TPS65981, TPS65982, and TPS65986 Firmware User's Guide

User's Guide

Literature Number: SLVUAH7B
June 2015–Revised July 2016

Contents

1 **Introduction** 5
- 1.1 Purpose and Scope 5
- 1.2 Related Documents 5

2 **Overview** 6
- 2.1 TPS65982 versus TPS65986 6
- 2.2 TPS65982 versus TPS65981 6

3 **Boot Code** 7
- 3.1 Boot Code 7
- 3.2 Initialization 8
- 3.3 I^2C Configuration 8
- 3.4 Dead Battery 9
- 3.5 Application Code 10
- 3.6 Flash Memory Read 11
- 3.7 Invalid Flash Memory 12
- 3.8 UART Download 13

4 **Application Code** 15
- 4.1 Overview 15
- 4.2 Application Code Boot Header 15
- 4.3 I^2C Host Interface 15
- 4.4 Updating Application Code 15
  - 4.4.1 Application Code Update through I^2C 16
  - 4.4.2 Application Code Update through External Device 17

5 **Power Management** 18
- 5.1 Power States 18
- 5.2 Activity Timer 19
- 5.3 System Power State 19
- 5.4 Power State Descriptions 19
  - 5.4.1 POWER OFF 19
  - 5.4.2 RESET 19
  - 5.4.3 SLEEP 19
  - 5.4.4 IDLE 20
  - 5.4.5 ACTIVE 21
  - 5.4.6 Dead Battery 21

6 **USB Type-C** 23
- 6.1 Overview 23
- 6.2 USB Type-C Port Configuration 23
  - 6.2.1 Source 23
  - 6.2.2 Sink 23
- 6.3 CC Detection 24
- 6.4 USB Type-C Connection State Machine 24

7 **Accessory Modes** 25
- 7.1 Audio Accessory Mode 25
- 7.2 Debug Accessory Mode 25

8 **Type-C Port Multiplexer Configurations** 26

9 USB Power Delivery 27
9.1 Overview 27
9.2 Protocol Layer 27
9.2.1 Control Messages 27
9.2.2 Data Messages 28
9.2.3 Reset 28
9.3 Policy Engine 28
9.3.1 Request Message 29

10 Alternate Modes 30
10.1 Overview 30
10.2 USB Billboard 30
10.3 Automatic Entry 30
10.4 DisplayPort Alternate Mode 30
10.5 PDIO Alternate Mode 31
10.5.1 Overview 31
10.5.2 PDIO GPIO Events 32
10.5.3 PDIO Signature 32
10.6 User Alternate Mode 32
10.6.1 Enter the Alternate Mode 32
10.6.2 Send Unstructured Vendor-Defined Message 33
10.6.3 Reconfigure the TPS65982 from Data Set 33
10.6.4 Execute Host-Interface Commands 33

11 Power Delivery Fault Handling 34

12 Charger Detection 35
12.1 Firmware Description 35
12.1.1 VBUS Detect 35
12.1.2 Data-Contact Detect 36
12.1.3 Primary Detection 36
12.1.4 Secondary Detection 37

13 Device Features 38
13.1 ADC 38
13.2 Digital I/O 38
13.3 Load App Config Set GPIO Events 41

Revision History 42

List of Figures

3-1. Boot-Code Sequence 8
3-2. I²C Address Configuration 9
3-3. Dead-Battery Process 10
3-4. TPS65982 Flash-Memory Organization 11
3-5. Flash Read Flow 12
3-6. Memory Invalid Flow 13
3-7. UART Download Process 14
5-1. Power State Diagram 18
10-1. DisplayPort Sink-Side Hardware Flow (HPD RX) 31
10-2. DisplayPort Sink-Side Firmware Flow (HPD RX) 31
10-3. PDIO Alternate Mode 31
12-1. Charger-Detection State Machine 35
12-2. Data-Contact Detect (DCD) 36
12-3. Primary Detection 37
12-4. Secondary Detection 37

List of Tables

4-1. Application Code Structure 15
5-1. Power States Summary 18
6-1. USB Type-C Port State Based on CC Terminations (Source Perspective) 24
6-2. USB Type-C Connection States Supported 24
8-1. USB Type-C Port Multiplexer Configurations 26
9-1. Control Messages 27
9-2. USB PD Message Sequences Supported by the TPS65982 28
10-1. PDIO Signature 32
11-1. Power Delivery Fault Conditions 34
13-1. GPIO Events 39

Introduction

1.1 Purpose and Scope

This document is the Firmware User's Guide for the TPS65981, TPS65982, and TPS65986 USB Type-C and USB Power Delivery (PD) controller, power switch, and high-speed multiplexer. The firmware for the TPS65981, TPS65982, and TPS65986 devices controls the port state, negotiates port-power levels, and allows for the entry and configuration of various alternate modes. These behaviors can be configured using the TPS6598x Configuration Tool.

This document intends to complement the standard specifications. Texas Instruments recommends to use the user's guide in conjunction with those standard specifications. If a conflict exists between this user's guide and any of the standard specifications, the standard specifications are to be referenced.

Similarly, despite selective inclusions from the TPS65982 device specification for the same reasons mentioned previously, the detailed description of the hardware features of TPS65982 device is beyond the scope of this user's guide and the TPS65982 data sheet, *TPS65982 USB Type-C and USB PD Controller, Power Switch, and High Speed Multiplexer*, must be referenced.

1.2 Related Documents

- *TPS65982 USB Type-C and USB PD Controller, Power Switch, and High Speed Multiplexer*
- *TPS65986 USB Type-C and USB PD Controller and Power Switch*
- *TPS65981 USB Type-C and USB PD Controller, Power Switch, and High Speed Multiplexer*
- TPS65981, TPS65982, and TPS65986 Host Interface Technical Reference Manual
- *Battery Charging Specification, Revision 1.2*, December 7, 2010 plus Errata.
- *DisplayPort Alt Mode Plug Requirement Corrections and Protocol Clarifications*
- *Universal Serial Bus 3.1 Specification, Revision 1.0*, July 26, 2013 plus ECN and Errata. www.usb.org/developers/docs
- *Universal Serial Bus Power Delivery Specification, Revision 2.0*, V1.1, May 7, 2015. www.usb.org/developers/docs
- *Universal Serial Bus Specification, Revision 2.0*, April 27, 2000 plus ECN and Errata. http://www.usb.org/developers/docs/usb20_docs/
- Universal Serial Bus Type-C Cable and Connector Specification, Revision 1.1, April 3, 2015. www.usb.org/developers/docs
- *VESA DisplayPort (DP) Standard, Version 1.3*, September 17, 2014.
- *VESA DisplayPort Alt Mode on USB Type-C Standard, Version 1.0*, September 22, 2014.

Overview

The TPS65982 firmware is responsible for controlling the various analog and digital components of the TPS65982 device. The TPS65982 firmware is divided into two sections: boot code and application code. The boot code is responsible for configuration of the device immediately after power application. The boot code is stored on internal device memory and cannot be altered. The TPS65982 application code is stored externally, and is loaded by the boot code. When the application code is loaded, this section of firmware is responsible for implementing the various required functionality for a USB Type-C device.

The TPS65982 device is the most integrated of the three devices (TPS65981, TPS65982, and TPS65986) that share the same core firmware. Because of the fact that the TPS65982 hardware contains all features that can be controlled by the firmware, the behavior of the firmware is commonly described by referring to the TPS65982 device only although it may also apply to the TPS65981, TPS65986, or both device.

2.1 TPS65982 versus TPS65986

Both the TPS65982 and TPS65986 devices are capable of controlling a USB Type-C port; however, the TPS65986 device does not contain the proper hardware for controlling an external power path or operating the Thunderbolt alternate mode. As such, any firmware behaviors or controls related to these features are not available on firmware offered for the TPS65986 device. The TPS65986 templates in the TPS6598x Configuration Tool accurately reflect the configurable parameters available for this device. For details on the pin-out of the TPS65986 device, refer to the TPS65986 data sheet, *TPS65986 USB Type-C and USB PD Controller and Power Switch*.

2.2 TPS65982 versus TPS65981

Both the TPS65982 and TPS65981 devices are capable of controlling a USB Type-C port; however, the TPS65981 device does not contain the proper hardware for operating the Thunderbolt alternate mode or allow two or more devices to share a single SPI flash IC. In addition, the TPS65981 device has four less GPIO pins and does not have an I2C_ADDR pin for setting the three least significant bits of the I^2C slave address. As such, any firmware behaviors or controls related to these features are not available on firmware offered for the TPS65981 device. The TPS65981 templates in the TPS6598x Configuration Tool accurately reflect the configurable parameters available for this device. For details on the pin-out of the TPS65981, refer to the TPS65981 Data sheet, *TPS65981 USB Type-C and USB PD Controller, Power Switch, and High Speed Multiplexer*.

Boot Code

3.1 Boot Code

When power is applied to TPS65982 device through VIN_3V3 or VBUS, LDO_3V3 is enabled and a power-on-reset (POR) signal is issued. The digital core receives this reset signal and in response loads and begins executing the boot code.

Figure 3-1 shows the TPS65982 boot-code sequence.

The TPS65982 boot code is loaded from internal memory on POR, and begins initializing TPS65982 settings. This initialization includes enabling and resetting internal registers, loading initial values, and configuring the device I^2C addresses.

The unique I^2C address is based on the DEBUG_CTLX pins, and resistor configuration on the I2C_ADDR pin.

When the initial device configuration is complete, the boot code determines if the TPS65982 device is booting under dead-battery condition (VIN_3V3 invalid, VBUS valid). If the boot code determines that the TPS65982 device is booting under dead-battery condition, the BUSPOWERZ pin is sampled to determine the appropriate path for routing VBUS power to the system. The dead-battery flag is set and the TPS65982 device continues through the boot flow and loads application code from attached flash memory.

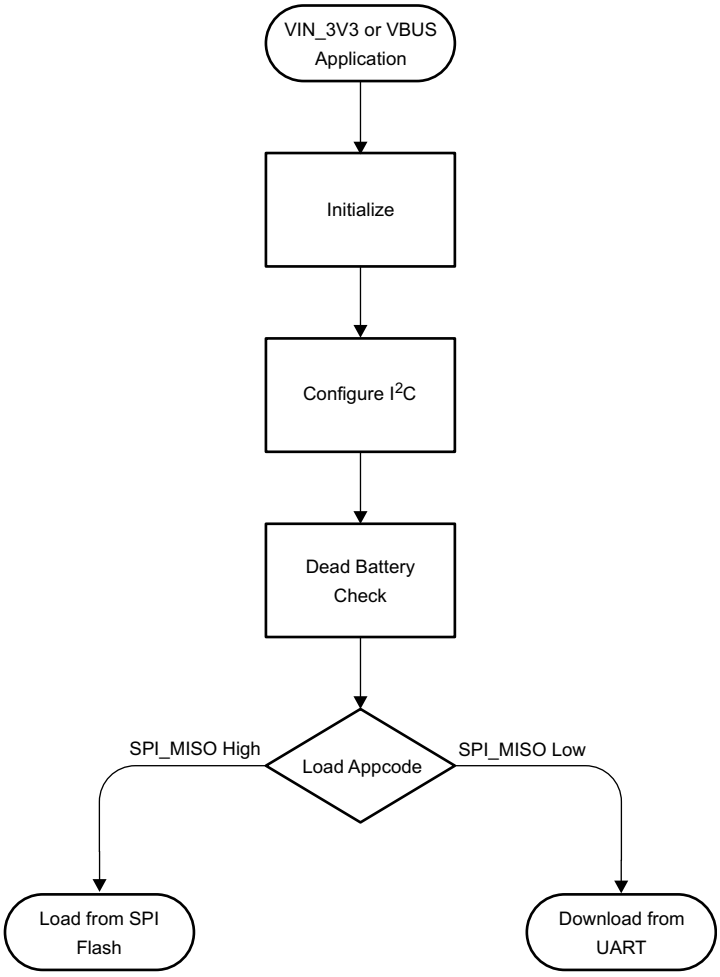

Figure 3-1. Boot-Code Sequence

3.2 Initialization

During initialization, the TPS65982 device enables the device internal hardware and loads the default configurations. The 48-MHz clock is enabled and the TPS65982 persistence counters begin monitoring the VBUS and VIN_3V3. These counters ensure the supply powering the TPS65982 device is stable before continuing the initialization process. The initialization concludes by enabling the thermal monitoring blocks and thermal-shutdown protection, along with the ADC, CRC, GPIO, and NVIC blocks.

