TPS25750 Host Interface Technical Reference Manual

Technical Reference Manual

Literature Number: SLVUC05A
NOVEMBER 2020 – REVISED JULY 2022

Table of Contents

List of Figures

List of Tables

This page intentionally left blank.

Read This First

About This Manual

This manual covers the features and peripherals supported by the TPS2575x family of USB Type-C® and PD controller devices. The document covers the GPIO Events, I2C Interrupts, Host Interface (4CC) Commands, and register read/write descriptions. This manual also covers the Patch Burst Mode process through the PBMx commands to program the PD controller configurations.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers can be shown with the suffix h or the prefix 0x. For example, the following number is 40 hexadecimal (decimal 64): 40h or 0x40.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties with default reset value below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure can have one of multiple meanings:
    - Not implemented on the device
    - Reserved for future device expansion
    - Reserved for TI testing
    - Reserved configurations of the device that are not supported
  - Writing nondefault values to the Reserved bits could cause unexpected behavior and should be avoided.

Glossary

TI Glossary	This glossary lists and explains terms, acronyms, and definitions.

Related Documents

- Universal Serial Bus Specification, Revision 2.0, April 27, 2000 plus ECN and Errata. http://www.usb.org/developers/docs/usb20_docs/
- Battery Charging Specification, Revision 1.2, December 7, 2010 plus Errata.
- Universal Serial Bus 3.1 Specification, Revision 1.0, July 26, 2013 and ECNs approved through August 11, 2014. www.usb.org/developers/docs
- USB Power Delivery Specification Revision 3.0, Version 1.2, June 21, 2018 www.usb.org/developers/docs
- USB Type-C Cable and Connector Specification Revision 1.3, July 14, 2017. www.usb.org/developers/docs

Support Resources

TI E2E™ support forums are an engineer's go-to source for fast, verified answers and design help — straight from the experts. Search existing answers or ask your own question to get the quick design help you need.

Linked content is provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

Trademarks

TI E2E™ is a trademark of Texas Instruments.
USB Type-C® is a registered trademark of USB Implementers Forum.
All trademarks are the property of their respective owners.

This page intentionally left blank.

1.1 Introduction

1.1.1 Purpose and Scope

Note

This section is for advanced users and the features listed here are only optional. This document is not necessary for simple power applications. An EC or Host is required in your system to implement some of the features described in this document.

This document describes the Host Interface for the TPS25750 Type-C Port Switch / Power Delivery (PD) Controller device. This document is aligned with patch F509.04.02 for TPS25750.

1.2 PD Controller Host Interface Description

1.2.1 Overview

The PD Controller provides one I2C slave. The I2Cs slave is meant to be connected to an Embedded Controller (EC).

The Host Interface defines how the registers are accessed from I2C slave port ands and all slave addresses. Slave Address #1 is selected by the customer using the ADCIN1 and ADCIN2 pins on the PD controller. See also Table 4-1 for more details about the slave addresses.

The Host Interface provides general status information to the master of these I2C interfaces about the PD Controller, ability to control the PD Controller, status of USB Type-C Port and communications to/from a connected device (Port Partner) and/or cable plug via USB PD messages. All Host Interface communication that uses the Unique I2C address is referred to as Unique Address Interface.

The PD Controller supports a register-based Unique Address Interface. Section 1.3.2 lists the Unique Address Interface registers and Chapter 2 provides detailed Unique Address Interface register descriptions.

The key to the protocol diagrams is in the SMBus Specification, version 2.0 and is repeated here in part in Figure 1-1.

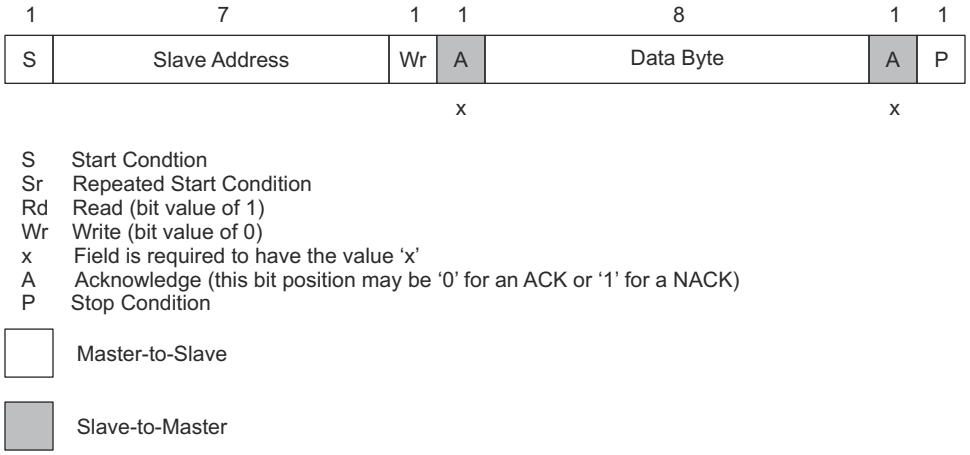


Figure 1-1. I2C Read/Write Protocol Key

1.2.2 Register and Field Notation

In this document the register names use an ALL CAPS notation, and the field names use a CamelBack notation. For example *TX_SOURCE_CAPS* refers to register 0x32, and *TX_SOURCE_CAPS.numValidPDOs* refers to a specific field in Byte 1 of that register.

Some registers have the same definition, but there are multiple instantiations at different register addresses. The following lists these registers.

- *INT_EVENT1* (0x14)
- *INT_MASK1* (0x16)
- *INT_CLEAR1* (0x18)
- *CMD1* (0x08)
- *DATA1* (0x09)

1.3 Unique Address Interface

1.3.1 Unique Address Interface Protocol

The Unique Address Interface allows for complex interactions between an I2C master and a single PD Controller. The I2C Slave unique address is used to receive or respond to Host Interface protocol commands. Figure 1-2 and Figure 1-3 show the write and read protocols, respectively. The Byte Count used during a register

write may be longer than the number of bytes actually written. In other words the, master may issue the stop bit without writing N bytes. Similarly, during a register read, the master may issue the stop bit before reading all N bytes.

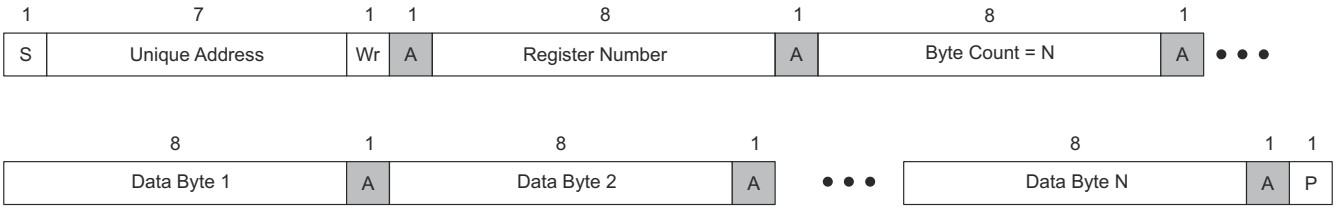


Figure 1-2. I2C Unique Address Write Register Protocol

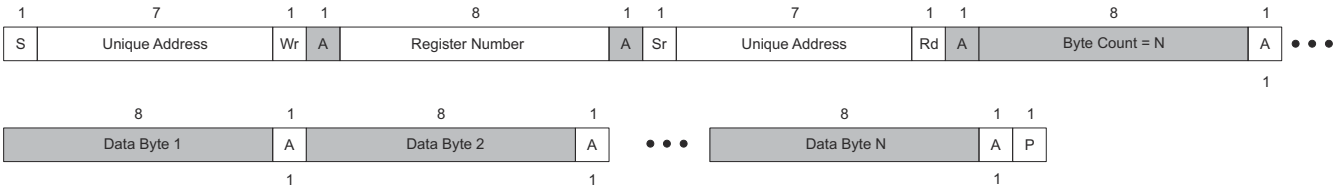


Figure 1-3. I2C Unique Address Read Register Protocol

1.3.2 Unique Address Interface Registers

The PD Controller supports Unique Address Interface registers (Unique Address Registers) provided in Table 1-1. Unless otherwise indicated, 2 or 4-byte registers are little endian (least significant byte in Data Byte 1). Registers that use four character codes (4CC) are defined where the first character corresponds to the ASCII value of Data Byte 1, the second character corresponds to the ASCII value of Data Byte 2, and so forth. Any 4CC codes that are less than 4 characters pad the tail with spaces (0x20).

For registers that are marked as not unique, the host may read that register using either slave address and it reads back the same value. For registers that are marked as not unique, the host may write that register using either slave address .

In the register map below, only the registers addresses shown are implemented. All other register addresses are Read-Only Reserved registers that must be ignored.

Table 1-1. Unique Address Interface Registers

Register Number(1)	Register Name	Access	# Data Bytes	Unique per Port	Description
0x02	Reserved				
0x03	MODE	RO	4	no	Indicates the operational state of the port. The PD controller has limited functionality in some modes. See Section 2.1
0x04	TYPE	RO	4	no	Default response is 'I2C ' (note space as 4th character).
0x06	CUSTUSE	RO	8	yes	These 8 bytes are allocated for customer use as needed. The PD controller does not use this register. This register may be changed during application customization.
0x07	Reserved				
0x08	CMD1	RW	4	yes	Command register for the primary command interface. Cleared to 0x0000_0000 by the PD Controller during initialization and after successful processing of every command. If an unrecognized command is written to this register, it is replaced by a 4CC value of "!CMD".
0x09	DATA1	RW	64	yes	Data register for the primary command interface (CMD1).
0x0A-0x0C	Reserved				
0x0D	DEVICE_CAPABILITIES	RO	4	no	Description of supported features. See Section 2.2
0x0E	Reserved				
0x0F	VERSION	RO	4	no	Binary Coded Decimal version number, bootloader/ application code version. Represented as VVVV.MM.RR with leading 0's removed.e.g. 65794d (decimal) to 0x00010102 to 0001.01.02 to 1.1.2 (version). The version information is returned in little Endian format i.e. byte 1 = RR, byte 2 = MM, etc. The 16-bit field RR is used as the FW version in the Source Capabilities Extended and Sink Capabilities Extended messages.
0x10-0x13	Reserved				

Table 1-1. Unique Address Interface Registers (continued)

Register Number[(1)]	Register Name	Access	# Data Bytes	Unique per Port	Description
0x14	INT_EVENT1	RO	11	yes	Interrupt event bit field for I2Cs_IRQ. If any bit in this register is 1, then the I2Cs_IRQ pin is pulled low. See Section 2.3
0x15	Reserved				
0x16	INT_MASK1	RW	11	yes	Interrupt mask bit field for INT_EVENT1. A bit in INT_EVENT1 cannot be set if it is cleared in this register. See Section 2.3
0x17	Reserved				
0x18	INT_CLEAR1	RW	11	yes	Interrupt clear bit field for INT_EVENT1. Bits set in this register are cleared from INT_EVENT1. See Section 2.3
0x19	Reserved				
0x1A	STATUS	RO	5	yes	Status bit field for non-interrupt events. See Section 2.4
0x1B-0x25	Reserved				
0x26	POWER_PATH_STATUS	RO	5	no	Power Path Status. See Section 2.5
0x27-0x28	Reserved				
0x29	PORT_CONTROL	RW	4	yes	Configuration bits affecting system policy. These bits may change during normal operation and are used for controlling the respective port. The PD Controller does not take immediate action upon writing. Changes made to this register take effect the next time the appropriate policy is invoked. Initialized by Application Customization. See Section 2.6
0x2A-0x2C	Reserved				
0x2D	BOOT_STATUS	RO	5	no	Detailed status of boot process. This register provides details on PD Controller boot flags, Customer OTP configuration, and silicon revision See Section 2.7
0x2E	BUILD_DESCRIPTION	RO	49	no	Build description. This is an ASCII string that uniquely identifies custom build information.
0x2F	DEVICE_INFO	RO	40	no	Device information. This is an ASCII string with hardware and firmware version information of the PD Controller.
0x30	RX_SOURCE_CAPS	RO	29	yes	Received Source Capabilties. This register stores latest Source Capabilities message received over BMC. See Section 2.8
0x31	RX_SINK_CAPS	RO	29	yes	Received Sink Capabilities. This register stores latest Sink Capabilities message received over BMC. See Section 2.9
0x32	TX_SOURCE_CAPS	RW	31	yes	Source Capabilities for sending. This register stores PDOs and settings for outgoing Source Capabilities PD messages. Initialized by Application Customization. See Section 2.10

Table 1-1. Unique Address Interface Registers (continued)

Register Number[1]	Register Name	Access	# Data Bytes	Unique per Port	Description
0x33	TX_SINK_CAPS	RW	29	yes	Sink Capabilities for sending. This register stores PDOs for outgoing Sink Capabilities USB PD messages. Initialized by Application Customization. See Section 2.11
0x34	ACTIVE_CONTRACT_PDO	RO	6	yes	Power data object for active contract. This register stores PDO data for the current explicit USB PD contract, or all zeroes if no contract. See Section 2.12
0x35	ACTIVE_CONTRACT_RDO	RO	4	yes	Power data object for the active contract. This register stores the RDO of the current explicit USB PD contract, or all zeroes if no contract. See Section 2.13
0x36-0x3E	Reserved				
0x3F	POWER_STATUS	RO	2	yes	Details about the power of the connection. This register reports status regarding the power of the connection. See Section 2.14
0x40	PD_STATUS	RO	4	yes	Status of PD and Type-C state-machine. This register contains details regarding the status of PD messages and the Type-C state machine. See Section 2.15
0x41-0x68	Reserved				
0x69	TYPEC_STATE	RO	4	yes	Contains current status of both CCn pins. See Section 2.17
0x6A-0x71	Reserved				
0x72	GPIO_STATUS	RO	8	no	Captures status and settings of all GPIO pins. See Section 2.19
0x73-0x7E	Reserved				

(1) Any register number not shown is reserved for proprietary use.

