

**ABSTRACT**

This document provides guidance regarding the configurable parameters for the TPS2576x-Q1 and TPS2577x-Q1 USB power delivery (PD) controller device family. Use this guide when configuring the TPS2576x-Q1 and TPS2577x-Q1 devices using the TPS257XX-Q1-GUI configuration tool (referred to as *GUI* throughout this document) available at [TPS257XX-Q1-GUI](#).

**Table of Contents**

<b>1 Introduction</b> .....	<b>2</b>
<b>2 Getting Started</b> .....	<b>3</b>
2.1 Related Documents.....	3
2.2 Hardware.....	3
2.3 Software.....	3
2.4 GUI Installation and Launch.....	3
<b>3 Application Configuration Overview</b> .....	<b>4</b>
3.1 Creating a New Application Configuration.....	5
3.2 Loading a Saved Configuration.....	5
<b>4 Configuration Parameters</b> .....	<b>5</b>
4.1 System Power.....	5
4.2 Internal & External DCDC.....	6
4.3 VIN Engine On or Off.....	7
4.4 Thermal Foldback.....	8
4.5 USB PORT(S).....	10
4.6 GPIO Configuration.....	10
4.7 I2C Configuration.....	10
4.8 Device IDs.....	10
4.9 DisplayPort Alt Mode.....	10
<b>5 Application Configuration Download</b> .....	<b>11</b>
5.1 Firmware Download Procedure.....	11
5.2 Secure Firmware Update.....	17
5.3 Optional USB Driver Installation.....	18
5.4 Direct EEPROM Programming.....	19
5.5 SSH Key Generation.....	22
<b>6 Telemetry</b> .....	<b>23</b>
<b>A TPS257XX-Q1 GUI Feature - CUSTOM ID (Version Control)</b> .....	<b>27</b>
<b>B VIN Engine On or Off (TPS257xC-Q1)</b> .....	<b>29</b>
<b>Revision History</b> .....	<b>30</b>

**List of Figures**

Figure 3-1. Typical GUI Flow.....	4
Figure 4-1. System Power Module.....	6
Figure 4-2. Engine On/Off Controls and Power Mode Operation (TPS257xxD-Q1).....	7
Figure 4-3. Thermistor Implementation Options: NTC and PTC Circuit Diagrams.....	8
Figure 4-4. Thermal Foldback Phase vs Maximum Port Power.....	9
Figure 5-1. GUI Firmware Download Page.....	11
Figure 5-2. Loading Keys.....	12
Figure 5-3. Upload Keys Page.....	12
Figure 5-4. BUILD GUI FLASH IMAGE Page.....	13
Figure 5-5. Generating a Low Region Bin as a C Header File.....	14
Figure 5-6. USB ENDPOINT FLASH Page.....	15
Figure 5-7. USB Endpoint Connection Status.....	16

Figure 5-8. SECURE FLASH - Complete.....	16
Figure 5-9. EEPROM FLASH Page.....	17
Figure 5-10. Zadig USB Driver Installer Dialog Window.....	18
Figure 5-11. PC Device Connection Status.....	18
Figure 5-12. Connection Error Message.....	18
Figure 5-13. Git Bash Key Generation - <i>Engineering</i> .....	22
Figure 5-14. Git Bash Key Generation - <i>Production</i> .....	22
Figure 5-15. Generated Keys.....	22
Figure 6-1. TIVA USB Port Connection.....	23
Figure 6-2. TIVA USB Disconnected.....	24
Figure 6-3. TIVA USB Connected.....	24
Figure 6-4. GUI Communication Port Menu.....	24
Figure 6-5. TPS257xx-Q1 Telemetry Interface.....	25
Figure 6-6. Telemetry Port Configuration.....	25
Figure 6-7. Telemetry <i>Status</i> View.....	26
Figure A-1. CUSTOM ID Menu.....	27
Figure A-2. Binary Files with CUSTOM ID Enabled vs. Disabled.....	28
Figure A-3. CUSTOM ID Data Headers.....	28
Figure B-1. Engine On/Off Controls and Power Mode Operation (TPS257x2C-Q1).....	29

## List of Tables

Table 4-1. PRESET1 (47k $\Omega$ $R_{NTC}$ With $R_B = 3k\Omega$ ).....	8
Table 4-2. PRESET2 - $R_{PTC}$ (TMP61) With $R_B = 10 k\Omega$ .....	9
Table B-1. VIN Engine On or Off Parameter Name Changes (GUI v1.4.0 to v2.1.0).....	29

## Trademarks

DisplayPort™ is a trademark of Video Electronics Standards Association (VESA®) in the United States and other countries.

USB-C™ is a trademark of USB Implementers Forum (USB-IF).

Total Phase™ and Aardvark I2C/SPI™ are trademarks of Total Phase, Inc..

Notepad++™ is a trademark of Don Ho.

Chrome® is a registered trademark of Google LLC.

Firefox® is a registered trademark of Mozilla Foundation.

Microsoft® and Windows® are registered trademarks of Microsoft Corporation.

USB Type-C® is a registered trademark of USB Implementers Forum.

All trademarks are the property of their respective owners.

## 1 Introduction

The TPS2576x-Q1 and TPS2577x-Q1 device family is user-configurable and can support a wide range of operational modes to meet specific use-case requirements. This programmable flexibility enables the same hardware design to support different requirements without changes to the physical hardware. Hardware system design of the [TPS25763Q1EVM](#), [TPS25762DQ1EVM](#), and [TPS25772DQ1EVM](#) include connection of an EEPROM memory to the TPS2576x-Q1 or TPS2577x-Q1 device through an I2C interface.

Application-specific device configuration and programming can be accomplished using the TPS257XX-Q1-GUI. This graphical user interface tool is configurable based on user requirements and generates corresponding binary files that can be downloaded into the EEPROM through an I2C interface. The GUI inherently supports the ability to update hardware using a USB endpoint connection, a mode typically used post-production once hardware has been deployed.

---

### Note

Some graphics contained in this document can differ slightly from the current version of the GUI available from [dev.ti.com](http://dev.ti.com) because regular maintenance releases can occur ahead of updates to this guide.

---

## 2 Getting Started

### 2.1 Related Documents

- Texas Instruments, [TPS25763-Q1 Automotive Single USB Type-C® Power Delivery Controller with Buck-Boost Regulator and DisplayPort Alt Mode](#) data sheet
- Texas Instruments, [TPS25762-Q1 Automotive Single USB Type-C® Power Delivery Controller with Buck-Boost Regulator](#) data sheet
- Texas Instruments, [TPS25772-Q1 Automotive Dual USB Type-C® Power Delivery Controller with Buck-Boost Regulator](#) data sheet
- Texas Instruments, [TPS25763Q1EVM User's Guide](#)
- Texas Instruments, [TPS25762DQ1EVM User's Guide](#)
- Texas Instruments, [TPS25772DQ1EVM User's Guide](#)
- Texas Instruments, [TPS25772Q1EVM-CD-150 User's Guide](#)
- Texas Instruments, [TPS257xx-Q1 System Power Management](#) application note

### 2.2 Hardware

The TI TPS2576x-Q1 and TPS2577x-Q1 devices and corresponding hardware evaluation modules (EVMs) can be used with this GUI configuration tool:

- TPS25763Q1EVM
- TPS25762DQ1EVM
- TPS25772DQ1EVM
- TPS25772Q1EVM-CD-150

### 2.3 Software

The following software packages are required to use the GUI configuration tool:

- TPS257xx-Q1 Firmware Package (*Request access from the GUI tool page. Use link below.*)
- [TPS257xx-Q1 Configuration GUI](#)

The GUI software is available in the Gallery at [dev.ti.com](http://dev.ti.com). Run the software from a web browser using Google Chrome® browser, Firefox®, or Microsoft® Edge. The TI Cloud Agent must be installed as a browser extension to enable USB port connection to the EVM. When the application is launched, instructions appear for installing the TI Cloud Agent.

The GUI software can also be run natively on the PC. If this is desired, the GUI application and GUI Composer Runtime package is installed. To install, click on the downwards-facing arrow inside the application dashboard listed in the Gallery to access the links for download. After selecting the correct operating system, open the installer and follow the prompts to install the programs.

### 2.4 GUI Installation and Launch

Launch the GUI tool either through a web browser or locally as a desktop application following these steps:

- Navigate to [TPS257XX-Q1-GUI](#) using a supported web browser
- Click the *Evaluate in the cloud* button
- A new tab opens with the application launched

If the TI Cloud Agent is not already installed, instructions appear for installing the required software.

To download and install the application locally on the PC follow these steps:

- Navigate to [TPS257XX-Q1-GUI](#) using a supported web browser
- Click the *Download* button
- If required, select the application dashboard that has the correct tool displayed
- Click on the downwards-facing arrow on the bottom left side of the dashboard, and look towards the top set of links that appear. Select the operating system, and open the installer.
- Once the installer is open, follow the directions to install the GUI application

The TPS257XX-Q1-GUI application configuration tool provides users with the following capabilities:

- Generate application configuration settings in a binary file format
- Load configuration settings to the onboard EEPROM of the EVM
- Load configuration settings to an EVM (or custom hardware) along with security keys using the USB Type-C® port
- Save configuration settings in JSON format locally for future use

### 3 Application Configuration Overview

After launching the GUI and selecting *Quick Start*, the user is prompted to select a device variant using the "Choose a device" drop-down menu.

After selecting the device and clicking *SELECT* in the *Start Configuration* box, the *Advanced Configuration* page appears (the *Simple Configuration* menu is recommended for new users and can be selected from the left sidebar). Using these input fields, the application can be configured based on system requirements by selecting the desired sub-system tabs in the GUI.

Once the application is configured, the firmware is ready to be downloaded into the system hardware.

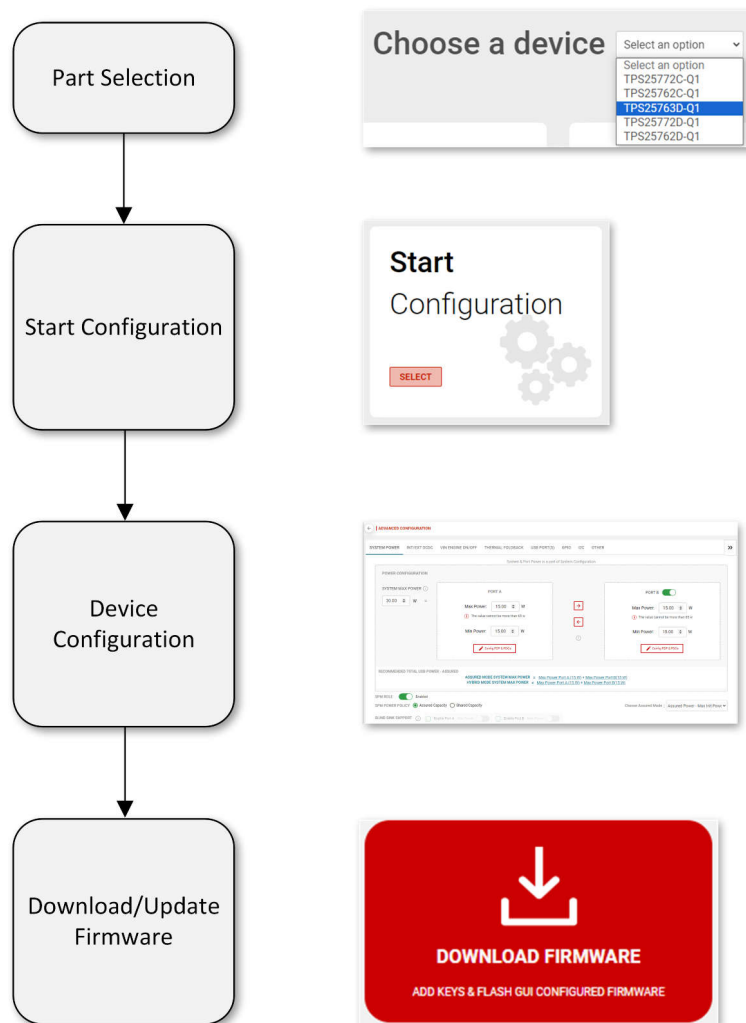


Figure 3-1. Typical GUI Flow

### 3.1 Creating a New Application Configuration

Starting from the initial GUI landing page, the following steps outline how to begin creating a new application configuration:

1. Click *Quick Start*
2. Click the dropdown menu next to *Choose a Device* to select device variant
3. Click *SELECT* in the *Start Configuration* box (the *Advanced Configuration* page appears)
4. Configure each module accordingly per system requirements, navigate using the the desired sub-system tabs
5. After configuration is complete, click the *SAVE CONFIGURATION* button to save the setup as a JSON file to reload the settings in a future GUI session

### 3.2 Loading a Saved Configuration

The TPS257XX-Q1-GUI supports the ability to load a previously saved configuration file. The file is saved as a .JSON file and can be loaded as described in the following list, starting from the initial GUI landing page:

1. Click *Quick Start*.
2. Click the *OPEN CONFIGURATION* button (upper right of the GUI screen).
3. Two options are presented:
  - a. *SAVE OLD CONFIGURATION*: this saves the current application configuration settings (if any) to a new JSON file
  - b. *LOAD NEW CONFIGURATION*: This opens a navigation window to select an existing JSON file
4. Click *LOAD NEW CONFIGURATION* and navigate to the desired JSON file. If the selected JSON file was generated using a previous version of the GUI, previous settings may not update as expected (click *Help*, located at the top of the GUI next to *Options*, then *Release Notes* for information on GUI versions and compatible firmware).
5. Once loaded, the *Advanced Configuration* page appears containing the restored application configuration settings.
6. Make any desired changes accordingly per system requirements, navigate using the button string graphic and the *NEXT* button in the GUI.
7. After configuration is complete, click the *SAVE CONFIGURATION* button to save the setup as a new JSON file.

