USB Power Delivery Exerciser Manual

Manual Version 1.3

For USB Protocol Suite Software Version 7.30 and above

April 2016

Document Disclaimer

The information contained in this document has been carefully checked and is believed to be reliable. However, no responsibility can be assumed for inaccuracies that may not have been detected.

Teledyne LeCroy reserves the right to revise the information presented in this document without notice or penalty.

Trademarks and Servicemarks

CATC Trace, Voyager M310C, Voyager ReadyLink, USB Protocol Suite, and *BusEngine* are trademarks of Teledyne LeCroy.

All other trademarks are property of their respective companies.

Copyright

Version

This is version 1.20 of the *USB Power Delivery Exerciser Manual*.

This manual applies to USB Protocol Suite software version 7.20 and higher.

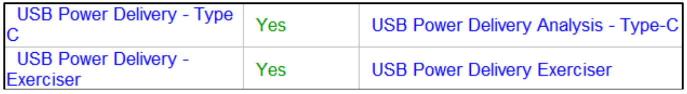
Contents

1 Introduction

Integrated in Teledyne LeCroy's Voyager M310C test platform, the Power Delivery exerciser supports traffic generation, including both provider and consumer device emulation. The Power Delivery exerciser continues to evolve with each software release. Be sure to check for updated software and firmware before getting started with the Exerciser.

Important Licensing Note:

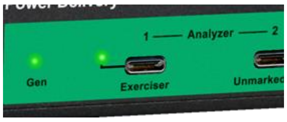
- Operating the PD Exerciser beta requires that the USB Power Delivery Exerciser option is enabled on the M310C base unit:



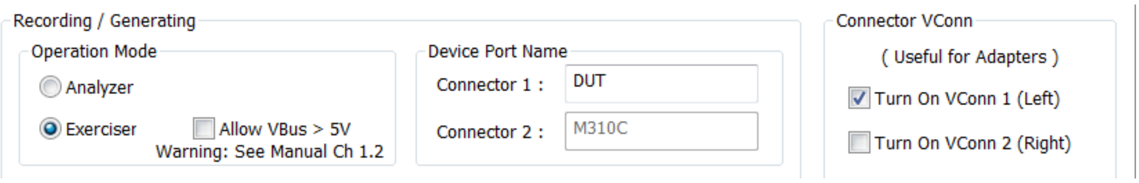
USB Power Delivery - Type C	Yes	USB Power Delivery Analysis - Type-C
USB Power Delivery - Exerciser	Yes	USB Power Delivery Exerciser

Getting Started:

- The "left" port of the Voyager should be used to connect DUT to the PD Exerciser. The PD exerciser also requires specific cable orientation (Red LED when connected wrong side-up).



- To enable the PD Trainer/Exerciser, use the PD Tab under *"Recording Options"* to select the Exerciser mode.



Recording / Generating

Operation Mode

○ Analyzer

● Exerciser □ Allow VBus > 5V
Warning: See Manual Ch 1.2

Device Port Name

Connector 1 : DUT

Connector 2 : M310C

Connector VConn

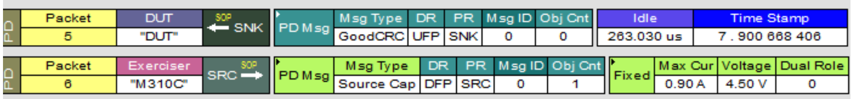
(Useful for Adapters)

☑ Turn On VConn 1 (Left)

□ Turn On VConn 2 (Right)

Note – *Allow VBUS > 5v* is a safety feature which prevents sourcing above 5V. When enabled, this mode will allow Voltage levels to be delivered to the DUT which may exceed their current carrying capabilities. While the M310C system is designed to tolerate higher current, these higher voltages may inadvertently cause damage to devices/cables under test.

Note – This window now allows *device port naming*. Use this option to add "alias labels" for your DUTs which will appear in the trace capture.

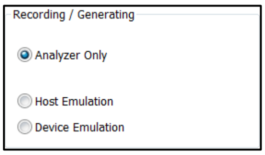


PD	Packet	DUT	SOP	PD Msg	Msg Type	DR	PR	Msg ID	Obj Cnt	Idle	Time Stamp
	5	"DUT"	← SNK		GoodCRC	UFP	SNK	0	0	263.030 us	7 . 900 668 406

PD	Packet	Exerciser	SOP	PD Msg	Msg Type	DR	PR	Msg ID	Obj Cnt	Fixed	Max Cur	Voltage	Dual Role
	6	"M310C"	SRC →		Source Cap	DFP	SRC	0	1		0.90 A	4.50 V	0

The Alias name is primarily for use in analyzer mode and requires that device names are added before recording traffic. The device naming can also be used in Exerciser mode; however message frames from the Voyager M310C will be always be labeled "M310C".

- Within the USB 3.1 tab – *"Recording/Generating"* option - leave in 'Analyzer Only' mode



Recording / Generating

● Analyzer Only

○ Host Emulation

○ Device Emulation

- Use the example PD Exerciser scripts to begin testing:

C:\Users\Public\Documents\LeCroy\USB Protocol Suite\Examples\Power Delivery Exerciser

Example Script	Behavior
Source Power Negotiate VDM.updg	Voyager as Source negotiates default Provider 900mA@4.5V then sends Discover-Id. Using Basic Commands.
High Level Negotiate with dynamic change cap.updg	Voyager as Source negotiates default Provider 1A@5V then broadcasts lower PDO 900mA@4.5V and re-negotiates. Using High Level Commands.
Discover Cable.updg	Voyager as Source programmatically turns on VCONN and sends Discover ID; then Discover/Enter Mode Cypress VDM (0x04B4); turns off VCONN. Using High Level Commands.
Sink Power Negotiate.updg	Voyager as Sink Waits to receive Source cap then negotiates as Sink - 900mA@5V. Using Basic Commands.
Apple VGA multiple Adaptor.updg	Voyager as Source enables VCONN and Sends Discover Id; Discover Mode for Apple SVID (0x05AC); Enter Mode (PD_DISPLAY_PORT_SVID) then Exit Mode; turns off VCONN. Using Basic Commands.
High Level Device Discovery.updg	Voyager as Source sends Discover Id; Discover SVIDs; Discover Modes for Display Port SVID (0xFF01); Enter Mode (0xFF01); Exit Mode (0xFF01); Discover Modes for Apple SVID (0x05AC); Enter Mode(0x5AC mode 1); Exit Mode(0x5AC mode 1); Enter Mode(0x5AC mode 2); Exit Mode(0x5AC mode 2); Using High Level Commands.
NegotiationSample_WithSwapPowerRole.updg	Voyager as Source sends SwapPowerRole; and negotiates as a Sink after power role swap. 1.5A@5V. Using High Level Commands.
Sink Auto Response.updg	Voyager as Sink will response to all incoming PD messages within 100s. Using Auto Response Command.

- To Run Sample Script – Connect Cable to Exerciser port; Click *Record*, wait a few seconds and Click *Run.* The PD Exerciser uses the sequence below at the beginning of each example script to simulate a re-connect event.

```
call PD_Disconnect()

call PD_SetResistorRp( PD_ON, PD_HARDWARE_VALUE_RP_DEFAULT )
call PD_SetVBus( PD_ON )
```

Note- it's also possible to execute the example scripts before the cable is connected to M310C then performing "hot-plug" (It's possible some issues may be seen with some devices not responding to exerciser in this case).
Note – some latency may be observed when activating/downloading PD exerciser scripts (Run button) This will be improved in a future release.

2 Packet Templates

List of available Power Delivery messages:

Packet Template Names	Description
PD_ControlMessage	
PD_GoodCrcMessage	
PD_GotoMinMessage	
PD_AcceptMessage	
PD_RejectMessage	
PD_PingMessage	
PD_PsRdyMessage	
PD_GetSourceCapMessage	
PD_GetSinkCapMessage	
PD_DataRoleSwapMessage	
PD_PowerRoleSwapMessage	
PD_VconnSwapMessage	
PD_WaitMessage	
PD_SoftResetMessage	
PD_SourceCapabilitiesMessage	
PD_PowerDataObjectFixedSupply_Source	Use as SourceCapabilitiesData for PD_SourceCapabilitiesMessage
Pd_PDOFixedSupplyNotVSafe5V_Source	Use as SourceCapabilitiesData for PD_SourceCapabilitiesMessage
PD_PowerDataObjectVariableSupply_Source	Use as SourceCapabilitiesData for PD_SourceCapabilitiesMessage
PD_PowerDataObjectBatterySupply_Source	Use as SourceCapabilitiesData for PD_SourceCapabilitiesMessage
PD_SinkCapabilitiesMessage	
PD_PowerDataObjectFixedSupply_Sink	Use as SinkCapabilitiesData for PD_SinkCapabilitiesMessage
PD_PowerDataObjectVariableSupply_Sink	Use as SinkCapabilitiesData for PD_SinkCapabilitiesMessage
PD_PowerDataObjectBatterySupply_Sink	Use as SinkCapabilitiesData for PD_SinkCapabilitiesMessage
PD_RequestPacket	
PD_RequestDataObject_Fixed_Variable_NoGiveBack	Use as data for PD_RequestDataObject
PD_RequestDataObject_Fixed_Variable_GiveBack	Use as data for PD_RequestDataObject
PD_RequestDataObject_Battery_NoGiveBack	Use as data for PD_RequestDataObject
PD_RequestDataObject_Battery_GiveBack	Use as data for PD_RequestDataObject
PD_BISTMessage	
PD_BISTCarrierModeMessage	
PD_BISTTestDataMessage	
PD_VDM_Unstructured_Header	Use as Header for Unstructured VDM messages
PD_VDM_Structured_Header	Will be used as Header for all Structured VDM messages which are described below
PD_VDM_Discover_Identity_Message	
PD_VDM_Discover_Identity_Response	
PD_VDM_Discover_Identity_ID_Header_VDO	Use as VDOs for PD_VDM_Discover_Identity_Response
PD_VDM_Discover_Identity_Cert_Stat_VDO	Use as VDOs for PD_VDM_Discover_Identity_Response
PD_VDM_Discover_Identity_Product_VDO	Use as VDOs for PD_VDM_Discover_Identity_Response
PD_VDM_Discover_Identity_Cable_VDO	Use as VDOs for PD_VDM_Discover_Identity_Response
PD_VDM_Discover_Identity_Alternate_Mode_Adapter_VDO	Use as VDOs for PD_VDM_Discover_Identity_Response