The PD Controller implements the Unique Address Interface Tasks defined in Table 1-1.

Table 1-2. Unique Address Interface Tasks

Command 4CC	Type	Command Summary	References
PBMs	Patch Bundle Update	Start Patch Burst Download Sequence	See Section 3.3.1
PBMc	Patch Bundle Update	Patch Burst Download Complete	See Section 3.3.2
PBMe	Patch Bundle Update	Patch Burst Mode Exit	See Section 3.3.3
GO2P	Patch Bundle Update	Forces PD controller to return to 'PTCH' mode and wait for patch over I2C.	See Section 3.3.5
GSkC	PD Message	PD Get Sink Capabilities	See Section 3.2.5
GSrC	PD Message	PD Get Source Capabilities	See Section 3.2.6
SSrC	PD Message	PD Send Source Capabilities	See Section 3.2.7
SWDF	PD Message	PD DR_Swap to DFP	See Section 3.2.3
SWSk	PD Message	PD PR_Swap to Sink	See Section 3.2.1
SWSr	PD Message	PD PR_Swap to Source	See Section 3.2.2

Table 1-2. Unique Address Interface Tasks (continued)

Command 4CC	Type	Command Summary	References
SWUF	PD Message	PD DR_Swap to UFP	See Section 3.2.4
DBfg	System	Clear Dead Battery Flag	See Section 3.4.1
I2Cr	System	Executes I2C read transaction on I2Cm.	See Section 3.4.2
I2Cw	System	Executes I2C write transaction on I2Cm.	See Section 3.4.3

This page intentionally left blank.

Chapter 2

Unique Address Interface Register Detailed Descriptions

2.1 0x03 MODE Register

The Mode register is a 32-bit register that returns 4 ASCII characters.

Table 2-1. 0x03 MODE Register

Address	Name	Access	Length	Unique per Port	Power-Up Default
0x03	MODE	RO	4	no	See table below.

Table 2-2. 0x03 MODE Register Bit Field Definitions

Bits	Name	Description
31:0	mode	The mode described in 4 ASCII characters. See table below for more details.

Table 2-3. Description of Device Modes

Mode register value	Description	$\overline{\text{I2Cs\_IRQ}}$ may be asserted	Register availability
'APP '	The PD Controller is fully functioning in the application firmware.	Yes	All registers fully available.
'BOOT'	Device booting in dead battery.	No	Limited (see list at the end of this table)
'PTCH'	Device in patch mode.	Yes	Limited (see list at the end of this table)
Any other value	The PD Controller is functioning in a limited capacity.	No	No

The host must not read or write most registers while the device is in the 'BOOT' or 'PTCH' mode. Only the following registers are available in 'BOOT' and 'PTCH' modes:

- the 4CC patch commands
- MODE (0x03)
- TYPE (0x04), VERSION (0x0F)
- CMD1 (0x08), DATA1 (0x09)
- DEVICE_CAPABILITIES (0x0D)
- INT_EVENT1 (0x14), INT_MASK1 (0x16), and INT_CLEAR1 (0x18)
- BOOT_STATUS (0x2D)
- DEVICE_INFO (0x2F)

2.2 0x0D DEVICE_CAPABILITIES Register

Table 2-4. 0x0D DEVICE_CAPABILITIES Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x0D	DEVICE_CAPABILITIES	RO	4	no	

Table 2-5. 0x0D DEVICE_CAPABILITIES Register Bit Field Definitions

Bits	Name	Description	
31:8	Reserved		
7	I2CmLevel	Pull-up voltage required for I2Cm port.	
		0b	1.8V or 3.3V
		1b	3.3V
6:5	BC1p2Supported	BC 1.2 support capability.	
		00b	Not supported.
		01b	Only source supported.
		10b	Reserved
		11b	
4:3	Reserved		
2	UsbPdCapability	USB Power Delivery capability.	
		0b	Supported
		1b	Not supported.
1:0	PowerRole	Power Role capability.	
		00b	Both source and sink roles supported (DRP).
		01b	Source-only.
		10b	
		11b	Source-only.

2.3 0x14 - 0x19 INT_EVENT, INT_MASK, INT_CLEAR Registers

Bytes 1 to 10 of this register are port-specific, but Byte 11 is common to all ports in the PD controller.

Table 2-6. 0x14 - 0x19 INT_EVENTX, INT_MASKX, INT_CLEARX Registers

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x14	INT_EVENT1	RO	11	yes	0
0x15	Reserved				
0x16	INT_MASK1	RW	11	yes	Initialized by Application Configuration
0x17	Reserved				
0x18	INT_CLEAR1	RW	11	yes	0
0x19	Reserved				

Table 2-7. 0x14 - 0x19 INT_EVENTX, INT_MASKX, INT_CLEARX Registers Bit Field Definitions

Bits	Name	Description
Byte 11: Patch Status (common to all slave ports)		
7:3	Reserved	
2	I2CMasterNACKed	A transaction on the I2C master was NACKed.
1	ReadyForPatch	Device ready for a patch bundle from the host.
0	PatchLoaded	Patch was loaded to the device.
Bytes 9-10:		
15:2	Reserved	
1	TXMemBufferEmpty	Transmit memory buffer empty.
Bytes 5-8:		
31:15	Reserved	
14	ErrorUnableToSource	The Source was unable to increase the voltage to the negotiated voltage of the contract.
13:12	Reserved	
11	PlugEarlyNotification	A connection has been detected but not debounced.
10	SnkTransitionComplete	This event only occurs when in source mode (PD_STATUS.PresentPDRole = 1b). It occurs tSrcTransition (ms) after sending an Accept message to a Request message, just before sending the PS_RDY message.
9	Reserved	
7	ErrorMessageData	An erroneous message was received.
6	ErrorProtocolError	An unexpected message was received from the partner device.
5	Reserved	
4	ErrorMissingGetCapMessage	The partner device did not respond to the Get_Sink_Cap or Get_Source_Cap message that was sent.
3	ErrorPowerEventOccurred	An OVP, or ILIM event occurred on VBUS. Or a TSD event occurred.
2	ErrorCanProvideVoltageOrCurrentLater	The USB PD Source can provide acceptable voltage and current, but not at the present time. A "wait" message was sent or received.
1	ErrorCannotProvideVoltageOrCurrent	The USB PD Source cannot provide an acceptable voltage and/or current. A Reject message was sent to the Sink or a Capability Mismatch was received from the Sink.
0	ErrorDeviceIncompatible	When set to 1, a USB PD device with an incompatible specification version was connected. Or the partner device is not USB PD capable.
Bytes 1-4:		
30	CMDComplete	Set whenever a non-zero value in CMD register is set to zero or ! CMD.

Table 2-7. 0x14 - 0x19 INT_EVENTX, INT_MASKX, INT_CLEARX Registers Bit Field Definitions (continued)

Bits	Name	Description
29	Reserved	
28	Reserved	
27	PDStatusUpdate	Set whenever contents of PD_STATUS register (0x40) change.
26	StatusUpdate	Set whenever contents of STATUS register (0x1A) change.
25	Reserved	
24	PowerStatusUpdate	Set whenever contents of POWER_STATUS register (0x3F) change.
23	PPswitchChanged	Set whenever contents of POWER_PATH_STATUS register (0x26) changes.
22	Reserved	
21	UsbHostPresentNoLonger	Set when STATUS.UsbHostPresent transitions to anything other than 11b.
20	UsbHostPresent	Set when STATUS.UsbHostPresent transitions to 11b.
19	Reserved	
18	DRSwapRequested	A DR swap was requested by the Port Partner.
17	PRSwapRequested	A PR swap was requested by the Port Partner.
16	Reserved	
15	Reserved	
14	SourceCapMsgRcvd	This is asserted when a Source Capabilities message is received from the Port Partner.
13	NewContractAsProv	An RDO from the far-end device has been accepted and the PD Controller is a Source. This is asserted after the PS_RDY message has been sent. See ACTIVE_CONTRACT_PDO register (0x34) and ACTIVE_CONTRACT_RDO register (0x35) for details.
12	NewContractAsCons	Far-end source has accepted an RDO sent by the PD Controller as a Sink. See ACTIVE_CONTRACT_PDO register (0x34) and ACTIVE_CONTRACT_RDO register (0x35) for details.
11:6	Reserved	
5	DRSwapComplete	A Data Role swap has completed. See STATUS register (0x1A) and PD_STATUS register (0x40) for port state.
4	PRSwapComplete	A Power role swap has completed. See STATUS register (0x1A) and PD_STATUS register (0x40) for port state.
3	PlugInsertOrRemoval	USB Plug Status has Changed. See Status register for more plug details.
2	Reserved	
1	PDHardReset	A PD Hard Reset has been performed. See PD_STATUS.HardResetDetails for more information.
0	Reserved	

2.4 0x1A STATUS Register

Table 2-8. 0x1A STATUS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x1A	STATUS	RO	5	yes	Never fully reset though many bits change during disconnect and connect.

Table 2-9. 0x1A STATUS Register Bit Field Definitions

Bits	Name	Description	
Byte 5:			
7:0	Reserved		
Bytes 1-4:			
30:28	Reserved		
27	Bist	Indicates if a BIST procedure is in progress.	
		0b	No BIST in progress.
		1b	BIST in progress. This may also be indicated by MODE register (0x03) reading 'BIST'.
26	Reserved		
25:24	ActingAsLegacy	Indicates when PD Controller has gone into a mode where it is acting like a legacy (non PD) device. It can take approximately 10 seconds for the PD controller to determine that it is attached to a legacy source or sink.	
		00b	PD Controller is not in a legacy (non PD) mode
		01b	PD Controller is acting like a legacy sink. It will not respond to USB PD message traffic.
		10b	PD Controller is acting like a legacy source. It will not respond to USB PD message traffic.
		11b	PD controller is acting as a legacy sink (non-PD) port until the dead battery flag is cleared. The PD controller enters this state if no Source Capabilities are received after the boot process is complete. After the dead-battery flag is cleared, the PD controller will send a Hard Reset.
23:22	UsbHostPresent	USB host attachment status.	
		00b	No host present. This means that no far-end device is presently providing VBUS or the PD Controller power role is Source.
		01b	VBUS is being provided by a Port Partner that is a PD device not capable of USB communications.
		10b	VBUS is being provided by a Port Partner that is not a PD device.
		11b	Host present. This means VBUS is being provided by a Port Partner that is USB PD capable and also capable of USB communications.
21:20	VbusStatus	Indicates the present state of VBUS.	
		00b	At vSafe0V (less than 0.8V)
		01b	At vSafe5V (4.75 V to 5.5 V).
		10b	Within expected limits. The limits are determined based on the USB PD negotiated value.
		11b	Not within any of the other specified ranges.
19:7	Reserved		
6	DataRole	PD controller data role. This is only valid after there is a connection.	
		0b	Upward-facing port (UFP)
		1b	Downward-facing port (DFP)
5	PortRole	Current state of PD Controller CCx terminations. This also indicates the PD Controller Power Role, after connected. This bit does not toggle during Unattached.* state transitions.	
		0b	PD Controller is in the Sink role. This means the CCx pull-down is active or the port is disabled/ disconnected.
		1b	PD Controller is Source (CCx pull-up active).
4	PlugOrientation	Plug orientation indicator. Indicates port orientation when known (requires connection).	
		0b	Upside-up orientation (plug CC on CC1). Can also be an unknown orientation or the port may be disabled/disconnected.
		1b	Upside-down orientation (plug CC on CC2).

Table 2-9. 0x1A STATUS Register Bit Field Definitions (continued)

Bits	Name	Description	
3:1	ConnState	Details of a connected plug.	
		000b	No connection.
		001b	Port is disabled.
		010b	Audio connection (Ra/Ra).
		011b	Debug connection (Rd/Rd).
		100b	No connection, Ra detected (Ra but no Rd).
		101b	Reserved (may be used for Rp/Rp Debug connection).
		110b	Connection present, no Ra detected. Can be an Rd (but no Ra) or an Rp detected with no previous Ra detection, includes PD Controller that connected in Attached.SNK.
		111b	Connection present, Ra detected. Can be Rd (and Ra) detected or Rp detected (with previous Ra detection, if the PD Controller started as Source and later swapped to Sink).
0	PlugPresent	Status of the plug	
		0b	No plug is connected.
		1b	A plug is connected.

