Programmer's Guide

TPS6521405 Non-Volatile Memory (NVM) Programming Guide

ABSTRACT

The TPS65214 power management integrated circuit (PMIC) includes a configurable non-volatile memory (NVM) space. This programmer's guide details the step by step instructions to define the PMIC default configuration and how to program the one time programmable (OTP) memory.

Table of Contents

List of Figures

List of Tables

Trademarks

TI E2E™ is a trademark of Texas Instruments.
All trademarks are the property of their respective owners.

1 Introduction

The configuration process to write to the NVM space is intended to be used in a production line or prototype board. The TPS65214 device has one time programmable (OTP) memory. After the device has been programmed once, the device cannot be reprogrammed again. Pre-configured NVMs that are programmed during manufacturing cannot be reprogrammed over.

The TPS6521405VAFR is an orderable part number, part of the TPS65214 family, created specifically to support custom NVM configuration. The -05 variant comes as a blank NVM with registers set to 00h, to be easily programmed over.

Figure 1-1 describes the supply options for pre-configured and custom NVMs based on volume. Design resources are available for pre-configured and high volume NVMs as well as low volume custom NVMs. These resources include application notes, user's guides, technical reference manuals, and NVM configuration files ready to be loaded into the PMIC NVM. Visit the TPS65214 product page on ti.com or use our PMIC TI E2E™ forum to ask about available resources.

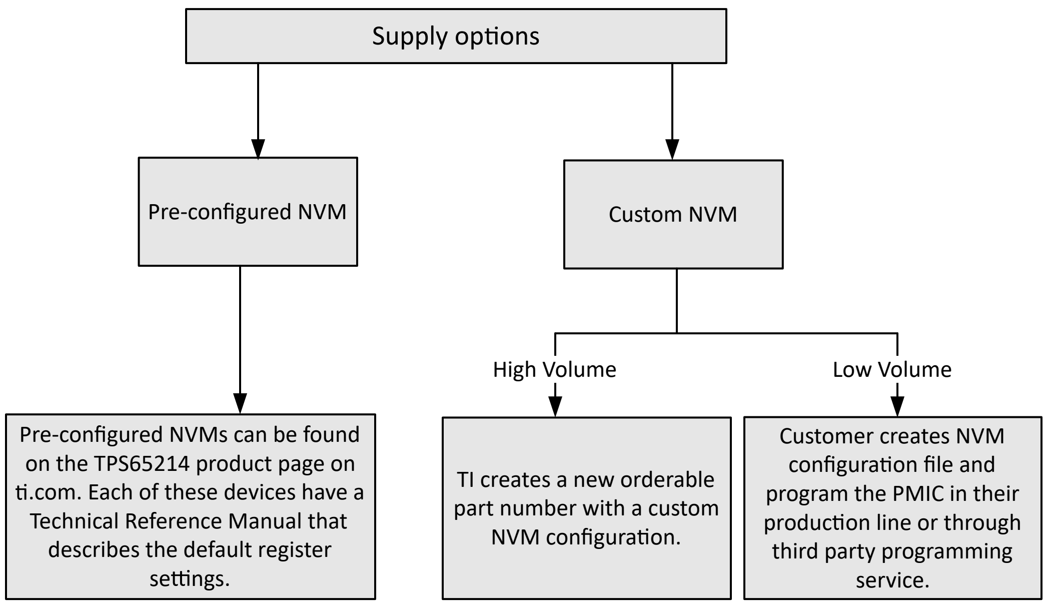

Figure 1-1. Supply Options

2 Hardware Requirements for NVM Programming

The PMIC has two memory spaces, the register map space and the NVM space. Re-programming the NVM is done by first writing to the register map through the serial interface (I2C) and then saving the register settings into the NVM. Because the configuration process involves writing to the register map, which controls the regulator and digital pins, the interface must not be dependent on the PMIC resources.

An external 3.3V power supply must be used to supply the pull-up resistors of the I2C pins instead of using one of the PMIC power resources while reprogramming the NVM. An external 8V supply is also needed to apply the proper voltage to the MODE/STBY pin during the programming step. An external LDO and boost converter are implemented on the TPS65214EVM-SKT board to satisfy these requirements. Table 2-1 and Figure 2-1 show the minimum hardware requirements for the hardware setup between the PMIC and the programming device.

Note

Other external components like inductors, capacitors, and so on are not needed to re-program the NVM in Initialize state. However, those components are needed for the PMIC operation in Active state and to validate NVM settings.

Table 2-1. Minimum Hardware Requirements for NVM Programming

Device pin	Required Connections
VSYS	VSYS voltage must be 3.3V or higher without exceeding the maximum recommended voltage in the spec.
	VSYS must have a minimum of 2.2uF capacitance.
VDD1P8	VDD1P8 must have a 2.2uF capacitance
I2C pins	Pull-up resistors on I2C pins (SDA/SCL) must be supplied by external 3.3V supply.
	I2C pins of the PMIC must be driven by an external I2C device that can communicate with the PMIC and write to the registers.
EN/PB/VSENSE	EN/PB/VSENSE pin must be connected to VSYS with a pull-up resistor.
MODE/STBY	MODE/STBY pin must be able to connect to a 8V source momentarily for the NVM programming.
AGND	AGND (pin# 15) must be connected to the PCB ground planes through a VIA . Keep the trace from the AGDN pin to the VIA short.
Thermal Pad	The package thermal pad must be connected to the PCB ground plane with a minimum of nine VIAS.

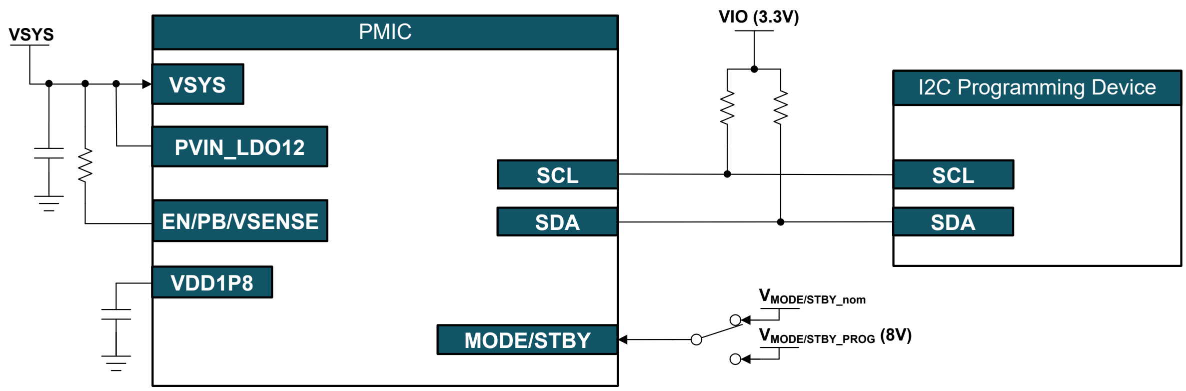

Figure 2-1. Hardware Setup for NVM Programming

3 Typical NVM Flow

This section describes the typical NVM definition flow which consists of the following steps: System requirements, Hardware setup, NVM programming and Test/Validation.

1. **System Requirements**

   Identify the system requirements and build a power distribution network (PDN). Voltage/Current, power-up/power-down sequence, low power modes, and load transient are typical requirements from processors, SoCs and peripherals.
2. **Hardware Setup**

   The TPS65214 can be programmed using the PMIC socketed EVM or a customer prototype board (in-circuit programming)

   - **Socketed EVM**: The socket board supports 5V USB input or an external power supply input. The TPS65214EVM-SKT board integrates a discrete 3.3V LDO that can supply the I2C pull-up resistors while the PMIC rails are OFF in Initialize state and an on-board 8V boost converter to apply $V_{MODE/STBY\_PROG}$. The socket board requires an external USB2ANY connection for I2C communication.