3.3 I²C Configuration

The TPS65982 device features dual I²C busses each with a configurable address. The I²C addresses are determined according to the flow shown in Figure 3-2. The address is configured by reading device GPIO states at boot (see the TPS65982 data sheet for hardware details). When the I²C addresses are established, the TPS65982 device enables a limited host interface to allow for communication with the device during the boot process.

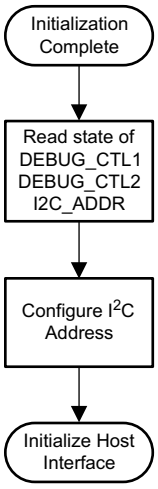

Figure 3-2. I²C Address Configuration

3.4 Dead Battery

After I²C configuration concludes, the TPS65982 device checks VIN_3V3 to determine the cause of device boot. If the device is booting from a source other than VIN_3V3, the dead-battery flow is followed to allow for the rest of the system to receive power. The state of the BUSPOWERZ pin is read to determine power path configuration for dead-battery operation. After the power path is configured, the TPS65982 device continues through the boot process. Figure 3-3 shows the full dead-battery process.

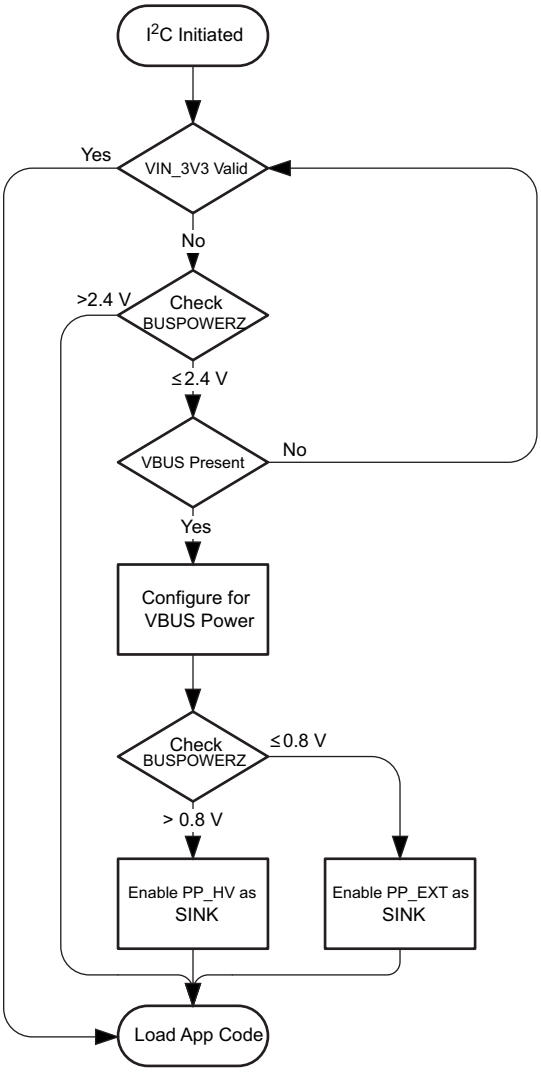

Figure 3-3. Dead-Battery Process

3.5 Application Code

The TPS65982 application code is stored in an external flash memory. The flash memory used for storing the TPS65982 application code can be shared with other devices in the system. The flash memory organization shown in Figure 3-4 supports the sharing of the flash as well as the TPS65982 device using the flash alone.

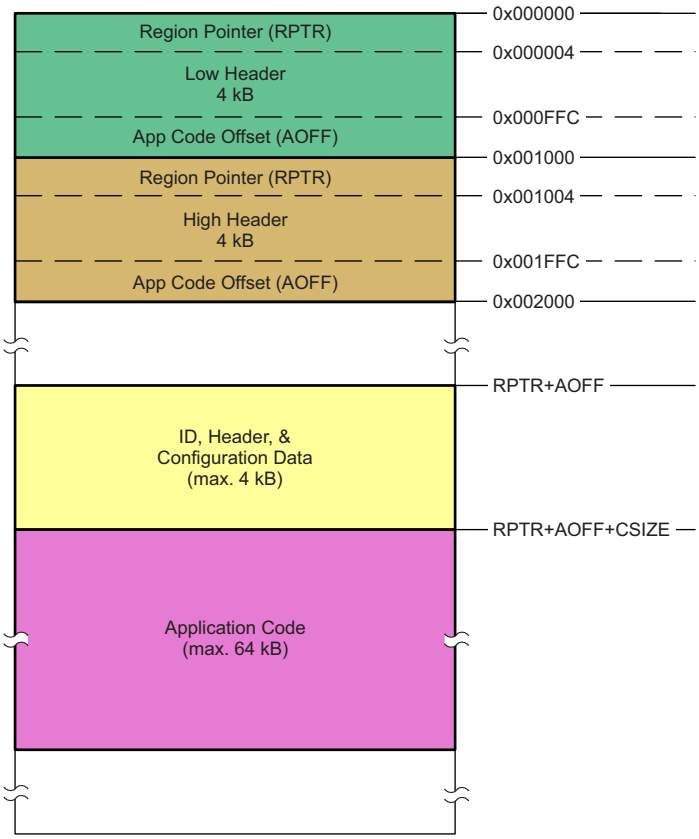

Figure 3-4. TPS65982 Flash-Memory Organization

The flash is divided into two separate regions: the low region and the high region. The size of this region is flexible and only depends on the size of the flash memory used. The two regions are used to allow updating the application code in the memory without overwriting the previous code. This ensures that the new updated code is valid before switching to the new code. For example, if a power loss occurred while writing new code, the original code is still in place and used at the next boot.

Figure 3-4 shows two 4kB header blocks starting at address 0x000000h. The low-header 4kB block is at address 0x000000h and the High-header 4kB block is at 0x001000h. Each header contains a region pointer (RPTR) that holds the address of the physical location in memory where the low-region application code resides. Each also contains an application-code offset (AOFF) that contains the physical offset inside the region where the TPS65982 application code resides. The TPS65982 firmware physical location in memory is RPTR + AOFF. The first sections of the TPS65982 application code contain device configuration settings where CSIZE is a maximum of 4kB. This configuration determines the default behavior of the device after power-up and can be customized using the TPS65982 Configuration Tool.

These pointers may be valid or invalid. The flash read flow (see Figure 3-5) handles reading and determining whether a region is valid and contains good application code.

3.6 Flash Memory Read

The TPS65982 device first attempts to load application code from the low region of the attached flash memory. If any part of the read process yields invalid data, the TPS65982 device aborts the low-region read and attempts to read from the high region. If both regions contain invalid data the device carries out the invalid memory flow (see Figure 3-6).

Figure 3-5 shows the flash memory read flow.

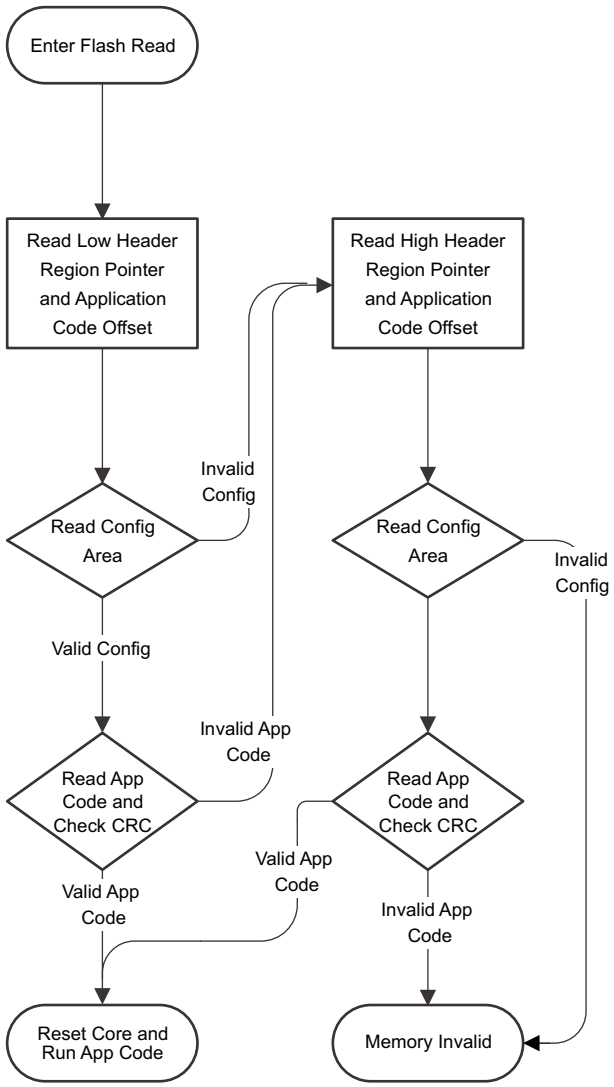

Figure 3-5. Flash Read Flow

3.7 Invalid Flash Memory

If the flash memory read fails because of invalid data, the TPS65982 device carries out the memory invalid flow and presents the SWD interface on the USB Type-C SBU.

Figure 3-6 shows the invalid memory process.

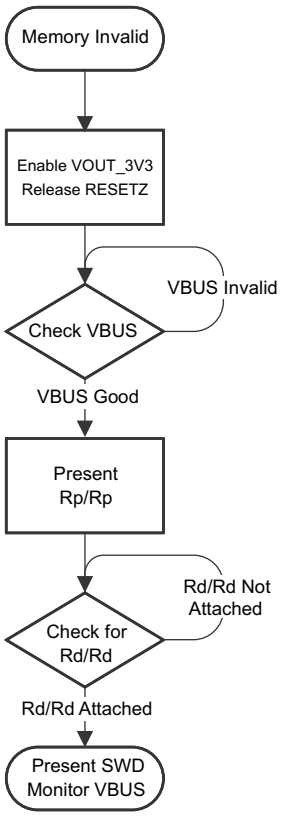

Figure 3-6. Memory Invalid Flow

3.8 UART Download

The TPS65982 device allows for multiple TPS65982 devices to be chained and configured from the same flash. In these applications the secondary TPS65982 downloads the required application code from the primary TPS65982 through UART. The download process is initiated when a secondary TPS65982 device sends a *Request Data* packet. The primary TPS65982 device responds to this request with a *Send Data* packet containing the first requested data block. This process continues until the secondary TPS65982 device sends a *Request Data* packet containing a completed block map. In response to the completed block map, the primary TPS65982 device sends a *Send CRC* packet containing the CRC for the application code along with the size of the application code. The secondary TPS65982 device then concludes by using the *Send CRC* packet to validate the downloaded code. Figure 3-7 shows this download process.