PD_VDM_Discover_Svids_Message	
PD_VDM_Discover_Svids_Response	
Discover_SVIDs_Responder_VDO	Use as DiscoverSVIDsResponderVDOs for PD_VDM_Discover_Svids_Response
PD_VDM_Discover_Modes_Message	
PD_VDM_Discover_Modes_Response	
PD_VDO	Use as VDO for PD_VDM_Enter_Mode_Message and PD_VDM_Attention_Message and PD_VDM_SVID_Specific_Message
PD_VDM_Enter_Mode_Message	
PD_VDM_Enter_Mode_Response	
PD_VDM_Exit_Mode_Message	
PD_VDM_Exit_Mode_Response	
PD_VDM_Attention_Message	
PD_VDM_DisplayPort_UpdateStatus_Message	
PD_VDM_DisplayPort_UpdateStatus_Response	
PD_VDM_DisplayPort_Configure_Message	
PD_VDM_DisplayPort_Configure_Response	
PD VDM DisplayPort DiscoverMode Vdo	Use in DiscoverModes Response
PD_VDM_DisplayPort_Status_VDO	Use in Display Port UpdateStatus Initiator and Responder
PD_VDM_DisplayPort_Configure_VDO	Use in Display Port Configure Initiator

3 Basic Commands

List of available Basic Commands:

Command Name	Description
PD_SendPacket	Sends the data payload towards the device. You can customize its behavior using provided settings.
PD_SendPacket_Cable	Sends a packet as a Marked Cable towards the device.
PD_ReceivePacket	Receives a packet from device. You can specify the packet type using its settings.
PD_SendSoftReset	Sends Soft Reset and performs the reset.
PD_SendHardReset	Sends Hard Reset and performs the reset.
PD_SendCableReset	Sends Cable Reset and performs the reset.
PD_Delay	Exerciser delay
PD_SetRoles	Sets Data Role and Power Role
PD_SetResistorRp	Sets resistor Rp On/Off
PD_SetResistorRd	Sets resistor Rd On/Off
PD_SetResistorRa	Sets resistor Ra On/Off
Error! Reference source not found.	Sets resistor Rp/Rd/Ra, On/Off for CC1/CC2 line
PD_SetVBusCap10MicroFarad	Sets the VBus Capacitor(10 Micro Farad) On/Off
PD_SetVBusCap1MicroFarad	Sets the VBus Capacitor(1 Micro Farad) On/Off
PD_SetVbus	Sets VBus On/Off.
PD_SetVConn	Sets VConn On/Off.
PD_SetLoadOnVbus	Sets load on VBus.
PD_Set	Using this command you can change necessary settings or variables inside the Exerciser.
IfMatched/ElseMatched	Using this command you can compare Exerciser settings or variables to other Exerciser settings or variables or to a constant.
PD_Loop	Using this command you can create a loop containing other Exerciser commands.
PD_BreakLoop	Breaks the PD_Loop command.
PD_ContinueLoop	Continue command for PD_Loop.
PD_Stop	Stops the Exerciser.
PD_Disconnect	Simulates cable detach.
PD_IncreaseMsgId	Increase Message ID(Exerciser mode: DFP/UFP)
PD_DecreaseMsgId	Decrease Message ID(Exerciser mode: DFP/UFP)
PD_IncreaseMsgId_Cable	Increase Message ID(Exerciser mode: Cable Emulator)
PD_DecreaseMsgId_Cable	Decrease Message ID(Exerciser mode: Cable Emulator)

3.1 PD_SendPacket

Sends specified packet to device.

Format

```
Call PD_SendPacket(PD_Packet $send_packet, PD_SendPacketSettings $settings)
```

Parameters

`$send_packet`

Defines the payload. It should be inherited from `PD_Packet` template (Packet Templates).

`$settings`

Settings for sending packet. It should be inherited from `PD_SendPacketSettings` template. Table below shows `PD_SendPacketSettings` structure in detail:

Field Name	Description
OrderedSetType	Defines Ordered set type. Possible values: PD_ORDERED_SET_TYPE_SOP(default) PD_ORDERED_SET_TYPE_SOP_PRIME PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME PD_ORDERED_SET_TYPE_HARDRESET

	PD_ORDERED_SET_TYPE_CABLERESET
WaitForGoodCrc	If the command should wait for peer GoodCrc message. Possible values: PD_TRUE(default) PD_FALSE
ResetOnError	Send Soft Reset if relative GoodCrc has not been received, in case of sending SoftReset failure, HardReset will be sent. Possible values: PD_TRUE(default) PD_FALSE
RetryCount	Indicates the Retry Count. Default: PD_DEFAULT_RETRY_COUNT(3)
RetryDelayTime	Delay time between two consecutive retries. Default: 0
AutoMessageId	To increase MessageId automatically. Possible values: PD_TRUE(default) PD_FALSE

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of possible result values:

Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received for sent packet
PD_SUBRESULT_HARDRESET	Subresult - Hardreset received
PD_SUBRESULT_SOFTRESET	Subresult - Softreset received

Examples

```
#send a discover identity command
#################################
$send_setting = PD_SendPacketSettings
{
    # could be PD_ORDERED_SET_TYPE_SOP_PRIME for cables
    OrderedSetType = PD_ORDERED_SET_TYPE_SOP
}
$discover_identity = PD_VDM_Discover_Identity_Message
Call PD_SendPacket( $discover_identity, $send_setting )

# Send Request message
######################
$request_data = PD_RequestDataObject_Fixed_Variable_NoGiveBack
{
    MaxOperatingCurrent_10mAUnits = 90
    OperatingCurrent_10mAUnits = 90
}
$request_packet = PD_RequestPacket
{
    Data = $request_data
}
#calling PD_SendPacket() command using default settings
$send_packet_settings = PD_SendPacketSettings
Call PD_SendPacket($request_packet, $send_packet_settings)
```

3.2 PD_SendPacket_Cable

Sends specified packet to device as a Marked Cable.

Format

```
Call PD_SendPacket_Cable( PD_Packet $send_packet,
                   PD_SendPacketSettings_Cable $settings )
```

Parameters

`$send_packet`

Defines the payload. It should be inherited from `PD_Packet` template (Packet Templates).

`$settings`

Settings for sending packet. It should be derived from `PD_SendPacketSettings_Cable` template. `PD_SendPacketSettings_Cable` is derived from `PD_SendPacketSettings` template. Default values for some fields is changed as below:

```
OrderedSetType = PD_ORDERED_SET_TYPE_SOP_PRIME
ResetOnErrors = PD_FALSE
RetryCount = 0
```

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of possible result values:

Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received for sent packet
PD_SUBRESULT_HARDRESET	Subresult - Hardreset received

Examples

```
#send a discover identity response
####################################
$send_setting = PD_SendPacketSettings_Cable

$header_vdo = PD_VDM_Discover_Identity_ID_Header_VDO
$stat_vdo = PD_VDM_Discover_Identity_Cert_Stat_VDO
$product_vdo = PD_VDM_Discover_Identity_Product_VDO
$cable_vdo = PD_VDM_Discover_Identity_Cable_VDO

$discover_identity_response = PD_VDM_Discover_Identity_Response
{
    VDOs = $header_vdo + $stat_vdo + $product_vdo + $cable_vdo
}
Call PD_SendPacket_Cable( $discover_identity_response, $send_setting )
```

3.3 PD_ReceivePacket

Wait to receive specified packet from device.

Format

```
Call PD_ReceivePacket( PD_ReceivePacketSettings $receive_Settings )
```

Parameters

`$receive_Settings`

Settings for receiving packet. The structure type should be `PD_ReceivePacketSettings`.
Table below shows this structure in detail:

Field Name	Description
OrderedSetType	Ordered set type for receiving message. Possible values: PD_ORDERED_SET_TYPE_SOP(default) PD_ORDERED_SET_TYPE_SOP_PRIME PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME PD_ORDERED_SET_TYPE_HARDRESET PD_ORDERED_SET_TYPE_CABLERESET
PacketType	Message type to receive. Possible values: PD_MESSAGE_TYPE_ANY(default) PD_MESSAGE_TYPE_GOODCRC PD_MESSAGE_TYPE_GOTO_MIN PD_MESSAGE_TYPE_ACCEPT