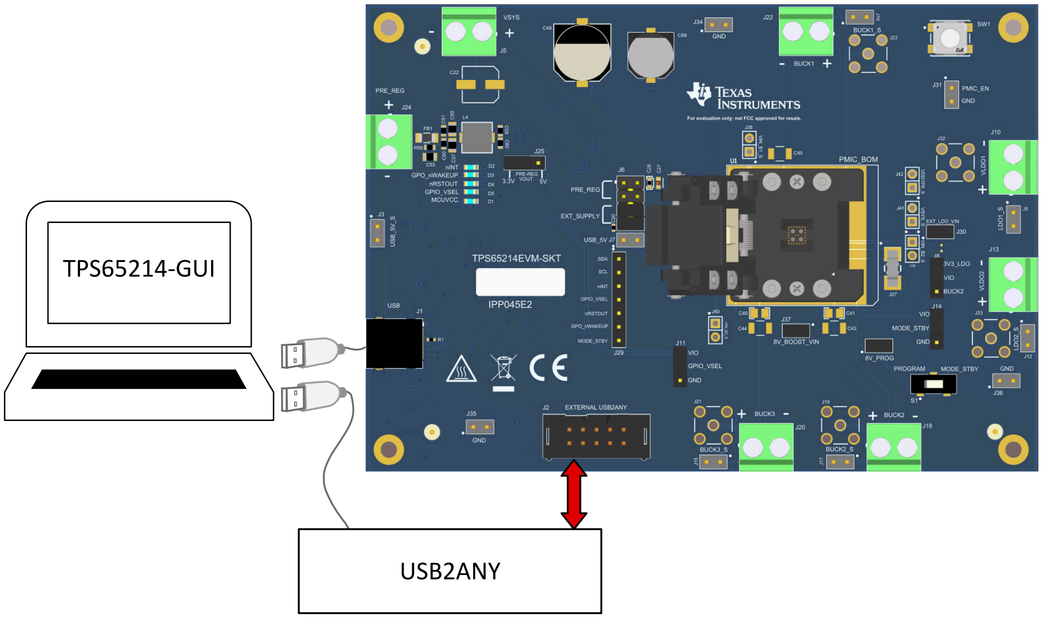

Figure 3-1. Socketed EVM

Note

When using the socketed EVM, ensure that the IC is inserted into the socket in the correct orientation, with the Pin 1 dot in the bottom right corner.

- **USB2ANY adapter**: The USB2ANY (available at ti.com) can be used to communicate with the PMIC and re-program the NVM settings.
- **Prototype board**: The user-programmable TPS6521405 NVM comes with all the power resources inactive by default and the EN/PB/VSENSE pin configured as push-button with without FSD (PU_ON_FSD = 0x0). If this pin is pulled up to VSYS, PMIC stays OFF (Initialize state) when a valid supply is connected to VSYS. This configuration allows the reprogramming of the NVM before the power-up sequence is executed. Figure 3-2 shows what customers need to include in the prototype board to re-program the PMIC NVM. The components required include three test points on GND, SCL, SDA, and two 1x3 single row header connectors. One of the header connectors must allow the pull-up supply to switch between the external 3.3V rail and the PMIC rail that supplies the I2C pins in the normal application. An additional header connector must be included to switch between the normal MODE/STBY operation and an external 8V source for the NVM programming step.

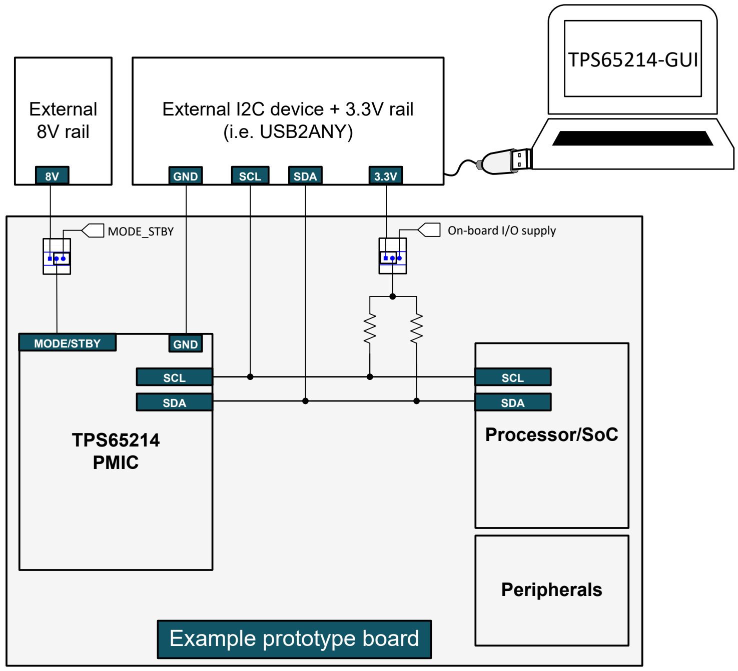

Figure 3-2. Prototype Example

Note

See section "Specifications" and "Detailed Design Procedure" in the device data sheet for information about recommended external components like inductors, output capacitance, and so on.

---

3. **NVM Programming**

   Follow the programming instructions in Section 4 and Section 5 to change the register settings. The TPS65214-GUI can be used with the socketed EVM (or a prototype board plus an external USB2ANY). Alternatively, customers can use their preferred I2C debugger tool to write to each of the NVM registers without using the TPS65214-GUI. After the desired settings are configured in the registers, follow the steps in Section 6 to save the settings into the NVM memory. Once the NVM is re-programmed, perform a power cycle to confirm the new register settings were burned correctly.

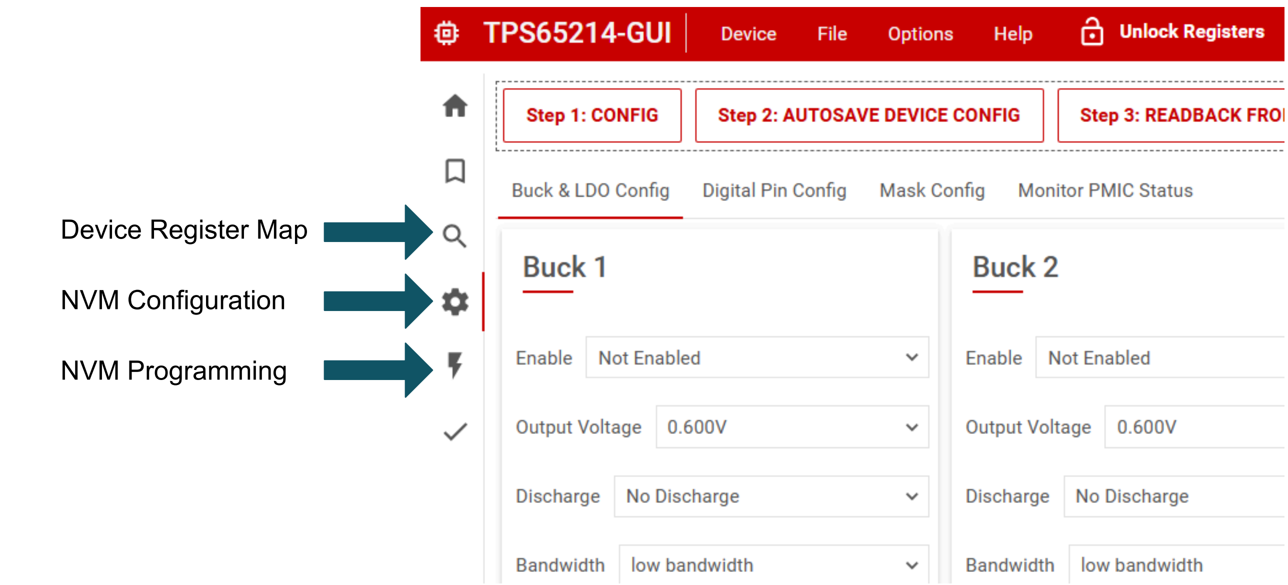

Figure 3-3. TPS65214-GUI

4. **NVM Testing**

   Test the NVM settings to confirm the expected PMIC behavior. The list below shows the minimum recommended tests. These tests can be performed with the socketed EVM or custom prototype board. If the socketed EVM was used to re-program the PMIC, the devices can be soldered down onto the customer prototype board to test and validate system level functionality. Alternatively, the device can be soldered onto the TPS65214EVM to test the custom NVM configuration by replacing the original IC on the board.