## 4 Configuration Parameters

### 4.1 System Power

Total system USB power and port power parameters are configurable in this page. PDPs and PDOs are auto-generated based on the power settings defined, but can be modified by clicking the *Config PDP & PDOs* button. When using a dual port controller device, the Port B configuration settings are available. See [Figure 4-1](#) below.

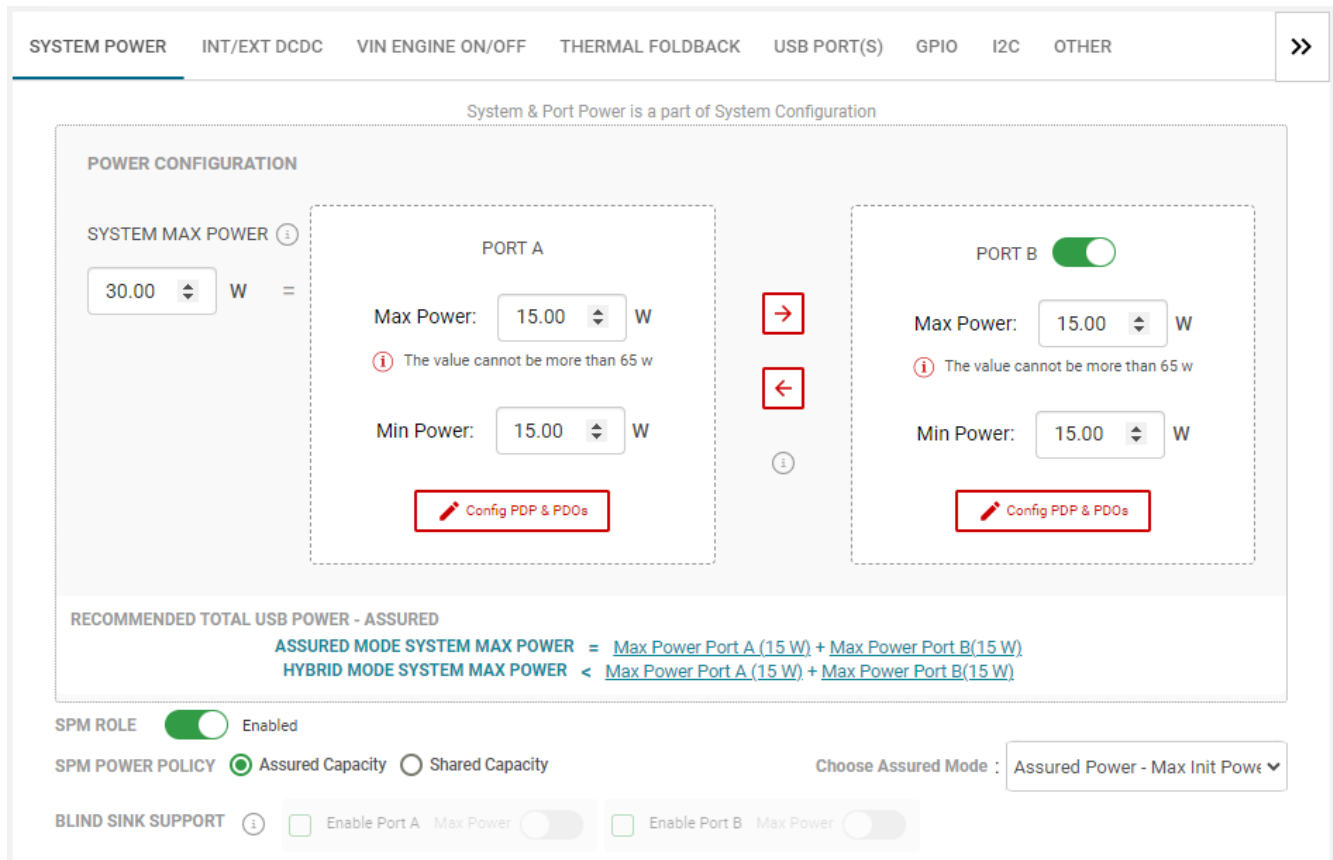


Figure 4-1. System Power Module

#### 4.1.1 System Power Management (SPM)

The System Power Management (SPM) parameters can also be configured in this page. Three SPM policies available:

- Assured Capacity - Source port has a fixed allocated amount of power and able to deliver its max power capacity independent of the second port's status
- Shared Capacity - Source port has a dynamically allocated amount of power and able to deliver up to its max power capacity depending on the remaining available system power that is shared with the second port.
- Hybrid Mode - An optional power policy that behaves as a hybrid of the Assured Capacity Policy and Shared Capacity Policy. It provides the benefit of the Assured Capacity Policy by initially advertising the Port Max Power when one Sink is connected, while allowing equal distribution of the total system USB power, similar to the Shared Capacity Policy, when two Sinks are connected.

For more information regarding SPM, see the [TPS257xx-Q1 System Power Management](#) application note.

#### 4.2 Internal & External DCDC

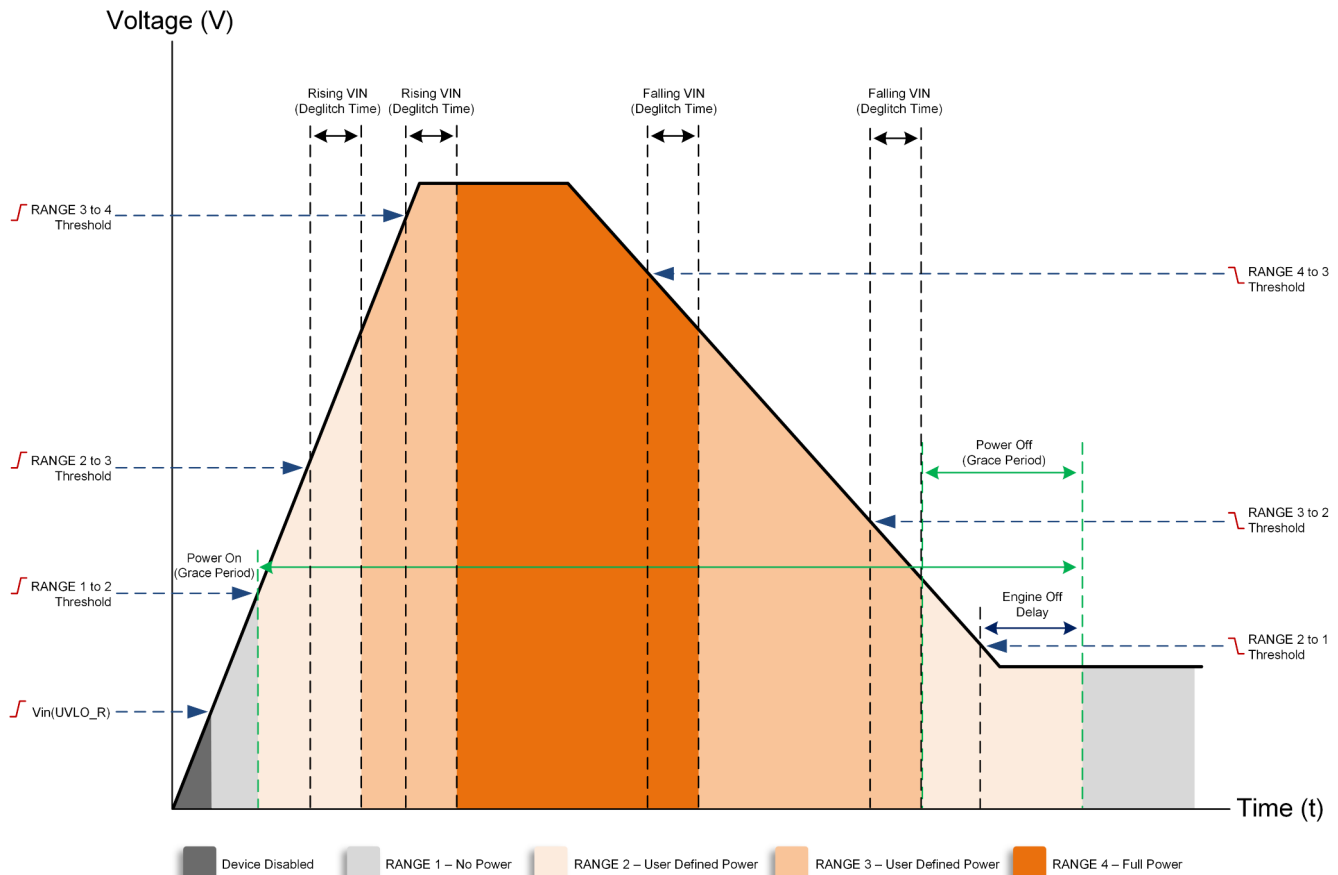
Internal DCDC parameters are configurable in this page. The inductor value chosen must match actual inductor values configured in the target system hardware.

For two-port device selection, external DCDC parameters are also configurable in this page for either the TPS55288-Q1 or TPS55289-Q1 buck-boost converters.

### 4.3 VIN Engine On or Off

VIN Engine On or Off power level parameters are configurable in this page.

The TPS2576x-Q1 and TPS257x-Q1 family's SPM supports autonomous power transition dependent upon the  $V_{IN}$  (for example,  $V_{BATT}$ ) voltage level per the diagram shown in Figure 4-2.



**Figure 4-2. Engine On/Off Controls and Power Mode Operation (TPS257xxD-Q1)**

#### Note

Figure 4-2 shows the VIN Engine On/Off VIN threshold and timer parameters of TPS257xxD-Q1 devices. The naming convention of the TPS257x2C-Q1 devices' VIN Engine On/Off parameters have changed moving from GUI v1.4.0 to v2.1.0 (see Appendix B).

The Engine On/Off voltage thresholds must be configured in sequential order (highest to lowest): RANGE 3 to 4 (rising) > RANGE 4 to 3 (falling) > RANGE 2 to 3 (rising) > RANGE 3 to 2 (falling) > RANGE 1 to 2 (rising) > RANGE 2 to 1 (falling).

Maximum power for the low-power range is a configurable total port power output. Program maximum power relative to total VBUS power.

Dual port example:

- Total USB VBUS Power = 100W
- VBUS Port A maximum power = 60W and VBUS Port B maximum power = 60W
- VBUS Port A minimum power = 15W and VBUS Port B minimum power = 15W
- Maximum power for low power = 30W:
  - Once the  $V_{IN}$  level falls below the RANGE 4 to 3 (falling) setpoint, the total USB VBUS power is reduced to 30W total for all ports.



VIN Engine On or Off is one of the SPM engine's power foldback policies. For more information regarding SPM, see the [TPS257xx-Q1 System Power Management](#).

**Note**

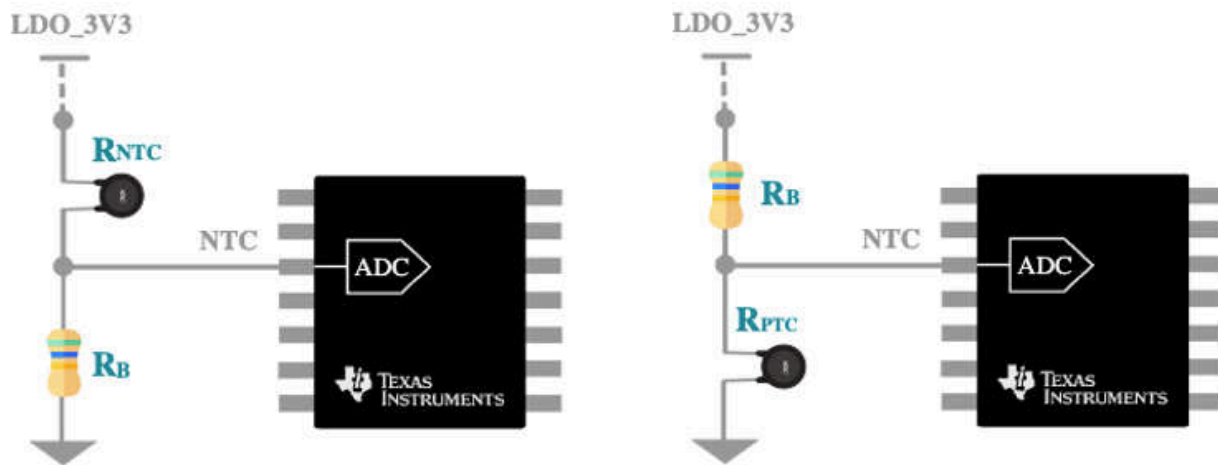
If the maximum power for low power is less than the total minimum port power (the sum of the minimum port power for ports) ports are disabled upon entry into the low-power range.