	PD_MESSAGE_TYPE_REJECT PD_MESSAGE_TYPE_PING PD_MESSAGE_TYPE_PS_RDY PD_MESSAGE_TYPE_GET_SOURCE_CAP PD_MESSAGE_TYPE_GET_SINK_CAP PD_MESSAGE_TYPE_DR_SWAP PD_MESSAGE_TYPE_PR_SWAP PD_MESSAGE_TYPE_VCONN_SWAP PD_MESSAGE_TYPE_WAIT PD_MESSAGE_TYPE_SOFT_RESET PD_MESSAGE_TYPE_SOURCE_CAP PD_MESSAGE_TYPE_REQUEST PD_MESSAGE_TYPE_BIST PD_MESSAGE_TYPE_SINK_CAP PD_MESSAGE_TYPE_VDM
VdmCommand	VDM command. Possible values: PD_VDM_COMMAND_ANY(default) PD_VDM_COMMAND_DISCOVER_IDENTITY PD_VDM_COMMAND_DISCOVER_SVIDS PD_VDM_COMMAND_DISCOVER_MODES PD_VDM_COMMAND_ENTER_MODE PD_VDM_COMMAND_EXIT_MODE PD_VDM_COMMAND_DISPLAYPORT_STATUS_UPDATE PD_VDM_COMMAND_DISPLAYPORT_CONFIGURE PD_VDM_COMMAND_ATTENTION
VdmCommandType	VDM command type. Possible values: PD_VDM_COMMAND_TYPE_INITIATOR(default) PD_VDM_COMMAND_TYPE_RESPONDER_ACK PD_VDM_COMMAND_TYPE_RESPONDER_NAK PD_VDM_COMMAND_TYPE_RESPONDER_BUSY PD_VDM_COMMAND_TYPE_ANY
AutoGoodCrc	Send GoodCrc on receiving a message, automatically. Possible values: PD_TRUE(default) PD_FALSE
DelayBeforeGoodCrc	Delay before sending GoodCrc message. Default: 0
WaitTimeOut	Receive timeout. Possible values: PD_DEFAULT_TIMEOUT_SENDER_RESPONSE(default) PD_DEFAULT_TIMEOUT_INFINIT
DiscardPrevReceived	Discards any (unprocessed) packet received before calling PD_ReceivePacket function. Possible values: PD_TRUE PD_FALSE(default)
ReturnOnUnexpectedPkt	If set to PD_TRUE, cause PD_ReceivePacket() function to return on receiving unexpected packet. Possible values: PD_TRUE PD_FALSE(default)

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - No packet received within specified time
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected packet received
PD_SUBRESULT_HARDRESET	Subresult - HardReset received
PD_SUBRESULT_SOFTRESET	Subresult - SoftReset received

Examples

```
#Receive source caps
####################
# Wait to receive source capability. GoodCRC is sent automatically.
$recv_settings = PD_ReceivePacketSettings
{
    WaitTimeOut = PD_DEFAULT_TIMEOUT_INFINIT
    PacketType = PD_MESSAGE_TYPE_SOURCE_CAP
}
call PD_ReceivePacket($recv_settings)
```

```
#Receive VDM message
####################
$receive_settings = PD_ReceivePacketSettings
{
    PacketType = PD_MESSAGE_TYPE_VDM
}
call PD_ReceivePacket( $receive_settings )
```

3.4 PD_SendSoftReset

Sends Soft Reset packet.

Format

```
Call PD_SendSoftReset( orderedset_type )
```

Parameters

`orderedset_type`

List of possible ordered set types:

OrderedSet Type	Description
PD_ORDERED_SET_TYPE_SOP	
PD_ORDERED_SET_TYPE_SOP_PRIME	
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME	

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive (accept packet) timeout
PD_SUBRESULT_HARDRESET	Subresult - HardReset received
PD_SUBRESULT_SOFTRESET	Subresult - SoftReset received

Examples

```
Call PD_SendSoftReset( PD_ORDERED_SET_TYPE_SOP )
```

3.5 PD_SendHardReset

Sends Hard Reset packet.

Note - If PD Exerciser Power Role Data is `PD_PORT_POWER_ROLE_SOURCE`, after sending Hard Reset user should manually call `Pd_NegotiatePower_Source()`.

Format

```
Call PD_SendHardReset()
```

Parameters

Result

Examples

```
Call PD_SendHardReset()
```

3.6 PD_SendCableReset

Sends Cable Reset packet.

Format

```
Call PD_SendCableReset()
```

Parameters

Result

Examples

```
Call PD_SendCableReset()
```

3.7 PD_Delay

Delays Exerciser for specified time.

Format

```
Call PD_Delay( delay_value )
```

Parameters

`delay_value`

Delay in micro seconds.

Result

Examples

```
#calling PD_Delay
Call PD_Delay(15000)
```

3.8 PD_SetRoles

Sets data role and power role of Exerciser initially.

Format

```
Call PD_SetRoles( DataRole, PowerRole )
```

Parameters

`DataRole`

Possible values:

```
PD_PORT_DATA_ROLE_UFP
PD_PORT_DATA_ROLE_DFP
```

`PowerRole`

Possible values:

```
PD_PORT_POWER_ROLE_SINK
PD_PORT_POWER_ROLE_SOURCE
```

Examples

```
Call PD_SetRoles( PD_PORT_DATA_ROLE_DFP, PD_PORT_POWER_ROLE_SOURCE )
```

3.9 PD_SetResistorRp

Format

```
Call PD_SetResistorRp( state, current, line )
```

Parameters

`state`

Possible values: PD_ON, PD_OFF

`current`

Possible values:

```
    CC_RP_CUR_DEFAULT
    CC_RP_CUR_1_5
    CC_RP_CUR_3_0
```

`line`

Possible values:

```
  CC_LINE_1
  CC_LINE_2
  CC_LINE_ALL
```

Examples

```
Call PD_SetResistorRP( PD_ON, CC_RP_CUR_1_5, CC_LINE_2 )

# default line is CC_LINE_1
Call PD_SetResistorRP( PD_OFF )
```

3.10 PD_SetResistorRd

Format

```
Call PD_SetResistorRd( state, line )
```

Parameters

`state`

Possible values:

```
    PD_ON
    PD_OFF
```

`line`

Possible values:

```
  CC_LINE_1
  CC_LINE_2
  CC_LINE_ALL
```

Examples

```
Call PD_SetResistorRd( PD_ON, CC_LINE_1 )

# CC_LINE_1 is default line
Call PD_SetResistorRd( PD_ON )
```

3.11 PD_SetResistorRa

Format

```
Call PD_SetResistorRa( state, line )
```

Parameters

`state`

Possible values:

```
    PD_ON
    PD_OFF
```

`line`

Possible values:

```
  CC_LINE_1
```

```
    CC_LINE_2
    CC_LINE_ALL
```

Examples

```
Call PD_SetResistorRa( PD_ON, CC_LINE_2 )

# CC_LINE_1 is default line
Call PD_SetResistorRa( PD_OFF )
```

3.12 PD_SetVBusCap10MicroFarad

Sets the VBus Capacitor(A) On or Off.

Format

```
Call PD_SetVBusCap10MicroFarad( state )
```

Parameters

`state`

Possible values:

```
PD_ON
PD_OFF
```

Examples

```
Call PD_SetVBusCap10MicroFarad( PD_ON )
```

3.13 PD_SetVBusCap1MicroFarad

Sets the VBus Capacitor(B) On or Off.

Format

```
Call PD_SetVBusCap1MicroFarad( state )
```

Parameters

`state`

Possible values:

```
PD_ON
PD_OFF
```

Examples

```
Call PD_SetVBusCap1MicroFarad( PD_ON )
```

3.14 PD_SetVbus

Format

```
Call PD_SetVbus( state, voltage_milli_volt )
```

Parameters

`state`

Possible values:

```
PD_ON
PD_OFF
```

`voltage_milli_volt`

The voltage which applied on VBus. Voltage should be in range of 5000 to 20500 mV. In order to apply voltages greater than 5V, the corresponding check box should be set in recording options.

Examples

```
Call PD_SetVBus( PD_ON, 5000 )
```

3.15 PD_SetVConn

Format

```
Call PD_SetVConn( state )
```

Parameters

`state`

Possible values:

PD_ON
PD_OFF

Examples

```
Call PD_SetVConn( PD_ON )
```

3.16 PD_SetLoadOnVbus

Format

```
Call PD_SetLoadOnVbus( state )
```

Parameters

`state`

Possible values:

PD_ON
PD_OFF

Examples

```
Call PD_SetLoadOnVbus( PD_ON )
```

3.17 PD_Set

Sets Exerciser settings.

Format

```
PD_Set $PdGlobalSettings.<field_name> = <value>
PD_Set $PdTimers.<field_name> = <value>
```

Parameters

List of `$PdGlobalSettings` fields:

Field Name	Description
PortDataRole	Defines port data role. Possible values: PD_PORT_DATA_ROLE_DFP(default) PD_PORT_DATA_ROLE_UFP
PortPowerRole	Defines port power role. Possible values: PD_PORT_POWER_ROLE_SINK(default) PD_PORT_POWER_ROLE_SOURCE
CheckMessageId	Enables/Disables received packet message id verification. *It should be set at the top of `Main` block*. Possible values: PD_FALSE(Default) PD_TRUE
SpecificationRevision	Changes the `SpecificationRevision` of all messages sent by the Exerciser. *It should be set at the top of `Main` block*. Possible values: PD_SPEC_REVISION_1 PD_SPEC_REVISION_2(Default) Or any user defined value.
EnableCableEmulator	Enables/Disables Cable Emulator engine in Exerciser. If enabled the Exerciser simulates a Marked Cable as well as source or sink PD Device. *It should be set at the top of `Main` block*. Possible values: PD_FALSE(Default)

	PD_TRUE

List of $PdTimers fields(for detailed description refer to Power Delivery Specification):

Field Name	Description
tTypeCSendSourceCap	Default: 150000 us
tPSTransition	Default: 550000 us
tPSSourceOff	Default: 900000 us
tPSSourceOn	Default: 450000 us
tSrcTransition	Default: 30000 us
tDiscoverIdentity	Default: 45000 us
tPSHardReset	Default: 30000 us
tSrcRecover	Default: 1000000 us
tReceive	Default: 1100 us
tVCONNSourceOff	Default: 25000 us
tVCONNSourceOn	Default: 100000 us
tSenderResponse	Default: 30000 us
tBISTContMode	Default: 60000 us
tVDMBusy	Default: 50000 us
tVDMWaitModeEntry	Default: 50000 us
tVDMWaitModeExit	Default: 50000 us
tDRP	Default: 80000 us
dcSRC_DRP	Default: 50(time percent)
tCCDebounce	Default: 150000 us
tCCDebounceMin	Default: 100000 us
tCCDebounceMax	Default: 200000 us

Note - $PdTimers fields should be set at the top of Main block only.