2.5 0x26 POWER_PATH_STATUS Register

Table 2-10. 0x26 POWER_PATH_STATUS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x26	POWER_PATH_STATUS	RO	5	no	0

Table 2-11. 0x26 POWER_PATH_STATUS Register Bit Field Definitions

Bits	Name	Description	
Bytes 4-5: PP and PP_CABLE over-current			
15:14	PowerSource	Indicates current PD Controller power source. NOTE: Because the Dead Battery flag forces PD Controller to be powered from VBUS, only 10b is valid when this flag is set. Any other setting indicates that the Dead Battery flag is not set.	
		00b	Reserved.
		01b	PD Controller is powered from VIN_3V3.
		10b	PD Controller is powered from VBUS. The Dead Battery flag is set.
		11b	Reserved.
13:12	Reserved	Reserved.	
10	PP_CABLE1_Overcurrent	PP_CABLE1 overcurrent indicator. Asserted if an overcurrent condition exists on PP_CABLE1 (VCONN).	
9:6	Reserved	Reserved.	
4	PP1_Overcurrent	PP_5V1 overcurrent indicator. Asserted if an overcurrent conditions exists on PP1 switch (PP_5V1).	
3:0	Reserved		
Bytes 1-3: PP and PP_CABLE Switch Status			
14:12	PP3switch	Indicates current state of PP3 (PP_EXT1).	
		0h	PP3 switch disabled.
		1h	PP3 switch currently disabled due to fault. The switch is a system output.
		2h	Reserved.
		3h	PP3 switch enabled (system input).
		4h-7h	Reserved.
8:6	PP1switch	Indicates current state of PP1 switch (PP_5V1).	
		0h	PP1 switch disabled.
		1h	PP1 switch currently disabled due to fault. The switch is a system output.
		2h	PP1 switch enabled (system output).
		3h-7h	Reserved.
5:4	Reserved		
1:0	PP_CABLE1_switch	Indicates current state of PP_CABLE1 switch.	
		00b	PP_CABLE1 switch disabled.
		01b	PP_CABLE1 switch currently disabled. The PD controller is waiting for PP5V pin to go high.
		10b	PP_CABLE1 switch CC1 enabled (system output).
		11b	PP_CABLE1 switch CC2 enabled (system output).

2.6 0x29 PORT_CONTROL Register

Table 2-12. 0x29 PORT_CONTROL Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x29	PORT_CONTROL	RW	4	yes	Initialized by Application Configuration

Table 2-13. 0x29 PORT_CONTROL Register Bit Field Definitions

Bits	Name	Description	
31:30	ChargerDetectEnable	Configure the types of legacy chargers to detect.	
		00b	Do not detect any legacy chargers.
		01b	Detect BC 1.2 chargers.
		10b	Reserved, do not use
		11b	Detect BC 1.2 and proprietary legacy chargers.
29	Reserved		
28:26	ChargerAdvertiseEnable	Configure the types of legacy chargers to emulate.	
		0h	Do not emulate any legacy charger. This means only the SDP mode will be used with legacy devices.
		1h	BC 1.2 CDP only.
		2h	BC 1.2 DCP only.
		3h	Reserved.
		4h	Reserved.
		5h	DCP Auto 1 (2.7V and DCP)
		6h	DCP Auto 2 (1.2V, 2.7V and DCP)
		7h	Reserved.
25	Reserved		
24	Resistor15kPresent	Configure D+ and D- termination. Assert this bit if there is a 15kOhm pull-down on D+ and D- (USB2.0 Host Phy pull-downs enabled). This must not be used for DCP or DCP Auto modes.	
		0b	System does NOT have 15 kOhm pull-down. The PD controller will apply 15-kOhm pull-downs as necessary for BC 1.2.
		1b	System has 15-kOhm pull-down. The PD controller will not apply 15 kOhm pull-downs.
23:2	Reserved		
1:0	TypeCCurrent	Type-C Current advertisement. This setting is ignored if a Source role is not enabled and active. This setting is also ignored during an explicit USB PD contract, where the Rp value is used for collision avoidance as required by the USB PD specification. Note that when PP5V is low, the FW will only use the default Type-C current regardless of the value in this field.	
		00b	USB Default Current.
		01b	1.5 A.
		10b	3.0 A.
		11b	Reserved.

2.7 0x2D BOOT_STATUS Register

Table 2-14. 0x2D BOOT_STATUS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x2D	BOOT_STATUS	RO	5	no	Context dependent (never reset)

Table 2-15. 0x2D BOOT_STATUS Register Bit Field Definitions

Bits	Name	Description	
Byte 5: Revision ID			
7:0	REV_ID	Revision ID for the PD controller.	
Bytes 1-4: Boot Flags (treated as a 32-bit little endian value)			
31:29	PatchConfigSource	Source of patch configuration. This field indicates the source of the configuration patch that has been successfully loaded.	
		0h	No configuration has been loaded.
		1h-4h	Reserved.
		5h	A configuration has been loaded from EEPROM.
		6h	A configuration has been loaded from I2C.
		7h	Reserved.
28	Reserved		
27	Reserved		
24	Reserved		
19	MasterTSD	Master thermal shut-down indicator. This bit is asserted if the PD controller is rebooting after the master thermal sensor caused a reset.	
18	Reserved		
16:12	Reserved		
11	Reserved		
10	patchdownloaderr	Asserted when a patch download error occurs.	
9:4	Reserved		
3	I2cEepromPresent	EEPROM presence indicator. This bit is asserted when an EEPROM device was discovered on I2Cm during boot.	
2	DeadBatteryFlag	Dead Battery flag indicator. This bit is asserted when the PD Controller booted in dead-battery mode.	
1	Reserved	Reserved.	
0	PatchHeaderErr	Asserted when a patch bundle header errors.	

2.8 0x30 RX_SOURCE_CAPS Register

Table 2-16. 0x30 RX_SOURCE_CAPS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x30	RX_SOURCE_CAPS	RO	29	yes	Cleared on disconnect, or Hard Reset.

Table 2-17. 0x30 RX_SOURCE_CAPS Register Bit Field Definitions

Bits	Name	Description
Bytes 26-29: PDO #7 (treated as a 32-bit little endian value)		
31:0	SourcePdo7	Seventh Source Capabilities PDO received.
Bytes 22-25: PDO #6 (treated as a 32-bit little endian value)		
31:0	SourcePdo6	Sixth Source Capabilities PDO received.
Bytes 18-21: PDO #5 (treated as a 32-bit little endian value)		
31:0	SourcePdo5	Fifth Source Capabilities PDO received.
Bytes 14-17: PDO #4 (treated as a 32-bit little endian value)		
31:0	SourcePdo4	Fourth Source Capabilities PDO received.
Bytes 10-13: PDO #3 (treated as a 32-bit little endian value)		
31:0	SourcePdo3	Third Source Capabilities PDO received.
Bytes 6-9: PDO #2 (treated as a 32-bit little endian value)		
31:0	SourcePdo2	Second Source Capabilities PDO received.
Bytes 2-5: PDO #1 (treated as a 32-bit little endian value)		
31:0	SourcePdo1	First Source Capabilities PDO received.
Byte 1: Header		
7:3	Reserved	
2:0	numValidPDOs	Number of valid PDOs in this register. Each PDO is 4 bytes (max of 7).

2.9 0x31 RX_SINK_CAPS Register

Table 2-18. 0x31 RX_SINK_CAPS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x31	RX_SINK_CAPS	RO	29	yes	Cleared on disconnect, or Hard Reset.

Table 2-19. 0x31 RX_SINK_CAPS Register Bit Field Definitions

Bits	Name	Description
Bytes 26-29: PDO #7 (treated as a 32-bit little endian value)		
31:0	SinkPdo7	Seventh Sink Capabilities PDO received.
Bytes 22-25: PDO #6 (treated as a 32-bit little endian value)		
31:0	SinkPdo6	Sixth Sink Capabilities PDO received.
Bytes 18-21: PDO #5 (treated as a 32-bit little endian value)		
31:0	SinkPdo5	Fifth Sink Capabilities PDO received.
Bytes 14-17: PDO #4 (treated as a 32-bit little endian value)		
31:0	SinkPdo4	Fourth Sink Capabilities PDO received.
Bytes 10-13: PDO #3 (treated as a 32-bit little endian value)		
31:0	SinkPdo3	Third Sink Capabilities PDO received.
Bytes 6-9: PDO #2 (treated as a 32-bit little endian value)		
31:0	SinkPdo2	Second Sink Capabilities PDO received.
Bytes 2-5: PDO #1 (treated as a 32-bit little endian value)		
31:0	SinkPdo1	First Sink Capabilities PDO received.
Byte 1: Header		
7:3	Reserved	
2:0	numValidPDOs	Number of valid PDOs in this register. Each PDO is 4 bytes (max of 7).

2.10 0x32 TX_SOURCE_CAPS Register

The PD controller will transmit Source Capabilities that are written to this register without verifying them (besides limiting current see below). The user is responsible to write this register correctly per the USB PD requirements. The PD controller will only use the first TXSourceNumPDOs PDO's, the host may write multiple PDO's during configuration then dynamically write TXSourceNumPDOs to change which PDO's are advertised. If this register is changed, the host must subsequently issue the 4CC command 'SSrC'. This will cause the PD controller to re-load this TX Source Capabilities register.

The PD controller will read the capabilities of the cable and limit the maximum current in each PDO to respect the cable's VBUS Current Handling Capability.

Table 2-20. 0x32 TX_SOURCE_CAPS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x32	TX_SOURCE_CAPS	RW	31	yes	Initialized by Application Configuration

Table 2-21. 0x32 TX_SOURCE_CAPS Register Bit Field Definitions

Bits	Name	Description	
Bytes 28-31: PDO #7 (treated as a 32-bit little endian value)			
31:0	TXSourcePDO7	Seventh Source Capabilities PDO contents. See Table 2-23.	
Bytes 24-27: PDO #6 (treated as a 32-bit little endian value)			
31:0	TXSourcePDO6	Sixth Source Capabilities PDO contents. See Table 2-23.	
Bytes 20-23: PDO #5 (treated as a 32-bit little endian value)			
31:0	TXSourcePDO5	Fifth Source Capabilities PDO contents. See Table 2-23.	
Bytes 16-19: PDO #4 (treated as a 32-bit little endian value)			
31:0	TXSourcePDO4	Fourth Source Capabilities PDO contents. See Table 2-23.	
Bytes 12-15: PDO #3 (treated as a 32-bit little endian value)			
31:0	TXSourcePDO3	Third Source Capabilities PDO contents. See Table 2-23.	
Bytes 8-11: PDO #2 (treated as a 32-bit little endian value)			
31:0	TXSourcePDO2	Second Source Capabilities PDO contents. See Table 2-23.	
Bytes 4-7: PDO #1 (treated as a 32-bit little endian value)			
31:0	TXSourcePDO1	First Source Capabilities PDO contents. See Table 2-22.	
Bytes 2-3: Power path configuration for each PDO.			
15:14	Reserved		
13:12	PowerPathForPDO7	Configures which PP to use for PDO7. Same format as PowerPathForPDO2.	
11:10	PowerPathForPDO6	Configures which PP to use for PDO6. Same format as PowerPathForPDO2.	
9:8	PowerPathForPDO5	Configures which PP to use for PDO5. Same format as PowerPathForPDO2.	
7:6	PowerPathForPDO4	Configures which PP to use for PDO4. Same format as PowerPathForPDO2.	
5:4	PowerPathForPDO3	Configures which PP to use for PDO3. Same format as PowerPathForPDO2.	
3:2	PowerPathForPDO2	Configures which PP to use for PDO2.	
		00b	Reserved\.
		01b	Reserved.
		10b	PP_EXT1 is used for this PDO.
1:0	PowerPathForPDO1	Configures which PP to use for PDO1.	
		00b	PP_5V1 is used for this PDO.
		10b	PP_EXT1 is used for this PDO.
Byte 1: Header			
7:3	Reserved		
2:0	numValidPDOs	Number of valid PDOs in this register. Each PDO is 4 bytes (max of 7).	

The PDO's in this register follow the definition in the USB PD specification. It is reproduced here for convenience, but for more details on each field refer to the USB PD specification.

Table 2-22. First PDO

Bits(s)	Description
31:30	Supply Type, this shall always be set to 00b (Fixed Supply).
29	Dual-Role Power, this is overridden by the logical OR of the ProcessSwapToSink, ProcessSwapToSource, InitiateSwapToSink, and InitiateSwapToSource fields in the PORT_CONTRL register.
28	USB Suspend Supported.
27:26	Reserved
25	Dual-Role Data, this is overridden by the logical OR of the ProcessSwapToUFP, ProcessSwapToDFP, InitiateSwapToUFP, and InitiateSwapToDFP fields in the PORT_CONTRL register.
24	Unchunked Extended Messages supported.
23:22	Reserved.
21:20	Peak Current.
19:10	Voltage.
9:0	Maximum Current.

Table 2-23. Other PDO's.

