer upstream.	7.5
6		Display 1 requests 15V@2A (30W) from Display 2.	Request received	Display 1 knows it needs 30W to operate so it requests this amount.	7.5
7		Accept received	Sends Accept	Display 2 accepts the offer since it is within its capabilities.	7.5
8		PS_RDY received. Display 1 starts drawing power and turns on.	Sends PS_RDY	Display 2 indicates its power supply is ready to offer the power.	30
Tablet – Power Role Swap					
9	Tablet is Attached to Display 1.	Attached, V_{BUS} powered.			30
10	Tablet sends out a set of capabilities including: 5V@0.5A (2.5W). The externally powered bit cleared and USB suspend bit set.	Capabilities received			30

Microchip Ex. 1020, p. 483
Microchip v. Aptiv
IPR2024-00496

Step	Tablet	Display 1	Display 2	Device Policy Manager	Power (W)
11	Request received	Display 1 requests 5V@0A from the Tablet. The externally powered and Dual-Role Power bits are set.		Display 1 has external power providing everything it needs so it does not request any more.	30
12	Sends Accept	Accept received.		No power has been requested from the Tablet so the tablet has no reason to Reject this.	30
13	Sends PS_RDY	PS_RDY received.		Table completes the Explicit Contract by sending PS_RDY.	30
14	Get Sink Capabilities received.	Sends Get Sink Capabilities		Display 1 has access to an external supply so it needs to check whether the Tablet upstream, which has no external supply, could use some power. Display 1 also knows that there is excess capacity, based on the last capabilities it received, which it is not currently using from Display 2.	30
15	The Tablet returns Sink Capabilities indicating that it is a Dual-Role and that it can use 5V@0.2A (1W) as a Sink.	Sink Capabilities received			30
16		Display 1 requests 15V@2.1A (31.5W) from Display 2.	Request received		30
17		Accept received	Sends Accept	Request is within the available power so Display 2 sends an accept.	30

Microchip Ex. 1020, p. 484
Microchip v. Aptiv
IPR2024-00496

Step	Tablet	Display 1	Display 2	Device Policy Manager	Power (W)
18		PS_RDY received	Sends PS_RDY	Display 2 indicates that the power supply is ready to supply the power.	31.5
19	PR_Swap received	Requests PR_Swap from Tablet.		Display 1 now offers to provide power to the Tablet by initiating a Power Role Swap.	31.5
20	Accept sent. Tablet turns off its V_{BUS} supply.	Accept received.		Tablet is happy to accept a Power Role Swap from any device offering it power.	31.5
21	Send PS_RDY	PS_RDY received. Display 1 turns on its V_{BUS} supply		Tablet indicates that its supply has been turned off.	31.5
22	PS_RDY received.	PS_RDY sent.		Display 1 indicates that its power supply is ready so the Tablet starts drawing power.	31.5
23	Source Capabilities received	Display 1 sends out a set of capabilities to the Tablet including: 5V@0.48A (2.4W), 12V@0.2A (2.4W) and 20V@0.12 (2.4W). The externally powered and USB suspend bits are set.			31.5
24	The Tablet requests 12V@0.2A.	Request received.		Tablet can now request the power it needs.	31.5
25	Accept received	Accept sent		Power is within the capabilities of Display 1 so it accepts the request.	31.5
26	PS_RDY received. The Tablet starts drawing 12V@0.2A.	PS_RDY sent		Display 1 indicates that its power supply is ready so the tablet starts drawing the power.	31.5

Microchip Ex. 1020, p. 485
Microchip v. Aptiv
IPR2024-00496

Step	Tablet	Display 1	Display 2	Device Policy Manager	Power (W)
Tablet – Charge					
27	Tablet requests 12V@0.2A (2.4W) from Display 1. The Tablet needs to charge and so sets the Capability Mismatch bit and the No USB Suspend bit.	Request received.		Tablet needs to charge but the power offered is not sufficient. Since Display 1 claims to have an external supply the Tablet will try to get more power using the Capability Mismatch Flag.	31.5
28	Accept received	Accept sent		A valid request has been made so Display 1 accepts the request.	31.5
29	PS_RDY received	PS_RDY received		Tablet indicates a capability mismatch to the user.	31.5
30	Get Sink Capabilities received.	Get Sink Capabilities sent		Due to the Capability Mismatch Flag Display 1 requests Sink Capabilities from the Tablet?	31.5
31	The Tablet returns Sink capabilities containing: 5V@2.4A (12W). The externally powered bit is cleared.	Sink Capabilities received			31.5