   - Measure all output voltages
   - Collect scope waveforms for power-up sequence (include GPIOs if enabled and nRSTOUT)
   - Collect scope waveforms for power-down sequence (include GPIOs if enabled and nRSTOUT)
   - Test EN/PB/VSENSE pin function and polarity to trigger ON and OFF request.
   - Test each multi-function pin (GPIO/VSEL, MODE/STBY) configuration and polarity. Pull these pins high or low and verify if PMIC behavior changes according to the configured pin function.

Note

The socketed EVM can be used for re-programming and basic tests (For example: measuring output voltages, colleting power-up sequence waveforms, and so on) but must not be used to test specific performance parameters like load transient and efficiency because the socket pogo pins and layout placement introduce higher parasitics that do not represent the design of a real application.

4 Programming Instructions

This section describes the steps required to program the PMIC NVM. The programming process consists of two primarily steps; changing the register settings and saving the new values into the NVM memory. Program the NVM in Initialize state, where VSYS is supplied but all of the PMIC outputs and monitors are OFF.

Figure 4-1 shows the steps to reprogram the device. Note that the device address for the TPS6521*505* PMIC is initially set to 0x48 by default. Ensure that the correct device address is set in the GUI in order to properly communicate with the PMIC.

The first command consists of an I2C OFF request to send the device to Initialize state. This command is only needed if the device is not in Initialize state. The second I2C command enables an internal oscillator for I2C communication during the Initialize state and disables the rail discharges. The third step requires updating register settings to match the desired application requirements. After the register settings are updated, the new values can be saved into the NVM by writing 0x0A to register address 0x34. The last step "Validation" is optional and consists of an I2C command that compares register settings with NVM content.

Note

The first I2C command (I2C OFF request) is only needed if the PMIC is not in Initialize state. The user programmable OPN TPS6521405 comes with the EN/PB/VSENSE pin configured as "push-button" with the FSD feature disabled by default. When configured as PB, the device detects an ON-request when the pin is pulled low. If this pin has a pull-up to VSYS, then PMIC stays in Initialize state after VSYS is supplied.

To verify if I2C communication is available in Initialize state, it is recommended to read the NVM ID register on address 0x01. The read back matches the two digits after the "TPS65214" in the part number. For example, when using TPS652140*5*, register 0x01 reads *05*.

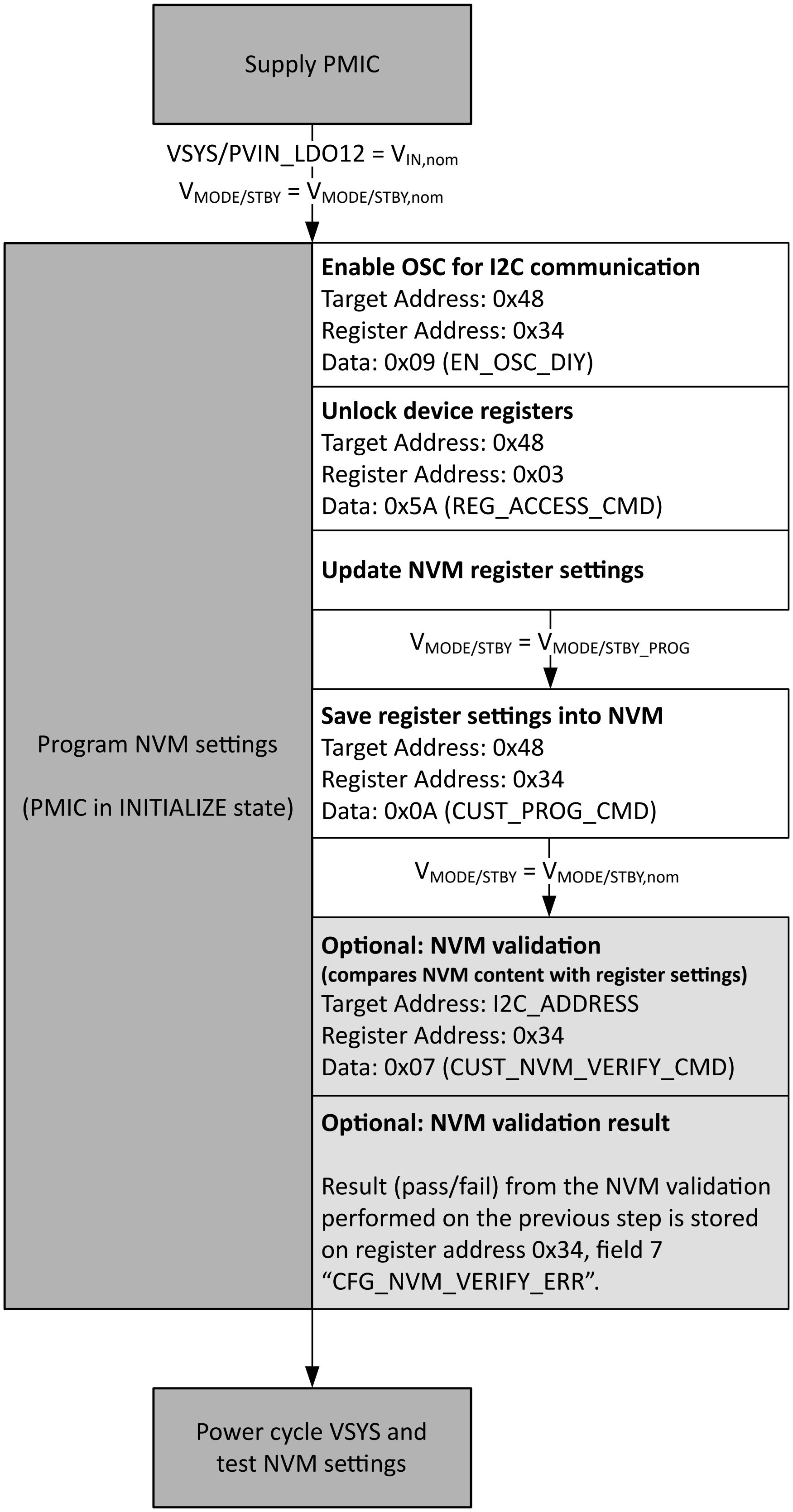

Figure 4-1. NVM Programming Steps

4.1 Connecting to the I2C address

Ensure that the correct device address is set in the GUI or respective I2C tool to properly communicate with the PMIC. The device address for the TPS6521405 PMIC is initially set to 0x48. Other TPS65214x devices, such as TPS6521401, may have the address typically set to 0x30.

In the TPS65214-GUI, go to the Device tab in the top bar shown in Figure 4-2, and highlight the I2C address field. This is set to 0x30 to default. Type in the target address "0x48" and hit enter. Check that the address is being acknowledged and the device is recognized as connected in the bottom left corner of the GUI.