Bits(s)	Description		
	Fixed Supply	Variable Supply	Battery Supply
31:30	00b	01b	10b
29:20	Reserved.	Maximum Voltage	Maximum Voltage
19:10	Voltage	Minimum Voltage	Minimum Voltage
9:0	Maximum Current	Maximum Current	Maximum Allowable Power

2.11 0x33 TX_SINK_CAPS Register

The PD controller transmits the contents of this register as a Sink_Capabilities message after receiving a Get_Sink_Cap message unless its configuration or USB PD rules require a different response in the context.

Note

Writes to this register have no immediate effect. The PD controller updates and uses this register each time it must send a *Sink Capabilities* message.

Table 2-24. 0x33 TX_SINK_CAPS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x33	TX_SINK_CAPS	RW	29	yes	Initialized by Application Configuration

Table 2-25. 0x33 TX_SINK_CAPS Register Bit Field Definitions

Bits	Name	Description
Bytes 26-29: PDO #7 (treated as a 32-bit little endian value)		
31:0	TXSinkPDO7	Seventh Sink Capabilities PDO contents. See Table 2-27.
Bytes 22-25: PDO #6 (treated as a 32-bit little endian value)		
31:0	TXSinkPDO6	Sixth Sink Capabilities PDO contents. See Table 2-27.
Bytes 18-21: PDO #5 (treated as a 32-bit little endian value)		
31:0	TXSinkPDO5	Fifth Sink Capabilities PDO contents. See Table 2-27.
Bytes 14-17: PDO #4 (treated as a 32-bit little endian value)		
31:0	TXSinkPDO4	Fourth Sink Capabilities PDO contents. See Table 2-27.
Bytes 10-13: PDO #3 (treated as a 32-bit little endian value)		
31:0	TXSinkPDO3	Third Sink Capabilities PDO contents. See Table 2-27.
Bytes 6-9: PDO #2 (treated as a 32-bit little endian value)		
31:0	TXSinkPDO2	Second Sink Capabilities PDO contents. See Table 2-27.
Bytes 2-5: PDO #1 (treated as a 32-bit little endian value)		
31:0	TXSinkPDO1	First Sink Capabilities PDO contents. See Table 2-26.
Byte 1: Header		
7:3	Reserved	
2:0	numValidPDOs	

Each PDO in this TX_SINK_CAPS register follows the definition from the USB PD specification, reproduced below for convenience. For more details on the meaning of each field refer to the USB PD specification.

Table 2-26. First PDO

Bits(s)	Description
31:30	Supply Type, this shall always be set to 00b (Fixed Supply)
29	Dual-Role Power, this is overridden by the logical OR of the ProcessSwapToSink, ProcessSwapToSource, InitiateSwapToSink, and InitiateSwapToSource fields in the PORT_CONTRL register.
28	Higher Capability
27:26	Reserved
25	Dual-Role Data, this is overridden by the logical OR of the ProcessSwapToUFP, ProcessSwapToDFP, InitiateSwapToUFP, and InitiateSwapToDFP fields in the PORT_CONTRL register.
24:20	Reserved
19:10	Voltage
9:0	Operational Current

Table 2-27. Other PDO's.

Bits(s)	Description			
	Fixed Supply	Variable Supply	Battery Supply	APDO (PPS)
31:30	00b	01b	10b	11b
29:28	Reserved.	Maximum Voltage	Maximum Voltage	00b
27:25				Reserved
24:20				MaxPpsVoltage
19:17	Voltage	Minimum Voltage	Minimum Voltage	
16				Reserved
15:10				MinPpsVoltage
9:8	Operational Current	Operational Current	Operational Power	
7				Reserved
6:0				MaxPpsCurrent

2.12 0x34 ACTIVE_CONTRACT_PDO Register

Table 2-28. 0x34 ACTIVE_CONTRACT_PDO Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x34	ACTIVE_CONTRACT_PDO	RO	6	yes	Cleared on disconnect, Hard Reset, or PR_Swap.

Table 2-29. 0x34 ACTIVE_CONTRACT_PDO Register Bit Field Definitions

Bits	Name	Description
Bytes 5-6: Source Properties		
15:10	Reserved	
9:0	firstPDOControlBits	Contains bits 29:20 of the first PDO. It does not matter which PDO was selected, this field is always drawn from the first PDO.
Bytes 1-4: Contract PDO (treated as 32-bit little endian value)		
31:0	ActivePDO	Power data object. This field contains the contents of the PDO Requested by PD Controller as Sink and Accepted by Source, after it is Accepted by Source.

2.13 0x35 ACTIVE_CONTRACT_RDO Register

Table 2-30. 0x35 ACTIVE_CONTRACT_RDO Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x35	ACTIVE_CONTRACT_RDO	RO	4	yes	Cleared on disconnect, Hard Reset, or PR_Swap.

Table 2-31. 0x35 ACTIVE_CONTRACT_RDO Register Bit Field Definitions

Bits	Name	Description
Bytes 1-4: Contract RDO (treated as 32-bit little endian value)		
31	Reserved	
30:28	ObjectPosition	As defined by USB PD.
27	GiveBackFlag	As defined by USB PD.
26	CapabilityMismatch	As defined by USB PD.
25	USBCommCapable	As defined by USB PD.
24	NoUSBSuspend	As defined by USB PD.
23	UnchunkedSupported	As defined by USB PD.
22:20	Reserved	
19:10	OperatingX	As defined by USB PD.
9:0	MaxMinOperatingX	As defined by USB PD.

2.14 0x3F POWER_STATUS Register

Table 2-32. 0x3F POWER_STATUS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x3F	POWER_STATUS	RO	2	yes	Cleared on disconnect.

Table 2-33. 0x3F POWER_STATUS Register Bit Field Definitions

Bits	Name	Description	
15:10	Reserved	Reserved.	
9:8	ChargerAdvertiseStatus(1) (2)	Charger Advertise Status.	
		00b	Charger advertise disabled or not run.
		01b	Charger advertisement in process.
		10b	Charger advertisement complete.
		11b	Reserved.
7:4	ChargerDetectStatus(1) (2)		
		0h	Charger detection disabled or not run.
		1h	Charger detection in progress.
		2h	Charger detection complete, none detected.
		3h	Charger detection complete, SDP detected.
		4h	Charger detection complete, BC 1.2 CDP detected.
		5h	Charger detection complete, BC 1.2 DCP detected.
		6h	Charger detection complete, Divider1 DCP detected.
		7h	Charger detection complete, Divider2 DCP detected.
		8h	Charger detection complete, Divider3 DCP detected.
		9h	Charger detection complete, 1.2-V DCP detected.
		Ah-Fh	Reserved.
3:2	TypeCCurrent	This field is redundant with PD_STATUS.CCPullUp in register 0x40 when SourceSink is 1b. This field is redundant with PORT_CONTROL.TypeCCurrent in register 0x29 when SourceSink is 0b. In the future, this redundant field may be removed. This field is intended for Type-C Sink operation. If the port is connected as source, the field is updated upon initial connection only.	
		00b	USB Default Current.
		01b	1.5 A.
		10b	3.0 A.
		11b	Explicit PD contract sets current. A PD contract was negotiated (see other PD registers for more details).
1	SourceSink	Source / Sink indicator. This bit is equivalent to PresentPDRole in register 0x40. It is replicated in this register for convenience. In the future, this redundant bit may be removed.	
		0b	Connection requests power. The PD Controller is the source.
		1b	Connection provides power (PD Controller as sink).
0	PowerConnection	Asserted if there is a connection. This bit is asserted when PlugPresent is TRUE and ConnState is greater than 5h. So it is redundant with information from register 0x1A. It is replicated in this register for convenience. In the future this redundant bit may be removed.	
		0b	No connection. The rest of bits in this register are not valid.
		1b	Connection present. See other bits in register for more details.

(1) This feature is not supported by TPS25750_F509.04.02.
(2) This feature is not supported by TPS25750_F509.04.02.

2.15 0x40 PD_STATUS Register

Table 2-34. 0x40 PD_STATUS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x40	PD_STATUS	RO	4	yes	Cleared on connect.

Table 2-35. 0x40 PD_STATUS Register Bit Field Definitions

Bits	Name	Description	
31	Reserved	Reserved.	
30:28	DataResetDetails	Reason for Data Reset.	
		0h	Reset value: no data reset.
		1h	Data Reset message received from port partner.
27:22	ErrorRecoveryDetails	2h-7h	Reserved
		Reason for Error Recovery.	
		00h	reset value: no error recovery.
		01h	System: over-temperature shut-down.
		02h	System: PP5V went low unexpectedly.
		03h	System: fault input GPIO was asserted. GPIO can be Fault_Input_Event.
		04h	System: Over-voltage detected on the VBUS pin.
		05h	Reserved.
		06h	System: ILIM on PP_5V.
		07h	System: ILIM on PP_CABLE.
		08h	System: OVP on CC detected.
		9h-Fh	Reserved.
		10h	Protocol error: invalid DR_Swap.
		11h	Protocol error: no Good_CRC during a PR_Swap sequence. This happens if the sink did not turn off in time.
		12h-14h	Reserved.
		15h	Policy Engine: NoResponse timer timed out.
		16h	Policy Engine: PSSourceOffTimer timed out during PR_Swap.
		17h	Policy Engine: PSSourceOnTimer timed out during PR_Swap.
		18h-21h	Reserved
		22h	HI: Swapping error during dead-battery. After Host cleared the dead-battery flag and the configuration requires operating as a source, but the PD controller is unable to swap to sink.
		23h-2Fh	Reserved.
		30h	Type-C: an error occurred in the Attached.SRC state.
		31h	Type-C: VCONN failed to discharge.
		32h-3Fh	Reserved.
21:16	HardResetDetails	Reason for Hard Reset	
		00h	Reset value, no hard reset.
		01h	Received from Port Partner.
		02h	Requested by host.
		03h	Invalid DR_Swap request during Active Mode
		04h	Required by policy engine: DischargeFailed.
		05h	Required by policy engine: NoResponseTimeOut.
		06h	Required by policy engine: SendSoftReset.
		07h	Required by policy engine: Sink_SelectCapability.

Table 2-35. 0x40 PD_STATUS Register Bit Field Definitions (continued)

Bits	Name	Description	
		08h	Required by policy engine: Sink_TransitionSink.
		09h	Required by policy engine: Sink_WaitForCapabilities.
		0Ah	Required by policy engine: SoftReset.
		0Bh	Required by policy engine: SourceOnTimeout.
		0Ch	Required by policy engine: Source_CapabilityResponse.
		0Dh	Required by policy engine: Source_SendCapabilities.
		0Eh	Required by policy engine: SourcingFault.
		0Fh	Required by policy engine: UnableToSource.
		10h	Reserved
		11h	Required by policy engine: Unexpected message
		12h	Required by policy engine: Failure to to complete the VCONN recovery sequence within 200ms after PP5V rising edge.
		13h-3Fh	Reserved.
15:13	Reserved		
12:8	SoftResetDetails	Reason for Soft Reset.	
		0h	Reset value, no soft reset.
		1h	Soft reset received from Port Partner.
		2h	Reserved.
		3h	Reserved.
		4h	Received source capabilities message was invalid.
		5h	Message retries were exhausted.
		6h	Received an accept message unexpectedly.
		7h	Received a control message unexpectedly.
		8h	Received a GetSinkCap message unexpectedly.
		9h	Received a GetSourceCap message unexpectedly.
		Ah	Received a GotoMin message unexpectedly.
		Bh	Received a PS_RDY message unexpectedly.
		Ch	Received a Ping message unexpectedly.
		Dh	Received a Reject message unexpectedly.
		Eh	Received a Request message unexpectedly.
		Fh	Received a Sink Capabilities message unexpectedly.
		10h	Received Source Capabilities message unexpected.
		11h	Received a Swap message unexpectedly.
		12h	Received a Wait Capabilities message unexpectedly.
		13h	Received an unknown control message.
		14h	Received an unknown data message.
		15h	To initialize SOP' controller in plug.
		16h	To initialize SOP'' controller in plug.
		17h	Received an Extended message unexpectedly.
		18h	Received an unknown Extended message.
		19h	Received a data message unexpectedly.
		1Ah	Received a Not Supported message unexpectedly.
		1Bh	Received a Get_Status message unexpectedly.
		1Ch-1Fh	Reserved.
7	Reserved		

Table 2-35. 0x40 PD_STATUS Register Bit Field Definitions (continued)

Bits	Name	Description	
6	PresentPDRole	Present PD power role. The PD Controller is acting under this PD power role.	
		0b	Sink.
		1b	Source.
5:4	PortType	Present Type-C power role. The PD Controller is acting under this Type-C power role.	
		00b	Sink/Source.
		01b	Sink.
		10b	Source.
		11b	Source/Sink.
3:2	CCPullUp	CC Pull-up value. The pull-up value detected by PD Controller when in CC Pull-down mode.	
		00b	Not in CC pull-down mode / no CC pull-up detected.
		01b	USB Default current.
		10b	1.5 A (SinkTxNG).
		11b	3.0 A (SinkTxOK).
1:0	Reserved		

2.16 GPIO Events

Table 2-36. GPIO Events

Event #	Event Name	I/O	Description
76	PdNegotiationInProcess	Output	When in source mode, this GPIO is asserted after a Request message is received, before sending the Accept message. The GPIO is de-asserted after the PS_RDY message is sent.When in sink mode, this GPIO is asserted right before sending a Request message, and de-asserted after a PS_RDY message is received.In either mode, the GPIO is de-asserted when a detach occurs.
75	AttachedAsSink	Output	When the PD controller has a port that is connected to a Source, this GPIO will be asserted. The GPIO is de-asserted upon disconnect, hard reset, during power-role swap only if none of the ports in the PD controller are connected to a source.
73	EnableSource	Output	When the PD controller sends an Accept message to start sourcing VBUS under a high-power contract this GPIO is asserted high. It will remain high as long as the high-power contract is active. If the contract transitions from a high-power contract to a low-power contract, this GPIO will have a high-to-low transition after the PS_Rdy message is sent.
71	Reserved		
69	MRESET	Input	Upon a rising edge on this GPIO the PD controller will drive a rising edge on the RESETZ GPIO after a delay . Upon a falling edge on this GPIO the PD controller will drive a falling edge on the RESETZ GPIO after a delay .
68	RESETZ	Output	This works in conjunction with MRESET.
67	Fault_Condition_Active_Low_Global	Output	Asserts low on an overcurrent event on Port .
61	Dp_Dm_Mux_Enable_Event	Output	This GPIO must be used to enable/disable a USB 2.0 D+/D- mux. The GPIO is driven high upon connection, and low upon disconnect.
52	VCONN_On_Event	Output	This GPIO is asserted when PP_CABLE1 is enabled.
50	Debug_Accessory_Mode_Event	Output	Output: This GPIO is asserted high when a Debug Accessory is attached.
48	Audio_Mode_Event	Output	Output: This GPIO is asserted high when an Audio Accessory (Ra/Ra) is attached.