Microchip Ex. 1020, p. 486
Microchip v. Aptiv
IPR2024-00496

Step	Tablet	Display 1	Display 2	Device Policy Manager	Power (W)
32		Display 1 requests 15V@2.8A (42W) from Display 2. The No Suspend Bit is set to reflect the request from the Tablet.	Request received	Since the Tablet requires an additional 12W of power, and Display 1 knows that this is available from Display 2 based on the last Capabilities received so it requests it. In addition the Request from the Tablet indicated that it wanted No Suspend so this is reflected upwards.	31.5
33		Accept received	Sends Accept	Display 2 has 42W available and so accepts the request.	42
34		PS_RDY received	Sends PS_RDY	Display 2 completes the Explicit Contract but at this point has not accepted that power can be drawn during suspend.	42
35		Source Capabilities received	Display2 sends out a new set of capabilities including: 5V@3A (15W), 9V@3A (27W), 15V@3A (45W) and 20V@3A (60W). The externally powered and USB suspend bits is now set to zero.	Based on the capabilities of the wall supply and its own needs Display 2 calculates what it can offer upstream. It decides that it can continue to supply the power even during USB suspend and so resets the USB suspend bit.	42

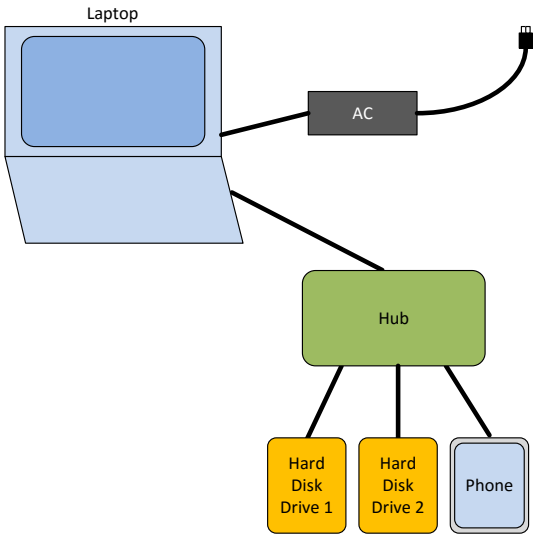
Microchip Ex. 1020, p. 487
Microchip v. Aptiv
IPR2024-00496

Step	Tablet	Display 1	Display 2	Device Policy Manager	Power (W)
36		Display 1 requests 15V@2.8A (42W) from Display 2. The No Suspend Bit is set to reflect the request from the Tablet.	Request received	Display 1 repeats its request since a new set of Capabilities have been sent out.	42
37		Accept received	Sends Accept	Display 2 has 42W available, even during suspend, and so accepts the request.	42
38		PS_RDY received	Sends PS_RDY	Display 2 completes the Explicit Contract.	42
39	Capabilities received	Display 1 sends out a set of capabilities to the Tablet including: 5V@2.4A (12W). The externally powered bit is set and USB suspend bit is cleared.		Display 1 now has the additional power available and so offers this to the Tablet.	42
40	Tablet requests 5V@2.4A (12W) from Display 1.	Request received.		Tablet is being offered the power it needs to charge and so the Tablet requests this from Display 1.	42
41	Accept received	Sends Accept		Request is within the available Display 1's available power and so it accepts the request.	42
42	PS_RDY received. Tablet starts drawing 5V@2.4A (12W) Display 1 and starts to charge.	Sends PS_RDY		Display 1 indicates its supply is ready to supply power.	42

Microchip Ex. 1020, p. 488
Microchip v. Aptiv
IPR2024-00496

B.3 Giving back power

Figure B-3 Giving Back Power



Configuration:

1. Laptop with an AC supply. AC supply provides sufficient power to charge the laptop and, in addition, to provide up to 60W PD Power downstream.
2. A Hub with 4 downstream ports which initially provides 1 unit load (150mA) per Port plus 1 unit load for its internal functions.
3. Two Hard Disk Drives both of which require 5V@2A (10W) to spin up and 5V@1A (5W) while being accessed.
4. A phone which uses 5V@2A (10W) to charge and can give back all of this power when requested.

Table B-3 Giving back power

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
Connect Hub					
1	Connected to wall supply	Detached	Detached		Default
2	Hub is Attached	Attached, V_{BUS} powered			Default

Microchip Ex. 1020, p. 489
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
3	Laptop sends out a set of capabilities including: 5V@3A (15W), 12V@3A (36W), and 20V@3A (60W). The externally powered and USB suspend bits are set.	Source Capabilities received		Laptop sends out details of all available power via external supply	Default
4	The Hub requests 5V@0.15A. This is the power for the Hubs internal operation.	Request received		Hub needs 1 unit load for its own operation and so requests this amount.	Default
5	Send Accept	Accept received		Laptop evaluates request and it is within its available power.	0.75
6	Send PS_RDY	PS_RDY received. Starts to draw 5V@0.15A		Laptop indicates that its power supply is ready.	0.75
Connect Hard Disk Drive 1					
7		Attached detected.	Hard Disk Drive 1 is Attached to one of the downstream ports of the Hub.		0.75
8	Request received	The Hub requests 5V@0.3A (1.5W) from the Laptop.		Hub needs 0.75W for its own operation plus 0.75W for USB communication on one Port.	0.75
9	Accept sent	Accept received		Request is within available power so the laptop accepts.	1.5
10	PS_RDY sent	PS_RDY received		Laptop indicates that its power supply is ready	1.5