Figure 4-2. Setting I2C address in the GUI

5 Register Configuration

This section describes how to configure the specific settings in the PMIC registers.

The TPS65214-GUI gathers all the configurable options in a single tab to make the configuration process easier. Register values can also be set individually using the register map tab or by writing to the respective registers if using a separate I2C interfacing tool from the designated GUI.

5.1 Configuring Enable Settings

The PMIC has an Active and Standby state where rails can be enabled or disabled. The state change can be triggered by the MODE/STBY pin when configured as STBY.

- Figure 5-1 shows the settings to be changed when using the configuration tab of the TPS65214-GUI.
- Table 5-1 show the register fields to be written when referring to the register map.

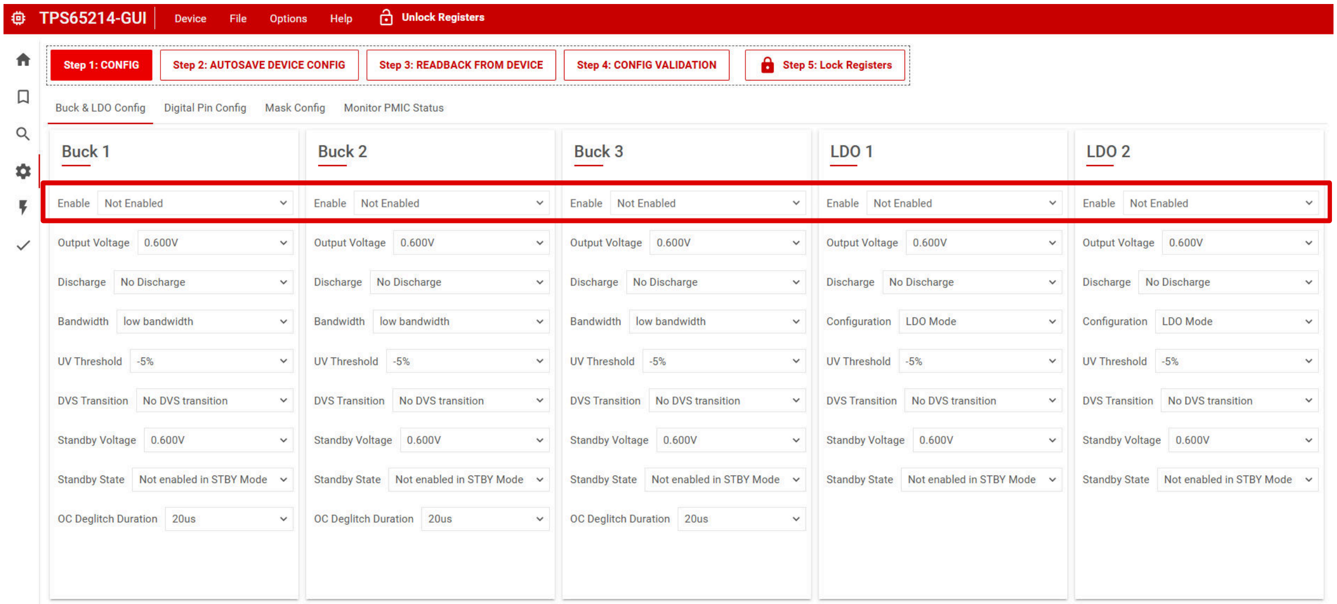

Figure 5-1. Enable Settings Using the TPS65214-GUI

Table 5-1. NVM Registers for Enable Settings

	Register Address	Bit		Settings
		Bit #	Field Name	
Enable rails in Active state	0x02	5	LDO1_EN	0h = Disabled 1h = Enabled
		4	LDO2_EN	0h = Disabled 1h = Enabled
		2	BUCK3_EN	0h = Disabled 1h = Enabled
		1	BUCK2_EN	0h = Disabled 1h = Enabled
		0	BUCK1_EN	0h = Disabled 1h = Enabled
Enable rails in Standby state	0x21	5	LDO1_STBY_EN	0h = Disabled 1h = Enabled
		4	LDO2_STBY_EN	0h = Disabled 1h = Enabled
		2	BUCK3_STBY_EN	0h = Disabled 1h = Enabled
		1	BUCK2_STBY_EN	0h = Disabled 1h = Enabled
		0	BUCK1_STBY_EN	0h = Disabled 1h = Enabled

5.2 Configuring the Bucks

There are several settings that can be programmed for the Buck converters. These include the output voltages, under voltage (UV) monitoring, and bandwidth among others.

- Figure 5-2 shows the settings to be changed when using the configuration tab of the TPS65214-GUI.
- Table 5-2, Table 5-3, and Table 5-4 show the register fields to be written when referring to the register map.

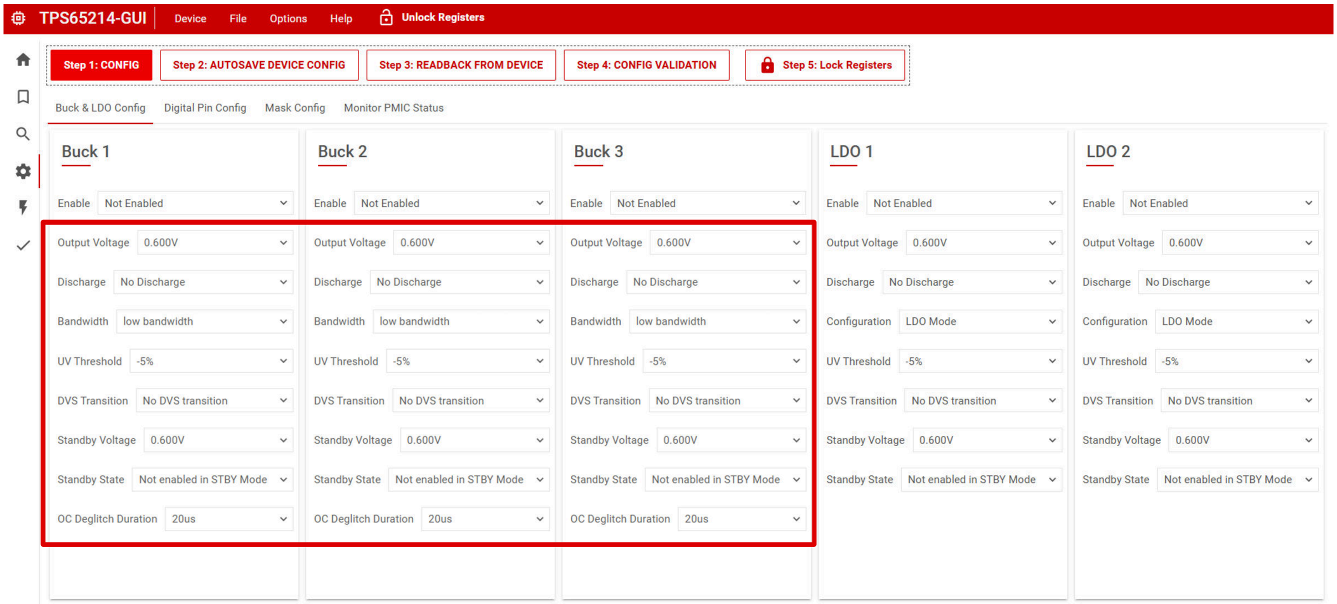

Figure 5-2. Buck Settings in the TPS65214-GUI

Table 5-2. NVM Registers for Buck1 Configuration

	Register Address	Bit		Settings
		Bit #	Field Name	
Bandwidth	0x0A	7	B1_BW_SEL	0h = low bandwidth 1h = high bandwidth
UV monitoring		6	B1_UV_THR_SEL	0h = -5% UV detection level 1h = -10% UV detection level
Output Voltage		5-0	B1_VSET	see register map on data sheet