**4.4 Thermal Foldback**

Thermal Foldback related parameters can be configured in this page. Thermal foldback allows the system to reduce power when defined device thermal thresholds are reached, helping to automatically manage power versus temperature system requirements.

Thermal Foldback is one of the SPM engine's power foldback policies. For more information regarding SPM, see the [TPS257xx-Q1 System Power Management](#).

Thermal phase is monitored via the NTC device pin (GPIO5) with a thermistor resistor network as described in [Figure 4-3](#). Only positive voltage slope (from lowest to highest temperature) is supported with NTC or PTC thermistor configurations.



**Figure 4-3. Thermistor Implementation Options: NTC and PTC Circuit Diagrams**

PHASE 1 is the first thermal foldback phase followed by PHASE 2 and finally PHASE 3. Voltage thresholds must be configured from lowest to highest temperature starting from PHASE 1. Input voltage levels can be calculated based upon the type of thermistor and external resistor values used in the target system hardware design. See the thermistor data sheet for voltage input calculations applicable for each phase as required for the given application operation. When enabled in the GUI, one of two provided preset values can be selected:

PRESET1 is based upon an NTC with  $R_B = 3\text{ k}\Omega$  and  $R_{NTC} = 47\text{ k}\Omega$ .

**Table 4-1. PRESET1 (47kΩ R<sub>NTC</sub> With R<sub>B</sub> = 3kΩ)**

Voltage Threshold	Voltage/Temp
Phase 1V <sub>THF</sub>	0.392V (40°C)
Phase 1V <sub>THR</sub>	0.574V (50°C)
Phase 2V <sub>THF</sub>	0.686V (55°C)
Phase 2V <sub>THR</sub>	0.938V (65°C)
Phase 3V <sub>THF</sub>	1.078V (70°C)
Phase 3V <sub>THR</sub>	1.386V (80°C)

PRESET2 is based upon a PTC (TMP61) with  $R_B = 10\text{ k}\Omega$ .

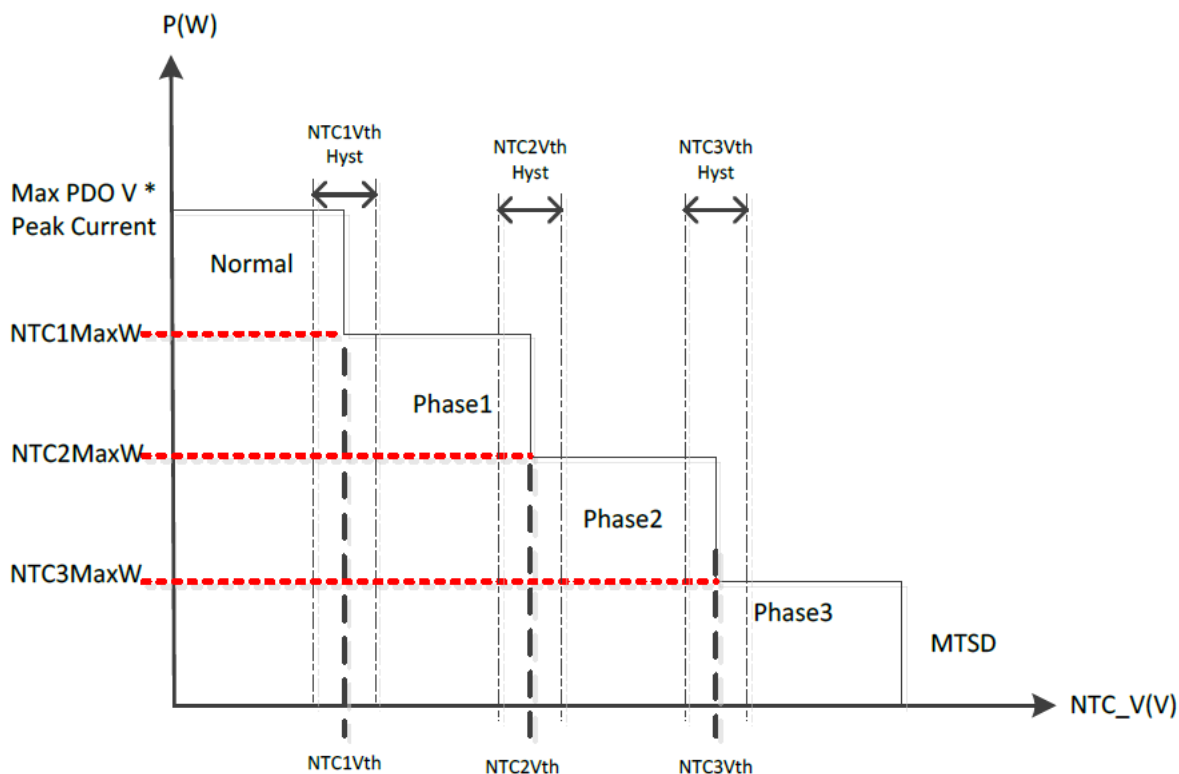


**Table 4-2. PRESET2 - R<sub>PTC</sub>(TMP61) With R<sub>B</sub> = 10 kΩ**

Voltage Threshold	Voltage/Temp
Phase 1V <sub>TH</sub> F	1.82V (40°C)
Phase 1V <sub>TH</sub> R	1.876V (50°C)
Phase 2V <sub>TH</sub> F	1.904V (55°C)
Phase 2V <sub>TH</sub> R	1.96V (65°C)
Phase 3V <sub>TH</sub> F	1.988V (70°C)
Phase 3V <sub>TH</sub> R	2.03V (80°C)

**Note**

The CUSTOM option is available to allow up to 6 PHASES and for user-defined setpoints based on voltage and temperature relationships for other thermistor resistor network designs.



**Figure 4-4. Thermal Foldback Phase vs Maximum Port Power**

Maximum power for each phase is a configurable total port power output for each thermal foldback phase. Program the maximum power relative to the total VBUS power. Note that if maximum power for a given thermal foldback phase is less than the user-defined total minimum port power (sum of minimum port power in dual port systems), the ports are disabled upon entry into the corresponding thermal foldback phase.

Dual port example:

- Total USB VBUS Power = 100W
- VBUS Port A maximum power = 60W and VBUS Port B maximum power = 60W
- VBUS Port A minimum power = 15W and VBUS Port B minimum power = 15W
- Maximum power for PHASE1 = 60W
  - Once this phase is entered, the total USB VBUS power is reduced to 60W total for ALL ports
- Maximum power for PHASE2 = 30W
  - Once this phase is entered, total USB VBUS power is reduced to 30W total for ALL ports

- Maximum power for PHASE3 = 15W
  - Once this phase is entered, the total USB VBUS power is reduced to 15W total for ALL ports, which is less than the sum of the minimum power of each port (15W + 15W = 30W). In this scenario, the maximum foldback power configured can meet the minimum 15-W requirement of a single port resulting in one port being disabled.

#### 4.5 USB PORT(S)

General parameters such as current limit delay, VBUS undervoltage protection, programmable power supply enable or disable, and USB legacy fast charge mode enable or disable are configurable in this page.

TPS25763-Q1 can be configured as Dual Role Power (DRP) or Source Only. DRP ensures the port can connect to source-only peripherals. If TPS25763D-Q1 is DRP and connects as sink, a 5V, 0A contract will be requested and the system should not sink significant current. DRP also ensures compliance to the DisplayPort™ over USB-C™ specification. Source Only implementation simplifies the hardware design and should be acceptable in the majority of applications.

Some USB peripherals require to be in the Downstream Facing Port (DFP) role to operate. TPS25763-Q1 can be configured to initiate a role swap to Upstream Facing Port (UFP) or handle this request when given from the port partner.

#### 4.6 GPIO Configuration

GPIO parameters are configurable in this page. Fixed values for each GPIO are shown for user information purposes. If GPIO is a fixed value, the pin configuration is not re-configurable. For example, in TPS25772-Q1, GPIO 0 and GPIO 1 are fixed as PB\_CC1 and PB\_CC2 for Port B USB Type-C PD communication purposes.

TPS25763-Q1 can control an external DisplayPort MUX via GPIO or I2C. This option is set in the “ALT MODE” pane. The GPIOs will automatically be assigned based on this selection. Channel 0 must be set to HPD for either case. For GPIO based control, one IO must be set as Output Event “Cable\_Orientation\_Event\_Port1” to tell the MUX which way the reversible Type-C connector was plugged in, one as “USB3\_Event\_Port1,” to tell the MUX the to configure lanes for USB SS communication, and one as “DP\_Mode\_Selection\_Event\_Port1” to tell the MUX to configure lanes for DisplayPort.

When I2C control is used for the MUX, TPS25763-Q1 uses I2C1 to communicate with the MUX, freeing up IOs for other functions.

#### 4.7 I2C Configuration

I2C parameters are configurable in this page. Fixed parameters are also shown for user information purposes.

TVSP CONFIG preference is determined by I2C target address setting. The target system hardware must have a matching TVSP configuration, otherwise it is possible the device does not function.

#### 4.8 Device IDs

Device IDs can be configured in this page. For test purposes, the fields can be left at their default values. Vendor IDs, XIDs, and Product IDs are unique IDs assigned by the USB Implementers Forum. See [usb.org](http://usb.org) for more information about obtaining IDs.

#### 4.9 DisplayPort Alt Mode

The “ALT MODE” pane allows further DisplayPort configuration of TPS25763-Q1. If MUX control is set to I2C, the MUX address and signal equalization settings must be set according to the system characteristics. These values are programmed to the TUSB564/1064-Q1 register space upon boot.

## 5 Application Configuration Download

Once the desired configuration is completed in the GUI, the binary file used to program the TPS257xx-Q1 device can be generated. This binary file is then downloaded to the hardware by one of two transport mechanisms:

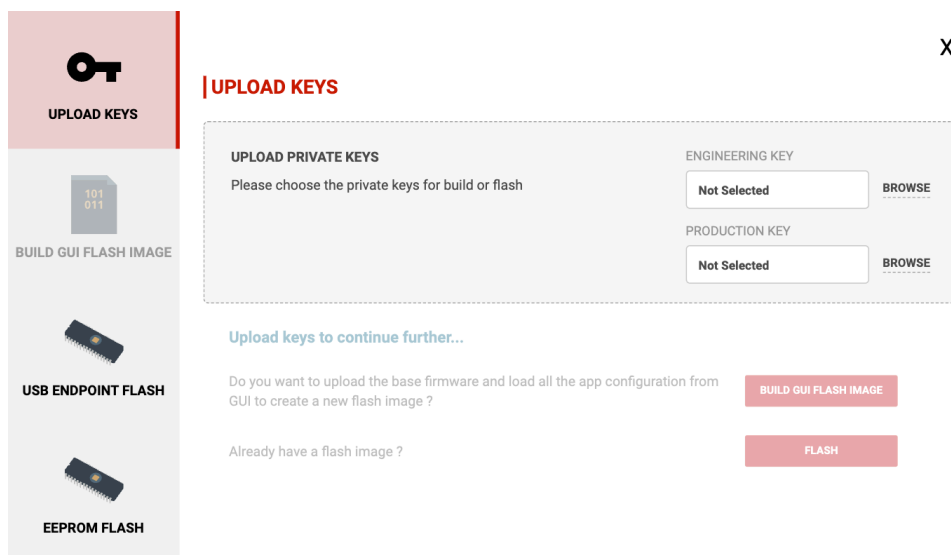
1. Direct programming of the binary image into onboard EEPROM using I2C:
  - a. This mode is the default method used during development
  - b. Programming using the built-in connections on the TPS257XX-Q1 EVM using the onboard TIVA MCU or directly using an external programming tool such as the Total Phase Aardvark I2C/SPI Host Adapter (see [Section 5.4](#))
  - c. This method of programming is required to load the production-ready binary image into a blank EEPROM device during production
2. Secure programming of the binary image into onboard EEPROM using USB:
  - a. This mode is accessed using the TPS257xx-Q1-GUI for post-production field updates of custom hardware (can also be used during development).
  - b. The TPS257xx-Q1 device must be powered up into *Firmware Update* mode (FWUP), configuring Port A as a USB endpoint connection
  - c. The binary image is programmed into EEPROM by the TPS257xx-Q1 device after verification of matching keys between the binary image and previously-programmed EEPROM image at production

Version control can be embedded manually to a GUI-generated binary file to uniquely identify and track changes. For more details, See [Appendix A](#).

The following sections describe the methods used to load the custom application configuration onto the hardware.

### 5.1 Firmware Download Procedure

Firmware download is supported by the GUI and accessed when clicking the *Download Firmware* button from the *Simple* or *Advanced* configuration pages.



**Figure 5-1. GUI Firmware Download Page**

From this page, the user can load TI-provided development keys or private keys. The binary image can then be generated and saved or directly loaded into hardware using the I2C or USB endpoint methods detailed in [Section 5.1.1.1](#) and [Section 5.1.1.2](#).

### 5.1.1 Key Upload and Binary File Generation

To create the configuration binary file, first select the private keys for use in the GUI. Default keys are provided in the TPS257xx-Q1 Firmware Package and can be used for evaluation purposes. Customer-specific private keys can also be selected (this is recommended before building and deploying a final production binary image). Upload the desired secure private keys in the *UPLOAD PRIVATE KEYS* window.