Table 2-36. GPIO Events (continued)

Event #	Event Name	I/O	Description
45	Prevent_DRSwap_To_UFP_Event	Input	When the GPIO is high, the PD controller will reject any DR_Swap messages from the Port Partner requesting to change the data-role from DFP to UFP.
44	UFP_Indicator_Event	Output	The GPIO is driven high when the data role of any port in the PD controller is UFP.
43	Barrel_Jack_Event	Input	When this GPIO is high, the PD controller interprets it to mean that a barrel-jack adaptor is connected and the system has power. Therefore, when this event is triggered, it will clear the dead-battery flag and attempt to perform a Power Role Swap to be a Source.
42:36	Reserved		
35	Fault_Condition_Active_Low_Event	Output	Asserts low on an overcurrent event.
33	Fault_Input_Event	Input	When set low by the system, Port1 enters the Type-C Error Recovery State. When set high, no action is taken.
32:30	Reserved		
29	UFP_DFP_Event	Output	Output: Asserted high when Port1 is operating as UFP. Asserted low when port is operating as DFP.
28:14	Reserved		
13	SourcePDOContractBit2	Output	Output: Bit2 of binary encoded outputs indicating when a Source PDO1 through PDO7 has been negotiated (the Accept message has been transmitted and the tSrcTransition timer has expired).
12	SourcePDOContractBit1	Output	Output: Bit1 of binary encoded outputs indicating when a Source PDO1 through PDO7 has been negotiated (the Accept message has been transmitted and the tSrcTransition timer has expired).
11	SourcePDOContractBit0	Output	Output: Bit0 of binary encoded outputs indicating when a Source PDO1 through PDO7 has been negotiated (the Accept message has been transmitted and the tSrcTransition timer has expired).
10	SourcePDO4Contract	Output	Output: Asserted high when a Source PDO4 has been negotiated (the Accept message has been transmitted and the tSrcTransition timer has expired). D-asserted when a PDO other than PDO2 has been negotiated.
9	SourcePDO3Contract	Output	Output: Asserted high when a Source PDO3 has been negotiated (the Accept message has been transmitted and the tSrcTransition timer has expired). D-asserted when a PDO other than PDO2 has been negotiated.
8	SourcePDO2Contract	Output	Output: Asserted high when a Source PDO2 has been negotiated (the Accept message has been transmitted and the tSrcTransition timer has expired). D-asserted when a PDO other than PDO2 has been negotiated.
7	SourcePDO1Contract	Output	Output: Asserted high when a Source PDO1 has been negotiated (the Accept message has been transmitted and the tSrcTransition timer has expired). D-asserted when a PDO other than PDO1 has been negotiated.
6:4	Reserved		
3	Cable_Orientation_Event	Output	Output: Indicates the plug orientation. Low when the plug is connected upside-up (CC1 connected to CC in cable) or disconnected. High when plug is connected upside-down (CC2 connected to CC in cable).
1	PlugEvent	Output	Output: Asserted high when plug event (attached state) has occurred, otherwise low.
0	NullEvent	NA	No event associated with this GPIO.

2.17 0x69 TYPEC_STATE Register

Table 2-37. 0x69 TYPEC_STATE Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x69	TYPEC_STATE	RO	4	yes	0

Table 2-38. 0x69 TYPEC_STATE Register Bit Field Definitions

Bits	Name	Description	
31:24	TypeCPortState	Present state of Type-C state-machine.	
		00h	Disabled.
		01h-04h	Reserved.
		05h	ErrorRecovery.
		06h-23h	Reserved.
		24h	Unattached.Accessory.
		25h-2Ah	Reserved.
		2Bh	AttachWait.Accessory.
		2Ch-44h	Reserved.
		45h	Try.SRC.
		46h-4Dh	Reserved.
		4Eh	TryWait.SNK.
		4Fh	Try.SNK.
		50h	TryWait.SRC.
		51-5Fh	Reserved.
		60h	Attached.SRC.
		61h	Attached.SNK.
		62h	AudioAccessory.
		63h	DebugAccessory.
		64h	AttachWait.SRC.
		65h	AttachWait.SNK.
		66h	Unattached.SNK.
		67h	Unattached.SRC.
		68h-FFh	Reserved.
23:16	Cc2PinState	State of CC2 pin	
		00h	Not connected.
		01h	Ra detected (Source only).
		02h	Rd detected (Source only)
		03h	USB Default Advertisement detected (Sink only).
		04h	1.5A Advertisement detected (Sink only).
		05h	3.0A Advertisement detected (Sink only).
		06h-FFh	Reserved.
15:8	Cc1PinState	State of CC1 pin	
		00h	Not connected.
		01h	Ra detected (Source only).
		02h	Rd detected (Source only).
		03h	USB Default Advertisement detected (Sink only).
		04h	1.5A Advertisement detected (Sink only).
		05h	3.0A Advertisement detected (Sink only).
		06h-FFh	Reserved.

Table 2-38. 0x69 TYPEC_STATE Register Bit Field Definitions (continued)

Bits	Name	Description	
7:0	CcPinForPd	CC pin used for PD communication.	
		00h	Not connected.
		01h	CC1 is used for USB PD communication.
		02h	CC2 is used for USB PD communication.
		03h-FFh	Reserved.

2.18 0x70 SLEEP_CONFIG Register

Table 2-39. 0x70 SLEEP_CONFIG Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x70	SLEEP_CONFIG	RW	1	no	Initialized by Application Configuration

Table 2-40. 0x70 SLEEP_CONFIG Register Bit Field Definitions

Bits	Name	Description
7:1	Reserved	
0	SleepModeAllowed	If this bit is asserted the PD controller will enter sleep modes after device is idle for Sleep Time.

2.19 0x72 GPIO_STATUS Register

Check the device-specific datasheet for the available GPIO because it may vary by device type.

Table 2-41. 0x72 GPIO_STATUS Register

Address	Name	Access	Length	Unique Per Port	Power-Up Default
0x72	GPIO_STATUS	RO	8	no	N/A

Table 2-42. 0x72 GPIO_STATUS Register Bit Field Definitions

Bits	Name	Description
Bytes 5-8: Direction Configuration		
31:9	Reserved	
8	GPIO8Dir(1)	This bit is asserted when this GPIO is configured as an output.
7	GPIO7Dir	This bit is asserted when this GPIO is configured as an output.
6	GPIO6Dir	This bit is asserted when this GPIO is configured as an output.
5	GPIO5Dir	This bit is asserted when this GPIO is configured as an output.
4	GPIO4Dir	This bit is asserted when this GPIO is configured as an output.
3	GPIO3Dir	This bit is asserted when this GPIO is configured as an output.
2	GPIO2Dir	This bit is asserted when this GPIO is configured as an output.
1	GPIO1Dir	This bit is asserted when this GPIO is configured as an output.
0	GPIO0Dir	This bit is asserted when this GPIO is configured as an output.
Bytes 1-4: Output Data		
31:13	Reserved	
12	GPIO12Data	Asserted if a logic high is detected on the GPIO.
11	Reserved	
10	Reserved	
9	Reserved	
8	GPIO8Data(1)	Asserted if a logic high is detected on the GPIO.
7	GPIO7Data	Asserted if a logic high is detected on the GPIO.
6	GPIO6Data	Asserted if a logic high is detected on the GPIO.
5	GPIO5Data	Asserted if a logic high is detected on the GPIO.
4	GPIO4Data	Asserted if a logic high is detected on the GPIO.
3	GPIO3Data	Asserted if a logic high is detected on the GPIO.
2	GPIO2Data	Asserted if a logic high is detected on the GPIO.
1	GPIO1Data	Asserted if a logic high is detected on the GPIO.
0	GPIO0Data	Asserted if a logic high is detected on the GPIO.

(1) This feature is not supported by TPS25750_F509.05.02.

3.1 Overview

Note

This section is for advanced users and the features listed here are only optional. An EC or Host is required in your system to implement the features described in the following section.

This section describes the 4CC Tasks defined by the PD Controller Host Interface. The Tasks are categorized into various sub-groups in this section. All Tasks that return data using the DATA registers always ensure the proper output data is loaded into those registers before setting the CMD register to 0 to indicate Task completion. DATA is never modified by PD Controller after CMD has been changed to 0, to ensure the Host can retrieve data from the previously-executed Task, and to ensure the Host can load these registers for a future Task without risk of overwriting. Note that other registers may continue to be updated after a Task completes, as Tasks may have additional side effects.

Many of the Tasks return a status code in the first byte of the DATA register. The standard Task response byte is defined in Table 3-1. The remaining DATA bytes may be used at each Task's discretion.

Table 3-1. Standard Task Response

Description	**Tasks are a special form of Tasks that return a status code in the first byte of the DATA register.**			
Output DATA	**Bit**	**Name**	**Description**	
	Byte 1: Task Return Code			
	7:4	Reserved	Reserved for standard Tasks. May be used by certain Tasks for Task-specific return codes. Successful return codes may use this byte provided TaskResult is 0x0.	
	3:0	TaskResult	Standard Task return codes.	
			0x0	Task completed successfully.
			0x1	Task timed-out or aborted by 'ABRT' Request.
			0x2	Reserved.
			0x3	Task rejected.
			0x4	Task rejected because the Rx Buffer was locked. This is for Tasks that require the PD controller to use the Rx Buffer.
			0x5-0xF	Reserved for standard Tasks. May be used by certain Tasks for Task-specific error codes. Treated as an error when encountered.

3.2 PD Message Tasks

3.2.1 'SWSk' - PD PR_Swap to Sink

Table 3-2. 'SWSk' - PD PR_Swap to Sink

Description	The 'SWSk' Task instructs PD Controller to attempt to become a Sink via PR_Swap at the first opportunity while maintaining policy engine compliance.
INPUT DATAX	None.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'SWSk' Task completes either when the PR_Swap is finished or it is otherwise determined to not be possible or fails. The Task may continue to run because of Wait messages being sent by the Source. The 'SWSk' Task shall be considered rejected if: • The Source indicated via Source Capabilities that it does not support Dual-Role Power. • The PR_Swap is Rejected. The 'SWSk' Task shall be considered timed-out if: • The PR_Swap is Accepted but failed to complete per the PD spec. The 'SWSk' Task shall be considered successful if: • PD Controller is already in the Sink power role. • The PR_Swap is Accepted and completes normally.
Side Effects	When the 'SWSk' Task completes successfully PD Controller will have transitioned to the Sink power role, which impacts other registers. If the PR_Swap fails after the Accept is sent then Soft and/or Hard Resets are likely to occur per PD spec requirements.
Additional Information	None.

3.2.2 'SWSr' - PD PR_Swap to Source

Table 3-3. 'SWSr' - PD PR_Swap to Source

Description	The 'SWSr' Task instructs PD Controller to attempt to become a Source via PR_Swap at the first opportunity while maintaining policy engine compliance.
INPUT DATAX	None.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'SWSr' Task completes either when the PR_Swap is finished or it is otherwise determined to not be possible or fails. The Task may continue to run because of Wait messages being sent by the Sink. The 'SWSr' Task shall be considered rejected if: • The Sink previously indicated via Sink or Source Capabilities that it does not support Dual-Role Power. • The PR_Swap is Rejected. The 'SWSr' Task shall be considered timed-out if: • The PR_Swap is Accepted but failed to complete per the PD spec. The 'SWSr' Task shall be considered successful if: • PD Controller is already in the Source power role. • The PR_Swap is Accepted and completes normally.
Side Effects	When the 'SWSr' Task completes successfully PD Controller will have transitioned to the Source power role, which impacts other registers. If the PR_Swap fails after the Accept is sent then Soft and/or Hard Resets are likely to occur per PD spec requirements.
Additional Information	None.