Table 5-3. NVM Registers for Buck2 Configuration

	Register Address	Bit		Settings
		Bit #	Field Name	
Bandwidth	0x09	7	B2_BW_SEL	0h = low bandwidth 1h = high bandwidth
UV monitoring		6	B2_UV_THR_SEL	0h = -5% UV detection level 1h = -10% UV detection level
Output Voltage		5-0	B2_VSET	see register map on data sheet

Table 5-4. NVM Registers for Buck3 Configuration

	Register Address	Bit		Settings
		Bit #	Field Name	
Bandwidth	0x08	7	B3_BW_SEL	0h = low bandwidth 1h = high bandwidth
UV monitoring		6	B3_UV_THR_SEL	0h = -5% UV detection level 1h = -10% UV detection level
Output Voltage		5-0	B3_VSET	see register map on data sheet

5.3 Configuring LDOs

There are several settings that can be programmed for the LDO regulators. These include the output voltages, and under voltage (UV) monitoring among others.

- Figure 5-3 shows the settings to be changed when using the configuration tab of the TPS65214-GUI.
- Table 5-5 and Table 5-6 show the register fields to be written when referring to the register map.

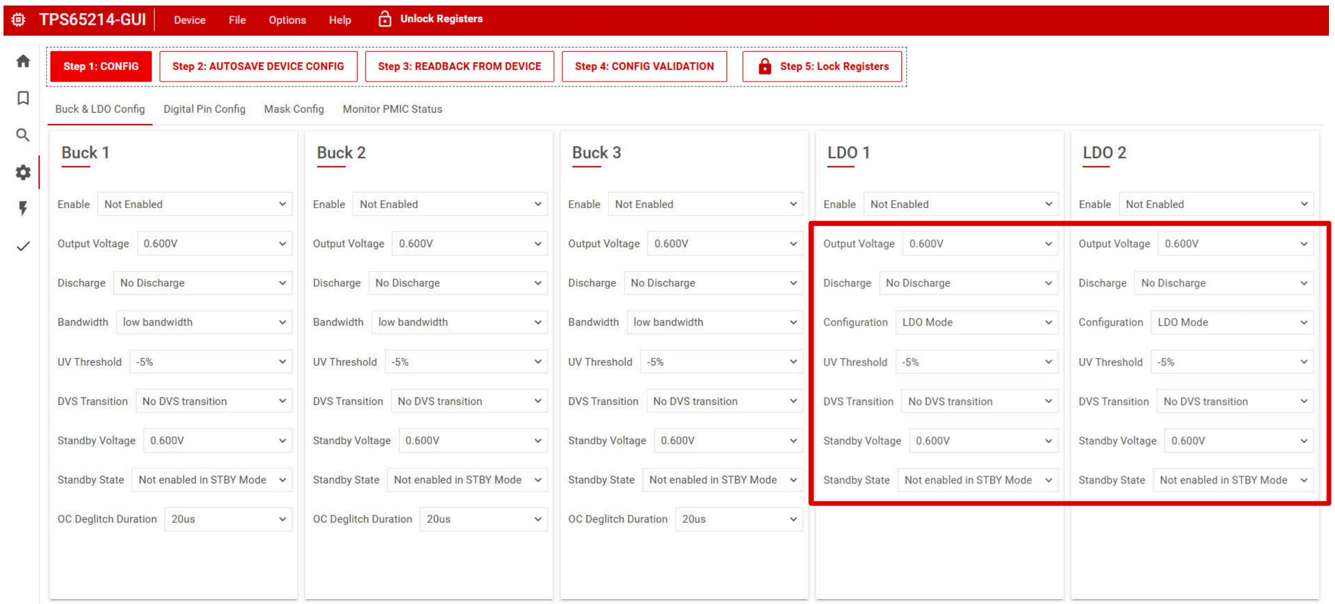

Figure 5-3. LDOs Settings Using the TPS65214-GUI

Table 5-5. NVM Registers for LDO1 Settings

	Register Address	Bit		Settings
		Bit #	Field Name	
Configuration	0x05	6	LDO1_LSW_CONFIG	0h = LDO1 NOT configured as load-switch 1h = LDO1 configured as Load-switch
Output Voltage		5-0	LDO1_VSET	see register map on data sheet
UV monitoring	0x1E	5	LDO1_UV_THR	0h = -5% UV 1h = -10% UV

Table 5-6. NVM Registers for LDO2 Settings

	Register Address	Bit		Settings
		Bit #	Field Name	
Configuration	0x06	7	LDO2_LSW_CONFIG	0h = LDO1 NOT configured as load-switch 1h = LDO1 configured as Load-switch
Output Voltage		5-0	LDO2_VSET	see register map on data sheet
UV Monitoring	0x1E	4	LDO2_UV_THR	0h = -5% UV 1h = -10% UV

5.4 Configuring Sequence

The process to configure the PMIC sequence consist of the following two steps:

1. Power-up/Power-down slot assignment: The slot assignment defines the order in which rails turn ON or OFF. Each of the PMIC rails must have a slot assigned. There are 16 slots available (0-15). Multiple rails (including GPIOs) can be assigned to the same slot so they be enabled at the same time.

2. Power-up/Power-down slot duration: The slot duration is the timing between the start of one slot to the start of the next slot. For example, if Buck1 is assigned to slot0 with a 3ms duration and Buck2 is assigned to slot 1, then Buck2 turns ON 3ms after Buck1.

Note

The slot duration does not dictate how long it takes for the rails to ramp. The slot duration only specifies how long the PMIC waits before enabling (or disabling) the rails that were assigned to the next slot.

- Table 5-7, Table 5-8, Table 5-9 and Table 5-10 show the register fields to be written when referring to the register map.

Table 5-7. Power-Up Sequence - Slot Assignments

	Register Address	Bit		Settings
		Bit#	Field Name	
Power-up Sequence Slot Assignment	0x11	7-4	BUCK1_SEQUENCE_ON_SLOT	see register map on data sheet
	0x10	7-4	BUCK2_SEQUENCE_ON_SLOT	see register map on data sheet
	0xF	7-4	BUCK3_SEQUENCE_ON_SLOT	see register map on data sheet
	0xE	7-4	LDO1_SEQUENCE_ON_SLOT	see register map on data sheet
	0xD	7-4	LDO2_SEQUENCE_ON_SLOT	see register map on data sheet
	0xC	7-4	LDO3_SEQUENCE_ON_SLOT	see register map on data sheet
	0xB	7-4	LDO4_SEQUENCE_ON_SLOT	see register map on data sheet
	0x15	7-4	GPO1_SEQUENCE_ON_SLOT	see register map on data sheet
	0x14	7-4	GPO2_SEQUENCE_ON_SLOT	see register map on data sheet
	0x13	7-4	GPIO_SEQUENCE_ON_SLOT	see register map on data sheet
	0x12	7-4	nRST_SEQUENCE_ON_SLOT	see register map on data sheet