Result

Examples

```
# Sets GoodCRC timeout
PD_Set $PdTimers.tReceive = 950

# Enables cable emulator
PD_Set $PdGLOBALSETTINGS.EnableCableEmulator = PD_TRUE

Main
{
    Call PD_WaitForDiscoverIdentity_Cable()
}
```

3.18 IfMatched/ElseMatched

Compares Exerciser settings, Received Packet Fields and Command Results to a desired value.

Format

```
Ifmatched(<1st_operand>, <2nd_operand>, <operator>)
{
    #command list
}
[
ElseMatched(<1st_operand>, <2nd_operand>, <operator>)
{
    #command list
}
#more optional ElseMatched(<1st_operand>, <2nd_operand>, <operator>) here
   .
   .
   .
ElseMatched
{
    #command list
}
]
IfMatchedEnd

* ElseMatched clause is optional
```

Parameters

`1st_operand`

1st operan should be in one of the following formats:

`$PdGlobalSettings.<field_name>` (Just `PortDataRole` and `PortPowerRole` fields)
`$PdResult.<field_name>`
`$<packet_variable>.<field_name>` (`packet_variable` should be instantiated from one of available PD Packet Templates)

List of `$PdResult` fields:

Field Name	Description
Result	Last executed command result
Subresult	Last executed command subresult (in case of failure, this field describes the reason)
LastReceivedPacketOrderedSet	Last received packet ordered set type
LastReceivedPacketType	Last received packet type
LastReceivedPacketPowerRole	Last received packet power role field value
LastReceivedPacketDataRole	Last received packet data role field value
LastReceivedPacketSentToCable	Indicates whether the last received packet has been sent to cable(packet towards the cable) or not
LastReceivedPacketMsgID	Last received packet `MessageId` field value
LastReceivedPacketVdmCommand	Last received packet VDM command value, if the packet is VDM packet
LastReceivedPacketVdmCommandType	Last received packet VDM command type value, if the packet is VDM packet
LastReceivedPacketVdmSVID	Last received packet SVID, if the packet is a VDM packet
LastReceivedPacketVdmObjPos	Last received packet `ObjetctPosition`, if the packet is a VDM packet
LastSelectedCapIndex	Last received packet selected capability index, if the packet is Request message
LastRequestHasMismatch	Last received packet `HasMismatch` field value, if the packet is Request message

For available `$PdGlobalSettings` fields refer to `PD_Set` Parameters.

`2nd_operand`

It could be as `<1st_operand>` or a constant `<value>`.

`operator`

List of possible values for operator:

Operator	Description
PD_COMPARE_EQUAL	Equal
PD_COMPARE_GREATER	Greater than
PD_COMPARE_LESS	Less than
PD_COMPARE_NOT_EQUAL	Not equal

Result

Examples

```
$send_setting = PD_SendPacketSettings
{
    ResetOnError = PD_FALSE
    OrderedSetType = PD_ORDERED_SET_TYPE_SOP
}
$receive_settings = PD_ReceivePacketSettings
{
    PacketType = PD_MESSAGE_TYPE_VDM
}
#send the packet
$discover_identity = PD_VDM_Discover_Identity_Message
Call PD_SendPacket( $discover_identity, $send_setting )

#check for result
IfMatched( $PdResult.Result, PD_RESULT_OK, PD_COMPARE_EQUAL )
{
    Call PD_ReceivePacket( $receive_settings )
}
ElseMatched( $PdResult.Result, PD_RESULT_FAILED, PD_COMPARE_EQUAL )
{
    Call PD_SendHardReset()
}
ElseMatched
{
    $ping_msg = PD_PingMessage
    Call PD_SendPacket( $ping_msg, $send_setting )
}
IfMatchedEnd
```

3.19 PD_Loop

Loop command.

Note - The limit for using nested `PD_Loop()` commands is 8.

Format

```
PD_Loop(count)
{
    #command list
}
```

Parameters

`count`
Loop count

Result

Examples

```
$send_setting = PD_SendPacketSettings
{
    OrderedSetType = PD_ORDERED_SET_TYPE_SOP
}
PD_Loop(3)
{
    $ping_msg = PD_PingMessage
    call PD_SendPacket( $ping_msg, $send_setting )
}
```

3.20 PD_BreakLoop

Format

```
PD_BreakLoop()
```

Parameters

Result

Examples

```
$send_setting = PD_SendPacketSettings
{
    OrderedSetType = PD_ORDERED_SET_TYPE_SOP
}
PD_Loop(3)
{
    PD_Loop(2)
    {
        $accept_msg = PD_AcceptMessage
        call PD_SendPacket( $accept_msg, $send_setting )
    }

    $ping_msg = PD_PingMessage
    call PD_SendPacket( $ping_msg, $send_setting )

    IfMatched( $PdResult.Result, PD_RESULT_OK, PD_COMPARE_EQUAL )
    {
        PD_BreakLoop()
    }
    IfMatchedEnd
}
```

3.21 PD_ContinueLoop

Format

```
PD_ContinueLoop()
```

Parameters

Result

Examples

```
$send_setting = PD_SendPacketSettings
{
    OrderedSetType = PD_ORDERED_SET_TYPE_SOP
}
PD_Loop(3)
{
    $ping_msg = PD_PingMessage
    call PD_SendPacket( $ping_msg, $send_setting )

    IfMatched( $PdResult.Result, PD_RESULT_OK, PD_COMPARE_EQUAL )
    {
            PD_ContinueLoop()
    }
    IfMatchedEnd

    Call PD_SendSoftReset( PD_ORDERED_SET_TYPE_SOP )
}
```

3.22 PD_Stop

Stops the Exerciser.

Format

```
Call PD_Stop( return_value )
```

Parameters

`return_value`
Value returned to Exerciser.

Result

Examples

```
Call PD_Stop(0)
```

3.23 PD_Disconnect

Simulates cable detach.

Format

```
Call PD_Disconnect()
```

Parameters

Result

Examples

```
Call PD_Disconnect()
```

3.24 PD_IncreaseMsgId

Increase Message ID for DFP/UFP Protocol Layer

Format

```
Call PD_IncreaseMsgId(OrderedSetType)
```

Parameters

`OrderedSetType`

Indicates the OrderedSet type. Possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

Result

Examples

```
Call PD_IncreaseMsgId(PD_ORDERED_SET_TYPE_SOP)
```

3.25 PD_DecreaseMsgId

Decrease Message ID for DFP/UFP Protocol Layer

Format

```
Call PD_DecreaseMsgId(OrderedSetType)
```

Parameters

`OrderedSetType`

Indicates the OrderedSet type. Possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

Result

Examples

```
Call PD_DecreaseMsgId(PD_ORDERED_SET_TYPE_SOP)
```

3.26 PD_IncreaseMsgId_Cable

Increase Message ID for Cable Protocol Layer

Format

```
Call PD_IncreaseMsgId_Cable(OrderedSetType)
```

Parameters

`OrderedSetType`

Indicates the OrderedSet type. Possible values:

```
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

Result

Examples

```
Call PD_IncreaseMsgId_Cable(PD_ORDERED_SET_TYPE_SOP_PRIME)
```

3.27 PD_DecreaseMsgId_Cable

Decrease Message ID for Cable Protocol Layer

Format

```
Call PD_DecreaseMsgId_Cable(OrderedSetType)
```

Parameters

`OrderedSetType`

Indicates the OrderedSet type. Possible values:

```
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

Result

Examples

```
Call PD_DecreaseMsgId_Cable(PD_ORDERED_SET_TYPE_SOP_PRIME)
```

4 Transaction Engine™

Power Delivery Transaction Engine™ includes high level commands and auto response capability.

4.1 High Level Commands

List of available high level commands:

Command Name	Description
PD_SetNegotiationSetting_Source	
PD_AddSourceCap	
PD_ResetSourceCaps	
PD_NegotiatePower_Source	
PD_SetNegotiationSetting_Sink	
PD_AddSinkCap	
PD_ResetSinkCaps	
PD_NegotiatePower_Sink	
PD_WaitForNegotiatePower	
PD_NegotiatePower	
PD_SetSwapPowerRole_Setting	
PD_SwapPowerRole	
PD_WaitForSwapPowerRole	
PD_SetSwapDataRole_Setting	
PD_SwapDataRole	
PD_WaitForSwapDataRole	
PD_SetSwapVconnSetting	
PD_SwapVconn	
PD_WaitForSwapVconn	
PD_GotoMin	
PD_WaitForGotoMin	
PD_GetSourceCapabilities	
PD_WaitForGetSourceCapabilities	
PD_GetSinkCapabilities	
PD_WaitForGetSinkCapabilities	
PD_SendBISTCarrierMode	
PD_SendBISTTestData	
PD_SetDiscoverIdentitySetting	
PD_AddDiscoverIdentityVDO	
PD_ResetDiscoverIdentityVDO	
PD_DiscoverIdentity	
PD_WaitForDiscoverIdentity	
PD_SetDiscoverSVIDSetting	
PD_AddSvid	
PD_ResetSvids	
PD_DiscoverSvids	
PD_WaitForDiscoverSvids	