3.2.3 'SWDF' - PD DR_Swap to DFP

Table 3-4. 'SWDF' - PD DR_Swap to DFP

Description	The 'SWDF' Task instructs PD Controller to attempt to become a DFPvia DR_Swap at the first opportunity while maintaining policy engine compliance.
INPUT DATAX	None.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'SWDF' Task completes either when the DR_Swap is finished or it is otherwise determined to not be possible or fails. The Task may continue to run because of Wait messages being sent by the DFP. The 'SWDF' Task shall be considered rejected if: • The UFP indicated via Source or Sink Capabilities that it does not support Data Role Swap. • The DR_Swap is Rejected. The 'SWDF' Task shall be considered successful if: • PD Controller is already in the DFP data role. • The DR_Swap is Accepted and completes normally.
Side Effects	When the 'SWDF' Task completes successfully PD Controller will have transitioned to the DFP data role, which impacts other registers. If the DR_Swap fails after the Accept is sent then Soft and/or Hard Resets are likely to occur per PD spec requirements.
Additional Information	None.

3.2.4 'SWUF' - PD DR_Swap to UFP

Table 3-5. 'SWUF' - PD DR_Swap to UFP

Description	The 'SWUF' Task instructs PD Controller to attempt to become a UFPvia DR_Swap at the first opportunity while maintaining policy engine compliance.
INPUT DATAX	None.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'SWUF' Task completes either when the DR_Swap is finished or it is otherwise determined to not be possible or fails. The Task may continue to run because of Wait messages being sent by the UFP. The 'SWUF' Task shall be considered rejected if: • The DFP indicated via Source or Sink Capabilities that it does not support Data Role Swap. • The DR_Swap is Rejected. The 'SWUF' Task shall be considered successful if: • PD Controller is already in the UFP data role. • The DR_Swap is Accepted and completes normally.
Side Effects	When the 'SWDF' Task completes successfully PD Controller will have transitioned to the UFP data role, which impacts other registers. If the DR_Swap fails after the Accept is sent then Soft and/or Hard Resets are likely to occur per PD spec requirements.
Additional Information	None.

3.2.5 'GSkC' - PD Get Sink Capabilities

Table 3-6. 'GSkC' - PD Get Sink Capabilities

Description	**The 'GSkC' Task instructs PD Controller to issue a Get_Sink_Capmessage to the Port Partner at the first opportunity while maintaining policyengine compliance.**
INPUT DATAX	None.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'GSkC' Task completes either when the Sink Capabilities message is received or the Task otherwise fails. • The Port Partner is a Source and indicated it was not Dual-Role Power. • The Port Partner responds to the Get_Sink_Cap message with a Reject or Not_Supported message. The 'GSkC' Task shall be considered timed-out if: • The Port Partner fails to respond within the time required by the PD spec. The 'GSkC' Task shall be considered successful if: • The Get_Sink_Cap message is sent, GoodCRC'ed and a Sink Capabilities response is received and processed.
Side Effects	When the 'GSkC' Task completes successfully the *RX_SINK_CAPS* register (0x31) will have been updated.
Additional Information	None.

3.2.6 'GSrC' - PD Get Source Capabilities

Table 3-7. 'GSrC' - PD Get Source Capabilities

Description	**The 'GSrC' Task instructs PD Controller to issue a Get_Source_Capmessage to the Port Partner device at the first opportunity while maintainingpolicy engine compliance.**
INPUT DATAX	None.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'GSrC' Task completes either when the Source Capabilities message is received or the Task otherwise fails. The 'GSrC' Task shall be considered rejected if: • The Port Partner is a Sink and indicated (via previous Source or Sink Capabilities) it was not Dual-Role Power. • The Port Partner responds to the Get_Source_Cap message with a Reject or Not_Supported message. The 'GSrC' Task shall be considered timed-out if: • The Port Partner fails to respond within the time required by the PD spec. The 'GSrC' Task shall be considered successful if: • The Get_Source_Cap message is sent, GoodCRC'ed and a Source Capabilities response is received and processed.
Side Effects	When the 'GSrC' Task completes successfully the *RX_SOURCE_CAPS* register (0x30) will have been updated.
Additional Information	None.

3.2.7 'SSrC' - PD Send Source Capabilities

Table 3-8. 'SSrC' - PD Send Source Capabilities

Description	**The 'SSrC' Task instructs the PD Controller to send a SourceCapabilities message at the first opportunity while maintaining policy enginecompliance.**
INPUT DATAX	None.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'SSrC' Task completes either when the Sink Capabilities message GoodCRC is received or the Task otherwise fails. The 'SSrC' Task shall be considered rejected if: • PD Controller is not in a Source role. The 'SSrC' Task shall be considered timed-out if: • The Source Capabilities message was sent but no GoodCRC was received. The 'SSrC' Task shall be considered successful if: • The Source Capabilities message was sent and a GoodCRC is received.
Side Effects	Other registers may change as a result of the contract negotiation that begins with the new Source Capabilities message.
Additional Information	None.

3.3 Patch Bundle Update Tasks

The 4CC commands and sequence for loading a patch bundle in burst mode to the PD controller from an external host over I2C is listed below.

The following tasks are used for updating a Patch Bundle.

3.3.1 'PBMs' - Start Patch Burst Mode Download Sequence

Table 3-9. 'PBMs' - Start Patch Burst Download Sequence

Description	**The 'PBMs' Task starts the patch loading sequence. This Task initializes the firmware in preparation for a patch bundle load sequence and indicates what the patch bundle will contain.**			
INPUT DATAX	**Bit**	**Name**	**Description**	
	Byte 6: Burst Mode Timeout			
	7:6	Reserved		
	5:0	Timeout value	Timeout value for this task. A non-zero value must be used, it is recommended to always use 0x32 in this field (5 seconds) (LSB of 100 ms).	
	Byte 5: I2C slave for downloading patch.			
	7	Reserved		
	6:0	I2C Slave Address	The following slave addresses are not valid: • 0x00. • The I2C1s slave address of any port selected using the ADCINx pins. Refer to data-sheet.	
	Bytes 0-3: Low Region Binary bundle size in of bytes: [Byte4, Byte3, Byte2, Byte1]			
	39:32	Byte4 of bundle size		
	31:24	Byte3 of bundle size		
	23:16	Byte2 of bundle size		
	15:8	Byte1 of bundle size		
OUTPUT DATAX	**Bit**	**Name**	**Description**	
	7:0	PatchStartStatus	Status of the patch start.	
			0x00	Patch start success.
			0x04	Invalid bundle size.
			0x05	Invalid slave address.
			0x06	Invalid Timeout value.
Task Completion	The 'PBMs' Task completes after output has a valid PatchStartStatus. If MODE register (0x03) is equal to 'APP ', then this Task will be rejected.			
Side Effects	When the 'PBMs' is successful, the second slave address will be set to the input value.			
Additional Information	The host can only issue a 'PBMs' Task to the I2Cs port of the PD controller. If the host issues 'PMBs' a second time, then the PD controller ignores the DATAX input, restarts the burst-mode timer, and resets the pointer to the beginning of the patch space in RAM. If the MODE register is 'APP ' indicating that the PD controller is in the APP mode, then it will reject the 'PBMs' Task.			

3.3.2 'PBMc' - Patch Burst Mode Download Complete

Table 3-10. 'PBMc' - Patch Burst Download Complete

Description	**The 'PBMc' Task ends the patch loading sequence. Send this Task after all patch data has been transferred. This Task will initiate the CRC check on the binary patch data that has been transferred, and if the CRC is successful, the patch_init function contained within the patch will be executed.**			
INPUT DATAX	None			
OUTPUT DATAX	**Bit**	**Name**	**Description**	
	319:288	acCalculatedCRC	The CRC calculated in FW for the configuration data.	
	287:256	acTransferredCRC	The CRC transferred along with the configuration data.	
	255:240	Reserved	reads as 0.	
	239:224	acIndicatedDataSize	The indicated DataSize in the transferred configuration data.	
	223:216	acHeaderVersion	The indicated header version in the transferred configuration data.	
	215:208	acFailCode	An error code indicating why the app config data failed to apply, if it failed to apply.	
			0x00	AC_FAIL_NONE: No failure.
			0x01	AC_FAIL_WRONG_HEADER_VERSION: The header version is expected to be 1 and was not.
			0x02	AC_FAIL_TOO_MUCH_DATA: The DataSize field indicates that you are trying to load more configuration data that there is allocated SRAM for.
			0x03	AC_FAIL_CRC_CHECK_FAIL: The CRC comparison failed.
	207:200	acState	The current internal state of the AppConfig state machine.	
			0x00	AC_NODATA: No configuration data found yet, because we haven't started looking.
			0x01	AC_LOADING_DEFAULT: Attempting to load configuration data from a factory default.
			0x02	AC_LOADING_SRAM: Attempting to load configuration data from SRAM.
			0x03	AC_LOADING_FLASH: Attempting to load configuration data from Flash.
			0x04	AC_LOADING_I2C: Attempting to load configuration data from I2C.
			0x05	AC_LOADING_DONE: Done loading configuration data, we found valid data.
			0x06	AC_ERROR: A generic error state.
			0x07	AC_DONE_SUCCESS: Completely done with the app customization process and the records were applied successfully.
			0x08	AC_DONE_FAIL: Completely done with the app customization process and the records were not applied.
	199:192	configBundleGood	1 if the top-level state machine found a valid configuration bundle, otherwise 0.	
	191:160	rpRomVersionExpected	The romVersionExpected in the transferred bundle's patch header	
	159:144	rpBundleTotalSize	The bundleTotalSize in the transferred bundle's patch header.	
	143:128	rpBundleFlags	The bundleFlags in the transferred bundle's patch header.	
	127:96	rpPatchBodyCrc	The patchBodyCrc in the transferred bundle's patch header.	
	95:64	rpPatchHeaderCrc	The patchHeaderCrc in the transferred bundle's patch header.	

Table 3-10. 'PBMc' - Patch Burst Download Complete (continued)

Description	**The 'PBMc' Task ends the patch loading sequence. Send this Task after all patch data has been transferred. This Task will initiate the CRC check on the binary patch data that has been transferred, and if the CRC is successful, the patch_init function contained within the patch will be executed.**			
OUTPUT DATAX	**Bit**	**Name**	**Description**	
	55:48	rpBundleSignature	The bundleSignature in the transferred bundle's patch header.	
	47:40	rpState	The current internal state of the RomPatch state machine.	
			0x00	RP_NOPATCH: No patch has been loaded.
			0x01	RP_LOADING: In the process of loading patch data.
			0x02	RP_LOADINGDONE: All patch data has been received.
			0x03	RP_RUNNING: A patch has been loaded and is running. Can also indicate that a NULL patch is active.
			0x04	RP_EARLYLOAD_SKIPPED: Indicates that the early boot process does not need to wait for a patch over I2C.
			0x05	RP_UARTBOOTED: Checking for a patch in RAM.
			0x06	RP_ERROR: A generic error state.
	39:32	patchBundleGood	0x01 if the top-level state machine found a good ROM patch, otherwise 0x00.	
	31:24	AppConfigPatchCompleteStatus	0x00	
			0x40	Warning.
			0x80	Failure.
	23:16	DevicePatchCompleteStatus	A return code indicating whether the RomPatch state machine executed successfully.This value is always valid, and reflective of the internal state of the RomPatch mechanism, but must only be considered if the bundle transferred did in fact include patch data.	
			0x00	Success.
			0x20	Not ready.
			0x40	Not a patch.
			0x41	Patch header checksum mismatch.
			0x42	Patch not compatible with this version of ROM.
			0x43	Patch code checksum mismatch.
			0x44	Null patch received.
			0x45	Error patch received.
	15:8	cpReturn	Always returns success, there is no way for it to fail.	
	Byte 1: Return Code			
	7:4	rpReturnIndicator	The most significant nibble of the rpReturn value.	
			0x0	Success.
			0x2	Informational.
			0x4	Warning.
			0x8	Error.
	3:0	acReturnIndicator	The most significant nibble of the acReturn value.	
			0x0	Success.
			0x2	Informational.
			0x4	Warning.
			0x8	Error.
Task Completion	The 'PBMc' Task completes as output has a valid DevicePatchCompleteStatus and AppConfigPatchCompleteStatus. This Task is rejected if the DATAX input does not contain the total patch size. If MODE register (0x03) is equal to 'APP ', then this Task will be rejected.			
Side Effects	Before this Task completes it will change the I2C slave address from the patch address back to the normal value. Upon successful completion of this Task the PD controller will change the MODE register (0x03) to 'APP ' and move to the application mode.			

Table 3-10. 'PBMc' - Patch Burst Download Complete (continued)

Description	**The 'PBMc' Task ends the patch loading sequence. Send this Task after all patch data has been transferred. This Task will initiate the CRC check on the binary patch data that has been transferred, and if the CRC is successful, the patch_init function contained within the patch will be executed.**
Additional Information	When the CMDx register goes to 0 check the Output DATAX register for status. If the MODE register is 'APP ' indicating that the PD controller is in the APP mode, then it will reject the 'PBMc' Task.