Table 5-8. Power-Up Sequence - Slot Duration

	Register Address	Bit		Settings
		Bit#	Field Name	
Power-up Sequence Slot Duration	0x16	7-6	POWER_UP_SLOT_0_DURATION	see register map on data sheet
		5-4	POWER_UP_SLOT_1_DURATION	see register map on data sheet
		3-2	POWER_UP_SLOT_2_DURATION	see register map on data sheet
		1-0	POWER_UP_SLOT_3_DURATION	see register map on data sheet
	0x17	7-6	POWER_UP_SLOT_4_DURATION	see register map on data sheet
		5-4	POWER_UP_SLOT_5_DURATION	see register map on data sheet
		3-2	POWER_UP_SLOT_6_DURATION	see register map on data sheet
		1-0	POWER_UP_SLOT_7_DURATION	see register map on data sheet
	0x18	7-6	POWER_UP_SLOT_8_DURATION	see register map on data sheet
		5-4	POWER_UP_SLOT_9_DURATION	see register map on data sheet
		3-2	POWER_UP_SLOT_10_DURATION	see register map on data sheet
		1-0	POWER_UP_SLOT_11_DURATION	see register map on data sheet
	0x19	7-6	POWER_UP_SLOT_12_DURATION	see register map on data sheet
		5-4	POWER_UP_SLOT_13_DURATION	see register map on data sheet
		3-2	POWER_UP_SLOT_14_DURATION	see register map on data sheet
		1-0	POWER_UP_SLOT_15_DURATION	see register map on data sheet

Table 5-9. Power-Down Sequence - Slot Assignments

	Register Address	Bit		Settings
		Bit#	Field Name	
Power-down Sequence Slot Assignment	0x11	7-4	BUCK1_SEQUENCE_OFF_SLOT	see register map on data sheet
	0x10	7-4	BUCK2_SEQUENCE_OFF_SLOT	see register map on data sheet
	0xF	7-4	BUCK3_SEQUENCE_OFF_SLOT	see register map on data sheet
	0xE	7-4	LDO1_SEQUENCE_OFF_SLOT	see register map on data sheet
	0xD	7-4	LDO2_SEQUENCE_OFF_SLOT	see register map on data sheet
	0xC	7-4	LDO3_SEQUENCE_OFF_SLOT	see register map on data sheet
	0xB	7-4	LDO4_SEQUENCE_OFF_SLOT	see register map on data sheet
	0x15	7-4	GPO1_SEQUENCE_OFF_SLOT	see register map on data sheet
	0x14	7-4	GPO2_SEQUENCE_OFF_SLOT	see register map on data sheet
	0x13	7-4	GPIO_SEQUENCE_OFF_SLOT	see register map on data sheet
	0x12	7-4	nRST_SEQUENCE_OFF_SLOT	see register map on data sheet

Table 5-10. Power-Down Sequence - Slot Duration

	Register Address	Bit		Settings
		Bit#	Field Name	
Power-down Sequence Slot Duration	0x1A	7-6	POWER_DOWN_SLOT_0_DURATION	see register map on data sheet
		5-4	POWER_DOWN_SLOT_1_DURATION	see register map on data sheet
		3-2	POWER_DOWN_SLOT_2_DURATION	see register map on data sheet
		1-0	POWER_DOWN_SLOT_3_DURATION	see register map on data sheet
	0x1B	7-6	POWER_DOWN_SLOT_4_DURATION	see register map on data sheet
		5-4	POWER_DOWN_SLOT_5_DURATION	see register map on data sheet
		3-2	POWER_DOWN_SLOT_6_DURATION	see register map on data sheet
		1-0	POWER_DOWN_SLOT_7_DURATION	see register map on data sheet
	0x1C	7-6	POWER_DOWN_SLOT_8_DURATION	see register map on data sheet
		5-4	POWER_DOWN_SLOT_9_DURATION	see register map on data sheet
		3-2	POWER_DOWN_SLOT_10_DURATION	see register map on data sheet
		1-0	POWER_DOWN_SLOT_11_DURATION	see register map on data sheet
	0x1D	7-6	POWER_DOWN_SLOT_12_DURATION	see register map on data sheet
		5-4	POWER_DOWN_SLOT_13_DURATION	see register map on data sheet
		3-2	POWER_DOWN_SLOT_14_DURATION	see register map on data sheet
		1-0	POWER_DOWN_SLOT_15_DURATION	see register map on data sheet

5.5 Configuring GPIOs

GPIOs can be used to enable external discrete components. GPIO can also be used for multi-PMIC configuration to sync the power-up and power-down sequence between two TPS65214 devices.

- Figure 5-4 shows the settings to be changed when using the configuration tab of the TPS65214-GUI.
- Table 5-11, Table 5-12 show the register fields to be written when referring to the register map.

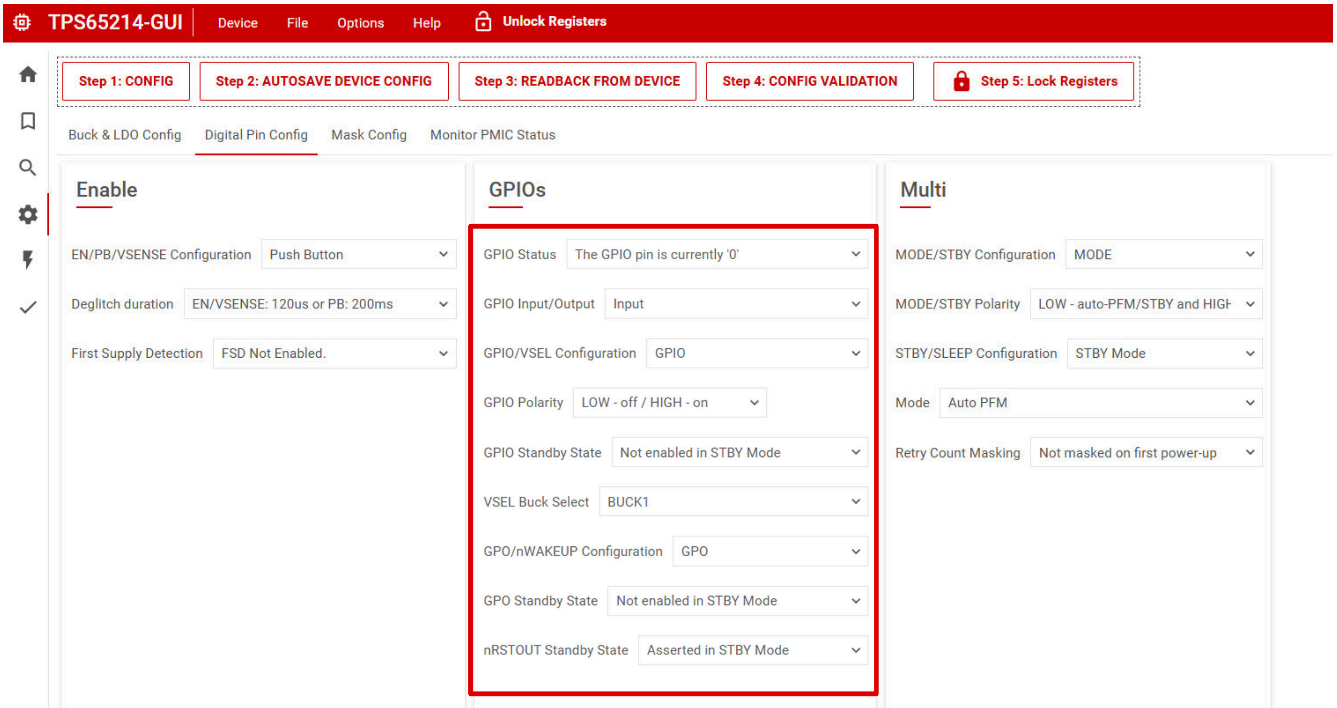

Figure 5-4. GPIOs Configuration

Table 5-11. NVM Registers for GPIO Enable Settings

	Register Address	Bit		Settings
		Bit #	Field Name	
Enable settings in Active state	0x1E	2	GPIO_EN	0h = Disabled. The output state is low. 1h = Enabled. The output state is Hi-Z.
		0	GPO_EN	0h = Disabled. The output state is low. 1h = Enabled. The output state is Hi-Z.
Enable settings in Standby state	0x22	2	GPIO_STBY_EN	0h = Disabled. The output state is low. 1h = Enabled. The output state is Hi-Z.
		0	GPO_STBY_EN	0h = Disabled. The output state is low. 1h = Enabled. The output state is Hi-Z.