PD_SetDiscoverModeSetting	
PD_AddMode	
PD_AddModeVDO	
PD_ResetModes	
PD_DiscoverModes	
PD_WaitForDiscoverModes	
PD_SetEnterModeSetting	
PD_EnterMode	
PD_EnterModeVdo	
PD_WaitForEnterMode	
PD_SetExitModeSetting	
PD_ExitMode	
PD_WaitForExitMode	
PD_Attention	
PD_AttentionVdo	
PD_SetDiscoveryProcessSetting	
PD_PerformDiscoveryProcess	
PD_SetDisplayPortSetting	
PD_DisplayPort_UpdateStatus	
PD_DisplayPort_Configure	
PD_WaitForDisplayPortStatus	
PD_WaitForDisplayPortConfigure	
PD_SetDiscoverIdentitySetting_Cable	
PD_WaitForDiscoverIdentity_Cable	
PD_AddDiscoverIdentityVDO_Cable	
PD_ResetDiscoverIdentityVDO_Cable	
PD_SetDiscoverSVIDSetting_Cable	
PD_WaitForDiscoverSvids_Cable	
PD_AddSvid_Cable	
PD_ResetSvids_Cable	
PD_SetDiscoverModeSetting_Cable	
PD_WaitForDiscoverModes_Cable	
PD_AddModeVDO_Cable	
PD_AddMode_Cable	
PD_ResetModes_Cable	
PD_SetEnterModeSetting_Cable	
PD_WaitForEnterMode_Cable	
PD_SetExitModeSetting_Cable	
PD_WaitForExitMode_Cable	

4.1.1 PD_SetNegotiationSetting_Source

Format

```
Call PD_SetNegotiationSetting_Source( PD_Negotiation_Source_Settings $settings )
```

Parameters

`$settings`

Defines negotiation settings for source. Should be in type of PD_Negotiation_Source_Settings template. Table below shows all available fields of PD_Negotiation_Source_Settings template:

Field Name	Description
NegotiationResponse	Indicates the response type. Possible values: PD_NEGOTIATION_ACCEPT(default) PD_NEGOTIATION_WAIT PD_NEGOTIATION_REJECT
SourceCapsRetryCount	Source capabilities retry count.
VBusVoltage_mv	VBus voltage in millivolt.

Note - If user sets the `VBusVoltage_mv`, then the PD Exerciser will set `VBusVoltage_mv` on the `VBus` regardless the actual voltage value which UUT selected during the negotiation process, otherwise the Exerciser will set the `VBus` using the voltage which UUT selected during the negotiation process.

Note - In order to apply voltages greater than 5V, the corresponding check box should be set in recording options (*Allow VBUS > 5v*).

Result

Examples

```
#set negotiation using default values
$settings = PD_Negotiation_Source_Settings
call PD_SetNegotiationSetting_Source( $settings )

#set negotiation using reject as response
$settings
{
    NegotiationResponse = PD_NEGOTIATION_REJECT
}
call PD_SetNegotiationSetting_Source( $settings )
```

4.1.2 PD_AddSourceCap

Before adding a group of source caps make sure that there is no unwanted source cap in the list by calling `PD_ResetSourceCaps()` function.

Note - By default there is one pre-defined source cap(vSafe5V) in the list.

Format

```
Call PD_AddSourceCap(PD_PowerDataObject $PowerDataObject)
```

Parameters

`$PowerDataObject`

One of source PowerDataObjects. List of available source PowerDataObjects:

Source PowerDataObject	Description
PD_PowerDataObjectFixedSupply_Source	
PD_PowerDataObjectVariableSupply_Source	
PD_PowerDataObjectBatterySupply_Source	

Result

Examples

```
local $power_data_object = PD_PowerDataObjectFixedSupply_Source
{
    MaxCurrent_10mAUnits = 20
    Voltage_50mVUnits = 250
}
call PD_AddSourceCap($power_data_object)
```

4.1.3 PD_ResetSourceCaps

Format

```
Call PD_ResetSourceCaps()
```

Parameters

Result

Examples

```
call PD_ResetSourceCaps()
```

4.1.4 PD_NegotiatePower_Source

Format

```
Call PD_NegotiatePower_Source()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_REQUEST_MSG_NOT_RECEIVED	Subresult - Request message not received
PD_SUBRESULT_REQUEST_MSG_INVALID_INDEX	Subresult - Invalid index in request message
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait has been sent as request message response
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject has been sent as request message response

Examples

```
call PD_NegotiatePower_Source()
```

4.1.5 PD_SetNegotiationSetting_Sink

Format

```
Call PD_SetNegotiationSetting_Sink( PD_Negotiation_Sink_Settings $settings )
```

Parameters

`$settings`
Should be from `PD_Negotiation_Sink_Settings` type.
Table below shows all available fields of `PD_Negotiation_Sink_Settings` template:

Field Name	Description
WaitTimeout	Indicates the wait timeout. (default = PD_DEFAULT_TIMEOUT_INFINIT)
SinkRequestData	Defines data object of Request message.
AutoSinkRequest	Builds the `SinkRequestData` automatically according to current sink capabilities and received source capabilities. Possible values: PD_TRUE(Default) PD_FALSE

Result

Examples

```
#Set sink negotiation settings as default
$settings = PD_Negotiation_Sink_Settings
call PD_SetNegotiationSetting_Sink( $settings )
```

4.1.6 PD_AddSinkCap

Before adding a group of sink caps make sure that there is no unwanted sink cap in the list by calling `PD_ResetSinkCaps()` function.

Note - By default there is one pre-defined sink cap in the list.

Format

```
Call PD_AddSinkCap(PD_PowerDataObject $PowerDataObject)
```

Parameters

`$PowerDataObject`

One of sink PowerDataObjects. List of available sink PowerDataObjects:

Sink PowerDataObject	Description
PD_PowerDataObjectFixedSupply_Sink	
PD_PowerDataObjectVariableSupply_Sink	
PD_PowerDataObjectBatterySupply_Sink	

Result

Examples

```
local $power_data_object = PD_PowerDataObjectFixedSupply_Sink
{
    OperationalCurrent_10mAUnits = 50
    Voltage_50mVUnits = 100
}
call PD_AddSinkCap($power_data_object)
```

4.1.7 PD_ResetSinkCaps

Format

```
Call PD_ResetSinkCaps()
```

Parameters

Examples

```
call PD_ResetSinkCaps()
```

4.1.8 PD_NegotiatePower_Sink

Format

```
Call PD_NegotiatePower_Sink()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.

List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed

PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_PSRDY_MSG_NOT_RECEIVED	Subresult - PS_RDY message not received
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject received as the response
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait received as the response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as the response

Examples

```
call PD_NegotiatePower_Sink()
```

4.1.9 PD_WaitForNegotiatePower

Format

```
Call PD_WaitForNegotiatePower()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_SOURCE_CAP_MSG_NOT_RECEIVED	Subresult - Source capabilities message not received
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_PSRDY_MSG_NOT_RECEIVED	Subresult - PS_RDY message not received
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject received as the response
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait received as the response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as the response

Examples

```
call Pd PD_WaitForNegotiatePower()
```

4.1.10 PD_NegotiatePower

Negotiates power with the peer port according to PD Exerciser current power role. If PD Exerciser operates as Source, this function starts power negotiation as Source and if the PD Exerciser operates as Sink, this function starts power negotiation as Sink.

Note - Both power negotiation settings can be applied to this function (by calling `PD_SetNegotiationSetting_Source()` or `PD_SetNegotiationSetting_Source()` functions).

Note - If PD Exerciser operates as Source this function returns same sub-results as `PD_NegotiatePower_Source()` funcion.

Note - If PD Exerciser operates as Sink this function returns same sub-results as `PD_WaitForNegotiatePower()` function.

Format

```
Call PD_NegotiatePower()
```

Parameters

Result

Examples

```
Call PD_NegotiatePower()
```

4.1.11 PD_SetSwapPowerRole_Setting

Format

```
Call PD_SetSwapPowerRole_Setting(PD_SwapResponse_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_SwapResponse_Settings` type.
List of `SwapResponse_Settings` fields:

Field Name	Description
SwapResponse	Defines the response type. Possible values: PD_SWAPPOWERROLE_ACCEPT(default) PD_SWAPPOWERROLE_WAIT PD_SWAPPOWERROLE_REJECT
SkipSendingPSRDY	If set to PD_TRUE, `PD_SwapPowerRole` will not send the PS_RDY message. Possible values: PD_TRUE PD_FALSE(Default)
SkipSwap	If set to PD_TRUE, `PD_SwapPowerRole` will not swap the power role. Possible values: PD_TRUE PD_FALSE(Default)
WaitTimeout	Timeout to wait in order to receive the PR_SWAP message

Result

Examples

```
#Set swap power role settings as default
$settings = PD_SwapResponse_Settings
call PD_SetSwapPowerRole_Setting( $settings )
```

4.1.12 PD_SwapPowerRole

Format

```
Call PD_SwapPowerRole()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description

PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - Response not received
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject received as response
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response
PD_SUBRESULT_PSRDY_MSG_NOT_RECEIVED	Subresult - PS_RDY message not received

Examples

```
call PD_SwapPowerRole()
```

4.1.13 PD_WaitForSwapPowerRole

Format

```
Call PD_WaitForSwapPowerRole()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_PRSWAP_MSG_NOT_RECEIVED	Subresult - PR_SWAP message not received
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait has been sent as response
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject has been sent as response
PD_SUBRESULT_PSRDY_MSG_NOT_RECEIVED	Subresult - PS_RDY message not received

Examples

```
call PD_WaitForSwapPowerRole()
```

4.1.14 PD_SetSwapDataRole_Setting

Format

```
Call PD_SetSwapDataRole_Setting( PD_SwapResponse_Settings $settings )
```

Parameters

`$settings`
Should be from `PD_SwapResponse_Settings` type. Table below describes the `PD_SwapResponse_Settings` template and settings related to Data Role Swap:

Field Name	Description
SwapResponse	Response type. Possible values: PD_MESSAGE_TYPE_ACCEPT(default) PD_MESSAGE_TYPE_REJECT PD_MESSAGE_TYPE_WAIT
WaitTimeout	Timeout to wait in order to receive DR_SWAP message