Microchip Ex. 1020, p. 490
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
11		Hub turns on V_{BUS} and sends out a set of capabilities to Hard Disk Drive 1 including: 5V@0.15A. The externally powered and USB suspend bits are set.	Source Capabilities received		1.5
12		Request received	Hard Disk Drive 1 requests 5V@0.15A from the Hub.	Hard Disk Drive 1 only needs one unit load when not operating so requests this.	1.5
13		Accept sent	Accept received	Request is within available power so the Hub accepts.	1.5
14		PS_RDY sent	PS_RDY received. The Hard Disk Drive starts drawing 1 unit load 5V@0.15A.	Laptop indicates its power supply is ready and the Hard Disk Drive starts drawing power.	1.5
Hard Disk Drive 1 spin up					
15		Request received	Hard Disk Drive 1 requests 5V@0.15A from the Hub but sets the Capability Mismatch bit.	Hard Disk Drive 1 needs 20V@0.5A to spin up but this is not available so it re-requests the available power flagging a capability mismatch.	1.5
16		Accept sent	Accept received	Request is within available power so the Hub accepts.	1.5
17		PS_RDY sent	PS_RDY received	Hard Disk Drive 1 indicates a capability mismatch to the user.	1.5

Microchip Ex. 1020, p. 491
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
18		The Hub requests the Sink Capabilities from Hard Disk Drive 1.	Get Sink Capabilities received	Due to the Capability Mismatch the Hub needs to determine what Hard Disk Drive 1 actually needs	1.5
19		Sink Capabilities received	Hard Disk Drive 1 returns capabilities indicating that it requires 5V@2A.		1.5
20	Request received	The Hub requests 5V@2.2A (11W) from the Laptop.		The Hub evaluates that it now needs 0.75W for the Hub and 10W for Hard Disk Drive 1.	1.5
21	Accept sent	Accept received		Power request from the Hub is within the Laptop's capabilities so the Laptop accepts the request.	11
22	PS_RDY sent	PS_RDY received		Laptop completes the Explicit Contract.	11
23		Hub sends out a set of capabilities to Hard Disk Drive 1 including: 5V@2A. The externally powered and USB suspend bits are set.	Source Capabilities received	Hub now offers Hard Disk Drive 1 what it needs.	11
24		Request received	Hard Disk Drive 1 requests 5V@2A operating current and indicates 5V@2A maximum current.	Hard Disk Drive 1 is operating at its maximum current to spin up so sets operating current = maximum current.	11
25		Accept sent	Accept received	Request is within the Hubs capabilities so it accepts.	11

Microchip Ex. 1020, p. 492
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
26		PS_RDY sent	PS_RDY received. Hard Disk Drive 1 starts to draw 5V@2A and spins up.	Hub indicates its power supply is ready so Hard Disk Drive 1 starts to draw power.	11
27		Request received	Once spun up Hard Disk Drive 1 requests 5V@1A operating current and 5V@2A maximum current.	Hard Disk Drive 1 is operating at a lower current so sets operating current < maximum current.	11
28		Accept sent	Accept received	The Hub will maintain a Power Reserve of 5V@1A (5W) for Hard Disk Drive 1 in addition to the 5V@1A (5W) it is currently using.	11
29		PS_RDY sent	PS_RDY received	Hub completes the Explicit Contract.	11
Hard Disk Drive 2 spin up					
30		Attach detected	Hard Disk Drive 2 is Attached to one of the downstream ports of the Hub.		11
31	Request received	The Hub requests 5V@2.3A (11.5W) from the Laptop.		The Hub needs 0.75W for itself, 0.75W for USB communication on one Port, 5W for Hard Disk Drive 1 operation and 5W for the Power Reserve.	11
32	Accept sent	Accept received		Power request from the Hub is within the Laptop's capabilities so it accepts the request.	11