#### Note

TI provides a set of keys for development purposes only. The end user is responsible for creating a secure set of private keys to generate binaries for final production.

### UPLOAD KEYS




Figure 5-2. Loading Keys

After uploading the keys, the user has the following options:

1. Select *BUILD GUI FLASH IMAGE* to build and save a binary file using the current configuration as defined within the GUI
2. Select *FLASH* which skips the build step and allows the user to select an existing binary file

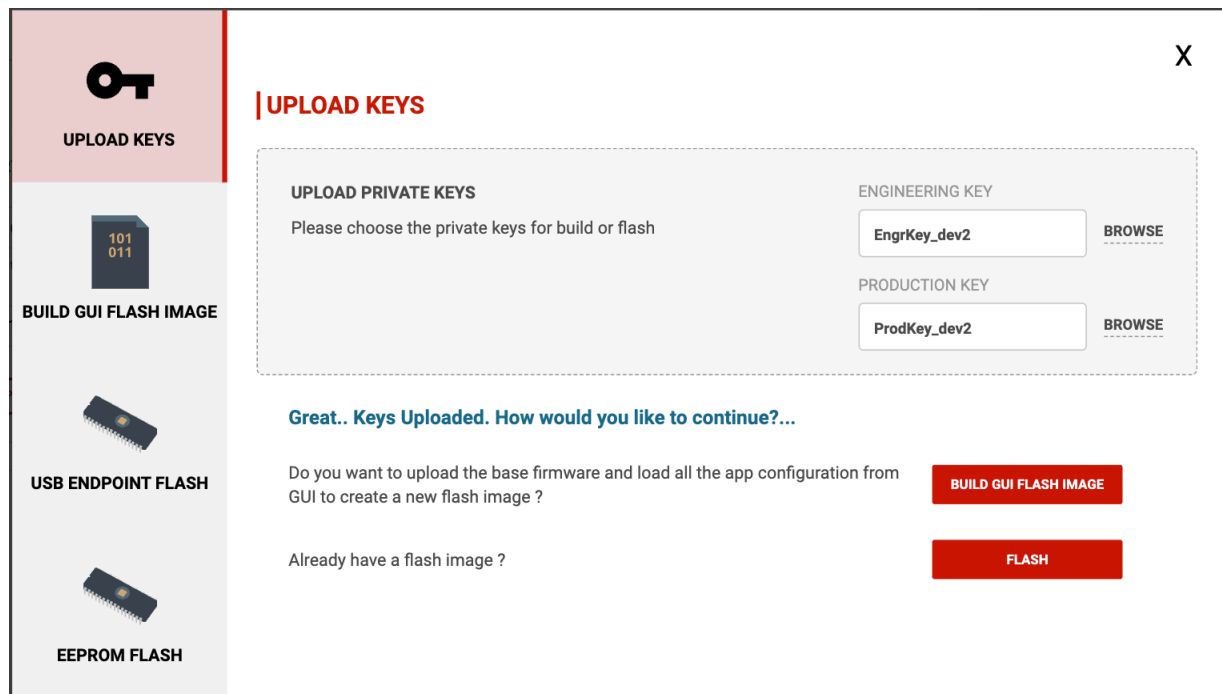


Figure 5-3. Upload Keys Page

If the desired binary file is already available, the file can be uploaded using one of two interface methods as described in [Section 5.1.1.1](#) or [Section 5.1.1.2](#) by clicking the *FLASH* button. If a new binary file needs to be built, click the *BUILD GUI FLASH IMAGE* button and follow the next steps described.

In the *BUILD GUI FLASH IMAGE* window, the user has the option to select the base firmware image binary - this file is provided from TI when the firmware release package is installed. A valid base firmware image must be selected or the device does not operate correctly (click *Help*, located at the top of the GUI next to *Options*, then *Release Notes* for information on GUI versions and compatible firmware). Next, select *Production Key* when uploading newer application configuration versions as needed. Alternatively, select *Engineering Key* when reverting to previous versions.

Figure 5-4 shows the selection of the engineering or production key.

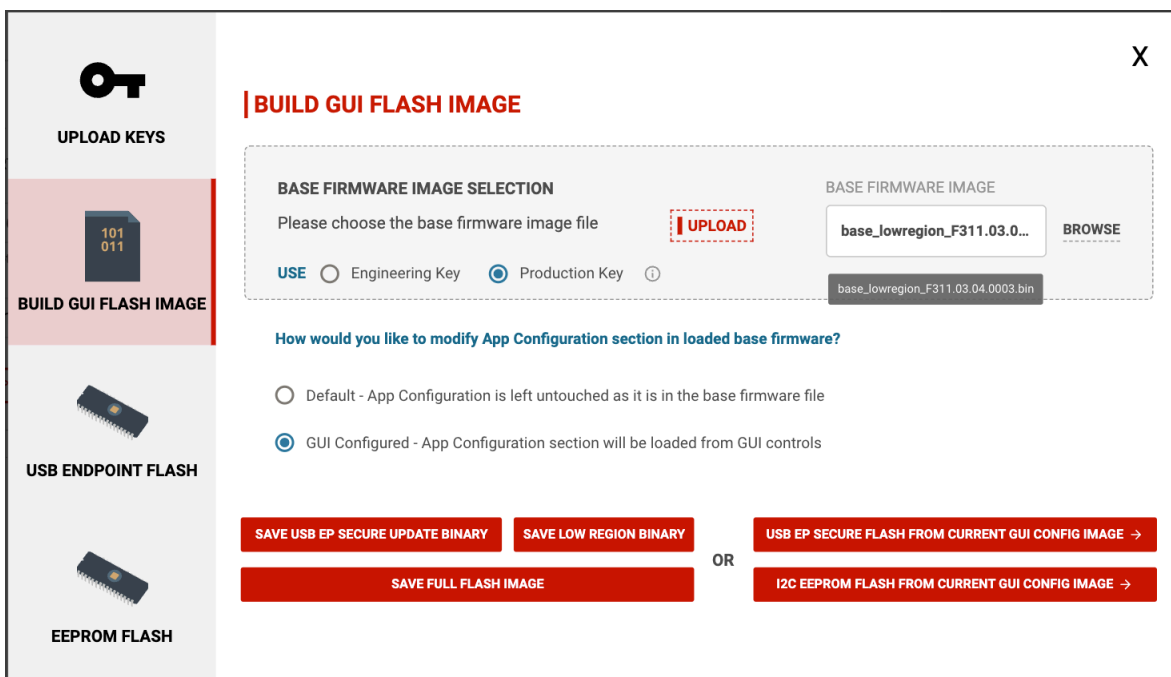


Figure 5-4. BUILD GUI FLASH IMAGE Page

After loading the base firmware image, the application configuration binary can be saved in three formats:

1. The first format option is *SAVE FULL FLASH IMAGE*.



Save in this format for update using I2C directly to the EEPROM. This is the method used to program a blank EEPROM and can be done using the TIVA USB interface designed on the EVM from TI or directly using an EEPROM programmer such as the Total Phase Aardvark programming tool. In both cases the binary image is loaded into the EEPROM device using an I2C interface.

**Note**

This method is required to create the initial image used to program a blank EEPROM device.

2. The second format option is *SAVE USB EP SECURE UPDATE BINARY*.

**SAVE USB EP SECURE UPDATE BINARY**

Save in this format for update via the USB Endpoint connection method. In this mode the PC is connected to the USB-C charger port ("Port A" for dual-port implementations) with the corresponding boot setting on power up or reset as described in the TPS25762/72-Q1 EVM User's Guide. The binary image is transferred over USB to the TPS257xx-Q1 device and then into the EEPROM device using I2C. Only images built with a valid private key that matches the image previously programmed into the EEPROM transfer successfully.

Once the desired binary format is saved, the file can then be selected from the *USB ENDPOINT FLASH* or *EEPROM FLASH* pages and programmed to the EEPROM in hardware.

**Note**

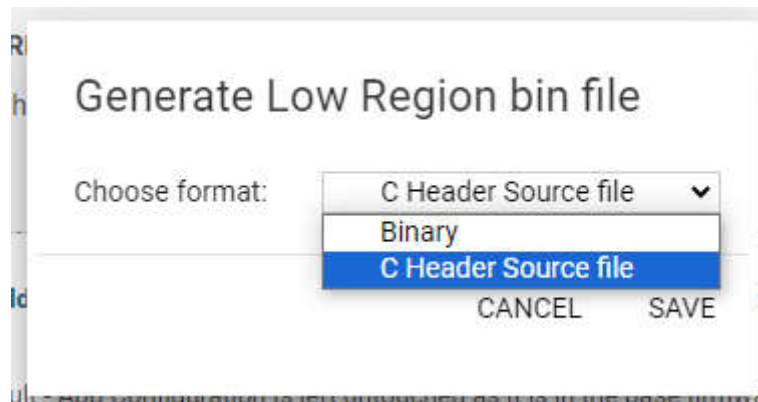
It is also possible to directly create the binary image and transition to the *USB ENDPOINT FLASH* or *EEPROM FLASH* pages to load the binary *without* saving the generated file. Loading without saving can be done for each format by selecting the *USB EP SECURE FLASH FROM CURRENT GUI CONFIG IMAGE* or *I2C EEPROM FLASH FROM CURRENT GUI CONFIG IMAGE* buttons. Selecting either button automatically creates the image and selects the image for download in the respective *FLASH* page without saving the image locally.

[Section 5.1.1.1](#) and [Section 5.1.1.2](#) describe the process to load the binary image into hardware using USB endpoint or I2C.

A third format option is *SAVE LOW REGION BINARY*.

**SAVE LOW REGION BINARY**

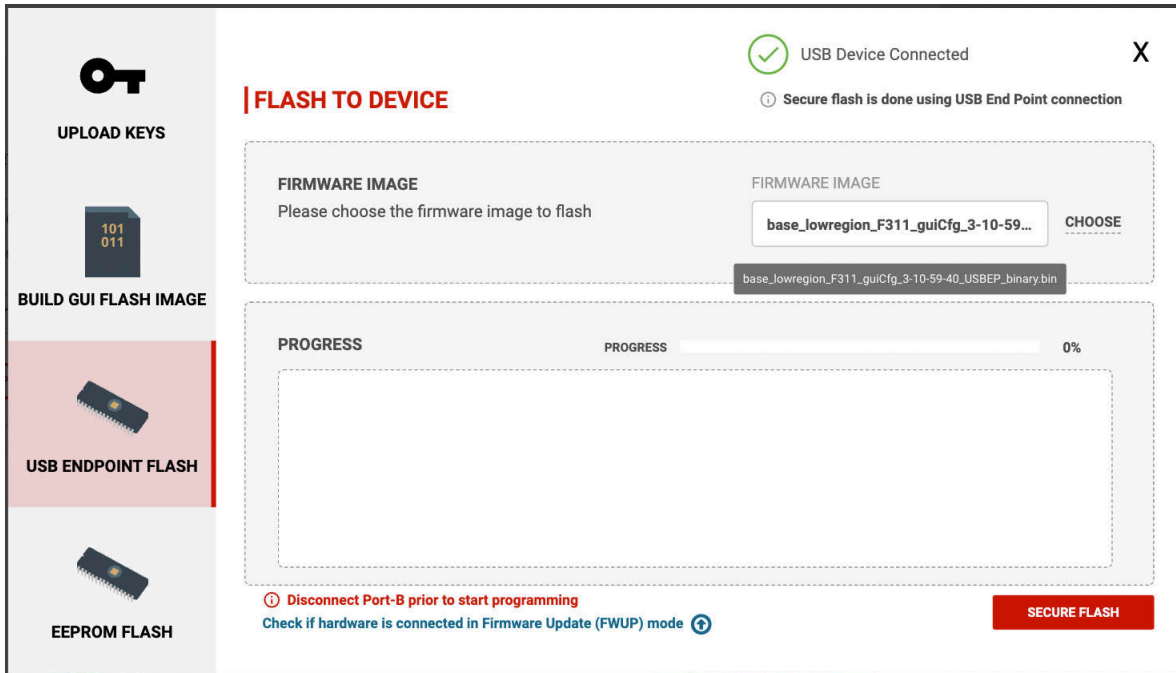
Selecting this option opens a menu to save the application configuration as a low region binary or a C header file (see [Figure 5-5](#)) that is intended to be used for the device's MCU/HUB Boot Mode. For more information regarding this boot mode, see the [TPS257x-Q1/77x-Q1 Firmware Update with a Host](#).



**Figure 5-5. Generating a Low Region Bin as a C Header File**

### 5.1.1.1 Firmware Update: USB Endpoint

In the *USB ENDPOINT FLASH* window, the selected USB EP binary image is loaded by USB connection to the TPS257xx-Q1 device to program the EEPROM.



**Figure 5-6. USB ENDPOINT FLASH Page**

Prior to initiating the download by selecting *SECURE FLASH*, verify the TPS257xx-Q1 device is booted into *Firmware Update* mode (for example, *FWUP* mode). This can be accomplished by making sure the TVSP pin is configured in hardware for FWUP operation. To program the EEPROM integrated on the TPS257xx-Q1 EVM, populate the TVSP jumper accordingly and power the device. Once in FWUP mode, connect the PC to the device using the USB Type-C charging port.