Result

Examples

```
$settings = PD_SwapResponse_Settings
{
    SwapResponse = PD_MESSAGE_TYPE_REJECT
}
call PD_SetSwapDataRole_Setting( $settings )
```

4.1.15 PD_SwapDataRole

Format

```
Call PD_SwapDataRole()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - Response not received
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject has been received
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait has been received
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message has been received

Examples

```
call PD_SwapDataRole()
```

4.1.16 PD_WaitForSwapDataRole

Format

```
Call PD_WaitForSwapDataRole()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DRSWAP_MSG_NOT_RECEIVED	Subresult - DR_SWAP message not received
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject has been sent as response

Examples

```
call PD_WaitForSwapDataRole()
```

4.1.17 PD_SetSwapVconnSetting

Format

```
Call PD_SetSwapVconnSetting( PD_SwapResponse_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_SwapResponse_Settings` type. Table below describes the `PD_SwapResponse_Settings` template and specific settings related to Swap VConn:

Field Name	Description
SwapResponse	Response type. Possible values: PD_MESSAGE_TYPE_ACCEPT(default) PD_MESSAGE_TYPE_REJECT PD_MESSAGE_TYPE_WAIT
WaitTimeout	Timeout to wait in order to receive VCONN_SWAP message

Result

Examples

```
#Using default settings
$settings = PD_SwapResponse_Settings
call PD_SetSwapVconnSetting( $settings )
```

4.1.18 PD_SwapVconn

Format

```
Call PD_SwapVconn()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - Response not received
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject has been received
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait has been received
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message has been received
PD_SUBRESULT_PSRDY_MSG_NOT_RECEIVED	Subresult - PS_RDY message not received

Examples

```
call PD_SwapVconn()
```

4.1.19 PD_WaitForSwapVconn

Format

```
Call PD_WaitForSwapVconn()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD SUBRESULT VCONNSWAP MSG NOT RECEIVED	Subresult - VCONN SWAP message not received
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait has been sent as response
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject has been sent as response
PD_SUBRESULT_PSRDY_MSG_NOT_RECEIVED	Subresult - PS_RDY message not received

Examples

```
call PD_WaitForSwapVconn()
```

4.1.20 PD_GotoMin

Format

```
Call PD_GotoMin()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received

Examples

```
call PD_GotoMin()
```

4.1.21 PD_WaitForGotoMin

Format

```
Call PD_WaitForGotoMin()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD RESULT OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_GOTOMIN_MSG_NOT_RECEIVED	Subresult - GOTOMIN message not received
PD_SUBRESULT_PSRDY_MSG_NOT_RECEIVED	Subresult - PS_RDY message not received

Examples

```
call PD_WaitForGotoMin()
```

4.1.22 PD_GetSourceCapabilities

Format

```
Call PD_GetSourceCapabilities()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No messages received as response

Examples

```
Call PD_GetSourceCapabilities()
```

4.1.23 PD_WaitForGetSourceCapabilities

Format

```
Call PD_WaitForGetSourceCapabilities()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_GETSOURCECAP_MSG_NOT_RECEIVED	Subresult - GETSOURCECAP message not received
PD_SUBRESULT_REQUEST_MSG_NOT_RECEIVED	Subresult - Request message not received
PD_SUBRESULT_REQUEST_MSG_INVALID_INDEX	Subresult - Invalid index in request message
PD_SUBRESULT_RESPONSE_WAIT	Subresult - Wait has been sent as request message response
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject has been sent as request message response

Examples

```
Call PD_WaitForGetSourceCapabilities()
```

4.1.24 PD_GetSinkCapabilities

Format

```
Call PD_GetSinkCapabilities()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_REJECT	Subresult - Reject received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No messages received as response

Examples

```
Call PD_GetSinkCapabilities()
```

4.1.25 PD_WaitForGetSinkCapabilities

Format

```
Call PD_WaitForGetSinkCapabilities()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_GETSINKCAP_MSG_NOT_RECEIVED	Subresult - GETSINKCAP message not received

Examples

```
Call PD_WaitForGetSinkCapabilities()
```

4.1.26 PD_SendBISTCarrierMode

Format

```
Call PD_SendBISTCarrierMode(OrderedSetType)
```

Parameters

`OrderedSetType`

Indicates the Ordered Set type

possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
```

PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed

Examples

```
Call PD_SendBISTCarrierMode(PD_ORDERED_SET_TYPE_SOP)
```

4.1.27 PD_SendBISTTestData

Format

```
Call PD_SendBISTTestData( OrderedSetType, PD_BISTTestData $test_data )
```

Parameters

`OrderedSetType`

Indicates the Ordered Set type

possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

`$test_data`

Defines the Test Data to be sent to the UUT

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received

Examples

```
$test_data = PD_BISTTestData
{
    TestData = { 00 00 00 00
                 AA AA AA AA
                 AA AA 00 00
                 AA AA AA AA
                 00 00 AA AA
                 AA AA AA AA }
}

Call PD_SendBISTTestData( PD_ORDERED_SET_TYPE_SOP_PRIME, $test_data )
```

4.1.28 PD_SetDiscoverIdentitySetting

Format

```
Call PD_SetDiscoverIdentitySetting( PD_DiscoverIdentity_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_DiscoverIdentity_Settings` type. Table below shows the available fields of `PD_DiscoverIdentity_Settings` template:

Field Name	Description
DiscoverIdentityResponse	Indicates the response type. possible values are: PD_DISCOVERIDENTITY_ACK(default) PD_DISCOVERIDENTITY_BUSY PD_DISCOVERIDENTITY_NAK
WaitTimeout	Timeout to wait for receiving Discover Identity Command

Result

Examples

```
#Using default settings
$settings = PD_DiscoverIdentity_Settings
call PD_SetDiscoverIdentitySetting( $settings )
```

4.1.29 PD_AddDiscoverIdentityVDO

Format

```
Call PD_AddDiscoverIdentityVDO( PD_DiscoverIdentity_VDO $vdo )
```

Parameters

`$vdo`

Should be from `PD_DiscoverIdentity_VDO` type.
List of predefined templates for this command:

Template Name	Description
PD_VDM_Discover_Identity_ID_Header_VDO	ID Header VDO
PD_VDM_Discover_Identity_Cert_Stat_VDO	Cert Stat VDO
PD_VDM_Discover_Identity_Product_VDO	Product VDO
PD_VDM_Discover_Identity_Cable_VDO	Product type VDO Cable
PD_VDM_Discover_Identity_Alternate_Mode_Adapter_VDO	Product type VDO Alternate Mode Adapter

Result

Examples

```
#Add a ID Header VDO
$vdo = PD_VDM_Discover_Identity_ID_Header_VDO
{
    IDHeaderVDO_USBVendorID = 0xFF01
    IDHeaderVDO_ModalOperationSupported = 1
    IDHeaderVDO_ProductType = PD_VDM_ID_HEADER_VDO_PRODUCT_TYPE_PERIPHERAL
    IDHeaderVDO_DataCapableAsUSBDevice = 1
}
call PD_AddDiscoverIdentityVDO( $vdo )
```

4.1.30 PD_ResetDiscoverIdentityVDO

Format

```
Call PD_ResetDiscoverIdentityVDO()
```

Parameters

Result

Examples

```
call PD_ResetDiscoverIdentityVDO()
```

4.1.31 PD_DiscoverIdentity

Format

```
Call PD_DiscoverIdentity( OrderedSetType )
```

Parameters

`OrderedSetType`

possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK received as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response

Examples

```
call PD_DiscoverIdentity(PD_ORDERED_SET_TYPE_SOP)
```

4.1.32 PD_WaitForDiscoverIdentity

Format

```
Call PD_WaitForDiscoverIdentity()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DISCOVERIDENTITY_CMD_NOT_RECEIVED	Subresult - DISCOVERIDENTITY command not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY has been sent as response

Examples

```
call PD_WaitForDiscoverIdentity()
```

4.1.33 PD_SetDiscoverSVIDSetting

Format

```
Call PD_SetDiscoverSVIDSetting( PD_DiscoverSvids_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_DiscoverSvids_Settings` type. Table below shows the available fields of `PD_DiscoverSvids_Settings` template:

Field Name	Description
DiscoverSvidsResponse	Indicates the response type. possible values are: PD_DISCOVERSVIDS_ACK(default) PD_DISCOVERSVIDS_BUSY PD_DISCOVERSVIDS_NAK
WaitTimeout	Timeout to wait for receiving Discover SVID Command

Result

Examples

```
#Using default settings
$settings = PD_DiscoverSvids_Settings
call PD_SetDiscoverSVIDSetting( $settings )
```

4.1.34 PD_AddSvid

Note - Up to 11 SVIDs can be added using this function.

Format

```
Call PD_AddSvid(value)
```

Parameters

`value`

SVID value to add

Result

Examples

```
call PD_AddSvid(0xFF01)
```

4.1.35 PD_ResetSvids

Format

```
Call PD_ResetSvids()
```

Parameters

Result

Examples

```
call PD_ResetSvids()
```

4.1.36 PD_DiscoverSvids

Note - PD Exerciser supports only one(first) DiscoverSVIDs Ack message.

Format

```
Call PD_DiscoverSvids(OrderedSetType)
```

Parameters

`OrderedSetType`

possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.

List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK received as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response

Examples

```
call PD_DiscoverSvids(PD_ORDERED_SET_TYPE_SOP)
```

4.1.37 PD_WaitForDiscoverSvids

Format

```
Call PD_WaitForDiscoverSvids()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.