Microchip Ex. 1020, p. 493
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
33	PS_RDY sent	PS_RDY received		Laptop indicates its power supply is ready.	11.5
34		Hub sends out a set of capabilities to Hard Disk Drive 2 including: 5V@0.15A. The externally powered and USB suspend bits are set.	Source Capabilities received by Hard Disk Drive 2	Hub offers Hard Disk Drive 2 enough power to enumerate.	11.5
35		Request received	Hard Disk Drive 2 requests 5V@0.15A from the Hub.		11.5
36		Accept sent to Hard Disk Drive 2	Accept received by Hard Disk Drive 2	Request is within available capabilities so the Hub accepts	11.5
37		PS_RDY sent to Hard Disk Drive 2.	PS_RDY received. Hard Disk Drive 2 starts drawing 5V@0.15A.	Hard Disk Drive 2 takes the power that it needs	11.5
Phone charge					
38		Attach detected	The phone is Attached to one of the downstream ports of the Hub.		11.5
39	Request received	The Hub Requests 5V@2.5A (12.5W) from the Laptop.		The Hub needs 0.75W for itself, 1.5W for USB communications on two ports (Hard Disk Drive 1 and the Phone), 5W for Hard Disk Drive 1 operation and 5W for the Power Reserve.	11.5

Microchip Ex. 1020, p. 494
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
40	Accept sent	Accept received		Request is within available capabilities so the Laptop accepts	12.5
41	PS_RDY sent	PS_RDY received		Laptop indicates that its power supply is ready.	12.5
42		The Hub powers V_{BUS} and sends out a set of capabilities to the Phone including: 5V@0.15A. The externally powered and USB suspend bits are set.	Source Capabilities received by the Phone	The Hub offers the Phone 1 unit load to enumerate.	12.5
43		Request received from the Phone	The Phone requests 5V@0.15A from the Hub but sets the Capability Mismatch bit.	The Phone would like to charge and so indicates this fact through the Capability Mismatch bit.	12.5
44		Accept sent	Accept received	Request is within available capabilities so the Hub accepts	12.5
45		PS_RDY sent	PS_RDY received	Hub indicates that its power supply is ready	12.5
46		The Hub requests the Sink Capabilities from the phone.	Get Sink Capabilities received by the Phone	Due to the Capability Mismatch the Hub needs to determine what the Phone actually needs	12.5
47		Sink Capabilities received from the Phone	The Phone returns capabilities indicating that it requires 5V@2A.	Phone returns the Capabilities it needs to charge	12.5

Microchip Ex. 1020, p. 495
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
48	Request received	The Hub Requests 9V@2.4A (21.6W) from the Laptop.		The Hub needs 0.75W for itself, 0.75W for Hard Disk Drive 2, 10W for the phone, 5W for Hard Disk Drive 1 operation and 5W for the Power Reserve.	12.5
49	Accept sent	Accept received		Request is within available capabilities so the Laptop accepts	12.5
50	PS_RDY sent	PS_RDY received		Laptop indicates that its power supply is ready.	21.6
51		The Hub sends out a set of capabilities to the Phone including: 5V@2A. The externally powered and USB suspend bits are set.	Source Capabilities received by the Phone	The Hub now has the power that the Phone needs and so sends out a new set of Capabilities.	21.6
52		Request received from the Phone	The Phone requests 5V@2A from the Hub and sets the No USB Suspend bit since it needs to charge constantly. It sets the GiveBack flag and sets the Minimum Operating Current to 5V@0A.	The Phone requests the power it needs to charge. It asks for the USB Suspend requirement to be removed.	21.6
53		Accept sent to the Phone	Accept received by the Phone		21.6
54		PS_RDY sent to the phone.	PS_RDY received by the phone. Phone starts to charge 5V@2A but has to follow USB Suspend rules		21.6

Microchip Ex. 1020, p. 496
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
55	Request received	The Hub Requests 9V@1.9A (17.1W) from the Laptop but sets the No USB Suspend bit.		The Hub needs 0.75W for itself, 0.75W for Hard Disk Drive 2, 10W for the phone (includes the Power Reserve of 5W), and 5W for Hard Disk Drive 1 operation. It requests for USB Suspend rule to be removed.	21.6
56	Accept sent	Accept received		Request is within available capabilities so the Laptop accepts. Note that the request for No Suspend has not been acted on by the Laptop. USB Suspend rules apply until the Laptop sends out new Source Capabilities with the USB Suspend bit cleared.	21.6
57	PS_RDY sent	PS_RDY received		Laptop indicates that its power supply is ready.	17.1
Hard Disk Drive 2 spin up					
58		Request received from Hard Disk Drive 2	Hard Disk Drive 2 requests 5V@0.15A from the Hub but sets the Capability Mismatch bit.	Hard Disk Drive 2 needs more power to spin up and so indicates a Capability Mismatch	17.1
59		Accept sent	Accept received	The request is within its capabilities so the Hub accepts.	17.1
60		PS_RDY sent	PS_RDY received	The Hub indicates that its power supply is ready.	17.1