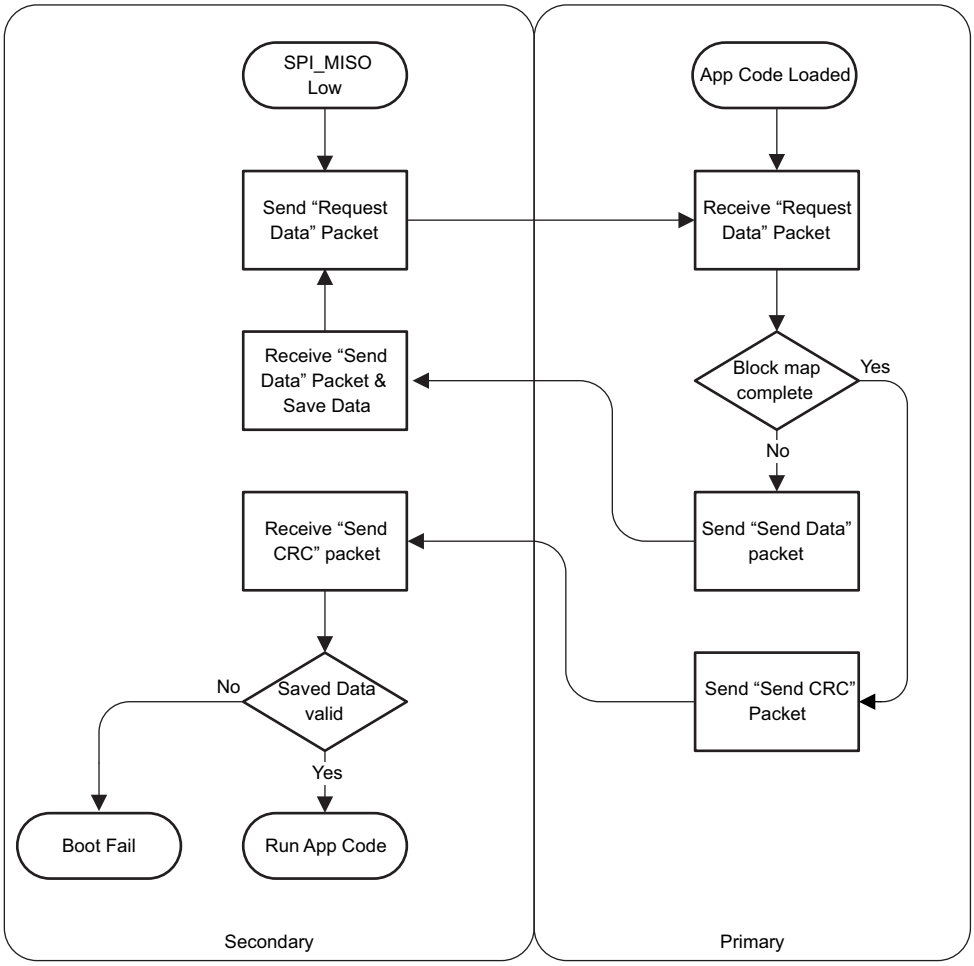

Figure 3-7. UART Download Process

Currently the TPS65982 firmware only supports 2 device (1 primary + 1 secondary) systems.

Application Code

4.1 Overview

The TPS65982 application code determines device configuration and behavior once Boot Code is complete. The TPS65982 application code is responsible for implementing the following device features:

- I²C host interface
- Power management states
- USB Type-C detection
- USB PD protocol layer and policy engine
- USB PD alternate modes
- Charger detection
- High-speed mux configuration

4.2 Application Code Boot Header

The first 4kB of the TPS65982 application code contains the application code boot header. This section includes information on the code size and device configuration, as well as the CRC for verifying valid application code. The application code binary follows the boot header. Table 4-1 lists the contents of the TPS65982 application-code boot header.

Table 4-1. Application Code Structure

Byte Address	Description	Size (Bytes)	Data
0x00000000	Device ID	4	0xACE00001
0x00000004	Reserved	4	0xFFFFFFFE
0x00000008	Boot config size	4	0x00001000
0x0000000C	TPS65982 binary size	4	(varies)
0x00000010	TPS65982 binary CRC	4	(varies)
0x00000014	Reserved	4	0x00000000
0x00000018	Reserved	40	0xFFFFFFFF
0x00000040	Device config pointer	4	(varies)
0x00000044	Reserved	4028	0xFFFFFFFF
0x00001000	TPS65982 binary		

4.3 I²C Host Interface

The TPS65982 host interface provides a method for external devices to communicate with the TPS65982 device. The host interface includes methods for reading and updating device configuration as well as commands for initiating various device functions. The full host interface is documented in the *TPS65981, TPS65982, and TPS65986 Host Interface Technical Reference Manual*.

4.4 Updating Application Code

The TPS65982 device only reads from an attached SPI flash device during boot. This process allows the flash to be shared with other devices in the system and allows external devices to write and update the flash memory. In systems where the flash is not shared, the TPS65982 can be used to update the flash memory through the host interface.

The dual-region memory structure allows for two approaches to application code storage:

1. Store firmware in only one half of the memory space. The header with the valid application code has correct pointers and the header with the invalid application code has pointers set to either 0x000000h or 0xFFFFFFh. When the application code is updated, the new application code is written to the unused half of the memory space. When validated, the region pointers are updated in the corresponding header and the old header region pointers are set to either 0x000000h or 0xFFFFFFh. Only one region contains the updated code and valid pointers which allows errors to occur during update without overwriting the current code.
2. Store firmware in both halves of the memory space. This method allows redundancy in the memory. Because both halves contain the exact same information, an error may occur in one copy but not in the other. The config page and application code validation checks catch the error and use the other copy. This method also allows protection during application code updating by only updating one copy and then validating this copy before updating the other copy. If an error occurs during write, when application code is loaded, the invalid application code is ignored.

4.4.1 Application Code Update through I^2C

The TPS65982 host interface contains two registers that can be used to execute various routines. The Cmd1 register (0x08) uses information stored in Data1 (0x09) when executing commands, while the Cmd2 register (0x10) uses the Data2 register (0x11). For more information on the host interface 4CC commands and their use, refer to the *TPS65981, TPS65982, and TPS65986 Host Interface Technical Reference Manual*.

NOTE: The CmdX register must be read back until it returns a value of 0x00 to determine the command is done executing and at least one byte of the DataX register must be read back and used to determine if the command was executed properly (least-significant bit = 0).

When updating the flash memory with the host interface, the following procedure should be followed:

Step 1. Determine which region will be updated. The DataX register should by populated with a value of 0x00 for the low region or 0x01 for the high region.

Step 2. Issue the FLrr command to the corresponding CmdX register and once complete read back the value of the DataX register. The value read back is the address of the chosen region.

Step 3. Erase the region header by writing a value of 0x00 for the low region or 0x01 for the high region to the DataX register and issuing the FLer command.

Step 4. Erase the selected region by writing the address from step 2 plus the number of 4kB sectors to erase to the DataX register and issuing the FLem command in the corresponding CmdX register.

Step 5. Write the 32-bit address of the application code location from step 2 to the DataX register in the host interface.

Step 6. Issue the FLad command using the appropriate CmdX register. This instruction sets the location stored in step 1 as the start location of the next flash write command.

Step 7. Write up to 64 bytes of the application code to be written to the DataX register.

Step 8. Issue the FLwd command to the CmdX register. This command will write the data stored in the previous step to flash memory starting at the location written in step 1.

Step 9. Repeat Step 7 and Step 8 until all of the application code is written. The FLwd command will auto-increment the write address after each 64-byte chunk. The first 4kB of application code must be a valid TPS65982 boot header.

Step 10. Write the boot header address of the updated region to the DataX register.

Step 11. Execute the FLvy command using the CmdX register. If this command returns 0x00 in the DataX register the update process was successful.

Step 12. If the update verified successfully, update the region pointer with the FLad and FLwd commands.

4.4.2 Application Code Update through External Device

The TPS65982 device only communicates with the flash memory containing the application code during boot or when a 4CC command has been issued through the host interface. Therefore an external device can update the application code while not in use by the TPS65982 device. Any updates made to the application code by an external device are required to follow the memory structure described in Section 3.5 and Section 4.2.

5.1 Power States

The TPS65982 device provides a flexible power and clock management architecture that allows the power to the analog and digital core to be turned on and off as well as clock dividing or gating to save power in digital circuits. This flexibility allows implementing various power states as shown in Figure 5-1 based on application needs.

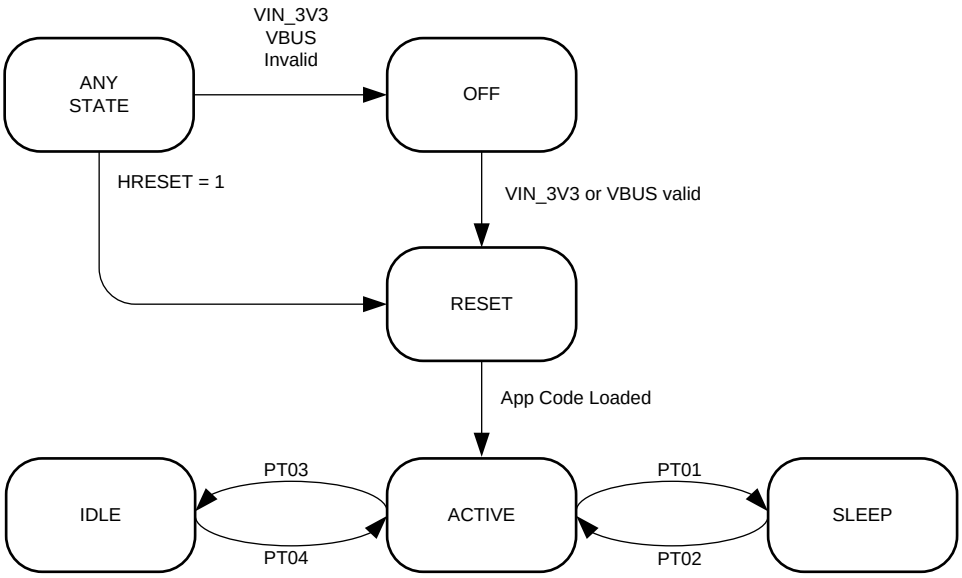

Figure 5-1. Power State Diagram

The TPS65982 firmware implements the SLEEP, IDLE, and ACTIVE states by programming the hardware resources. As shown, all entry into low-power states must originate from the ACTIVE state. Similarly, all low-power states transition to the ACTIVE state upon exiting.