**Note**

In a dual-port system, use *Port A* for the USB endpoint connection and make sure the other port is not connected during the firmware update process.



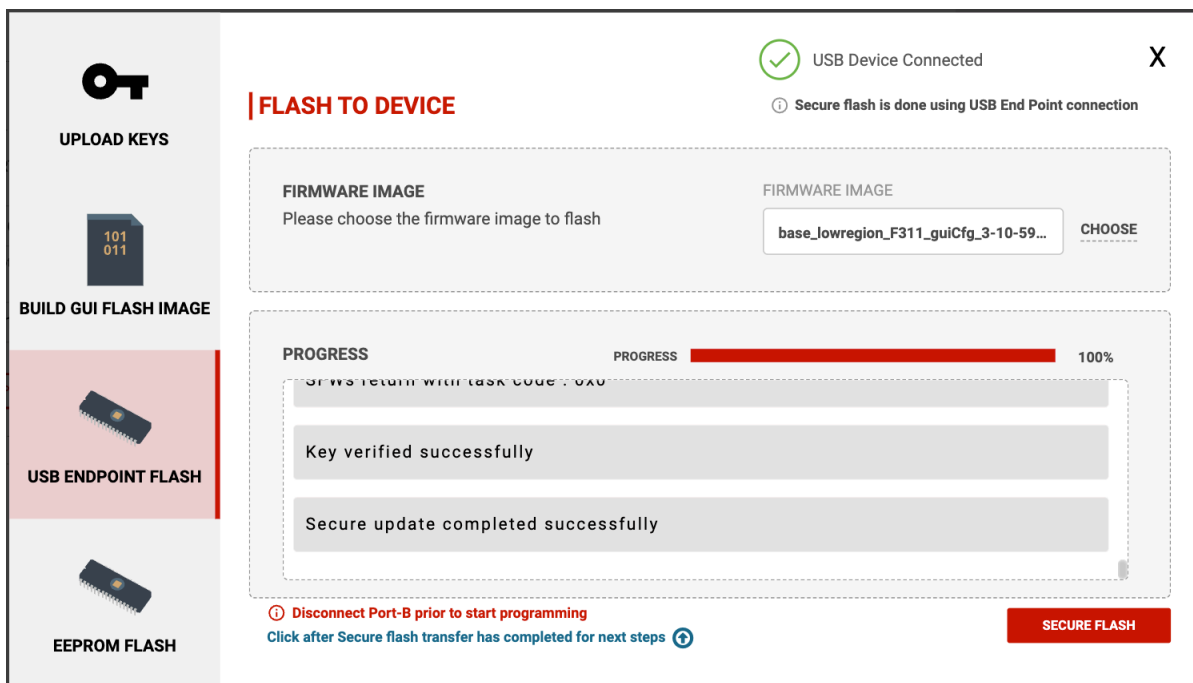
Once connected, the device appears under device manager as *TPS DMC Family* and the GUI indicates *USB Device Connected* status upon a successful connection. Figure 5-7 shows the successful connection image.



**Figure 5-7. USB Endpoint Connection Status**

Once a proper connection is established with the target device in FWUP mode, start the update using the *SECURE FLASH* button. Progress is shown in the GUI until completion.

Once the firmware update is complete, the message *Secure update completed successfully* is returned and the cable can be disconnected.



**Figure 5-8. SECURE FLASH - Complete**

Release the TPS257xx-Q1 device from FWUP mode by changing the TVSP setting followed by a power cycle or device reset.

### 5.1.1.2 Firmware Update: I2C

In the *EEPROM FLASH* window (see Figure 5-9), the selected low region or full flash binary image is programmed into the EEPROM by direct I2C connection.

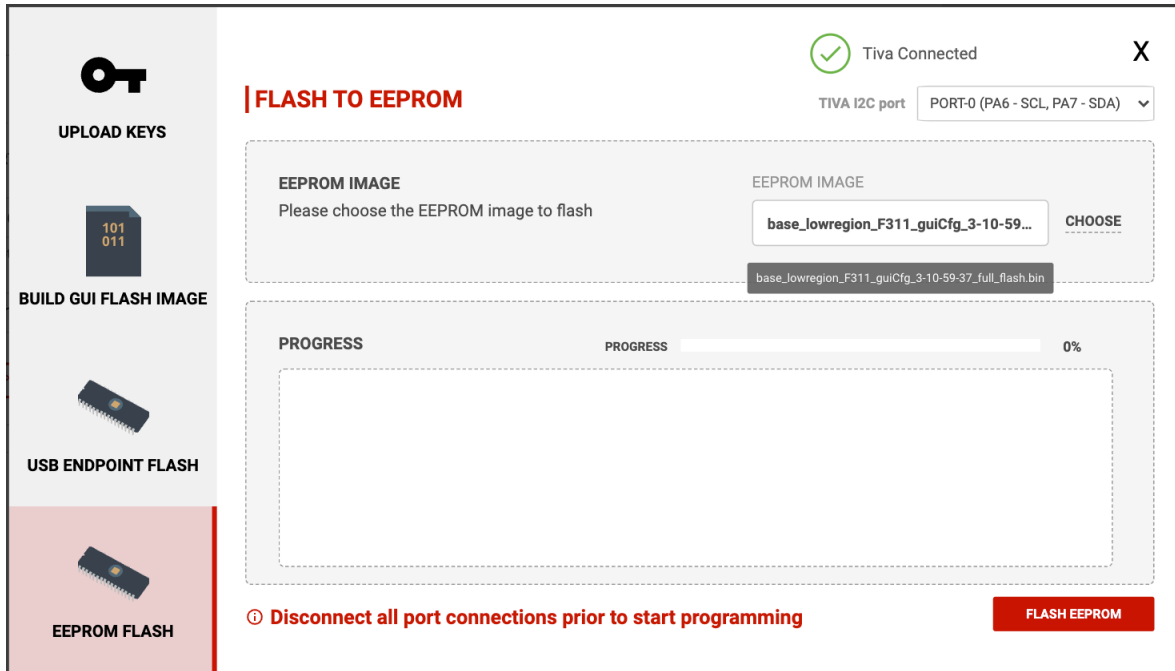


Figure 5-9. EEPROM FLASH Page

When evaluating using the EVM from TI, using this method programs the system EEPROM through the USB connection to the onboard TIVA microcontroller. Ensure that the correct serial port is configured if the TIVA USB connection cannot be established (see Figure 6-4). In addition, any I2C-capable programming tool can be used to program a saved binary file generated by the TPS257XX-Q1-GUI tool. This is the method used in custom-built hardware when programming the EEPROM via I2C.

Once the I2C connection is established and the desired binary file is chosen in the *EEPROM IMAGE* field, selecting the *FLASH EEPROM* button downloads the binary image. Progress is displayed in the GUI until the update is completed.

#### Note

Disconnect all USB-C devices from ports prior to programming the EEPROM. All ports must remain disconnected during programming.

## 5.2 Secure Firmware Update

TPS257xx-Q1 devices support secure firmware updates over USB using the previously-described USB endpoint method. The secure firmware update makes sure that only the binary image signed with the correct set of keys can be used to reprogram the EEPROM device. The GUI and firmware use the SHA-256 algorithm to hash and sign the binary image with RSA-PSS generated keys. A set of private and public key pairs can be generated using RSA-PSS for end-customer (OEM) development and production purposes.

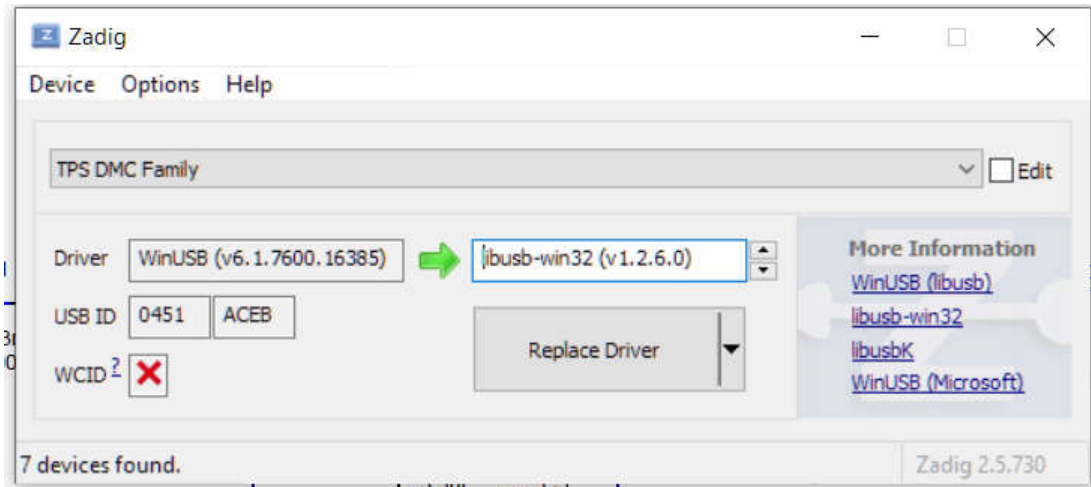
During end-product production, initial programming of the EEPROM must be performed using I2C with the *Full Flash* binary image built with the desired set of keys. Field updates can then be done using the TPS257XX-Q1-GUI tool over a USB connection to the primary charging port with the *USB EP* binary image that was built with the same set of keys. The customer-specific private keys must be kept secure and need to be loaded at the time of USB field update to reprogram the device; otherwise, the secure firmware update process does not complete.

### 5.3 Optional USB Driver Installation

The computer OS system must be able to connect as a USB endpoint to use the USB endpoint firmware update capability. Install a generic libusb-win32 USB device driver to connect to the USB endpoint of the TPS257xx-Q1 device. This connection can be accomplished by downloading a driver installation tool such as [Zadig](#).

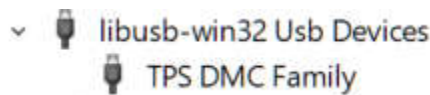
Complete the following after installing Zadig:

1. Connect Port A of the device to be programmed to a PC. Make sure the TVSP pin is configured for FWUP mode.
2. Download then run the Zadig executable, then choose the libusb-win32 option (see [Figure 5-10](#)).



**Figure 5-10. Zadig USB Driver Installer Dialog Window**

Once enumerated correctly, the device end-point enumerates in the device manager as [Figure 5-11](#) shows.



**Figure 5-11. PC Device Connection Status**

Some PC systems operate with the *WINUSB* vs *LIBUSB* driver. If the *LIBUSB error* is received upon USB EP firmware update connection, update the PC driver with the WINUSB driver, then reboot the computer system. The WINUSB driver can be installed with the Zadig tool by choosing the winusb option instead of libusb.

Error Status: Error: LIBUSB\_ERROR\_NOT\_FOUND

**Figure 5-12. Connection Error Message**

### 5.4 Direct EEPROM Programming

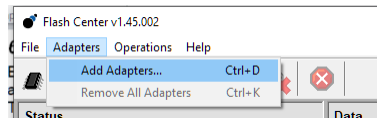
The EEPROM device can be programmed with the *Full Flash* binary file generated using the GUI with a valid set of keys as detailed in Section 5.1.1. While the EEPROM on the TPS257xx-Q1 EVM can be programmed using the onboard TIVA MCU, it is also possible to use a standard EEPROM programmer interfaced with the proper pins on the EVM. Use of an EEPROM programmer can also be used when programming the EEPROM of a custom hardware design. This is required for the initial programming of a blank EEPROM.