Microchip Ex. 1020, p. 497
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
61		The Hub requests the Sink Capabilities from Hard Disk Drive 2.	Get Sink Capabilities received by Hard Disk Drive 2	Due to the Capability Mismatch the Hub has to determine what Hard Disk Drive 2 needs	17.1
62		Sink Capabilities received	Hard Disk Drive 2 returns capabilities indicating that it requires 20V@0.5A maximum current.		17.1
63		The Hub instructs the Phone to Goto Minimum operation.	Goto Min received by the Phone	Hub assess that there is additional power available from the Phone and so tells it to Goto Min. In this case it is reallocating the Phone's Charging power as the Power Reserve for the Hard Disk Drives.	17.1
64			The Phone drops to zero current draw.		17.1
65		PD_RDY sent	PS_RDY received.	Hub indicates that its power supply has changed to the new level.	17.1
66	Request received	The Hub Requests 9V@2.4A (21.6W) from the Laptop		The Hub has an additional 10W from the Phone but needs 5W more to maintain its Power Reserve. The Hub needs 0.75W for itself, 10W for Hard Disk Drive 2, 5W for the Power Reserve, 5W for Hard Disk Drive 1 operation.	17.1

Microchip Ex. 1020, p. 498
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
67	Accept sent	Accept received		Request is within available capabilities so the Laptop accepts.	17.1
68	PS_RDY sent	PS_RDY received		Laptop indicates that its power supply is ready.	21.6
69		Hub sends out a set of capabilities to Hard Disk Drive 2 including: 5V@0.5A and 20V@0.5A. The externally powered and USB suspend bits are set.	Source Capabilities received by Hard Disk Drive 2	The Hub now has the power that Hard Disk Drive 2 needs so it sends out new Capabilities.	21.6
70		Request received from Hard Disk Drive 2	Hard Disk Drive 2 requests 20V@0.5A operating current and 20V@0.5A.	Hard Disk Drive 2 requests what it needs to spin up.	21.6
71		Accept sent to Hard Disk Drive 2	Accept received by Hard Disk Drive 2	The Hub assesses that the request is within its Capabilities so it accepts.	21.6
72		PS_RDY sent.	PS_RDY sent. Hard Disk Drive 2 starts to draw 20V@0.5A and spins up.		21.6
73		Request received from Hard Disk Drive 2	Once spun up Hard Disk Drive 2 requests 20V@0.25A operating current and 20V@0.5A maximum current.	Hard Disk Drive 2 no longer needs the additional power so it gives back what it does not need.	21.6
74		Accept sent to Hard Disk Drive 2	Accept received by Hard Disk Drive 2	The Hub assesses that the request is within its Capabilities so it accepts.	21.6
75		PS_RDY sent to Hard Disk Drive 2.	PS_RDY received by Hard Disk Drive 2.	The Hub indicates that its power supply is ready.	21.6

Microchip Ex. 1020, p. 499
Microchip v. Aptiv
IPR2024-00496

Step	Laptop	Hub	Peripherals	Device Policy Manager	Hub Power (W)
76		The Hub sends out a set of capabilities to the Phone including: 5V@2A. The externally powered bit is set and the USB suspend bit is set.	Source Capabilities received by the Phone	The Hub now has the power available to charge the phone so it sends out new Capabilities	21.6
77		Request received from the Phone	The Phone requests 5V@2A operating current from the Hub and sets the No USB Suspend bit since it needs to charge constantly. It sets the GiveBack flag and sets the Minimum Operating Current to 5V@0A.	The Phone requests the power it needs to charge. It asks for the USB Suspend requirement to be removed.	21.6
78		Accept sent to the Phone	Accept received by the Phone	The Hub assesses that the request is within its Capabilities so it accepts but maintains USB Suspend rules.	21.6
79		PS_RDY sent to the Phone.	PS_RDY received by the Phone. The phone starts to draw 5V@2A but has to follow USB Suspend.	The Hub has allocated 0.75W for itself, 5W for Hard Disk Drive 2, 10W for the Phone (including 5W for the Power Reserve), and 5W for Hard Disk Drive 1 operation.	21.6

Microchip Ex. 1020, p. 500
Microchip v. Aptiv
IPR2024-00496

C. VDM Command Examples

C.1 Discover Identity Example

C.1.1Discover Identity Command request

Table C-1 below shows the contents of the key fields in the Message Header and VDM header for an Initiator sending a ***Discover Identity*** Command request.

Table C-1 Discover Identity Command request from Initiator Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	0xFF00 (***PD SID***)
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	000b
B7..6	Command Type	00b (Initiator)
B5	***Reserved***	0
B4..0	Command[1]	1 (***Discover Identity***)

C.1.2Discover Identity Command response – Active Cable

Table C-2 shows the contents of the key fields in the Message Header and VDM header for a Responder returning VDOs in response to a ***Discover SVIDs*** Command request. In this illustration, the responder is an active Gen2 cable which supports Modal Operation.