Table 5-1 summarizes the state of the power supplies, oscillators, and functionality that can be supported in each power state.

Table 5-1. Power States Summary

	Power Off	Dead Battery[1]	SLEEP	IDLE	ACTIVE
VIN_3V3	Not Valid	Not Valid	Valid	Valid	Valid
VBUS	Not Valid[1] or Valid[2]	Valid	Do not care	Do not care	Do not care
LDO_3V3	Disabled	Enabled	Enabled	Enabled	Enabled
LDO_1V8D					
LDO_1V8A					
FOSC_100K	OFF	ON	ON	ON	ON
FOSC_48M	OFF	ON	OFF	ON	ON
USB Type-C Detection (Cable attach/detach)	No	Yes	Yes	Yes	Yes

(1) Assumes dead-battery support is enabled through RPD_CCn configuration.
(2) Assumes dead-battery support is disabled through RPD_CCn configuration.

Table 5-1. Power States Summary (continued)

	Power Off	Dead Battery[1]	SLEEP	IDLE	ACTIVE
I^2C	No	Yes	No[3]	Yes	Yes
UART	No	Yes	No	No	Yes
SPI	No	Yes	No	No	Yes
USB PD	No	Yes	No	No	Yes

(3) Wake up from SLEEP to ACTIVE upon an I^2C message is supported, however, the first I^2C message is lost.

5.2 Activity Timer

The device uses a programmable timer to monitor activity that is occurring while operation is in the ACTIVE state. The counter is reset automatically to its programmed value because of the following events, and will begin counting again:

- Upon entry into the ACTIVE state
- I^2C activity
- UART activity
- PD modem activity

If no activity is detected within the programmed time, the activity timer times out, indicating to the firmware that the device can exit the active mode and transition to a lower power mode when possible.

The activity timer can be programmed through the host interface using the Sleep Configuration register. Refer to the *TPS65981, TPS65982, and TPS65986 Host Interface Technical Reference Manual* for details.

5.3 System Power State

The TPS65982 host interface contains a register for storing the system power state. This state does not reflect the power state of the TPS65982 device but rather the state of the surrounding system. The TPS65982 power-management firmware compares the state stored in the System Power State register with the state stored in the second byte of the Sleep Configuration register when attempting to enter a sleep state. If the system power state is equal to or less than the state of the Sleep Configuration register, the TPS65982 device enters the lowest power sleep state. If the stored value is greater than that of the Sleep Configuration register, the TPS65982 device enters the higher power idle state.

NOTE: Higher hex values correspond to lower power states. Such that 0x00 is the highest power state (S0) and 0xFE is the lowest power state (S254).

5.4 Power State Descriptions

5.4.1 POWER OFF

The TPS65982 device is in the POWER OFF state when VIN_3V3 and VBUS are not valid.

5.4.2 RESET

The TPS65982 device has a POR (power-on-reset) circuit that initializes the device when VIN_3V3 or VBUS are valid. While in the RESET state the TPS65982 device carries out boot code and RESETZ is asserted until VOUT_3V3 is valid and TUVRDELAY has elapsed. After application code has loaded, the TPS65982 device transitions into the ACTIVE state.

5.4.3 SLEEP

Sleep is the low-power state of TPS65982 device. Sleep state can only be entered while the device is unattached or operating in a legacy 5-V application. During SLEEP state the device operates from the 100-kHz oscillator to monitor for wake-up events and communication with the device is disabled.

5.4.3.1 Entry to the SLEEP (Power Transition 01 – PT01)

Entry to the SLEEP state is only possible from the ACTIVE state. Entry to the SLEEP occurs because of the following events:

- The system power state is equal to or less than state set in the Sleep Configuration Register (controlled by the host interface).
- No cable attached or cable detach event occurs.
- The device is a source connected to a non-PD capable sink or device is configured as a non-PD capable sink (controlled by the host interface).
- The activity timer times out (controlled by the host interface).

5.4.3.2 Exit from SLEEP (Power Transition 02 – PT02)

Exit from the SLEEP state is always to ACTIVE. Exit from SLEEP occurs because of the following events:

- Any reset event
- I^2C bus activity
- Any enabled interrupt event (I^2C interrupt request, supervisor events, CC attach or detach events, and so forth)

NOTE: For DRP and sink with accessory ports that require DRP toggle and accessory toggle operations, respectively, the TPS65982 device supports the toggle operations completely while operating in SLEEP. Therefore, no transition to ACTIVE is required to perform the toggle operation which enables significant power consumption savings for DRP and sink with accessory ports while in the Unattached.SNK and Unattached.SRC states.

Upon exiting SLEEP, the TPS65982 device transitions to ACTIVE. The TPS65982 device enables FOSC_48M and waits for it to stabilize before releasing it to the digital core.

5.4.4 IDLE

The IDLE state is a low-power state similar to SLEEP, except that the high-speed oscillator is kept active to allow the TPS65982 device to continue to respond immediately to I^2C commands. While in IDLE, processing is enabled, however, with a clock frequency of 6 MHz. The TPS65982 device advertises itself on CC1 and CC2 according to the configuration and monitors USB Type-C Port for attach or detach.

5.4.4.1 Entry to the IDLE (Power Transition 03 – PT03)

Entry to the IDLE state is only possible from the ACTIVE state. Entry to the IDLE occurs because of the following events:

- The system power state is greater than state set in Sleep Configuration Register (controlled by the host interface).
- No cable attached or cable detach event occurs.
- The device is a source connected to a non-PD capable sink or device is configured as a non-PD capable sink (controlled by the host interface).
- The activity timer times out (controlled by the host interface).

5.4.4.2 Exit from IDLE (Power Transition 04 – PT04)

Exit from the IDLE state is always to ACTIVE. Exit from IDLE occurs because of the following events:

- Any enabled interrupt event (I^2C interrupt request, supervisor events, CC detach events, and so forth) except host command events.
- Any reset event

5.4.5 ACTIVE

The ACTIVE state is an operational state where either USB PD or USB2.0 data transmission activity happens on the USB Type-C Port and the TPS65982 device responds to configuration and status commands from the host through the I^2C interface. The TPS65982 device is usually in one of attached (DFP, UFP, or Alternate Mode) USB Type-C Port states in the ACTIVE state. The TPS65982 device advertises itself on CC1 and CC2 as according to the configuration and monitors USB Type-C Port for attach or detach. The TPS65982 device runs the policy engine and all associated hardware and software logic if USB PD communication is required.

5.4.5.1 Entry to the ACTIVE

Entry to the ACTIVE state occurs because of the following events:

- Exit from all low-power states transition to ACTIVE
- Reset event

5.4.5.2 Exit from ACTIVE

Exit from the ACTIVE state to any low-power states meeting the criteria for entry into the respective low-power state.

5.4.6 Dead Battery

In systems where the battery is unable to provide adequate power to the TPS65982 VIN_3V3 supply, the TPS65982 dead-battery mode allows the device to power from VBUS. VBUS power can also be passed to the system to allow for battery charging. While in the DEAD BATTERY state, the TPS65982 device operates as if the Type-C connection is the only source of power to the system.

5.4.6.1 Entry to the Dead Battery

The TPS65982 dead-battery behavior is defined by the configuration of RPD_G1 and RPD_G2 pins. These pins can be connected to C_CC1 and C_CC2, respectively, to enable dead-battery support. Alternately, they may be connected to ground to disable dead-battery support.

If dead-battery support is enabled, when connected to a source, an unpowered TPS65982 powers the Rd resistors from the C_CC1 and C_CC2 pins and advertises itself as a Sink.

In response, the source provides VBUS power and the TPS65982 device initiates the boot flow. During boot, the TPS65982 device samples the BUSPOWERZ pin to determine if VBUS is received by the system through the PP_EXT path, or the PP_HV path. The device then continues through the RESET state, carrying out the boot flow and loading the application code.

Dead-battery operation is indicated by the dead-battery flag located in register 0x2D of the host interface. The flag is set by the TPS65982 bootloader upon detection of dead-battery conditions.

5.4.6.2 Dead Battery Restrictions

While in dead-battery mode certain functions of the TPS65982 device are restricted to ensure power from VBUS is not lost. The TPS65982 port type is restricted to sink only, and all power role-swap requests are rejected. Additionally only the dead-battery power switch (configured through BUSPOWERZ) is allowed to close. Because of this behavior, TI recommends that the sink switch defined in the device system configuration match the switch enabled by BUSPOWERZ.

The TPS65982 device does not source VCONN while operating in dead-battery mode, and rejects any VCONN swap requests.

5.4.6.3 Exit from Dead Battery

Dead-battery operation can be exited in one of the following ways:

- Executing the *DBfg* host interface 4CC command
- High edge occurring on a GPIO configured with BARREL_JACK_EVENT

- VBUS removal during dead-battery operation
- Thermal protection event occurring

The DBfg 4CC command upon execution clears the dead-battery flag used to indicate dead-battery operation and exits the dead-battery mode. The *DBfg* command is primary method for exiting dead-battery mode, and should be issued by an attached embedded controller once the system is self-sustainable and able to provide VIN_3V3.

The GPIO event labeled, *BARREL_JACK_EVENT*, issues the *DBfg* command upon detection of a rising edge. This event can be used to clear the dead-battery flag upon application of an external power source. The external power source should be tied to the GPIO through a resistor divider such that the GPIO is at the programed I/O voltage when the external source is present. For more information on the barrel jack GPIO event see Table 13-1.

If VBUS is removed during dead-battery operation, the TPS65982 device issues a cold device reset. This reset causes the device to restart the boot process, forcing a reevaluation of the VIN_3V3 state.

USB Type-C

6.1 Overview

Main functionality supported:

- USB Type-C port configuration
- CC detection
- USB Type-C connection state machines for:
  - Downstream facing port (DFP)
  - Upstream facing port (UFP)
  - UFP with accessory support
  - Dual-role port (DRP)
  - DRP with accessory and Try.SRC support
- Accessory modes
  - Audio Adapter Accessory Mode
  - Debug Accessory Mode

6.2 USB Type-C Port Configuration

The TPS65982 firmware supports configuring the USB Type-C port based on the needs and capabilities of the system. These USB Type-C port configurations include:

- Power capabilities of the port (sink, source, DRP)
- Receptacle type
- USB Type-C current advertisement for a port that has power sourcing capabilities (source, DRP)
- VCONN support modes
- VBUS power switch settings
- VCONN power switch settings

The device configuration is initially loaded from the configuration data loaded along with the application code. Additionally the TPS65982 host interface allows access to a System Configuration register where these USB Type-C port configurations can be written to or read from. For more information about all the USB Type-C port configurations offered, see the System Configurations register bit field definitions in the *TPS65981, TPS65982, and TPS65986 Host Interface Technical Reference Manual*.

6.2.1 Source

When configured as a source by the application code Config Data, the TPS65982 device disables the VBUS power path and VCONN power path and enables the CC pin pullup current sources. The device then enters the USB Type-C state machine in the Unattached.SRC state and waits for a connection on the USB Type-C port.

6.2.2 Sink

When configured as a sink by the application-code configuration data, the TPS65982 device enables the pulldown resistors on the CC pins and the VBUS detection circuitry. The device then enters the USB Type-C state machine in the Unattached.SNK state, and waits for a connection on the USB Type-C port.