3.3.3 'PBMe' - End Patch Burst Mode Download Sequence

Table 3-11. 'PBMe' - Patch Burst Mode Exit

Description	The 'PBMe' Task ends the patch loading sequence. This Task instructs the PD controller to complete the patch loading process.
INPUT DATAX	.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'PBMe' Task completes after it has ended the patch loading sequence. If MODE register (0x03) is equal to 'APP ', then this Task will be rejected.
Side Effects	When the 'PBMe' is successful, the second slave address will be restored to the value configured by the ADCINx pins. The PD controller leaves the MODE register (0x03) as 'PTCH' and will wait for the patching process to restart.
Additional Information	If the MODE register is 'APP ' indicating that the PD controller is in the APP mode, then it will reject the 'PBMe' Task.

3.3.4 Patch Burst Mode Example

Commands

- PBMs: The 'PBMs' Task starts the patch loading sequence. This task initializes the firmware in preparation for a patch bundle load sequence and indicates what the patch bundle contains.
- PBMc: The 'PBMc' Task ends the patch loading sequence. This command shall be executed after all patch data has been transferred. This task initiates the CRC check on the binary patch data that has been transferred, and if the CRC is successful, the patch_init function contained within the patch is executed. If this Task is sent prior to a 'PBMs' start Task, it indicates to the PD Controller that no patch is available and bypasses the patch process.
- PBMe: The 'PBMe' Task ends the patch loading sequence. This Task instructs the PD controller to complete the patch loading process.
- PTCr: This task resets the patch firmware to the no patch state.

1. The device generates an I2C interrupt, INT_EVENT.ReadyForPatch, indicating that it's ready for patch. The host shall start the patch download process only after receiving this notification from the device.

2. The host shall start the patch download process by sending a PBMs command to the device.

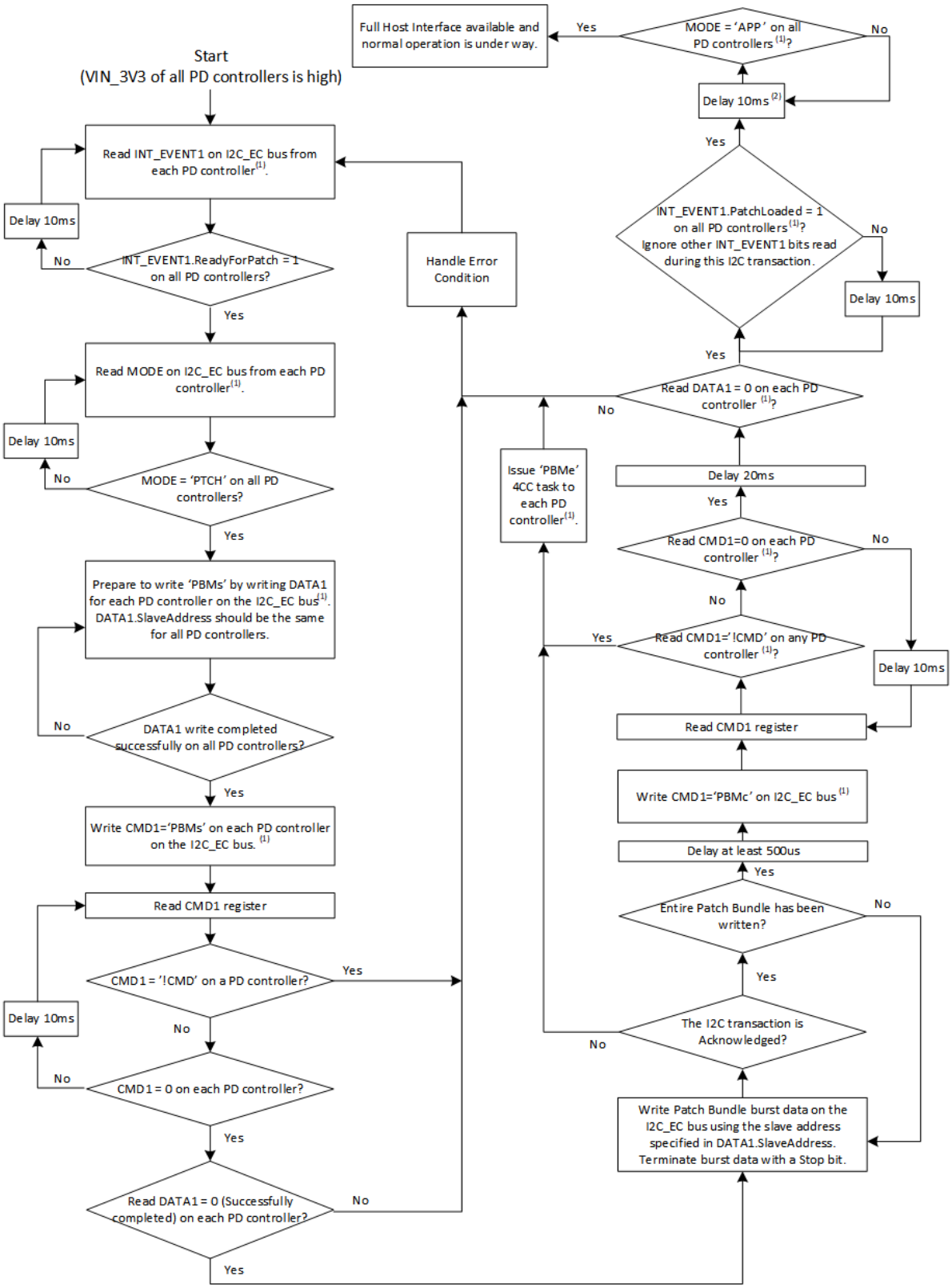
3. The host shall then transmit the entire patch-bundle data. The host may send the entire Patch Bundle in a single I2C transaction, or it may break it into multiple transactions. The PD controller increments the pointer into its patch memory space with each byte received on the Patch Slave address that was configured as part of the 'PBMs' 4CC Task.

4. The host shall then send the PBMc command to indicate the PD Controller that the patch load sequence is complete. After successful boot sequence, the PD Controller begins application execution.

To execute a 4CC command, the host application shall follow the below sequence:

- If the 4CC command requires an input, the application shall first write the input data into the DataX (0x09 or 0x11) register.
- The application shall then write the 4CC command characters into the corresponding CmdX (0x08 or 0x10) register.
- The application can register and waits for CmdXComplete event or repeatedly read the four byte content of CmdX register until it reads
  - 0x00' indicating that the command is successfully executed.
  - or, '!CMD' indicating that the command's execution has failed.
- If the command is successfully executed, the application shall proceed to read the 'n' byte content of the DataX register which will contain the output.

Execution Flow



(1) Use the fundamental I2C slave address of each PD controller.
(2) This delay before reading the MODE register, is optional but recommended.

Example Code

1.

```
void AsyncEvtHandlerP11()
{
s_AppContext *const pCtx = &gAppCtx;
I2C_IF_TPS6598xIntDisable(pCtx->i2cPort);
ProcessEvent();
}
static int32_t ProcessEvent()
{
s_AppContext *const pCtx = &gAppCtx;
s_AppData *const pData = &gAppData[pCtx->i2cPort];
uint8_t outdata[MAX_BUF_BSIZE] = {0};
uint8_t indata[MAX_BUF_BSIZE] = {0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF};
s_TPS_IntEvent *pSetEvent = NULL;
s_TPS_IntEvent *pClrEvent = NULL;
int32_t retVal = -1;
#ifdef WORKAROUND_WAIT_AFTER_GAID
if(0 == wait_after_gaid)
{
wait_after_gaid += 1;
/*
* The 100ms delay here is to allow the device to ACK the I2C transactions
* after a cold start
*/
Board_IF_Delay(100000);
}
#endif /* WORKAROUND_WAIT_AFTER_GAID */
retVal = ReadReg(pData->event_reg, REG_LEN_INTEVENT1, &outdata[0]);
RETURN_ON_ERROR(retVal);
/*
* output[0] = Register's size in bytes
* output[1..REG_LEN_INTEVENT1] = Register's content, and hence '&outdata[1]' below.!
*/
pSetEvent = (s_TPS_IntEvent *)&outdata[1];
pClrEvent = (s_TPS_IntEvent *)&indata[0];
if(0 != pSetEvent->cmd1complete)
```

```
{
pData->cmdExeInProgress = 0;
pClrEvent->cmd1complete = 1;
}
if(0 != pSetEvent->readyforpatch)
{
pClrEvent->readyforpatch = 1;
SignalEvent(APP_EVENT_READY_FOR_PATCH);
}
if(0 != pSetEvent->patchloaded)
{
pClrEvent->patchloaded = 1;
SignalEvent(APP_EVENT_PATCH_LOADED);
}
/* App events are triggred - The corresponding events of the device can now be cleared */ retVal =
WriteReg(pData->clear_reg, REG_LEN_INTCLEAR1, &indata[0]); RETURN_ON_ERROR(retVal);
return 0;
}
```

2.

```
static int32_t StartPatchDownloadBurstMode()
{
uint8_t outdata[MAX_BUF_BSIZE] = {0};
s_PBMs_InData pbmsInData = {0};
s_PBMs_OutData *p_pbmsOutData = NULL;
int32_t retVal = -1;
UART_PRINT("Starting Patch Download in Burst Mode\n\r");
retVal = ReadMode();
RETURN_ON_ERROR(retVal);
retVal = ReadVersion(); RETURN_ON_ERROR(retVal); /* If the device is in APP mode, don't allow burst-mode
patch download */
if(isDeviceInAppMode())
{
UART_PRINT("Burst mode patching not allowed in APP mode\n\r");
retVal = -1;
goto error;
} /* Wait for CmdComplete event when executing the 4CC commands */
```

```
retVal = MaskCmdComplete(1);
RETURN_ON_ERROR(retVal); /* Fill the input, and execute 'PBMs' */
pbmsInData.lrBinSize = gSizeLowregionArray;
pbmsInData.slaveAddress = SLAVE_ADDR_FOR_PATCH_DATA;
pbmsInData.timeout = BURST_MODE_TIMEOUT;
retVal = ExecCmd(PBMs, sizeof(pbmsInData), (uint8_t *)&pbmsInData, \ PBMs_OUTPUT_SIZE, &outdata[0]);
RETURN_ON_ERROR(retVal); /* 'outdata' will contain the command's output after the successful execution of PBMs. */
p_pbmsOutData = (s_PBMs_OutData *)&outdata[1];
if(PBMs_SUCCESS != p_pbmsOutData->patch_start_status)
{
ERR_PRINT(p_pbmsOutData->patch_start_status);
retVal = -1; goto error;
}
error: return retVal;
}
```

3.

Low region binary 'C' array format can be generated using the Application Customization GUI. tps6598x_lowregion_array and gSizeLowregionArray referenced in the below snippet are taken from this generated file.

```
static int32_t DownloadDataBurstMode()
{
int32_t retVal = 0;
#ifdef WORKAROUND_IGNORE_CMDxCMPLT_BURST_TX
/*
* Workaround: Device was erroneously sending an unexpected CmdXComplete
* event occationally while in the middle of the patch download process,
* causing the SM to go for a toss.!
*/
MaskCmdComplete(0);
#endif /* WORKAROUND_IGNORE_CMDxCMPLT_BURST_TX */
/*
* tps6598x_lowregion_array and gSizeLowregionArray are defined
* in the 'C' array file that can be created using the
* Application customization GUI
*/
```

```
UART_PRINT(".");
SendDataToSlave(SLAVE_ADDR_FOR_PATCH_DATA, gSizeLowregionArray, (uint8_t
*)&tps6598x_lowregion_array[0]);
/* Proceed to the next step after the patch is successfully downloaded */
UART_PRINT("\n\r");
SignalEvent(APP_EVENT_PATCH_DOWNLOADED);
return retVal;
}
```

4.

```
static int32_t PatchDownloadCompleteBurstMode()
{
s_AppContext *const pCtx = &gAppCtx;
/* PBMc return structure and statuses are exactly same as PTCc */
s_PTCc_OutData *p_pbmcOutData = NULL;
uint8_t outdata[MAX_BUF_BSIZE] = {0};
int32_t retVal = -1;
UART_PRINT("Patch Download Complete\n\r");
#ifdef WORKAROUND_IGNORE_CMDxCMPLT_BURST_TX
MaskCmdComplete(1);
#endif /* WORKAROUND_IGNORE_CMDxCMPLT_BURST_TX */
/* Execute PBMc to indicate to the device that the patch is successfully downloaded */
retVal = ExecCmd(PBMc, 0, NULL, PTCc_OUTPUT_SIZE, &outdata[0]);
RETURN_ON_ERROR(retVal);
/* 'outdata’ will contain the command's output after the successful execution of PBMc. */
p_pbmcOutData = (s_PTCc_OutData *)&outdata[3];
if(p_pbmcOutData->patch_complete_status != PTCc_SUCCESS)
{
ERR_PRINT(p_pbmcOutData->patch_complete_status);
retVal = -1;
goto error;
}
/* Expecting 'PatchLoaded' event */
retVal = I2C_IF_TPS6598xIntEnable(pCtx->i2cPort);
RETURN_ON_ERROR(retVal);
error: return retVal;
}
```

3.3.5 'GO2P' - Go to Patch Mode

Table 3-12. 'GO2P' - Forces PD Controller to Return to 'PTCH' Mode and Wait for Patch Over I2C

Description	The 'GO2P' Task causes the PD controller to re-enter the patch mode (MODE = 'PTCH').
INPUT DATAX	None.
OUTPUT DATAX	Byte 1: Standard Task Return Code. See also Table 3-1.
Task Completion	The 'GO2P' Task completes after the PD controller has re-entered the patch mode. • If the PD controller has re-entered the patch mode and the MODE register reads as 'PTCH'. The 'GO2P' Task is considered rejected if: • The PD controller did not enter the 'APP ' mode without receiving a patch over I2C. • BOOT_STATUS.PatchConfigSource does not read as 3h or 4h.
Side Effects	When the 'GO2P' Task is successful, the MODE register will read as 'PTCH' and the USB PD PHY will be disabled. The PD Controller may temporarily NAK I2C transactions. The host must wait for the IRQ signal to assert (because INT_EVENT1.ReadyForPatch is asserted), and then push the patch as soon as possible.
Additional Information	The 'GO2P' Task must only be used when the ADCINx configuration option NegotiateHighVoltage is used.