For example, EEPROM programming can be done using the Total Phase™ Aardvark I2C/SPI™ Host Adapter. The following steps are required to use the programmer:

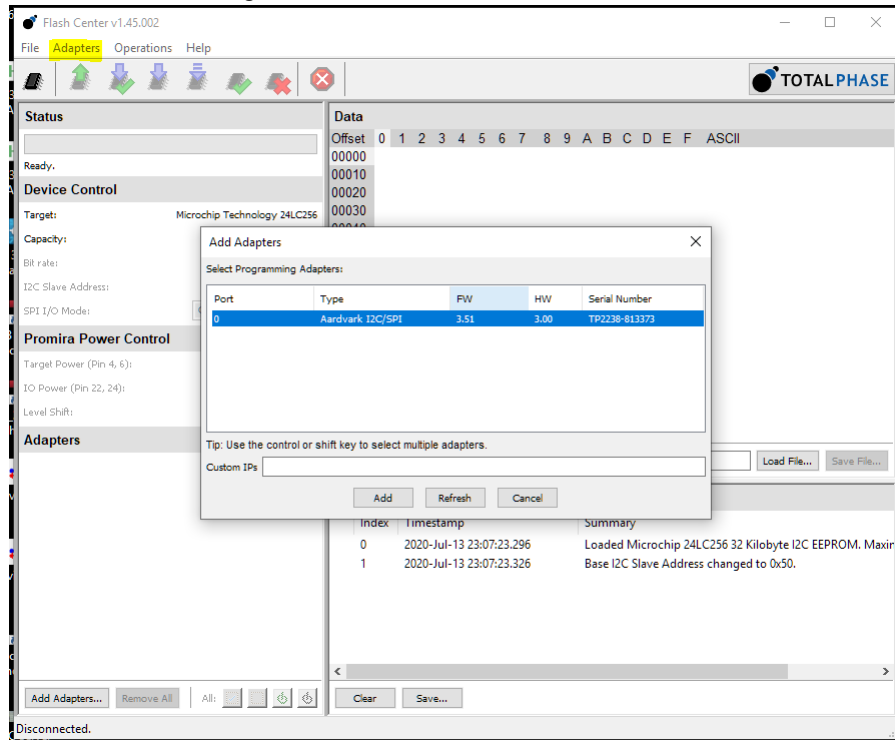
1. Install the Total Phase Flash Center Software
2. Connect Aardvark I2C/SPI Host adapter to the PC
3. Install the USB driver for the Aardvark I2C/SPI Host Adapter

Once setup is completed, use the following steps to program the EEPROM:

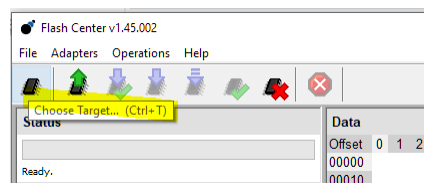
1. Disconnect all devices from the TPS257xx-Q1 device USB charge ports. All USB ports must remain disconnected throughout the programming process.
2. Invoke the *Flash Center Software*
3. Select: *Adapters > Add Adapters*



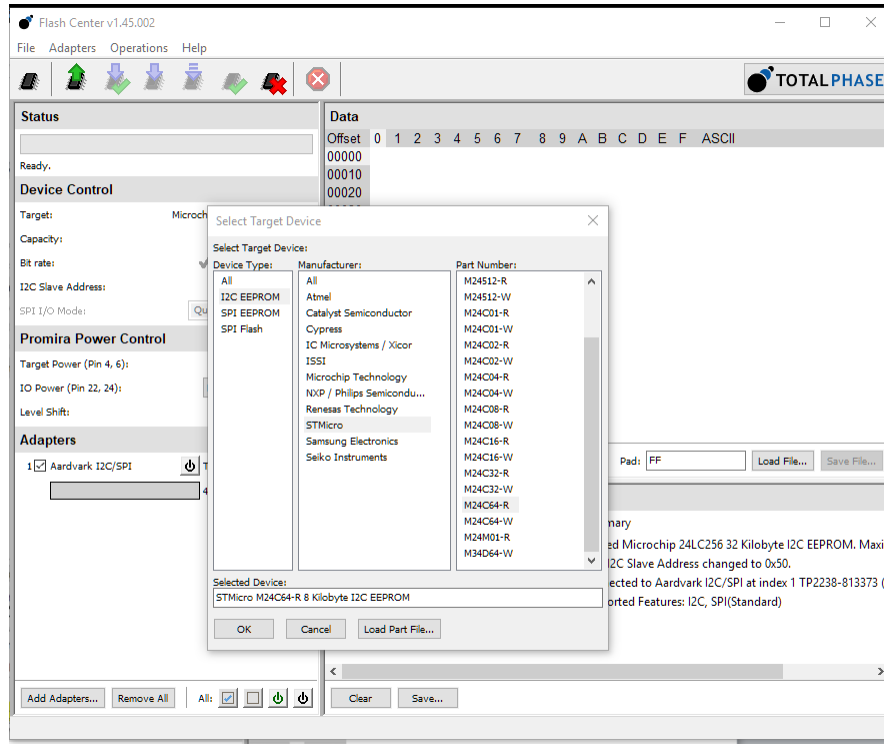
4. *Add Adapters* leads to the following screen. Click the *Add* button



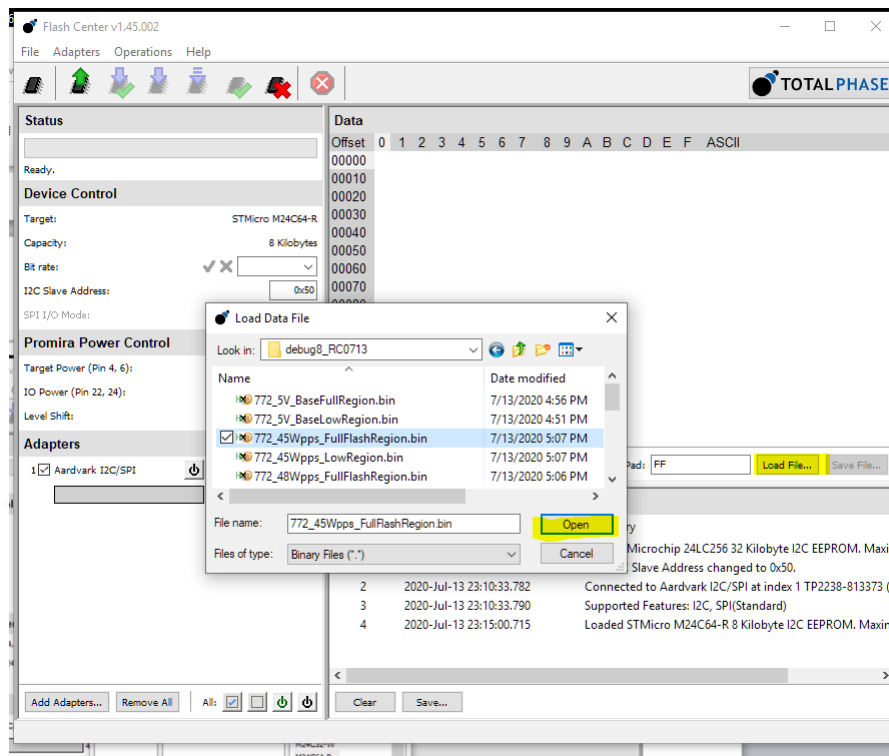
5. Click the *Choose Target* icon



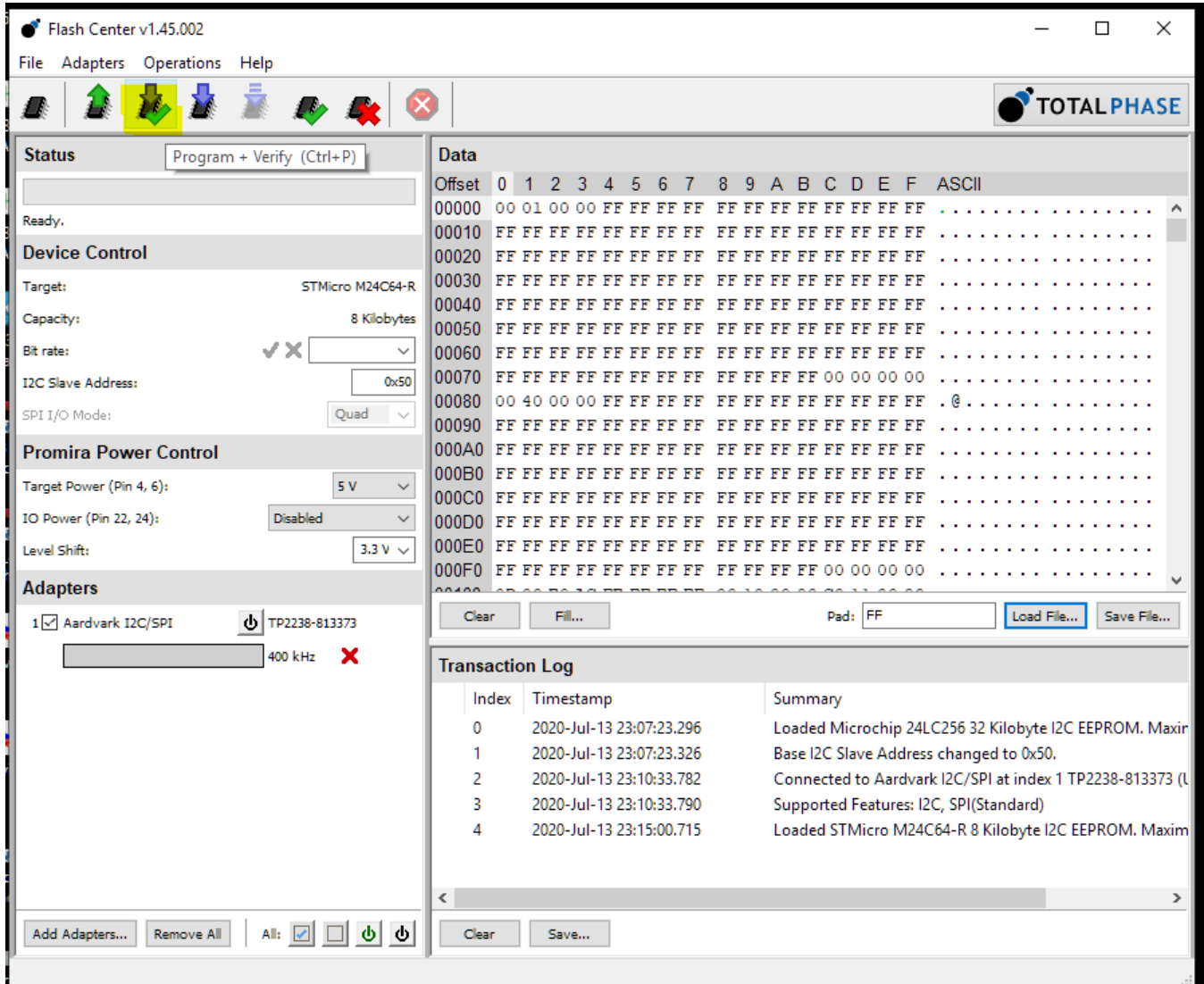
- Choose the target device to be programmed. If the part number for the device is not shown, choose a target device with the same memory capacity.



- Select **Load File > Open** and navigate to the **Full Flash** binary example file provided by TI or a custom binary file created with the GUI.



## 8. Select Program and Verify



Flash Center v1.45.002

File Adapters Operations Help

**Status** Program + Verify (Ctrl+P)

Ready.

**Device Control**

Target: STMicro M24C64-R

Capacity: 8 Kilobytes

Bit rate:   [v]

I2C Slave Address: 0x50

SPI I/O Mode: Quad

**Promira Power Control**

Target Power (Pin 4, 6): 5 V

IO Power (Pin 22, 24): Disabled

Level Shift: 3.3 V

**Adapters**

Aardvark I2C/SPI TP2238-813373

[v] 400 kHz

Add Adapters... Remove All All:

**Data**

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ASCII	
00000	00	01	00	00	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
00010	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
00020	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
00030	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
00040	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
00050	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
00060	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
00070	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	00	00	00	00	.....
00080	00	40	00	00	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	@.....
00090	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
000A0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
000B0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
000C0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
000D0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
000E0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	.....
000F0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	00	00	00	00	.....

Clear Fill... Pad: FF Load File... Save File...

**Transaction Log**

Index	Timestamp	Summary
0	2020-Jul-13 23:07:23.296	Loaded Microchip 24LC256 32 Kilobyte I2C EEPROM. Maxim
1	2020-Jul-13 23:07:23.326	Base I2C Slave Address changed to 0x50.
2	2020-Jul-13 23:10:33.782	Connected to Aardvark I2C/SPI at index 1 TP2238-813373 (L
3	2020-Jul-13 23:10:33.790	Supported Features: I2C, SPI(Standard)
4	2020-Jul-13 23:15:00.715	Loaded STMicro M24C64-R 8 Kilobyte I2C EEPROM. Maxim

Clear Save...

## 5.5 SSH Key Generation

Creating RSA Keys using ssh-keygen. SSH key pairs can be generated using the following command lines:

1. `mkdir keys`
2. `ssh-keygen -t rsa -b 3072 -m PEM -f keys/rsa3072engr`
3. `ssh-keygen -t rsa -b 3072 -m PEM -f keys/rsa3072prod`

```
$ ssh-keygen -t rsa -b 3072 -m PEM -f keys/rsa3072engr1
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in keys/rsa3072engr1.
Your public key has been saved in keys/rsa3072engr1.pub.
The key fingerprint is:
SHA256:3B1cJTFvQlXFGp7BRH2PSzstnkKpYe8HrkVVkvV+eHo a0214798@lta0214798
The key's randomart image is:
+---[RSA 3072]-----+
|
|      ***B*
|      . o.+oo=
|      O..O+.+
|      . . o.+o.
|      S O .+.+=
|      o.+ *oo
|      . *.o.+E
|      ..+ +.
|      .o.o
+-----[SHA256]-----+
```

Figure 5-13. Git Bash Key Generation - *Engineering*

```
$ ssh-keygen -t rsa -b 3072 -m PEM -f keys/rsa3072prod1
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in keys/rsa3072prod1.
Your public key has been saved in keys/rsa3072prod1.pub.
The key fingerprint is:
SHA256:L9ZMEfwykZ7WxrX7Iz3cdKRx6wsKDFxwsFPTh2s+w6I a0214798@lta0214798
The key's randomart image is:
+---[RSA 3072]-----+
|
|      o.=o..
|      = +. . .
|      o o.=o ..
|      . o *+o o.o
|      oS.=+ . =o
|      o* =. .oo
|      +o= o.+.+
|      E .. ...*o
|      . ..+
+-----[SHA256]-----+
```

Figure 5-14. Git Bash Key Generation - *Production*

Generated keys are found in the `keys` directory created via the `mkdir keys` command.

```
$ dir keys
rsa3072engr      rsa3072engr1      rsa3072prod      rsa3072prod1
rsa3072engr.pub  rsa3072engr1.pub  rsa3072prod.pub  rsa3072prod1.pub
```

Figure 5-15. Generated Keys



For a Microsoft® Windows® PC, *Git installation with SSH* can be necessary to generate keys as instructed in this section.