Table C-2 Discover Identity Command response from Active Cable Responder Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	5 (VDM Header + ID Header VDO + Cert Stat VDO + Product VDO + Cable VDO)
11..9	***MessageID***	0..7
8	***Cable Plug***	1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)

Microchip Ex. 1020, p. 501
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Value
VDM Header		
B31..16	Standard or Vendor ID (SVID)	0xFF00 (***PD SID***)
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	000b
B7..6	Command Type	01b (Responder ACK)
B5	***Reserved***	0
B4..0	Command	2 (***Discover Identity***)
ID Header VDO		
B31	Data Capable as USB Host	0 (not data capable as a Host)
B30	Data Capable as a USB Device	0 (not data capable as a Device)
B29..27	Product Type	100b (Active Cable)
B26	Modal Operation Supported	1 (supports Modes)
B25..16	***Reserved***. Shall be set to zero.	0
B15..0	16-bit unsigned integer. USB Vendor ID	USB-IF assigned VID for this cable vendor
Cert Stat VDO		
B31..0	32-bit unsigned integer	USB-IF assigned XID for this cable
Product VDO		
B31..16	16-bit unsigned integer. USB Product ID	Product ID assigned by the cable vendor
B15..0	16-bit unsigned integer. bcdDevice	Device version assigned by the cable vendor
Cable VDO (returned for Product Type "Active Cable")		
B31..28	HW Version	Cable HW version number (vendor defined)
B27..24	Firmware Version	Cable FW version number (vendor defined)
B23..21	VDO Version	00b (Version 1.0)
B20	***Reserved***	0
B19..18	USB Type-C plug to USB Type-C/Captive	10b (USB Type-C)
B17	***Reserved***	0
B16..13	Cable Latency	0001b (<10ns (~1m))
B12..11	Cable Termination Type	11b (Both ends Active, VCONN required)
B10..9	Maximum V_{BUS} Voltage	00b (20V)
B8..7	***Reserved***	0
B6..5	V_{BUS} Current Handling Capability	01b (3A)
B4	V_{BUS} through cable	1 (Yes)
B3	SOP" controller present?	1 (SOP" controller present)
B2..0	USB SuperSpeed Signaling Support	010b (***[USB 3.1]*** Gen1 and Gen2)

C.1.3Discover Identity Command response – Hub

Table C-2 shows the contents of the key fields in the Message Header and VDM header for a Responder returning VDOs in response to a ***Discover SVIDs*** Command request. In this illustration, the responder is a Hub

Microchip Ex. 1020, p. 502
Microchip v. Aptiv
IPR2024-00496

Table C-3 Discover Identity Command response from Hub Responder Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	4 (VDM Header + ID Header VDO + Cert Stat VDO + Product VDO)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	0xFF00 (***PD SID***)
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	000b
B7..6	Command Type	01b (Responder ACK)
B5	***Reserved***	0
B4..0	Command	2 (***Discover Identity***)
ID Header VDO		
B31	Data Capable as USB Host	0 (not data capable as a Host)
B30	Data Capable as a USB Device	1 (data capable as a Device)
B29..27	Product Type	001b (Hub)
B26	Modal Operation Supported	0 (doesn't support Modes)
B25..16	***Reserved***. Shall be set to zero.	0
B15..0	16-bit unsigned integer. USB Vendor ID	USB-IF assigned VID for this hub vendor
Cert Stat VDO		
B31..0	32-bit unsigned integer	USB-IF assigned XID for this hub
Product VDO		
B31..16	16-bit unsigned integer. USB Product ID	Product ID assigned by the hub vendor
B15..0	16-bit unsigned integer. bcdDevice	Device version assigned by the hub vendor

Microchip Ex. 1020, p. 503
Microchip v. Aptiv
IPR2024-00496

C.2 Discover SVIDs Example

C.2.1Discover SVIDs Command request

Table C-4 below shows the contents of the key fields in the Message Header and VDM header for an Initiator sending a ***Discover SVIDs*** Command request.

Table C-4 Discover SVIDs Command request from Initiator Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	0xFF00 (***PD SID***)
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	000b
B7..6	Command Type	00b (Initiator)
B5	***Reserved***	0
B4..0	Command[1]	2 (***Discover SVIDs***)

C.2.1Discover SVIDs Command response

Table C-5 shows the contents of the key fields in the Message Header and VDM Header for a Responder returning VDOs in response to a ***Discover SVIDs*** Command request. In this illustration, the value 3 in the Message Header indicates that one VDO containing the supported SVIDs would be returned followed by a terminating VDO. Note that the last VDO returned (the terminator of the transmission) contains zero value SVIDs. If a SVID value is zero, it is not used.