6.3 CC Detection

When configured as a source, the TPS65982 device continually monitors the state of the CC pins. The possible detected configurations are summarized in Table 6-1.

Table 6-1. USB Type-C Port State Based on CC Terminations (Source Perspective)

CC1	CC2	State
Open	Open	Nothing attached
Rd	Open	Sink attached
Open	Rd	
Open	Ra	Powered cable without sink attached
Ra	Open	
Rd	Ra	Powered cable with sink or VCONN-powered accessory attached
Ra	Rd	
Rd	Rd	Debug accessory attached
Ra	Ra	Audio Adapter Accessory Mode attached

6.4 USB Type-C Connection State Machine

The universal serial bus Type-C cable and connector specification define the mandatory and optional states for each type of port. While the TPS65982 device supports all mandatory states, the TPS65982 USB Type-C connection state machine can support optional states selectively based on OTP configuration bits as shown in Table 6-2.

Table 6-2. USB Type-C Connection States Supported

	Source(1)	Sink(1)	DRP(1)	USB PD Communication
Disabled	○	○	○	Not permitted
ErrorRecovery	○	○	○	Not permitted
Unattached.SNK	N/A	●	●	Not permitted
AttachWait.SNK	N/A	●	●	Not permitted
Attached.SNK	N/A	●	●	Permitted
Unattached.SRC	●	N/A	●	Not permitted
AttachWait.SRC	●	N/A	●	Not permitted
Attached.SRC	●	N/A	●	Permitted
Try.SRC	N/A	N/A	○	Not permitted
TryWait.SNK	N/A	N/A	○	Not permitted
AudioAccessory	○	○	○	Not permitted
DebugAccessory	○	○	○	Permitted
Unattached.Accessory	N/A	○	N/A	Not permitted
AttachWait.Accessory	N/A	○	N/A	Not permitted
Powered.Accessory	N/A	○	N/A	Permitted
Unsupported.Accessory	N/A	○	N/A	Not permitted
PowerDefault.SNK	N/A	●	●	Permitted
Power1.5.SNK	N/A	○	○	Permitted
Power3.0.SNK	N/A	○	○	Permitted

(1) ○: optional state supported by the TPS65982, ●: mandatory state supported by the TPS65982, N/A: not applicable

Accessory Modes

7.1 Audio Accessory Mode

The TPS65982 enters Audio Adapter Accessory Mode when it detects the states of both CC pins at SRC.Ra (that is, the analog audio adapter identifies itself by presenting a resistance to ground of ≤ Ra on both CC and VCONN pin of the USB Type-C plug).

When in Audio Adaptor Accessory Mode, the USB Type-C port mux is unused. Because the audio signals are routed externally to the Type-C connector, the USB Type-C port mux must be set to high impedance (Hi-Z).

7.2 Debug Accessory Mode

The TPS65982 device enters Debug Accessory Mode when it detects the states of both CC pins at SRC.Rd range (that is, the debug accessory identifies itself by presenting a resistance to Rd on both CC and VCONN pin of the USB Type-C plug).The TPS65982 device configures the USB Type-C port multiplexor. The TPS65982 device requires no checks for proper orientation of the debug accessory. The user is assumed to be responsible for providing the proper orientation of the debug accessory.

The system port signals, UART_TX and UART_RX, are rerouted by the digital crossbar inside the digital core. The UART_TX and UART_RX signals are level shifted and buffered, routed through the cross bar mux, and level shifted to the USB Type-C port signals C_USB_BP and C_USB_BN, respectively. In addition, the system port signals USB_RP and USB_RN are routed as analog signals to the USB Type-C port signals C_USB_TP and C_USB_TN, respectively. Lastly, the system port signals SWD_CLK and SWD_DIO are routed as analog signals to the USB Type-C port signals SBU1 and SBU2, respectively.

Type-C Port Multiplexer Configurations

The default configurations of the USB Type-C multiplexor are determined by the USB Type-C port states as shown in Table 8-1.

Table 8-1. USB Type-C Port Multiplexer Configurations

USB Type-C Port Pin	Unattached	Attached.SRC or Attached.SNK		Alternate Modes	Audio Adapter Accessory Mode	Debug Accessory Mode	Alternate Debug Modes
		Plug CC = CC1	Plug CC = CC2				
C_USB_TP	Hi-Z	USB_EP_P or USB_RP_P(1)	Hi-Z	Configurable through I²C register settings or Structured VDMs	Hi-Z	USB_RP_P	Configurable through I²C register settings
C_USB_TN	Hi-Z	USB_EP_N or USB_RP_N(1)	Hi-Z		Hi-Z	USB_RP_N	
C_USB_BP	Hi-Z	Hi-Z	USB_EP_P or USB_RP_P(1)		Hi-Z	UART_TX	
C_USB_BN	Hi-Z	Hi-Z	USB_EP_N or USB_RP_N(1)		Hi-Z	UART_RX	
SBU1	Hi-Z	Hi-Z	Hi-Z		Hi-Z	SWD_CLK	
SBU2	Hi-Z	Hi-Z	Hi-Z		Hi-Z	SWD_DATA	

(1) USB_EP or USB_RP connection dependent on system configuration settings.

USB Power Delivery

9.1 Overview

The TPS65982 USB Power Delivery firmware allows pairs of directly attached ports to negotiate voltage, current, direction of power flow over the USB cable or all of these, using the CC wire as the communications channel.

The Physical Layer firmware handles transmission and reception of bits on the CC wire.

The Protocol Layer firmware enables messages to be exchanged between a Source Port and a Sink Port.

The Policy Engine firmware implements the Local Policy for the Port.

The TPS65982 USB Power Delivery firmware supports Standard and Vendor defined Modal Operation.

9.2 Protocol Layer

The TPS65982 Protocol Layer firmware forms the messages used to communicate information between a pair of ports. The firmware is responsible for forming capabilities messages, requests and acknowledgments. Additionally, the firmware forms messages used to swap roles and maintain presence. The firmware receives inputs from the policy engine indicating which messages to send and indicates the responses back to the policy engine.

The basic protocol uses a push model where the source pushes the capabilities to the sink that, in turn, responds with a request based on the offering. However, the sink can asynchronously request the present capabilities of the source and can select another voltage or current.

The TPS65982 protocol layer implements the following according to the Universal Serial Bus Power Delivery Specification:

- Control messages
- Data messages
- Timers
- Counters
- Reset

9.2.1 Control Messages

Table 9-1 summarizes control messages supported by the TPS65982 protocol layer.

Table 9-1. Control Messages

Control Message	Sent by
GoodCRC	Source, sink or cable plug
GoToMin	Source only
Accept	Source, sink or cable plug
Reject	Source or sink
Ping	Source only
PS_RDY	Source or sink
Get_Source_Cap	Source or sink
Get_Sink_Cap	Source or sink
DR_Swap	Source or sink

Table 9-1. Control Messages (continued)

Control Message	Sent by
PR_Swap	Source or sink
VCONN_Swap	DFP
Wait	Source or sink
Soft Reset	Source or sink

9.2.2 Data Messages

The TPS65982 firmware supports the following data message types:

- Source capabilities
- Request
- BIST
- Sink capabilities
- Vendor defined

9.2.2.1 Power Data Objects

Power data objects (PDOs) are used by USB Power Delivery messages to communicate the power capabilities or power requirements of a source. The TPS65982 host interface and system configuration allows for seven PDOs to be implemented and stored on the device. When defining device PDOs, care should be taken to avoid overlap of the voltage capabilities of the PDO as the TPS65982 device does not support these configurations.

9.2.3 Reset

The TPS65982 firmware implements both soft reset and hard reset as defined by the USB Power Delivery Specification.

A soft reset message is used to cause a soft reset of the protocol communication when it has broken down in some way. The soft reset does not have any impact on power supply operation and may be triggered by either port partner in response to an error.

A hard reset is signaled by an ordered set. Both the sender and recipient reset both the power supplies and protocol.

9.3 Policy Engine

The Universal Serial Bus Power Delivery Specification defines and provides detailed message sequences and associated timing requirements. Because the message sequences are explicitly described in the Universal Serial Bus Power Delivery Specification, this document establishes a framework for supported message sequences and refers to the Universal Serial Bus Power Delivery Specification referenced in Section 1.2. The USB standards documents should be referenced for the latest information.

Table 9-2. USB PD Message Sequences Supported by the TPS65982

Message sequence	Message sub-sequence
Power Negotiation	
Reclaiming Power with GoToMin message	
Soft Reset	
Hard Reset	Source Initiated Hard Reset
	Sink Initiated Hard Reset
	Source Initiated Hard Reset – Sink Long Reset
Type-C Power Role Swap	Type-C Source Initiated Power Role Swap without subsequent Power Negotiations

Table 9-2. USB PD Message Sequences Supported by the TPS65982 (continued)

Message sequence	Message sub-sequence
	Type-C Sink Initiated Power Role Swap without subsequent Power Negotiation
Type-C Data Role Swap	Type-C Data Role Swap, Initiated by UFP Operating as Sink
	Type-C Data Role Swap, Initiated by UFP Operating as Source
	Type-C Data Role Swap, Initiated by DFP Operating as Source
	Type-C Data Role Swap, Initiated by DFP Operating as Sink
Type-C VCONN	Type-C DFP to UFP VCONN Source Swap
	Type-C UFP to DFP VCONN Source Swap
Structured VDM	DFP to UFP Discover Identity
	Source Port to Cable Plug Discover Identity
	DFP to Cable Plug Discover Identity
	DFP to UFP Enter Mode
	DFP to UFP Exit Mode
	DFP to Cable Plug Enter Mode
	DFP to Cable Plug Exit Mode
	UFP to DFP Attention
	Cable Plug to DFP Attention
Built in Self-Test (BIST)	BIST Receiver Mode
	BIST Transmit Mode
	BIST Test Patterns

9.3.1 Request Message

During power negotiation, the sink port sends a *Request* message to request power in response to the most recent *Source Capabilities* message. The *Request* message returns one sink request data object (RDO) that identifies the power data object (PDO) being requested. The USB Power Delivery specification describes the various types of Request messages depending on the type of supply (fixed, battery, or variable).

When the TPS65982 device is operating as a sink, the policy engine firmware selects the source PDO of matching supply type that will deliver maximum power. When the sink cannot satisfy its power requirements from the capabilities offered by the source, the sink sets the Capability Mismatch bit in RDO.

9.3.1.1 Automatic Request Negotiation

When operating as a sink, the TPS65982 device can automatically select the best source capability and send the appropriate request message. The TPS65982 device determines the sink capability from the power data objects stored in the 0x33 TX Sink Capabilities Register. The settings stored in the0x37 Auto Negotiate Sink Register are then used to prioritize and select the best received source PDO. The selected PDO is determined using the following process:

Step 1. Attempt to determine the best PDO using the user defined priority set in the Offer Priority field of the Auto Negotiate Sink Register.

Step 2. If multiple PDOs exist of equal priority, select based on supply type: fixed supply, then variable supply, then battery supply.

Step 3. Select PDO based on highest offered peak current.

Alternate Modes

10.1 Overview

The Universal Serial Bus Type-C Cable and Connector Specification provides support for alternate modes using the USB Type-C connector and cables. In an alternate mode various pins on the USB Type-C connector may be reconfigured to support interfaces outside the scope of USB Type-C.