To install Git:

1. Download and initiate the [Git installer](#).
2. When prompted, accept the default components by clicking the *Next* button.
3. Choose the default text editor. If you have Notepad++™ installed, select Notepad++ and click the *Next* button.
4. Select *Use Git from the Windows Command Prompt* and click the *Next* button.
5. Select *Use OpenSSL library* and click the *Next* button.
6. Select *Checkout Windows-style, commit Unix-style line endings* and click the *Next* button.
7. Select *Use MinTTY* (The default terminal of mYSYS2) and click the *Next* button.
8. Accept the default extra option configuration by clicking the *Install* button.
9. When the installation completes, restart Windows, if needed.

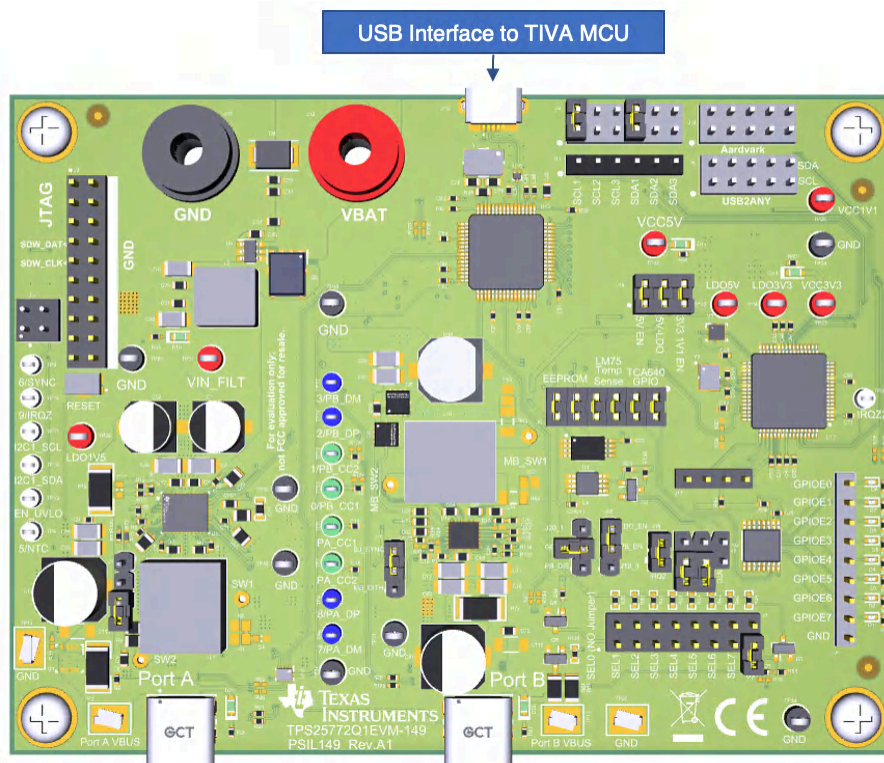
To launch GitBash:

1. Invoke the command line window: `cmd.exe`.
2. Run `bash.exe` from the Git installation folder: (that is, `C:\Program Files\Git\bin`)

## 6 Telemetry

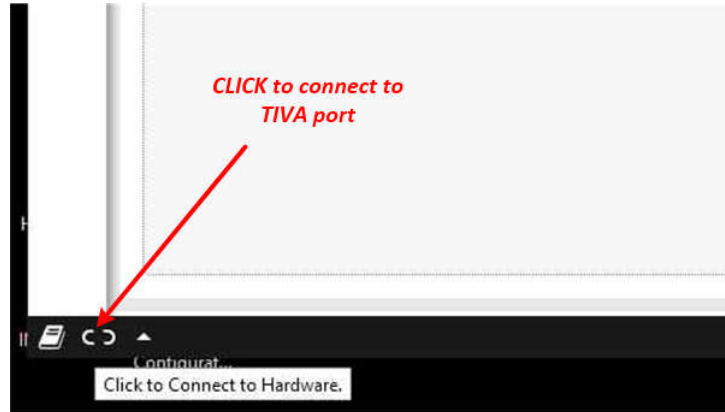
The EVM supports device communication over USB to the integrated TIVA MCU. Connect the USB cable to the micro USB port of the EVM as shown in [Figure 6-1](#). In addition to allowing EEPROM programming over I2C, the USB connection also provides a data and control communication path between the EVM and PC. Using the USB interface, the user has control over the following actions:

- Device reset
- I2C port configuration
- Telemetry data detailing the operational status of the charging ports
- Device status



**Figure 6-1. TIVA USB Port Connection**

The connection status is indicated in the bottom left corner of TPS257XX-Q1-GUI, see [Figure 6-2](#) and [Figure 6-3](#). The status icon can also be used to connect or disconnect the TIVA USB port by clicking the icon. If not connected, click the icon to establish connection.

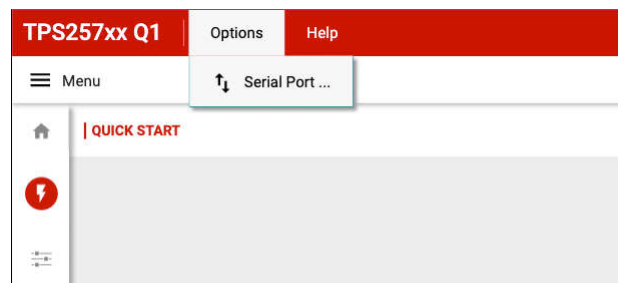


**Figure 6-2. TIVA USB Disconnected**



**Figure 6-3. TIVA USB Connected**

If the connection cannot be established, check the serial port configuration in the control at the upper left of the GUI, see [Figure 6-4](#).

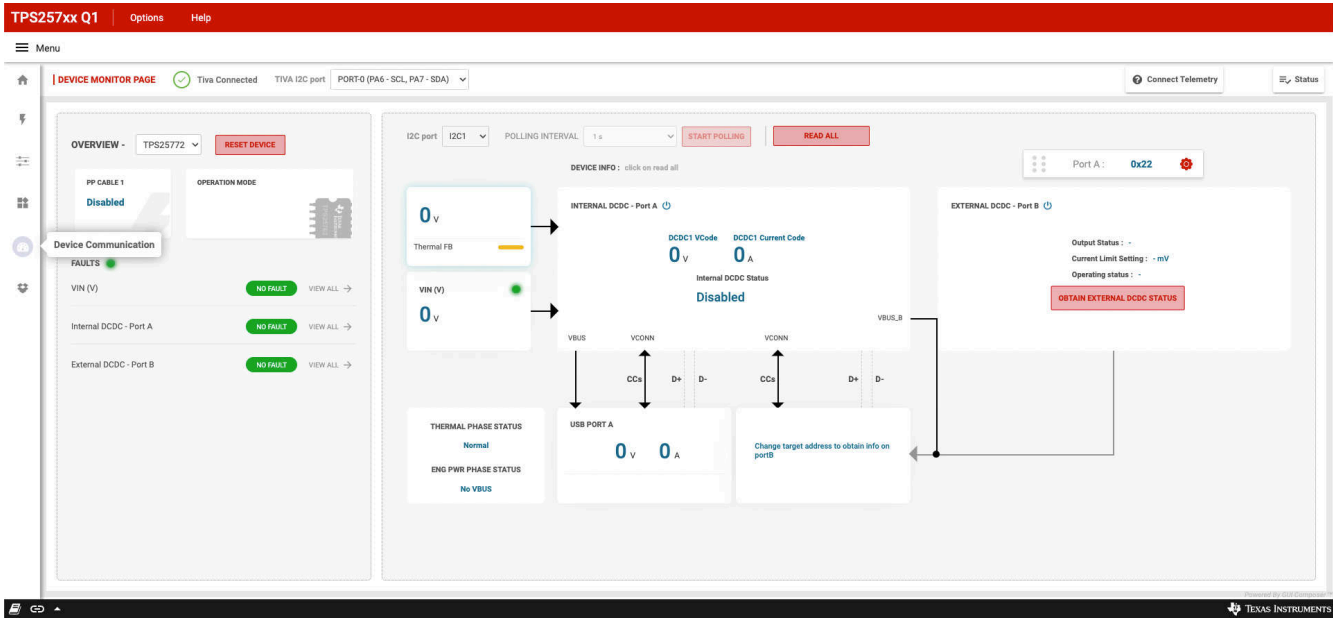


**Figure 6-4. GUI Communication Port Menu**

#### Note

When the I2C1 port is used for device communication (default), data cannot be read or written real-time as intended due to continuous bus traffic between I2C controller and connected target peripheral devices in hardware. Alternatively, when the I2C2 port is used for device communication, corresponding GPIOs (GPIO2 and GPIO3) must be configured as I2C\_SCL2 and I2C\_SDA2. (For dual-port devices, GPIO2 and GPIO3 are configured as Port B D+ and Port B D– and therefore are not intended to be used for I2C communication.)

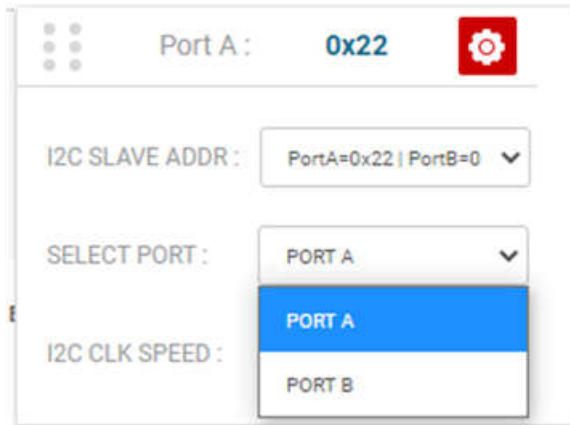
The telemetry view in the GUI is selected by clicking the *Device Communication* button in the sidebar, see [Figure 6-5](#).



**Figure 6-5. TPS257xx-Q1 Telemetry Interface**

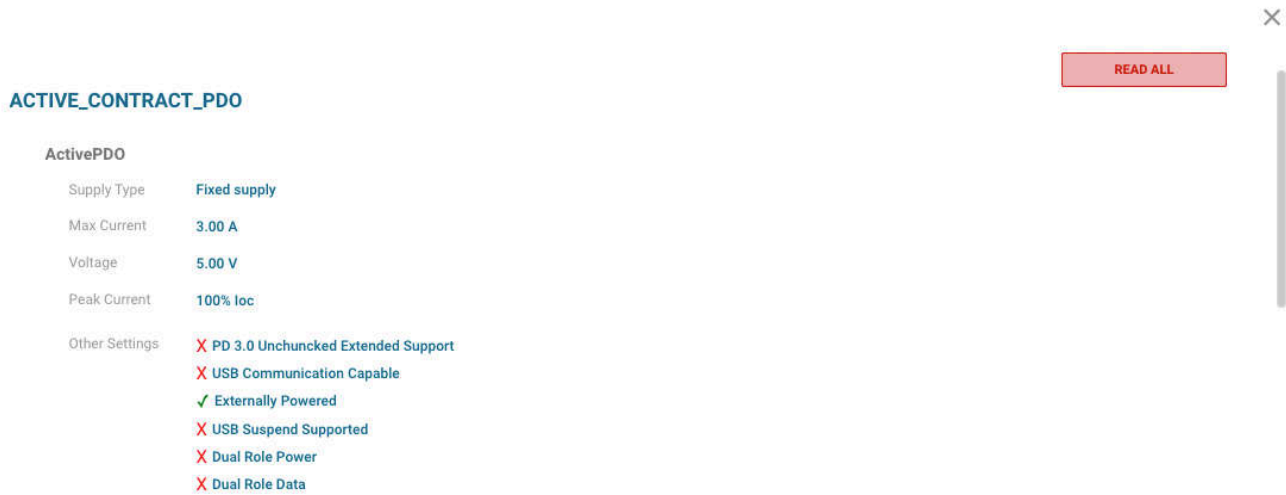
Make sure the correct target address is selected to access device telemetry information. The address is determined by the TVSP pin configuration as described in the device-specific data sheet. The target address is 0x22 or 0x23 for Port A and 0x26 or 0x27 for Port B (on a dual-port device). Only one port status can be read at a time.

The GUI provides Port A and Port B target address selection controls to align with the hardware configuration, see [Figure 6-6](#).



**Figure 6-6. Telemetry Port Configuration**

In addition to the main telemetry information, clicking the *Status* button in the upper right of the GUI page opens the quick status window displaying additional information about the port charging status when the *READ ALL* button is selected, see [Figure 6-7](#).