Table C-5 Discover SVIDs Command response from Responder Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	3 (VDM Header + 2*VDO)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	0xFF00 (***PD SID***)
B15	VDM Type	1 (Structured VDM)

Microchip Ex. 1020, p. 504
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Value
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b (***Reserved***)
B10..8	Object Position	000b
B7..6	Command Type	01b (Responder ACK)
B5	***Reserved***	0
B4..0	Command	2 (***Discover SVIDs***)
VDO 1		
B31..16	SVID 0	SVID value
B15..0	SVID 1	SVID value
VDO 2		
B31..16	SVID 2	0x0000
B15..0	SVID 3	0x0000

Microchip Ex. 1020, p. 505
Microchip v. Aptiv
IPR2024-00496

C.3 Discover Modes Example

C.3.1Discover Modes Command request

Table C-6 shows the contents of the key fields in the Message Header and VDM header for an Initiator sending a ***Discover Modes*** Command request. The Initiator of the ***Discover Modes*** Command sequence sends a Message Header with the ***Number of Data Objects*** field set to 1 followed by a VDM Header with the Command Type (B7..6) set to zero indicating the Command is from an Initiator and the Command (B4..0) is set to 3 indicating Mode discovery.

Table C-6 Discover Modes Command request from Initiator Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for which Modes are being requested
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	000b
B7..6	Command Type	00b (Initiator)
B5	***Reserved***	0
B4..0	Command[1]	3 (***Discover Modes***)

C.3.2Discover Modes Command response

The Responder to the ***Discover Modes*** Command request returns a Message Header with the ***Number of Data Objects*** field set to a value of 1 to 7 (the actual value is the number of Mode objects plus one) followed by a VDM Header with the Message Source (B5) set to 1 indicating the Command is from a Responder and the Command (B4..0) set to 2 indicating the following objects describe the Modes the device supports. If the ID is a VID, the structure and content of the VDO is left to the vendor. If the ID is a SID, the structure and content of the VDO is defined by the Standard.

Table C-7 shows the contents of the key fields in the Message Header and VDM Header for a Responder returning VDOs in response to a ***Discover Modes*** Command request. In this illustration, the value 2 in the Message Header indicates that the device is returning one VDO describing the Mode it supports. It is possible for a Responder to report up to six different Modes.

Table C-7 Discover Modes Command response from Responder Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	2 (VDM Header + 1 Mode VDO)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1

Microchip Ex. 1020, p. 506
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Value
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	Message Type	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for which Modes were requested
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	000b
B7..6	Command Type	01b (Responder ACK)
B5	***Reserved***	0
B4..0	Command	3 (***Discover Modes***)
Mode VDO		
B31..0	Mode 1	Standard or Vendor defined Mode value

Microchip Ex. 1020, p. 507
Microchip v. Aptiv
IPR2024-00496

C.4 Enter Mode Example

C.4.1Enter Mode Command request

The Initiator of the ***Enter Mode*** Command request sends a Message Header with the ***Number of Data Objects*** field set to 1 followed by a VDM Header with the Message Source (B5) set to zero indicating the Command is from an Initiator and the Command (B4..0) set to 4 to request the Responder to enter its mode of operation and sets the Object Position field to the desired functional VDO based on its offset as received during Discovery.

Table C-8 shows the contents of the key fields in the Message Header and VDM Header for an Initiator sending an ***Enter Mode*** Command request.

Table C-8 Enter Mode Command request from Initiator Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for the Mode being entered
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	001b (a one in this field indicates a request to enter the first Mode in list returned by Discover Modes)
B7..6	Command Type	00b (Initiator)
B5	***Reserved***	0
B4..0	Command	4 (***Enter Mode***)

C.4.2Enter Mode Command response

The Responder that is the target of the ***Enter Mode*** Command request shall send a Message Header with the ***Number of Data Objects*** field set to a value of 1 followed by a VDM Header with the Command Source (B5) set to 1 indicating the response is from a Responder and the Command (B4..0) set to 4 indicating the Responder has entered the Mode and is ready to operate.

Table C-9 shows the contents of the key fields in the Message Header and VDM Header for a Responder sending an ***Enter Mode*** Command response with an ACK.

Table C-9 Enter Mode Command response from Responder Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1

Microchip Ex. 1020, p. 508
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Value
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for the Mode entered
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	001b (offset of the Mode entered)
B7..6	Command Type	01b (Responder ACK)
B5	***Reserved***	0
B4..0	Command	4 (***Enter Mode***)

C.4.1Enter Mode Command request with additional VDO

The Initiator of the ***Enter Mode*** Command request sends a Message Header with the ***Number of Data Objects*** field set to 2 indicating an additional VDO followed by a VDM Header with the Message Source (B5) set to zero indicating the Command is from an Initiator and the Command (B4..0) set to 4 to request the Responder to enter its mode of operation and sets the Object Position field to the desired functional VDO based on its offset as received during Discovery.