List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DISCOVERSVIDS_CMD_NOT_RECEIVED	Subresult - DISCOVERSVIDS message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY has been sent as response

Examples

```
call PD_WaitForDiscoverSvids()
```

4.1.38 PD_SetDiscoverModeSetting

Format

```
Call PD_SetDiscoverModeSetting( PD_DiscoverModes_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_DiscoverModes_Settings` type. Table below describes the `PD_DiscoverModes_Settings` template:

Field Name	Description
DiscoverModesResponse	Response type. Possible values are PD_DISCOVERMODES_ACK(default) PD_DISCOVERMODES_BUSY PD_DISCOVERMODES_NAK
WaitTimeout	Timeout to wait for receiving Discover Modes command

Result

Examples

```
#Using default settings
$settings = PD_DiscoverModes_Settings
call PD_SetDiscoverModeSetting( $settings )
```

4.1.39 PD_AddMode

Format

```
Call PD_AddMode(Mode)
```

Parameters

`Mode`

Mode to add

Result

Examples

```
call PD_AddMode(0x00000001)
```

4.1.40 PD_AddModeVDO

Format

```
Call PD_AddModeVDO(PD_Generic_VDO $ModeVdo)
```

Parameters

`$ModeVdo`

Should be from `PD_Generic_VDO` type. Table below describes the `PD_VDO`(inherited from `PD_Generic_VDO`) template that can be use as ModeVdo:

Filed Name	Description
Data	

Result

Examples

```
local $vdo_1 = PD_VDO
{
    Data = 0x01
}
call PD_AddModeVDO($vdo_1)
```

4.1.41 PD_ResetModes

Format

```
Call PD_ResetModes()
```

Parameters

Result

Examples

```
call PD_ResetModes()
```

4.1.42 PD_DiscoverModes

Format

```
Call PD_DiscoverModes(OrderedSetType, selectedSvid)
```

Parameters

`OrderedSetType`

possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

`selectedSvid`

Indicates the SVID value

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK received as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response

Examples

```
call PD_DiscoverModes(PD_ORDERED_SET_TYPE_SOP,0xFF00)
```

4.1.43 PD_WaitForDiscoverModes

Format

```
Call PD_WaitForDiscoverModes()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DISCOVERMODES_CMD_NOT_RECEIVED	Subresult - DISCOVERMODES message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY has been sent as response

Examples

```
call PD_WaitForDiscoverModes()
```

4.1.44 PD_SetEnterModeSetting

Format

```
Call PD_SetEnterModeSetting( PD_EnterMode_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_EnterMode_Settings` type. Table below describes the `PD_EnterMode_Settings` template:

Field Name	Description
EnterModeResponse	Response type. Possible values : PD_ENTERMODE_ACK(default) PD_ENTERMODE_NAK
WaitTimeout	Timeout to wait for receiving Enter Mode command

Result

Examples

```
#Using default setting
$settings = PD_EnterMode_Settings
call PD_SetEnterModeSetting( $settings )
```

4.1.45 PD_EnterMode

Format

```
Call PD_EnterMode(OrderedSetType, selectedSvid, modeIndex)
```

Parameters

`OrderedSetType`

possible values:

PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME

`selectedSvid`

Indicates the SVID

`modeIndex`

Indicates the mode index for the specified SVID

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.

List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded

PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response

Examples

```
call PD_EnterMode(PD_ORDERED_SET_TYPE_SOP,0xFF00, 1)
```

4.1.46 PD_EnterModeVdo

Format

```
Call PD_EnterModeVdo( OrderedSetType, selectedSvid, modeId, PD_Generic_VDO $Vdo
)
```

Parameters

`OrderedSetType`

Indicates the ordered set type. Possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

`selectedSvid`

Indicates the SVID

`modeId`

Indicates the mode index related to the specified SVID

`$Vdo`

Vendor defined data object. Should be from `PD_VDO`(Inherited from `PD_Generic_VDO`) type.

Field Name	Description
Data	VDO data

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response

Examples

```
$vdo = PD_VDO
{
    Data = 0x00
}
call PD_EnterModeVdo(PD_ORDERED_SET_TYPE_SOP, 0xFF01, 1, $vdo)
```

4.1.47 PD_WaitForEnterMode

Format

```
Call PD_WaitForEnterMode()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_ENTERMODE_CMD_NOT_RECEIVED	Subresult - ENERMODE message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response

Examples

```
call PD_WaitForEnterMode()
```

4.1.48 PD_SetExitModeSetting

Format

```
Call PD_SetExitModeSetting( PD_ExitMode_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_ExitMode_Settings` type. Table below describes the `PD_ExitMode_Settings` template:

Field Name	Description
ExitModeResponse	Indicates the response type. Possible values : PD_EXITMODE_ACK(default) PD_EXITMODE_NAK
WaitTimeout	Timeout to wait for receiving the Exit Mode command

Result

Examples

```
#Using default settings
$settings = PD_ExitMode_Settings
call PD_SetExitModeSetting( $settings )
```

4.1.49 PD_ExitMode

Format

```
Call PD_ExitMode(OrderedSetType, selectedSvid, modeIndex)
```

Parameters

`OrderedSetType`

possible values:

PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME

`selectedSvid`

Indicates the SVID

`modeIndex`

Indicates the mode index related to the specified SVID

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response

Examples

```
call PD_ExitMode(PD_ORDERED_SET_TYPE_SOP, 0xFF00, 1)
```

4.1.50 PD_WaitForExitMode

Format

```
Call PD_WaitForExitMode()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_EXITMODE_CMD_NOT_RECEIVED	Subresult - EXITMODE message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response

Examples

```
call PD_WaitForExitMode()
```

4.1.51 PD_Attention

Format

```
Call PD_Attention( OrderedSetType, selectedSvid, modeIndex )
```

Parameters

`OrderedSetType`

Indicates the ordered set type. possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

`selectedSvid`

Indicates the SVID

`modeIndex`

Indicates the mode index related to the specified SVID

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received

Examples

```
call PD_Attention(PD_ORDERED_SET_TYPE_SOP, 0xFF01, 1 )
```

4.1.52 PD_AttentionVdo

Format

```
Call PD_AttentionVdo( OrderedSetType, selectedSvid, modeIndex, PD_Generic_VDO
$Vdo )
```

Parameters

`rderedSetType`

Indicates the ordered set type. Possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

`selectedSvid`

Indicates the SVID

`modeId`

Indicates the mode index related to the specified SVID

`$Vdo`

Vendor defined data object. Should be from `PD_VDO`(Inherited from `PD_Generic_VDO`) type.

Field Name	Description
Data	VDO data

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received

Examples

```
$vdo = PD_VDO
{
    Data = 0x00
}
call PD_AttentionVdo(PD_ORDERED_SET_TYPE_SOP, 0xFF01, 1, $vdo)
```

4.1.53 PD_SetDiscoveryProcessSetting

Format

```
Call PD_SetDiscoveryProcessSetting(PD_DiscoveryProcess_Settings $settings)
```

Parameters

`$settings`

Should be from `PD_DiscoveryProcess_Settings` type. Table below describes the `PD_DiscoveryProcess_Settings` template:

Field Name	Description
Discover_SOP_PP_During_SOP_P	Indicates whether perform SOP Double Prime discovery during SOP Prime discovery process or not. Possible Values: `PD_TRUE` `PD_FALSE`(Default)

Result

Examples

```
#Using default settings
$settings = PD_DiscoveryProcess_Settings
call PD_SetDiscoveryProcessSetting( $settings )
```

4.1.54 PD_PerformDiscoveryProcess

Performs full discovery process.

Note - PD Exerciser supports only one(first) DiscoverSVIDs Ack message (up to 12 SVIDs).

Format

```
Call PD_PerformDiscoveryProcess( OrderedSetType )
```

Parameters

`OrderedSetType`

Indicates the ordered set type. Possible values:

```
PD_ORDERED_SET_TYPE_SOP
PD_ORDERED_SET_TYPE_SOP_PRIME
PD_ORDERED_SET_TYPE_SOP_DOUBLE_PRIME
```

Result

Examples

```
call PD_PerformDiscoveryProcess(PD_ORDERED_SET_TYPE_SOP)
```

4.1.55 PD_SetDisplayPortSetting

Format

```
Call PD_SetDisplayPortSetting( PD_DisplayPort_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_DisplayPort_Settings` type. Table below describes the `PD_DisplayPort_Settings` template:

Field Name	Description
ConfigureResponse	Indicates the response for incoming Display Port Configure command. Possible values: `PD_DISPLAYPORT_ACK`(Default) `PD_DISPLAYPORT_NAK`
DisplayPortModeIndex	Mode index related to the Display Port SVID. (Default: `0x01`).
StatusVdo	Indicates the Display Port Status Vendor Defined Data Object which can be used in Display Port Update Status initiator or responder messages.
ConfigureVdo	Indicates the Display Port Configure Vendor Defined Data Object which can be used in Display Port Configure initiator message.

WaitTimeout	Timeout to wait for receiving Display Port Update Status or Configure command.

Result

Examples

```
#Using default settings
#################################
$settings = PD_DisplayPort_Settings
call PD_SetDisplayPortSetting($settings)

#Set the StatusVdo
#################################
$update_status = PD_VDM_DisplayPort_Status_VDO
{
    DFPD_UFPD_Connected      = PD_DISPLAYPORT_DFPD_CONNECTED
    PowerLow                 = 0x00
    AdaptorEnabled           = 0x01
    MultiFunctionPreferred   = 0x01
    UsbConfigurationRequest  = 0x00
    ExitDisplayModeRequest   = 0x00
    HPD_State                = 0x00
    IRQ_HPD                  = 0x00
    Reserved_DPS_1           = 0x00
}
$settings
{
    StatusVdo = $update_status
}
Call PD_SetDisplayPortSetting($settings)

#Set the ConfigureVdo to default
#################################
$config = PD_VDM_DisplayPort_Configure_VDO
$settings
{
    ConfigureVdo = $config
}
Call PD_SetDisplayPortSetting($settings)
```

4.1.56 PD_DisplayPort_UpdateStatus

Format

```
Call PD_DisplayPort_UpdateStatus()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response