Table 5-12. NVM Register for GPIO Configuration

	Register Address	Bit		Settings
		Bit #	Field Name	
GPIO/VSEL configuration	0x1E	1	GPIO_CONFIG	0h = Configured as an Input 1h = Configured as an Output
	0x1F	3	GPIO_VSEL_CONFIG	0h = Configured as GPIO 1h = Configured as VSEL

5.6 Configuring Multi-Function Pins

The TPS65214 PMIC has two configurable multi-function pins. MODE/STBY can be configured as MODE to select the switching or as STBY to trigger a transition to Standby state. When configured as VSEL, the GPIO/VSEL pin can be configured to set the output voltage on BUCK1 or BUCK3. Refer to the data sheet for information on pin polarity.

- Figure 5-5 shows the settings to be changed when using the configuration tab of the TPS65214-GUI
- Figure 5-5 show the register fields to be written when referring to the register map.

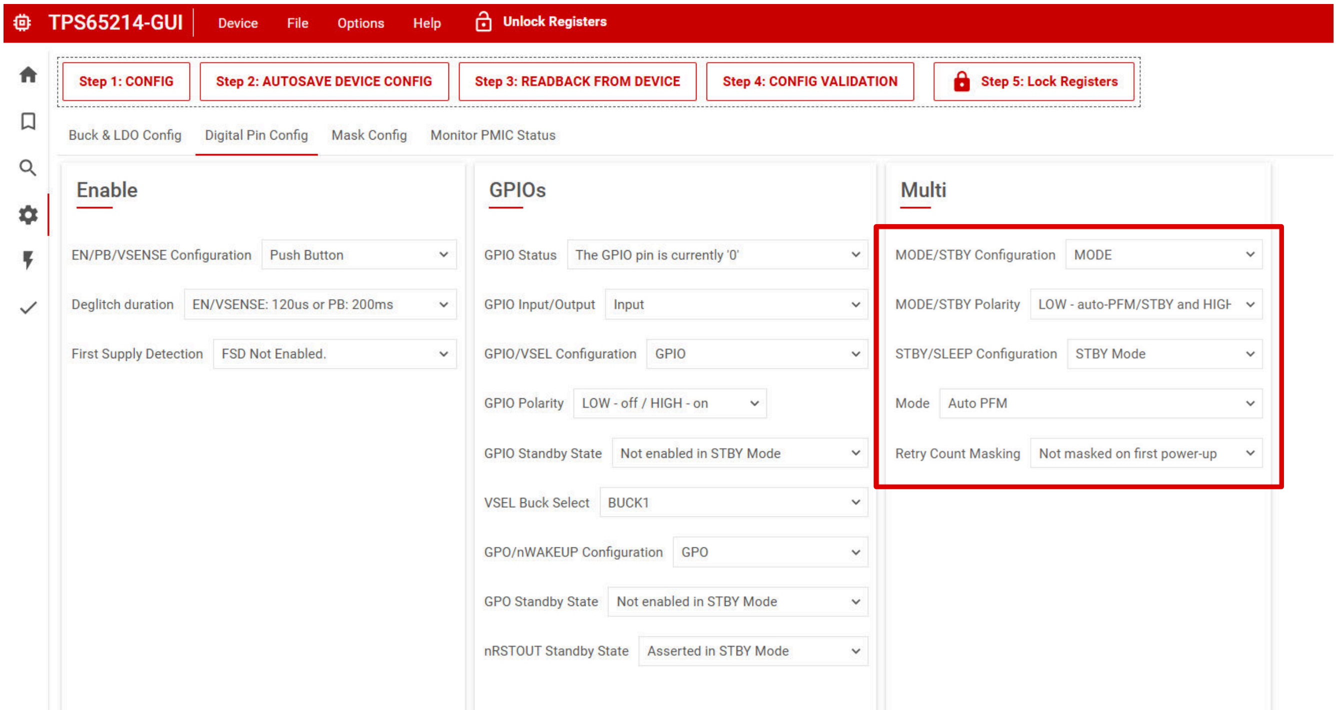

Figure 5-5. Multi-Function Configuration using the TPS65214-GUI

Table 5-13. NVM Registers for VSEL

	Register Address	Bit		Settings
		Bit #	Field Name	
Pin Function	0x1F	3	GPIO_VSEL_CONFIG	0h = Configured as GPIO 1h = Configured as VSEL
		2	VSEL_RAIL	0h = BUCK1 1h = BUCK3
Pin polarity		1	VSEL_SD_POLARITY	0h = • LOW: 1.8V • HIGH: LDOx_VOUT register 1h = • HIGH: 1.8V • LOW: LDOx_VOUT register

Table 5-14. NVM Registers for MODE / STBY

	Register Address	Bit		Settings
		Bit #	Field Name	
Pin Function	0x20	1-0	MODE_STBY_CONFIG	0h = MODE 1h = STBY 2h = MODE and STBY 3h = MODE
Pin Polarity	0x1F	4	MODE_STBY_POLARITY	see register map on data sheet

Table 5-15. NVM Registers for GPO / nWAKEUP

	Register Address	Bit		Settings
		Bit #	Field Name	
Pin Function	0x20	2	MODE_RESET_CONFIG	0h = MODE 1h = RESET
RESET config		6	WARM_COLD_RESET_CONFIG	0h = COLD RESET 1h = WARM RESET
Pin Polarity	0x1F	5	MODE_RESET_POLARITY	see register map on data sheet

5.7 Configuring the EN/PB/VSENSE Pin

The enable pin of the PMIC can be configured as Enable, Push-Button, or VSENSE. In addition to the function, the deglitch can also be configured. Additionally, this pin has the option for first supply detection (FSD) to ignore the state of the EN/PB/VSENSE pin during the first power-up.

- Figure 5-6 shows the settings to be changed when using the configuration tab of the TPS65214-GUI.
- Table 5-16 show the register fields to be written when referring to the register map.