3.4 System Tasks

3.4.1 'DBfg' - Clear Dead Battery Flag

Table 3-13. 'DBfg' - Clear Dead Battery Flag

Description	The 'DBfg' Task is used to clear the dead battery flag. This Task does not disable the PP_EXT input switch that may have been enabled during dead battery operation.
INPUT DATAX	None.
OUTPUT DATAX	None.
Task Completion	The 'DBfg' Task completes after the effects of clearing the Dead Battery Flag are complete.
Side Effects	The Dead Battery Flag causes the PD Controller to take specific actions, so clearing this flag will have side effects. PD Controller 's power input is forced to VBUS until the Dead Battery Flag is cleared, so executing this Task will change PD Controller 's power input.
Additional Information	None.

There are several limitations placed on the PD controller while the Dead-Battery Flag is asserted (PowerPathStatus.PowerSource = 10b).

- A Hard Reset will not be transmitted while in the sink role (on either port).
- VBUS is selected as the main supply for the PD controller, even if the 3.3 V input is present.
- The PD controller will reject PR_Swap requests to become source (on either port).
- The 2nd port in the PD controller that is unconnected will only offer the USB Type-C Default Rp (PortControl.TypeCCurrent is ignored) if it connects as a source.
- A port connected to a source will only act as a Type-C sink regardless of the configuration.
- If no Source Capabilities message is received after the boot process is complete (Status.ActingAsLegacy=11b), the PD controller will not send a Hard Reset until the Dead-Battery Flag is cleared even if the SinkWaitCapTimer expires.

3.4.2 'I2Cr' - I2C Read Transaction

Table 3-14. 'I2Cr' - Executes I2C Read Transaction on I2Cm

Description	**The 'I2Cr' task may be used to cause the PD controller to read from a specified slave address and register offset using a I²C read transaction via the I2Cm_SDA and I2Cm_SCL pins.**		
INPUT DATA	**Bit**	**Name**	**Description**
	Byte 3: Number of bytes to read from the slave.		
	7:0	NumBytes.	
	Byte 2: Register offset to use in the I2C read transaction.		
	7:0	RegisterOffset.	
	Byte 1: Slave Address.		
	7	Reserved.	
	6:0	Slave to use for the transaction.	
OUTPUT DATA	**Bit**	**Name**	**Description**
	Bytes 2-65: Data Bytes read from the slave (in order received).		
	511:0	Data.	
	Byte 1: Standard Task Return Code. See also Table 3-1.		
Task Completion	The PD controller completes after it has succesfully read the specified number of bytes, or the I²C transaction terminated for some other reason.		
Side Effects	This task will cause the PD controller to issue a command on the I2Cm port. It can result in INT_EVENT. I2CMasterNACKed being asserted.		
Additional Information	None.		

3.4.3 'I2Cw' - I2C Write Transaction

Table 3-15. 'I2Cw' - Executes I2C Write Transaction on I2Cm

Description	**The 'I2Cw' task may be used to cause the PD controller to write a particular I²C transaction using I2Cm_SDA and I2Cm_SCL.**		
INPUT DATA	**Bit**	**Name**	**Description**
	Bytes 5-14: Payload for the I2C transaction.		
	Byte 4: Register Offset for the I2C transaction.		
	7:0	Register offset.	
	Bytes 2-3: Length.		
	15:8	Reserved.	
	7:0	Number of bytes in the transaction payload.	
	Byte 1: Slave Address.		
	7	Reserved.	
	6:0	Slave to use for the transaction.	
OUTPUT DATA	Byte 1: Standard Task Return Code. See also Table 3-1.		
Task Completion	The PD controller maintains a queue of transactions to send on the I2Cm port. If the PD controller has been configured to send transactions upon certain events, it is possible there is a transaction in the queue when the 'I2Cw' task is received. In that case the task will complete successfully after the transaction is inserted into the queue. If the PD controller fails to insert the task into the queue for any reason, the task is rejected. Therefore, when this task is completed successfully it does not ensure that the I2C transaction is complete. If possible, the host must use the 'I2Cr' 4CC task to confirm the write was successful.		
Side Effects	When successful, this task will cause the PD controller to issue a command on the I2Cm port. This can result in INT_EVENT.I2CMasterNACKed being asserted.		
Additional Information	If the DATA register is written with more than 14 bytes, all bytes beyond byte 14 are ignored. The PD controller has a limit on the maximum length of the I²C write transaction.		

4.1 PD Controller Application Customization

The PD Controller application binary may be pushed over I2C using the I2Cs port, or the PD controller can read it from an external EEPROM at slave address 0x50 on the I2Cm port. The PD Controller application binary provides a way to customize and initialize the settings of the PD Controller. It allows for any register bit accessible via the Host Interface to be changed *before* the PD Controller application starts normal operation, to configure system-related settings that must be correct before any application decision is made. TI provides a GUI tool to create the PD Controller application binary.

4.2 Loading a Patch Bundle

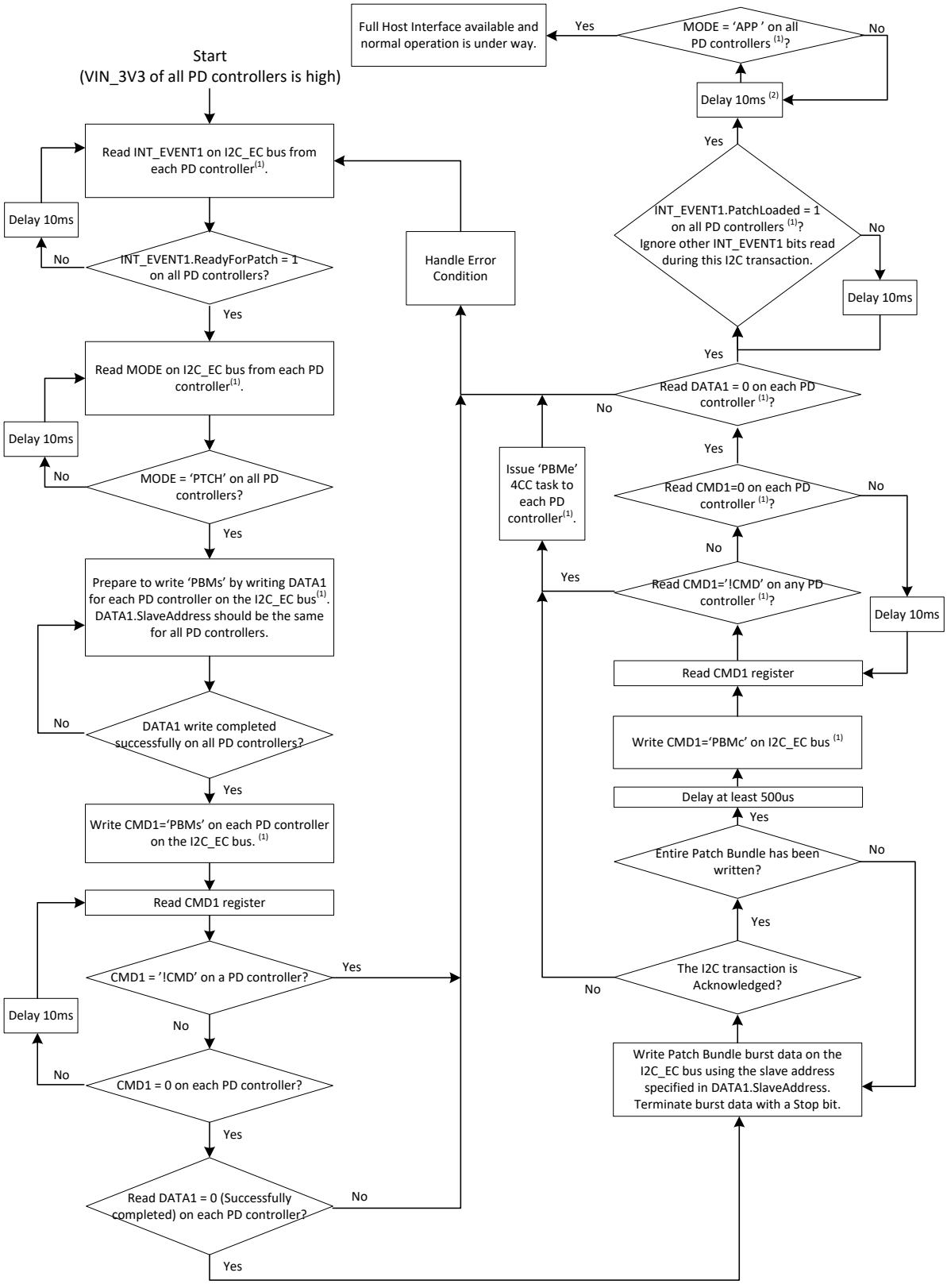
The patch bundle may contain Application Customization data and a Patch binary that modifies the default application firmware in the PD controller. This section describes how the host can load the patch bundle. The host uses the I2Cs bus for all transactions related to loading the patch bundle. As noted in the flow diagram below, the I2C slave address varies depending upon which mode the PD controller is in. The Patch Burst Mode allows the host to push the Patch Bundle to multiple PD controllers simultaneously.

The following flow diagram illustrates the normal successful patch loading process. Other error handling steps may be necessary depending upon the nature of the errors encountered for a particular system. The EC may reset and restart the patch process by issuing a 'PBMe' 4CC Task.

Table 4-1. Usage of Slave Addresses During Different Modes of Operation.

MODE register read-back value	I2Cs
	Slave Address #1
'BOOT'	As configured by ADCINx pins for Port A. This is the "Fundamental" I2C slave address.
'PTCH'(1)	
'APP '(2)	

(1) A successful 'PBMs' Task puts the PD controller into the 'PTCH' mode.
(2) A successful 'PBMc' Task puts the PD controller into the 'APP ' mode.



(1) Use the fundamental I2C slave address of each PD controller.
(2) This delay before reading the MODE register, is optional but recommended.

Figure 4-1. Flow for Pushing a Patch Bundle Over the I2Cs Bus to Multiple PD Controllers at the Same Time

While the host is writing the Patch Bundle burst data, the I2C protocol in the following figure must be followed. The host may send the entire Patch Bundle in a single I2C transaction, or it may break it up into multiple transactions. The PD controller increments the pointer into its patch memory space with each byte received on the Patch Slave address that was configured by DATA1.SlaveAddress as part of the 'PBMs' 4CC Task. The EC may re-issue a 'PBMs' 4CC Task or it may issue a 'PBMe' 4CC Task to reset the pointer.

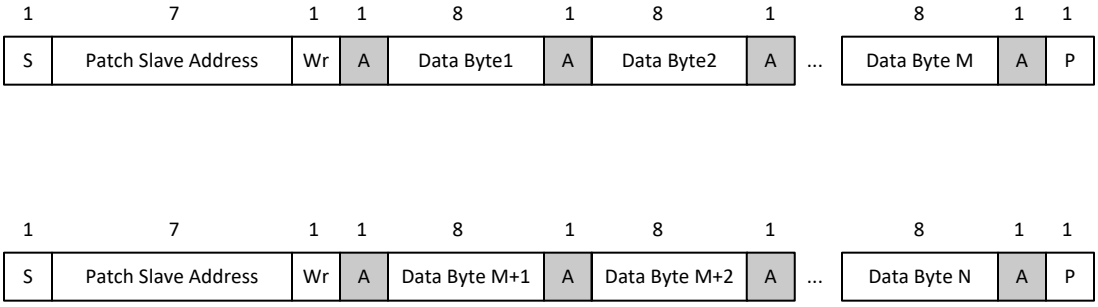


Figure 4-2. Protocol of Patch Bundle Burst Data Assuming it is Broken into Two Transactions

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision * (November 2020) to Revision A (July 2022)	Page
• Added *Preface* section	7
• Added PBMx method and example	9

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265