The TPS65982 device implements the discovery process as outlined in the Universal Serial Bus Type-C Cable and Connector Specification (and Universal Serial Bus Power Delivery Specification, which it leverages) for discovering the support of alternate modes in connected devices, including the method for switching into and out of a mode.

10.2 USB Billboard

The integrated USB low-speed endpoint of the TPS65982 device allows the device to comply with USB Type-C standards without needing additional external billboard devices. After a UFP attach event, if an alternate mode is not entered after one second has elapsed, the TPS65982 device exposes the USB billboard. The USB billboard can be provided by the integrated USB endpoint of the TPS65982 device or by an externally provided endpoint on the devices USB_RP pins. The TPS65982 firmware only supports the EP0 control endpoint.

10.3 Automatic Entry

When attached to a port partner supporting alternate mode, the TPS65982 device automatically attempts to negotiate mode entry for discovered alternate modes.

For modes that require the use of connector resources, such as the SBU lines, or are mutually exclusive, the Alternate Mode Automatic Entry Sequence Register of the host interface should be used. When negotiating alternate modes the TPS65982 device prioritizes mode entry based on mode order in this register. When a mode in the register is entered, mode entry for other modes listed will not be attempted.

10.4 DisplayPort Alternate Mode

The TPS65982 device supports DisplayPort as found in the DisplayPort Alt Mode Standard and contains hardware to support HPD handling. Figure 10-1 shows the TPS65982 process for handling HPD as a DP sink (UFP_D).

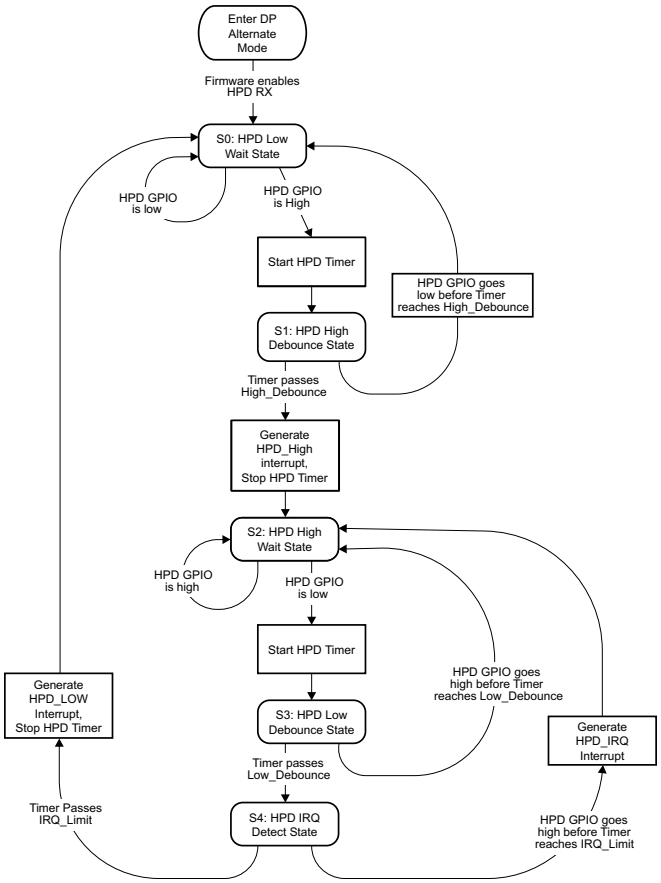

Figure 10-1. DisplayPort Sink-Side Hardware Flow (HPD RX)

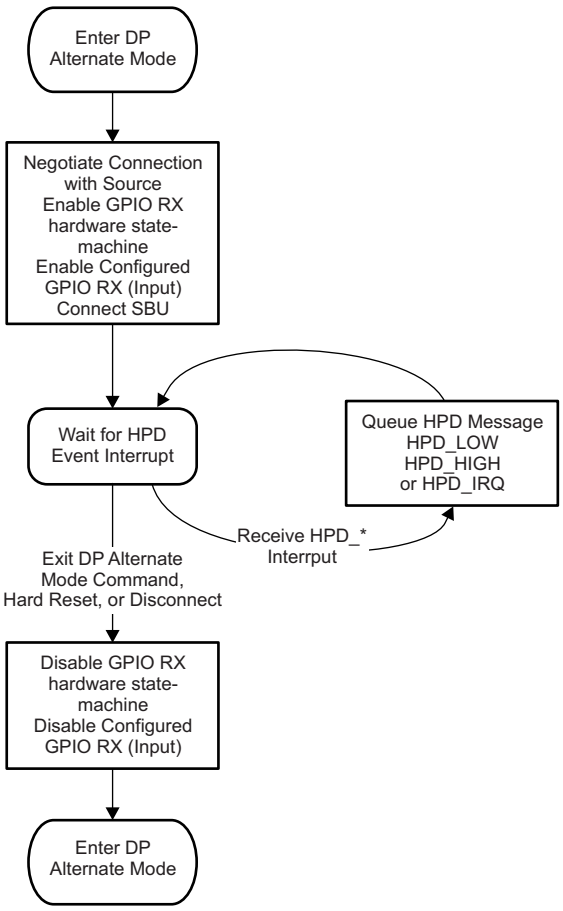

Figure 10-2. DisplayPort Sink-Side Firmware Flow (HPD RX)

10.5 PDIO Alternate Mode

10.5.1 Overview

The PDIO alternate mode provides a method for remotely controlling a GPIO over a USB PD connection. The mode pairs an input and an output GPIO, such that if an external signal drives the input GPIO high, the paired output GPIO is drives high in response. This implementation allows for very-slow speed signals, such as pushbutton events and LED enables, to be communicated over a USB-Type C connection.

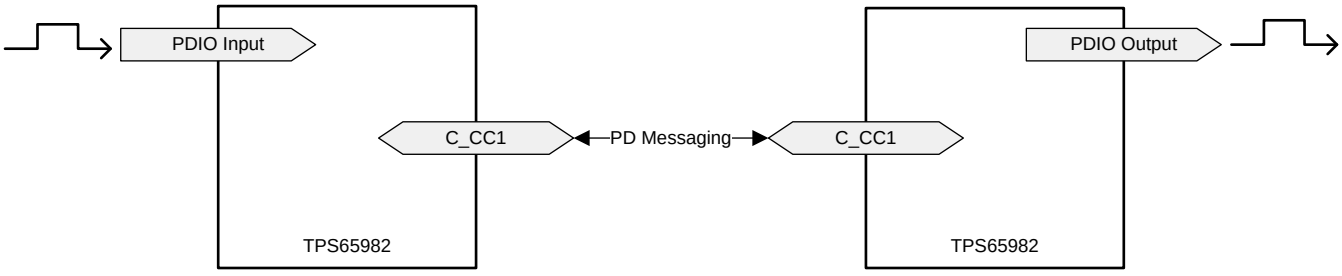

Figure 10-3. PDIO Alternate Mode

10.5.2 PDIO GPIO Events

The PDIO alternate mode uses four pairs of GPIO events to configure the TPS65982 GPIOs as either a PDIO input or PDIO output. The events are pair such that the state of the PDIO_OUT0 GPIO mirrors any changes made to the PDIO_IN0 GPIO. Each TPS65982 device can have four GPIOs configured as PDIO inputs and four GPIOs configured as PDIO outputs. For more information on GPIO events see Table 13-1.

10.5.3 PDIO Signature

To ensure PDIO operation only occurs between compatible devices, the TPS65982 device configuration includes a 32-bit PDIO signature. This signature can be configured through the TPS6598x Configuration Tool. The upper 16 bits should be a unique product ID defined by the user, while the lower 16 bits should be set to the USB vendor ID of the device as shown in Table 10-1.

Table 10-1. PDIO Signature

Bit(s)	Field	Value
31:16	Product ID	16 bit product ID 00h = Virtual wire not supported XXh = User defined product ID
15:0	Device VID	16 bit USB VID of product vendor

10.6 User Alternate Mode

The User Alternate mode allows users to configure an arbitrary SVID with up to four independently configurable mode numbers. If enabled, this custom SVID is added to the list of supported SVIDs used to respond to a *Discover SVIDs* PD command and the modes will be entered, assuming that the SVID and mode numbers configured are also supported by the far-end device.

When the TPS65982 device is operating autonomously as a stand-alone port controller, the User Alternate Mode feature executes four steps in sequential order:

1. Enter the user-defined alternate mode using SVID and mode number.
2. Send a predefined unstructured VDM (optional).
3. Reconfigure the TPS65982 device from an application configuration data set (optional).
4. Execute up to two host interface commands upon mode entry (optional).

Step 2, Step 3, and Step 4 are listed as *optional* because they are essential to the User Alternate Mode when the TPS65982 device is a stand-alone PD port controller, but when an external system controller (I^2C master) is also used in the application steps 2-4 or even more steps can be executed dynamically based on other system conditions any time after the User Alternate Mode is successfully entered. When an external system controller is present, the User Alternate Mode can be exited and Step 3 and Step 4 can be repeated with different behavior.

When the TPS65982 device is executing the User Alternate Mode autonomously, the reconfiguration of the TPS65982 device is loaded from an additional application configuration data set and two different host interface commands can be executed.

10.6.1 Enter the Alternate Mode

The DFP sends a *Discover SVIDs* PD command to identify Alternate Modes supported by the UFP. When it is determined that both port partners support the SVID associated with the User Alternate Mode, the DFP sends a *Discover Modes* command before entering the mode. The User Alternate Mode can support up to four (4) mode numbers and each mode number is 32-bits long. Although it is common for SVIDs to start numbering modes at decimal 1 and increment by one digit at a time (1→2→3→4), the first mode number in the list may be 0x11111111 or any other 32-bit number and the mode numbers are not restricted to incrementing one digit at a time. The only restriction is that both port partners support the mode number to enter the alternate mode.

10.6.2 Send Unstructured Vendor-Defined Message

The ability to send a pre-defined unstructured vendor-defined message (VDM) upon mode entry is generally used to advertise an identity. One practical example of a custom alternate mode is communicating between battery-powered products and compatible power supplies. A power supply that does not contain a dedicated processor can use the integrated processor of the TPS65982 to automatically send an unstructured VDM advertising the power supply's model number, revision, serial number, and others.

10.6.3 Reconfigure the TPS65982 from Data Set

The ability to reconfigure the TPS65982 device allows modification of any of the configuration registers of the host interface automatically upon mode entry from a data set stored in flash memory. This can be used, for instance, to modify the power sourcing and sinking capabilities of the PD port when compatible products are attached. Any register listed as read/write (R/W) in the *TPS65981, TPS65982, and TPS65986 Host Interface Technical Reference Manual* and accessible in the TPS6598X Application Customization Tool can be reconfigured upon mode entry and the registers that need to be modified will vary based on the application.

10.6.4 Execute Host-Interface Commands

After reconfiguration of the host interface registers, up to two host-interface 4CC commands may be executed. The first 4CC command that is executed is an internal command that modifis the behavior of the TPS65982 device. A practical example is driving a GPIO high or low to indicate alternate mode entry or clearing the dead-battery flag to indicate external power is available. The second 4CC command that is executed can be an internal command or a PD task. A practical example of a PD task is forcing renegotiation of the PD power contract or issuing a data-role or power-role swap request.