Examples

```
call PD_DisplayPort_UpdateStatus()
```

4.1.57 PD_DisplayPort_Configure

Format

```
Call PD_DisplayPort_Configure()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_RESPONSE_TIMEOUT	Subresult - No response received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK received as response
PD_SUBRESULT_UNEXPECTED_MSG_RECEIVED	Subresult - Unexpected message received as response

Examples

```
call PD_DisplayPort_Configure()
```

4.1.58 PD_WaitForDisplayPortStatus

Format

```
Call PD_WaitForDisplayPortStatus()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DISPLAYPORT_STATUS_CMD_NOT_RECEIVED	Subresult - UPDATE_STATUS message not received

Examples

```
call PD_WaitForDisplayPortStatus()
```

4.1.59 PD_WaitForDisplayPortConfigure

Format

```
Call PD_WaitForDisplayPortConfigure()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DISPLAYPORT_CONFIGURE_CMD_NOT_RECEIVED	Subresult - Configure message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response

Examples

```
call PD_WaitForDisplayPortConfigure()
```

4.1.60 PD_SetDiscoverIdentitySetting_Cable

Format

```
Call PD_SetDiscoverIdentitySetting_Cable( PD_DiscoverIdentity_Settings $settings
)
```

Parameters

`$settings`

Should be from `PD_DiscoverIdentity_Settings` type. Table below shows the available fields of `PD_DiscoverIdentity_Settings` template:

Field Name	Description
DiscoverIdentityResponse	Indicates the response type. possible values are: PD_DISCOVERIDENTITY_ACK(default) PD_DISCOVERIDENTITY_BUSY PD_DISCOVERIDENTITY_NAK
WaitTimeout	Timeout to wait for receiving Discover Identity Command

Result

Examples

```
#Using default settings
$settings = PD_DiscoverIdentity_Settings
call PD_SetDiscoverIdentitySetting_Cable( $settings )
```

4.1.61 PD_WaitForDiscoverIdentity_Cable

Format

```
Call PD_WaitForDiscoverIdentity_Cable()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DISCOVERIDENTITY_CMD_NOT_RECEIVED	Subresult - DISCOVER_IDENTITY message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY has been sent as response

Examples

```
Call PD_WaitForDiscoverIdentity_Cable()
```

4.1.62 PD_AddDiscoverIdentityVDO_Cable

Format

```
Call PD_AddDiscoverIdentityVDO_Cable( PD_DiscoverIdentity_VDO $vdo )
```

Parameters

`$vdo`

Should be from `PD_DiscoverIdentity_VDO` type.
List of predefined templates for this command:

Template Name	Description
PD_VDM_Discover_Identity_ID_Header_VDO	ID Header VDO
PD_VDM_Discover_Identity_Cert_Stat_VDO	Cert Stat VDO
PD_VDM_Discover_Identity_Product_VDO	Product VDO
PD_VDM_Discover_Identity_Cable_VDO	Product type VDO Cable
PD_VDM_Discover_Identity_Alternate_Mode_Adapter_VDO	Product type VDO Alternate Mode Adapter

Result

Examples

```
#Add a Cable VDO
$vdo = PD_VDM_Discover_Identity_Cable_VDO
call PD_AddDiscoverIdentityVDO_Cable( $vdo )
```

4.1.63 PD_ResetDiscoverIdentityVDO_Cable

Format

```
call PD_ResetDiscoverIdentityVDO_Cable()
```

Parameters

Result

Examples

```
call PD_ResetDiscoverIdentityVDO_Cable()
```

4.1.64 PD_SetDiscoverSVIDSetting_Cable

Format

```
Call PD_SetDiscoverSVIDSetting_Cable( PD_DiscoverSvids_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_DiscoverSvids_Settings` type. Table below shows the available fields of `PD_DiscoverSvids_Settings` template:

Field Name	Description
DiscoverSvidsResponse	Indicates the response type. possible values are: PD_DISCOVERSVIDS_ACK(default) PD_DISCOVERSVIDS_BUSY PD_DISCOVERSVIDS_NAK
WaitTimeout	Timeout to wait for receiving Discover SVID Command

Result

Examples

```
#Using default settings
$settings = PD_DiscoverSvids_Settings
call PD_SetDiscoverSVIDSetting_Cable( $settings )
```

4.1.65 PD_WaitForDiscoverSvids_Cable

Format

```
Call PD_WaitForDiscoverSvids_Cable()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DISCOVERSVIDS_CMD_NOT_RECEIVED	Subresult - DISCOVER_SVIDS message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY has been sent as response

Examples

```
Call PD_WaitForDiscoverSvids_Cable()
```

4.1.66 PD_AddSvid_Cable

Format

```
Call PD_AddSvid_Cable(value)
```

Parameters

`value`
SVID value to add

Result

Examples

```
call PD_AddSvid_Cable(0xFF81)
```

4.1.67 PD_ResetSvids_Cable

Format

```
Call PD_ResetSvids_Cable()
```

Parameters

Result

Examples

```
call PD_ResetSvids_Cable()
```

4.1.68 PD_SetDiscoverModeSetting_Cable

Format

```
Call PD_SetDiscoverModeSetting_Cable( PD_DiscoverModes_Settings $settings )
```

Parameters

`$settings`

Should be from `PD_DiscoverModes_Settings` type. Table below describes the `PD_DiscoverModes_Settings` template:

Field Name	Description
DiscoverModesResponse	Response type. Possible values are PD_DISCOVERMODES_ACK(default) PD_DISCOVERMODES_BUSY PD_DISCOVERMODES_NAK
WaitTimeout	Timeout to wait for receiving Discover Modes command

Result

Examples

```
#Using default settings
$settings = PD_DiscoverModes_Settings
call PD_SetDiscoverModeSetting_Cable( $settings )
```

4.1.69 PD_WaitForDiscoverModes_Cable

Format

```
Call PD_WaitForDiscoverModes_Cable()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_DISCOVERMODES_CMD_NOT_RECEIVED	Subresult - DISCOVER_MODES message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response
PD_SUBRESULT_RESPONSE_BUSY	Subresult - BUSY has been sent as response

Examples

```
Call PD_WaitForDiscoverModes_Cable()
```

4.1.70 PD_AddModeVDO_Cable

Format

```
Call PD_AddModeVDO_Cable(PD_Vdo $ModeVdo)
```

Parameters

`$ModeVdo`

Should be from `PD_Vdo` type. Table below describes the `PD_VDO` template that can be use as ModeVdo:

Filed Name	Description
Data	

Result

Examples

```
local $vdo_1 = PD_VDO
{
    Data = 0x01
}
call PD_AddModeVDO_Cable($vdo_1)
```

4.1.71 PD_AddMode_Cable

Format

```
Call PD_AddMode_Cable(Mode)
```

Parameters

`Mode`
Mode to add

Result

Examples

```
call PD_AddMode_Cable(0x00000001)
```

4.1.72 PD_ResetModes_Cable

Format

```
Call PD_ResetModes_Cable()
```

Parameters

Result

Examples

```
Call PD_ResetModes_Cable()
```

4.1.73 PD_SetEnterModeSetting_Cable

Format

```
Call PD_SetEnterModeSetting_Cable( PD_EnterMode_Settings $settings )
```

Parameters

`$settings`
Should be from `PD_EnterMode_Settings` type. Table below describes the `PD_EnterMode_Settings` template:

Field Name	Description
EnterModeResponse	Response type. Possible values : PD_ENTERMODE_ACK(default) PD_ENTERMODE_NAK
WaitTimeout	Timeout to wait for receiving Enter Mode command

Result

Examples

```
#Using default setting
$settings = PD_EnterMode_Settings
call PD_SetEnterModeSetting_Cable( $settings )
```

4.1.74 PD_WaitForEnterMode_Cable

Format

```
Call PD_WaitForEnterMode_Cable()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_ENTERMODE_CMD_NOT_RECEIVED	Subresult - ENTER_MODE message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response

Examples

```
Call PD_WaitForEnterMode_Cable()
```

4.1.75 PD_SetExitModeSetting_Cable

Format

```
Call PD_SetExitModeSetting_Cable( PD_ExitMode_Settings $settings )
```

Parameters

`$settings`
Should be from `PD_ExitMode_Settings` type. Table below describes the `PD_ExitMode_Settings` template:

Field Name	Description
ExitModeResponse	Indicates the response type. Possible values : PD_EXITMODE_ACK(default) PD_EXITMODE_NAK
WaitTimeout	Timeout to wait for receiving the Exit Mode command

Result

Examples

```
#Using default settings
$settings = PD_ExitMode_Settings
call PD_SetExitModeSetting_Cable( $settings )
```

4.1.76 PD_WaitForExitMode_Cable

Format

```
Call PD_WaitForExitMode_Cable()
```

Parameters

Result

User can evaluate the command results(including sub-results) using `IfMatched/ElseMatched` command.
List of result values:

Result Value	Description
PD_RESULT_OK	Command succeeded
PD_RESULT_FAILED	Command failed
PD_SUBRESULT_NO_GOODCRC	Subresult - No GoodCRC received
PD_SUBRESULT_RECEIVE_TIMEOUT	Subresult - Receive timeout
PD_SUBRESULT_EXITMODE_CMD_NOT_RECEIVED	Subresult - EXIT_MODE message not received
PD_SUBRESULT_RESPONSE_NAK	Subresult - NAK has been sent as response

Examples

```
Call PD_WaitForExitMode_Cable()
```

4.2 Auto Responses Capability

To gain auto response capability, use below command.

4.2.1 PD_DelayAutoResponse

Format

```
Call PD_DelayAutoResponse( duration_micro_Sec )
```

Parameters

`duration_micro_Sec`

Command waits for maximum specified duration and responses to received packet automatically.

Examples

```
call PD_DelayAutoResponse( 1000 )
```