Table C-8 shows the contents of the key fields in the Message Header and VDM Header for an Initiator sending an ***Enter Mode*** Command request with an additional VDO.

Table C-10 Enter Mode Command request from Initiator Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for the Mode being entered
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	001b (a one in this field indicates a request to enter the first Mode in list returned by Discover Modes)
B7..6	Command Type	00b (Initiator)
B5	***Reserved***	0
B4..0	Command	4 (***Enter Mode***)
Including *Optional* Mode specific VDO		
B31..0	Mode specific	

Microchip Ex. 1020, p. 509
Microchip v. Aptiv
IPR2024-00496

C.5 Exit Mode Example

C.5.1Exit Mode Command request

The Initiator of the ***Exit Mode*** Command request sends a Message Header with the ***Number of Data Objects*** field set to 1 followed by a VDM Header with the Message Source (B5) set to zero indicating the Command is from an Initiator and the Command (B4..0) set to 5 to request the Responder to exit its Mode of operation.

Table C-11 shows the contents of the key fields in the Message Header and VDM header for an Initiator sending an ***Exit Mode*** Command request.

Table C-11 Exit Mode Command request from Initiator Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for the Mode being exited
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	001b (identifies the previously entered Mode by its Object Position that is to be exited)
B7..6	Command Type	00b (Initiator)
B5	***Reserved***	0
B4..0	Command	5 (***Exit Mode***)

C.5.2Exit Mode Command response

The Responder that receives the ***Exit Mode*** Command request sends a Message Header with the ***Number of Data Objects*** field set to a value of 1 followed by a VDM Header with the Message Source (B5) set to 1 indicating the Command is from a Responder and the Command (B4..0) set to 5 indicating the Responder has exited the Mode and has returned to normal USB operation.

Table C-12 shows the contents of the key fields in the Message Header and VDM header for a Responder sending an ***Exit Mode*** Command ACK response.

Table C-12 Exit Mode Command response from Responder Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1

Microchip Ex. 1020, p. 510
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Value
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for the Mode exited
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	001b (offset of the Mode to be exited)
B7..6	Command Type	01b (Responder ACK)
B5	***Reserved***	0
B4..0	Command	5 (***Exit Mode***)

Microchip Ex. 1020, p. 511
Microchip v. Aptiv
IPR2024-00496

C.6 Attention Example

C.6.1Attention Command request

The Initiator of the ***Attention*** Command request sends a Message Header with the ***Number of Data Objects*** field set to 1 followed by a VDM Header with the Message Source (B5) set to zero indicating the Command is from an Initiator and the Command (B4..0) set to 6 to request attention from the Responder.

Table C-11 shows the contents of the key fields in the Message Header and VDM header for an Initiator sending an ***Attention*** Command request.

Table C-13 Attention Command request from Initiator Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	1 (VDM Header)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for which attention is being requested
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	001b (offset of the Mode requesting attention)
B7..6	Command Type	000b (Initiator)
B5	***Reserved***	0
B4..0	Command	6 (***Attention***)

C.6.2Attention Command request with additional VDO

The Initiator of the ***Attention*** Command request sends a Message Header with the ***Number of Data Objects*** field set to 2 indicating an additional VDO followed by a VDM Header with the Message Source (B5) set to zero indicating the Command is from an Initiator and the Command (B4..0) set to 6 to request attention from the Responder.

Table C-11 shows the contents of the key fields in the Message Header and VDM header for an Initiator sending an ***Attention*** Command request with an additional VDO.

Table C-14 Attention Command request from Initiator with additional VDO Example

Bit(s)	Field	Value
Message Header		
15	***Reserved***	0
14..12	***Number of Data Objects***	2 (VDM Header + VDO)
11..9	***MessageID***	0..7
8	***Port Power Role***	0 or 1
7..6	***Specification Revision***	10b (Revision 3.0)
5..4	***Reserved***	0

Microchip Ex. 1020, p. 512
Microchip v. Aptiv
IPR2024-00496

Bit(s)	Field	Value
3..0	***Message Type***	1111b (Vendor Defined Message)
VDM Header		
B31..16	Standard or Vendor ID (SVID)	SVID for which attention is being requested
B15	VDM Type	1 (Structured VDM)
B14..13	Structured VDM Version	01b (Version 2.0)
B12..11	***Reserved***	00b
B10..8	Object Position	001b (offset of the Mode requesting attention)
B7..6	Command Type	000b (Initiator)
B5	***Reserved***	0
B4..0	Command	6 (***Attention***)
Including *Optional* Mode specific VDO		
B31..0	Mode specific	

Microchip Ex. 1020, p. 513
Microchip v. Aptiv
IPR2024-00496