Power Delivery Fault Handling

The USB Type-C port manager can detect a number of power delivery fault conditions. The USB Type-C port manager directs the USB Type-C connection state machine to the DISABLED state and attempts to enter a safe state.

Table 11-1 lists fault conditions that can be detected by the TPS65982 device when the port is configured for a particular data role or power role.

Table 11-1. Power Delivery Fault Conditions

Fault Condition	Data Role	Power Role
VBUS overcurrent	DFP, DRP	Source
VBUS reverse current	DFP, UFP, DRP	Source, sink
VBUS overvoltage	DFP, UFP, DRP	Source, sink
VBUS undervoltage	DFP, UFP, DRP	Source, sink
VCONN overcurrent	DFP, UFP, DRP	Source, sink

Charger Detection

12.1 Firmware Description

Charger-detection firmware implements a state machine to detect standard chargers that follow the USB Battery Charging Specification v1.2 as shown in Figure 12-1.

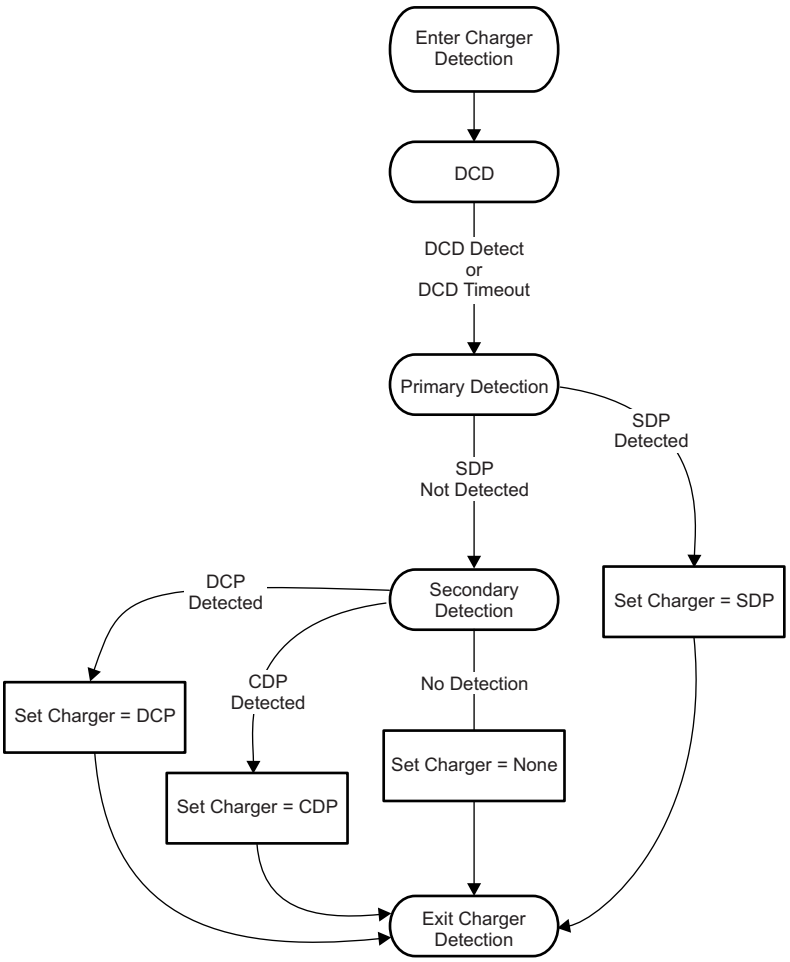

Figure 12-1. Charger-Detection State Machine

12.1.1 VBUS Detect

When a valid VBUS is detected, the charger detection firmware sets the DCD timer and proceeds to the DCD charger detection.

12.1.2 Data-Contact Detect

Data-contact detect (DCD) uses a current source to detect when the data pins have made contact during an attach event for most cases (SDP, CDP, most DCP cases). Because DCD does not work in all cases, a DCD timer is implemented to proceed with primary detection after it reaches the DCD timeout value. The primary benefit of DCD is that it allows starting primary detection as soon as the data pins have made contact, and then connects without having to wait for the DCD timer to expire. DCD logic is implemented in the firmware as shown in Figure 12-2.

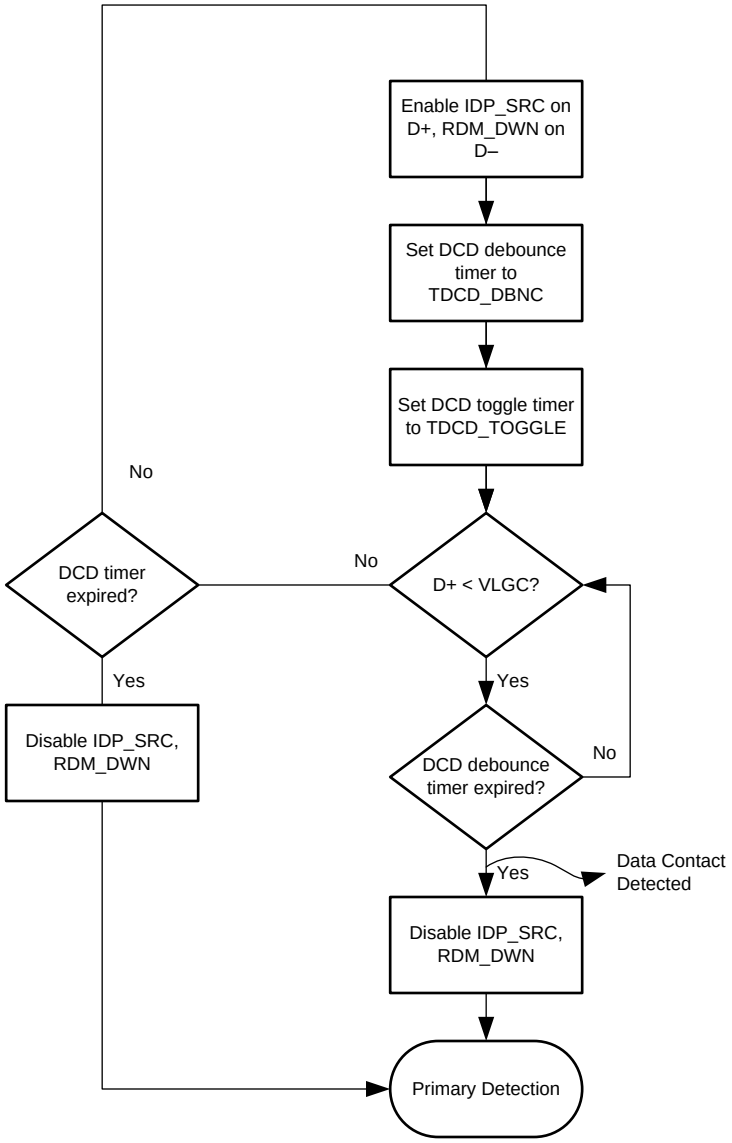

Figure 12-2. Data-Contact Detect (DCD)

12.1.3 Primary Detection

Primary detection is used to distinguish between an SDP and different types of charging ports.

Primary-detection logic is implemented in the firmware as shown in Figure 12-3.

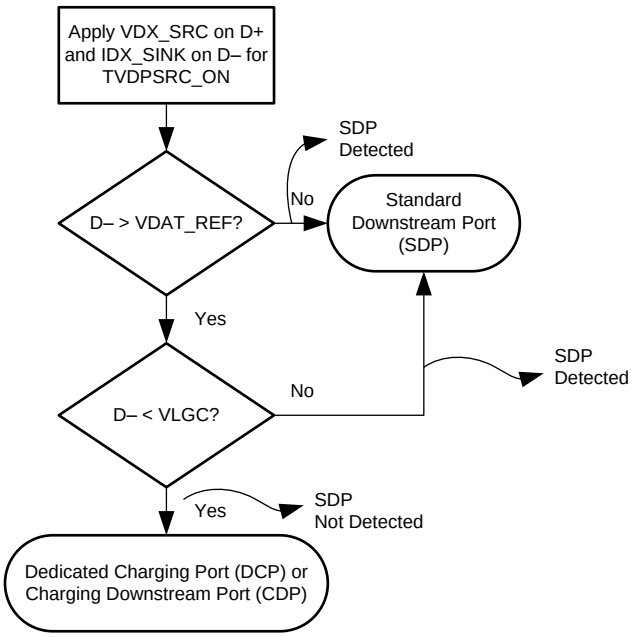

Figure 12-3. Primary Detection

12.1.4 Secondary Detection

Secondary detection is used to distinguish between a DCP and a CDP.

Secondary-detection logic is implemented in the firmware as shown in Figure 12-4.

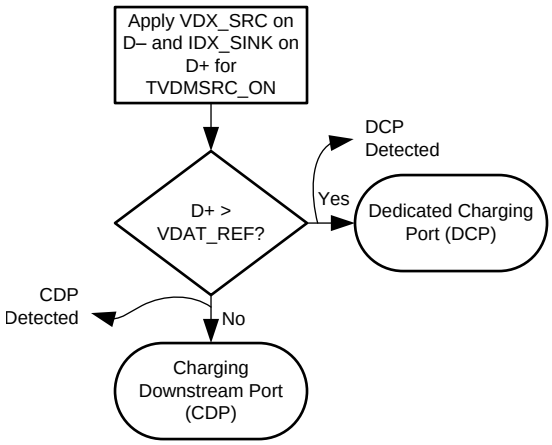

Figure 12-4. Secondary Detection

Device Features

13.1 ADC

The TPS65982 device features an integrated ADC which monitors various device voltages and currents. The host interface includes commands thatcan be used to read and report these values over the I^2C interface. When an ADC conversion is complete, Equation 1 can be used to convert an ADC current reading to its respective value and Equation 2 can be used for a temperature reading.

$$I = \frac{1.2}{1023} \times ADC_{dec} \times Isense_{ACC} \quad (1)$$

where

- I = Current in Amps
- ADC_{dec} = ADC reading in decimal
- $Isense_{ACC}$ = Current-sense accuracy

$$T = \frac{\frac{1.2}{1023} \times ADC_{dec} + 0.6 - T_{VO}}{T_{Gain}} \quad (2)$$

where

- T = Die temperature in degrees Celsius
- T_{V0} = 0.823 V
- T_{Gain} = 0.003095 V/°C

13.2 Digital I/O

The TPS65982 device features 19 configurable GPIOs. Each GPIO output can be configured as open-drain or push-pull, and use either LDO_3V3 or VDDIO as the supply. Each GPIO can also be configured with a weak internal pullup resistor, pulldown resistor, or both types of resistors enabled.

The firmware also specifies specific events that can be tied to GPIOs. These events dictate the behavior of a specified GPIO in response to a defined hardware or USB event. The TPS65982 Configuration Tool can be used to assign events to specific GPIOs. Table 13-1 specifies the events that are available in firmware for use with the GPIOs and their behavior.