**ACTIVE\_CONTRACT\_PDO**

**ActivePDO**

Supply Type	Fixed supply
Max Current	3.00 A
Voltage	5.00 V
Peak Current	100% loc

**Other Settings**

- X PD 3.0 Unchunked Extended Support
- X USB Communication Capable
- ✓ Externally Powered
- X USB Suspend Supported
- X Dual Role Power
- X Dual Role Data

**Figure 6-7. Telemetry Status View**

## A TPS257XX-Q1 GUI Feature - CUSTOM ID (Version Control)

There are two ways to version control a GUI-generated binary file. The first method is automatically implemented when a binary is generated, as covered in [Section 5](#), and a timestamp will be included within the file name. In some cases, however, it is desirable to be able to uniquely identify a binary file that is loaded into an EEPROM using an EEPROM reader such as an Aardvark adapter (see [Section 5.4](#)). This can be done by integrating a user-defined version control to the binary file using the CUSTOM ID feature on the OTHER tab from the *Advanced Configuration GUI* page (see [Figure A-1](#)).

Other System Config is a part of System Configuration

**DEVICE IDS**

Port A

VENDOR ID : 0x 451

XID ⓘ : 0x 00

PRODUCT ID : 0x 00

Port B

VENDOR ID : 0x 451

XID ⓘ : 0x 00

PRODUCT ID : 0x 00

**LDO**

SELECT LDO  EXTERNAL LDO  INTERNAL LDO

[CONFIGURE IN USB PORT CONFIG](#)

**CUSTOM ID**  Enabled ⓘ

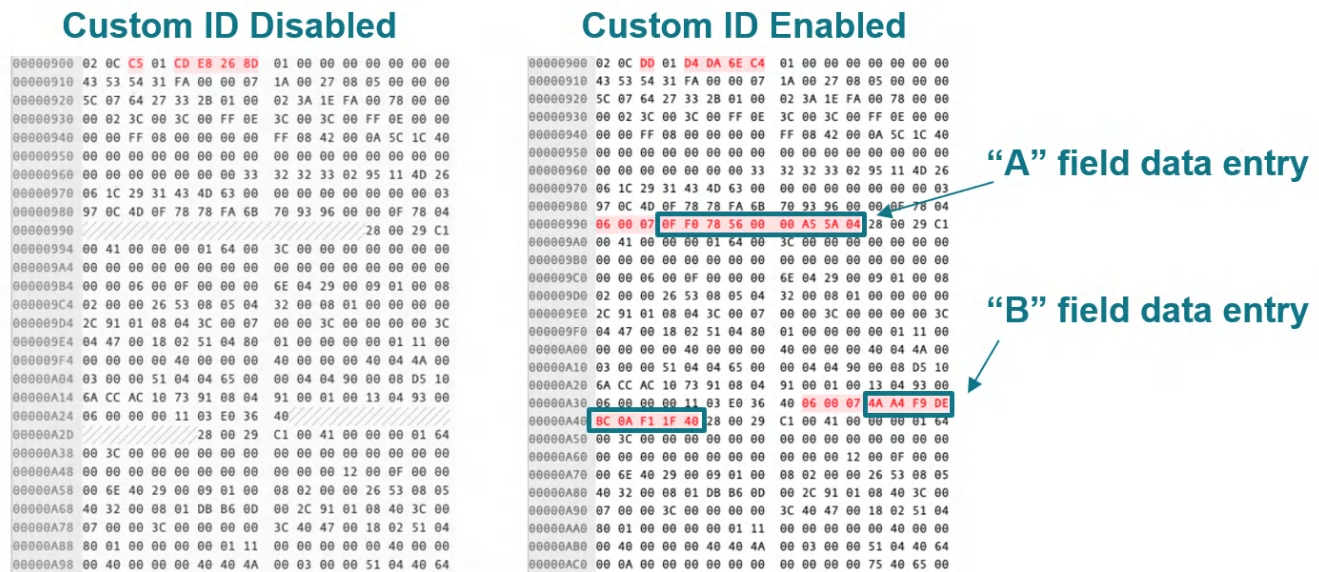
A	0x 5AA5	0x 0000	0x 5678	0x F00F
B	0x 1FF1	0x 0ABC	0x DEF9	0x A44A

**Figure A-1. CUSTOM ID Menu**

The CUSTOM ID feature is disabled by default. By enabling it, custom-defined values can be added to four input boxes per port. A 16-bit value can be entered in each box represented in hexadecimal format, so values up to 8 bytes for TPS25762-Q1 ("A" field only) and 16 bytes for TPS25772-Q1 ("A+B" fields) are possible.

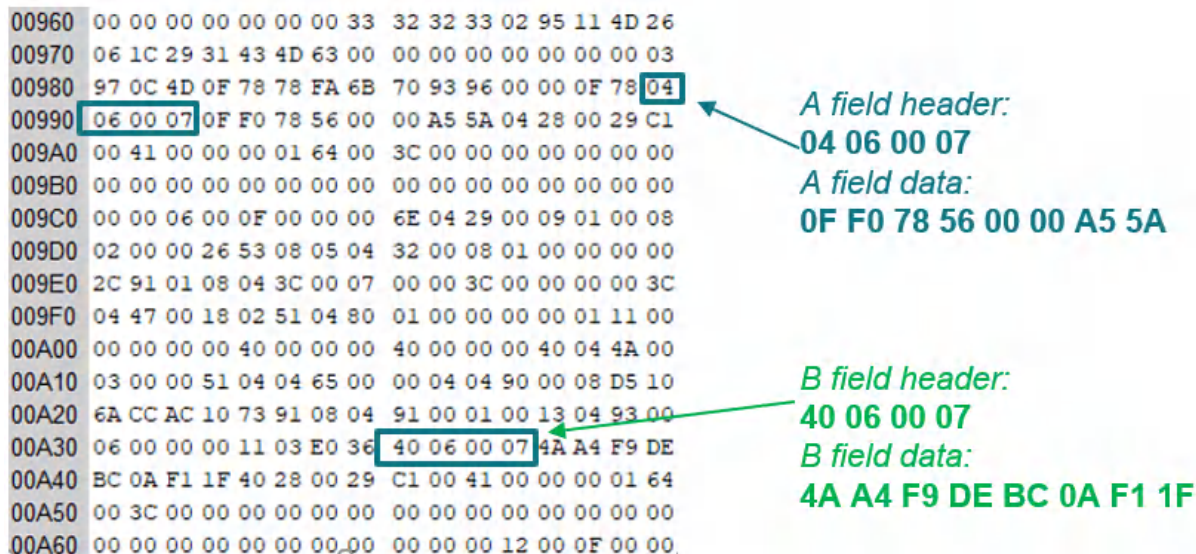
[Figure A-2](#) shows a comparison of two binary output files with the CUSTOM ID feature disabled and enabled. The same "A" and "B" field data entries from [Figure A-1](#) are highlighted in the CUSTOM ID enabled binary shown below. Note that the custom-defined values in the binary are assembled least significant byte first.





**Figure A-2. Binary Files with CUSTOM ID Enabled vs. Disabled**

The byte position of the CUSTOM ID data may vary in the binary file depending on how the device features are configured. The "A" field data entry and "B" field data entry are preceded with their respective headers that are always the same (see [Figure A-3](#)); the header for the "A" field is 04 06 00 07 and the header for the "B" field is 40 06 00 07. Searching for the header within the binary file will allow for easy identification of the CUSTOM ID data.

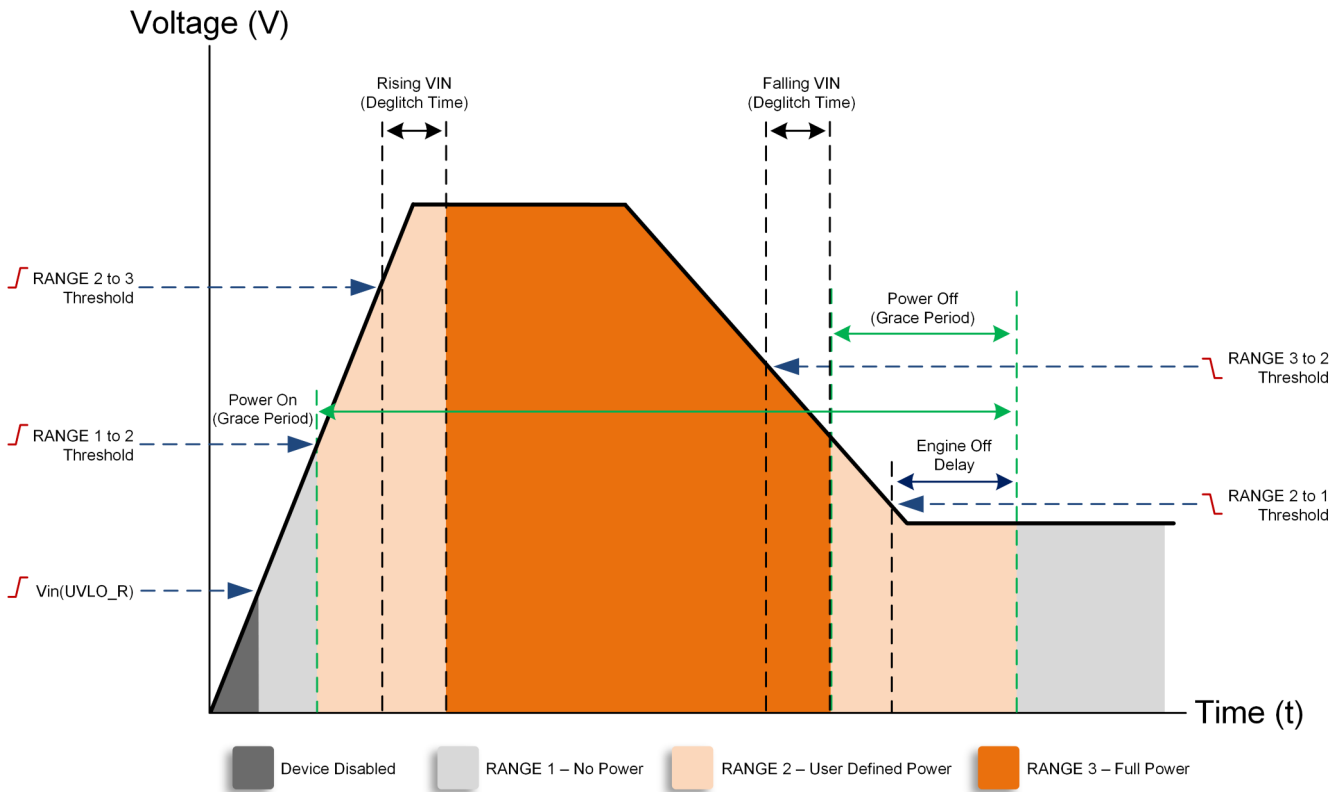


**Figure A-3. CUSTOM ID Data Headers**

Although a binary file is time stamped in the file name when it is generated using the GUI, the CUSTOM ID feature is a more reliable way to embed a custom revision/version identifier. It is integrated into the binary file generation and it can be read out of the EEPROM directly for verification purposes.

## B VIN Engine On or Off (TPS257xC-Q1)

The naming convention of the TPS257x2C-Q1 devices' VIN Engine On/Off parameters have changed moving from GUI v1.4.0 to v2.1.0. No changes were made functionally. The new VIN Engine On or Off diagram for the TPS257x2C-Q1 devices is shown in Figure B-1.



**Figure B-1. Engine On/Off Controls and Power Mode Operation (TPS257x2C-Q1)**

Table B-1 below shows the VIN Engine On or Off parameter name changes implemented when updating the GUI from v1.4.0 to v2.1.0.

**Table B-1. VIN Engine On or Off Parameter Name Changes (GUI v1.4.0 to v2.1.0)**

GUI v1.4.0	GUI v2.1.0
EngOnVth	RANGE 2 to 3 Threshold
EngOffNomVth	RANGE 3 to 2 Threshold
EngineOnMinVth	RANGE 1 to 2 Threshold
EngineOffMinVth	RANGE 2 to 1 Threshold
EngOn Deglitch	Rising VIN (Range 2 to 3)
EngOff Deglitch	Falling VIN (Range 3 to 2)
EngOnTimer(Grace Period)	Power On (Range 1 to 2)
EngOffTimer(Grace Period)	Power Off (Range 3 to 2)



## Revision History

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

<b>Changes from Revision B (February 2024) to Revision C (August 2024)</b>	<b>Page</b>
• Changed "TPS257xx-Q1-GUI" to "TPS257XX-Q1-GUI" throughout the document.....	1
• Removed all mentions and guidance of Simple Configuration View throughout the document.....	1
• Updated list of EVMs to latest ROM4 in <a href="#">Section 1</a> .....	2
• Added ROM4 product and tool pages to list .....	3
• Changed list to ROM4 EVMs in <a href="#">Section 2.2</a> .....	3
• Updated <a href="#">Section 2.4</a> .....	3
• Updated <a href="#">Section 3</a> . Updated to show the "Choose device" selection in the new GUI v2.0.4.....	4
• Updated <a href="#">Section 3.1</a> . .....	5
• Updated <a href="#">Section 3.2</a> .....	5
• Updated <a href="#">Section 4.1</a> .....	5
• Updated <a href="#">Section 5</a> .....	11
• Added <a href="#">Appendix B</a> .....	29

---

## 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](https://www.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  
Copyright © 2024, Texas Instruments Incorporated