Table 13-1. GPIO Events

Event Name	I/O	Active State	Behavior
ATTACHED_H (PLUG_EVENT[1])	Output	High	• Asserted high when a Type-C electrical connection is made at either the CC1 or CC2 pin; • Low when disconnected (opposite polarity of ATTACHED_L)
CC2_CONN (CABLE_ORIENTATION[1])	Output	High	• Asserted high when an upside-down port connection is made (at the CC2 pin); • Low when port is disconnected or a right-side up port connection is made
PD_SOURCE_SINK_DISC (PROVIDER_CONSUMER_HIGH_Z[1])	Output	N/A (Tri-State)	• Asserted high when USB PD contract negotiated as Source; • Low when USB PD contract negotiated as Sink; • High-Z when port is disconnected or no PD contract is active (tri-state capable with equal value external pullup and pulldown resistors)
FAULT_CONDITION_L	Output	Low	• Asserted low when an over-current fault condition occurs on any power path (PP_5V0, PP_HV, or PP_EXT) as a Source (USB Type-C or PD) or 5 V cannot be provided to VBUS on initial connection (short on contact); • High during normal operation
DP_OR_USB3_H	Output	High	• Asserted high when data connection is DisplayPort or USB3; • Low if neither data mode is active or port is disconnected (opposite polarity of DP_OR_USB3_L)
DP_MODE_SELECTION	Output	High	• Asserted high when data connection is DisplayPort (either 4-Lane mode or 2-Lane+USB3 mode); • Low when Type-C port is disconnected or DisplayPort mode is not active
SUPPLY_P5V	Output	High	• Asserted high when PP_5V0 path is enabled; • Low when PP_5V0 path is disabled (independent of other power paths)
SUPPLY_PHV	Output	High	• Asserted high when PP_HV path is enabled; • Low when PP_HV path is disabled (independent of other power paths)
SUPPLY_PHVE	Output	High	• Asserted high when PP_EXT path is enabled; • Low when PP_EXT path is disabled (independent of other power paths)
SUPPLY_PPCABLE	Output	High	• Asserted high when PP_CABLE path is enabled and supplying VCONN to either CC1 or CC2, depending on connection orientation; • Low when PP_CABLE path is disabled (independent of other power paths)
ATTACHED_L	Output	Low	• Asserted low when a Type-C electrical connection is made at either the CC1 or CC2 pin; • High when Type-C port is disconnected (opposite polarity of ATTACHED_H)
VBUS_DET	Output	High	• Asserted high when voltage is present on VBUS and Power Status (USB Type-C or PD) is Sink; • Low when port is disconnected and set low when connection is lost and VBUS approaches GND
P5V_OVERCURRENT	Output	Low	• Asserted low when over-current fault condition occurs on PP_5V0 path as a Source (USB Type-C or PD); • Low during normal operation
PWR_SINK_SOURCE	Output	High	• Asserted high when Power Status is Sink (USB Type-C or PD); • Low when Power Status is Source (Type-C or USB PD) or port disconnected
USB3_H	Output	Hi-Z	• High-Z when data connection requires USB3 (fixed open-drain configuration, requires pullup resistor for High state to operate correctly); • Low when USB3 data is not required or supported (for example, 4-Lane DisplayPort mode entered or USB3 support de-activated by firmware configuration)
USB2	Output	High	• Asserted high when data connection is USB2; • Low when Type-C port is disconnected or USB2 data is not required or supported
DPx2_MODE	Output	High	• Asserted high when 2-Lane DisplayPort and USB3 mode is supported and entered; • Low when Type-C port is disconnected, DisplayPort mode is not entered, or 4-Lane DisplayPort mode is entered

(1) GPIO Event names in parentheses are the original names and have been replaced by the new name for clarity and consistency. For example, events with opposite polarity may have similar names ending in 'H' or 'L' to indicate the GPIO's active state.

Table 13-1. GPIO Events (continued)

Event Name	I/O	Active State	Behavior
PD_SINK_SOURCE (CONSUMER_PROVIDER[(1)])	Output	High	• Asserted high when USB PD contract negotiated as Sink; • Low when USB PD contract negotiated as Source, no PD contract is active, or port is disconnected (opposite polarity of PD_SOURCE_SINK_DISC but not tri-state capable)
AMSEL	Output	N/A (Tri-State)	• High when 4-Lane DisplayPort mode; • Low when 2-Lane DisplayPort and USB3 mode is supported and entered; • High-Z when Type-C port is disconnected or USB3 data is required without DisplayPort mode entry (tri-state capable with equal value pullup and pulldown resistors)
SINK_LESS_12V	Output	High	• Asserted high when in an active PD contract and sinking less than 12 V; • Low when any other sink or source PD contract is active, no PD contract is active, or port is disconnected
SINK_12V	Output	High	• Asserted high when in an active PD contract and sinking 12 V; • Low when any other sink or source PD contract is active, no PD contract is active, or port is disconnected
SINK_MORE_12V	Output	High	• Asserted high when in an active PD contract and sinking more than 12 V; • Low when any other sink or source PD contract is active, no PD contract is active, or port is disconnected
USB3_L (HS_SEL0[(1)])	Output	High	• Asserted low when data connection is USB3; • High when USB3 data is not required or supported (opposite polarity of USB3_H)
UFP_DFP	Output	High	• Asserted high when data role is UFP or no connection at Type-C port; • Low when data role is DFP
DP_OR_USB3_L (HS_N_EN[(1)])	Output	Low	• Asserted low when data connection is DisplayPort or USB3; • High if neither data mode is active or port is disconnected (opposite polarity of DP_OR_USB3_H)
AC_DETECT	Input	High	• When signal is asserted high, CONSUMER_NO_AC is asserted low (indicating AC Adapter is present and external power is available) • If low when TPS65982 becomes a Sink (Type-C or PD), then CONSUMER_NO_AC is asserted high
CONSUMER_NO_AC	Output	High	• Asserted high when AC_DETECT is low as TPS65982 becomes a Sink; • Low when AC_DETECT is asserted high or when AC_DETECT is low and TPS65982 becomes a Source
CC1_CONN	Output	High	• Asserted high when a right-side up port connection is made (at the CC1 pin); • Low when port is disconnected or upside-down port connection is made
BARREL_JACK_DET	Input	High	**Upon Rising Edge (Barrel Jack detected):** • Clear Dead Battery Flag • Set Externally Powered = 1 • Swap to Source. **Upon Falling Edge (Barrel Jack removed):** • Set Externally Powered = 0 • Swap to Sink
PDIO_IN0 PDIO_IN1 PDIO_IN2 PDIO_IN3	Input	N/A	Input GPIO event for PDIO Alternate Mode (when supported by both port partners and mode is entered). A change in state of PDIO_INx will trigger a PDIO Alternate Mode message to be sent to the port partner. PDIO_OUTx will reflect the value of this signal after the PDIO Alternate Mode message is received by the port partner. These events do not have a pre-determined active state
PDIO_OUT0 PDIO_OUT1 PDIO_OUT2 PDIO_OUT3	Output	N/A	Output GPIO event for PDIO alternate mode. When PDIO Alternate Mode is supported by both port partners and entered, output follows GPIO pin mapped to PDIO_INx event on port partner.
SOURCE_PDO0_NEGOTIATED SOURCE_PDO1_NEGOTIATED SOURCE_PDO2_NEGOTIATED SOURCE_PDO3_NEGOTIATED	Output	High	• Asserted high when the corresponding Source PDO # (Power Delivery Object) becomes the active contract (after *Accept* PD message is sent but before *PS_Ready* PD message is sent); • Low when no PD contract is active or one of the other 3 Source PDO events is active (these 4 GPIOs are mutually exclusive and only 1 can be active at any time)

Table 13-1. GPIO Events (continued)

Event Name	I/O	Active State	Behavior
SOURCE_PDO_NEGOTIATED_TT_BIT0 SOURCE_PDO_NEGOTIATED_TT_BIT1 SOURCE_PDO_NEGOTIATED_TT_BIT2	Output	High	These 3 Events combine to form a 3-bit truth table to allow digital outputs indicating the active state of up to 7 PDOs. Bit 2 is the most-significant bit (MSB) and Bit 0 is the least significant bit (LSB) ● 000b → Indicates that the Type-C port is disconnected, a USB PD contract has not been negotiated or a 5-V PD contract is active ● 001b-110b → The decimal equivalent of PDOx active as configured in the Host Interface ● 111b → Un-used output state (cannot occur)
VBUS_UVP_QUICK_DETECT	Output	High	● Asserted high when TPS65982 is a Sink and VBUS rises above the UVP threshold of the active Type-C connection or PD contract; ● Low when port is disconnected and set low immediately after VBUS falls below UVP threshold of the active Type-C connection or PD contract
LOAD_APPCONFIG_SET_1 LOAD_APPCONFIG_SET_2 LOAD_APPCONFIG_SET_3	Input	—	**Upon Rising Edge:** ● App Config Set for GPIO = High will be loaded as the active configuration ● 1st 4CC Data and Command is written to selected CMDX register (optional) ● 2nd 4CC Data and Command (or PD Task) is written to selected CMDX register (optional) **Upon Falling Edge:** ● App Config Set for GPIO = Low will be loaded as the active configuration ● 1st 4CC Data and Command is written to selected CMDX register (optional) ● 2nd 4CC Data and Command (or PD Task) is written to selected CMDX register (optional)
USBEP_ENABLE_EVENT	Input	High	● When signal is asserted high, the Host Interface will be exposed through the USB2.0 Low Speed Endpoint. The TPS65982 Endpoint (EP) driver can be used to debug or to perform a FW update from a USB Host connected to the port with a Type-C cable. ● When signal is low the Host Interface cannot be accessed by the USB EP. In this case, the EP may be used to Billboard or the USB2.0 through path may be enabled based on the active data connection.
SINK_HVEXT	Output	High	● Asserted high when either the PP_HV or PP_EXT switch is enabled as the Sink path (Type-C or PD, after Soft Start is complete; ● Low when port is disconnected or any switch is enabled a Source (PP_5V0, PP_HV, or PP_EXT)
THERM_PROT_EXT_SW_IN	Input	—	Configurable polarity (active-high or active-low) ● When this signal transitions to the active state it indicates an overtemperature event for the external PP_EXT switch path and immediately opens the switch to stop the flow of current while keeping the connection or contract active. ● When the overtemperature event no longer exists (signal transitions to inactive state), the PP_EXT switch is closed and current flow resumes without requiring a new port connection

13.3 Load App Config Set GPIO Events

The GPIO events named *Load App Config Set X* are used to load a modified application-configuration (App Config) data set into the settings of the TPS65982 device to modify the behavior of the device based on the requirements of an application that change in real-time. The GPIO events LOAD_APPCONFIG_SET_X in Table 13-1 have the same capabilities as the Section 10.6 feature except they are triggered from the transition of an external digital-logic signal. The transition of the GPIO input signal from low-to-high is analogous to entering the User Alternate Mode and the transition of the GPIO input signal from high-to-low is analogous to exiting the User Alternate Mode. These events can be used to produce behavior similar to the static BARREL_JACK_DET event with more functionality or to reconfigure the TPS65982 device to meet other requirements of the application.

Revision History

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

Changes from A Revision (November 2015) to B Revision	Page
• Added the TPS65981 device to the list of supported products	5
• Added the description of the User Alternate Mode feature	32
• Added , deleted, and modified GPIO Events	39
• Added the *Load App Config Set GPIO Events* section	41

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

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