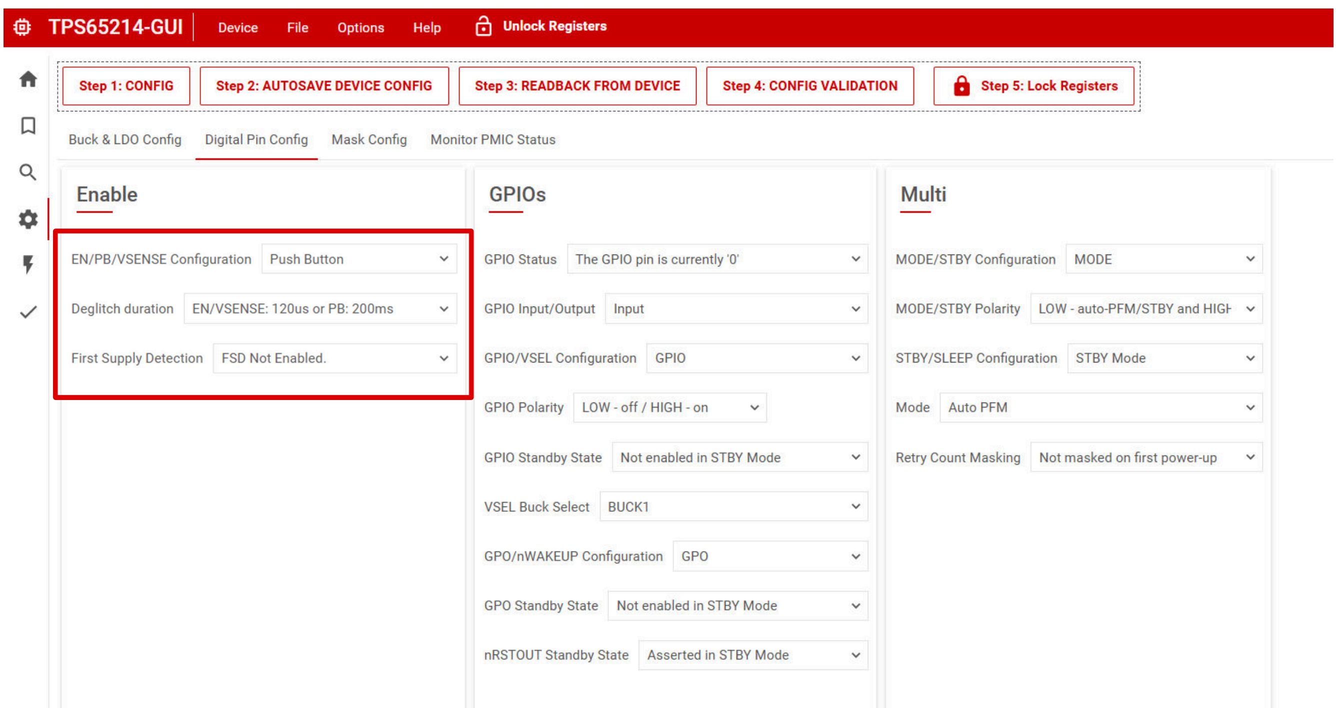

Figure 5-6. EN/PB/VSENSE Configuration Using the TPS65214-GUI

Table 5-16. NVM Registers for EN / PB / VSENSE

	Register Address	Bit		Settings
		Bit #	Field Name	
First Supply Detection	0x20	7	PU_ON_FSD	0h = FSD Disabled 1h = FSD Enabled
Pin Configuration		5-4	EN_PB_VSENSE_CONFIG	0h = Enable 1h = Push Button 2h = VSENSE 3h = Enable
Deglitch Time		3	EN_PB_VSENSE_DEGL	see register map on data sheet

5.8 Changing I2C Address

The TPS6521*405* has the default I2C address configured as 0x30. This configuration can be changed if needed in the *I2C_ADDRESS_REG*, at register 0x26 and changing the default 0x30 address as shown in Figure 5-7. Once the register is changed, the new value must be saved into the NVM by writing 0x0A to register 0x34.

Note

When using multiple TPS65214 devices in multi-PMIC configuration, each device must have a unique I2C address. The I2C address for the 2nd, 3rd and other PMICs must be changed from the default 0x30 to a new value.

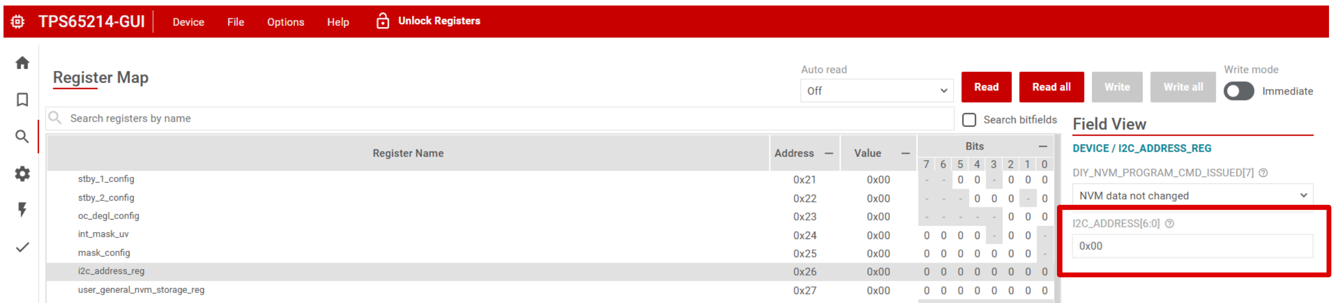

Figure 5-7. I2C_ADDRESS_REG

Table 5-17. I2C_ADDRESS_REG

Register Address	Bit	
	Bit#	Field Name
0x26	6-0	I2C_ADDRESS

5.9 Configuring Mask Settings

There are several interrupt settings that can be masked to bypass specific PMIC monitoring features or modify how PMIC reacts when interrupts are detected. The interrupts that can be masked include undervoltage monitoring, temperature monitoring, among others. Figure 5-8 shows the mask settings in the configuration tab of the GUI.

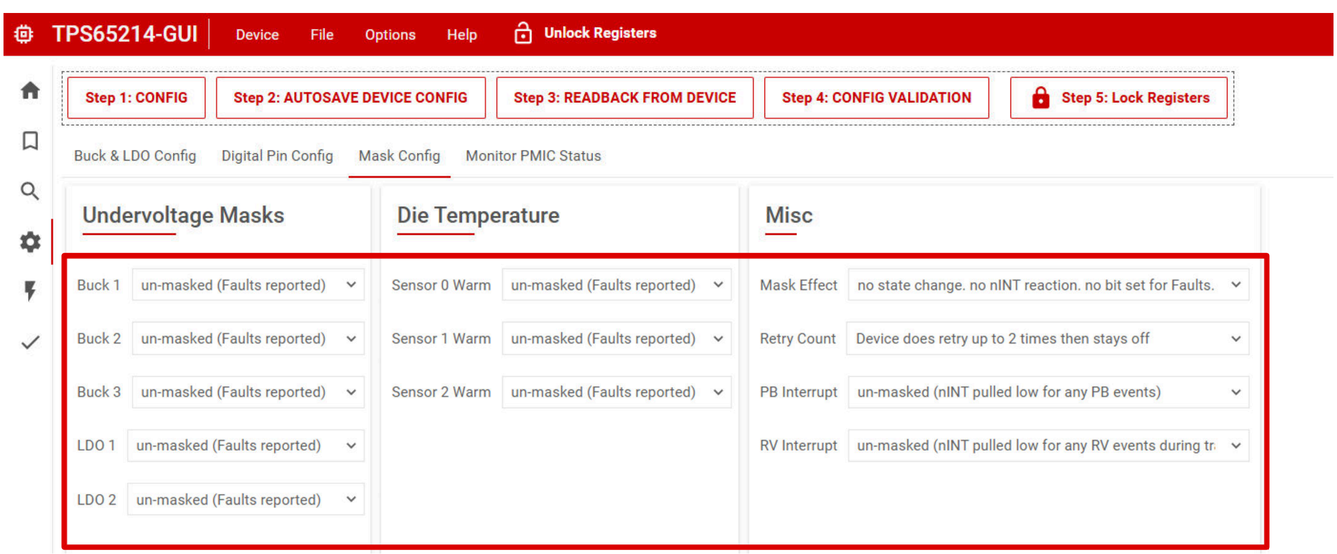

Figure 5-8. Mask Settings in TPS65214-GUI

Table 5-18. MASK Settings on Register 0x1E

Register Address	Bit	
	Bit#	**Field Name**
0x1E	7	BYPASS_RV_FOR_RAIL_ENABLE

Table 5-19. MASK Settings on Register 0x1E

Register Address	Bit	
	Bit#	**Field Name**
0x24	7	MASK_RETRY_COUNT
	6	B3_UV_MASK
	5	B2_UV_MASK
	4	B1_UV_MASK
	2	LDO1_UV_MASK
	1	LDO2_UV_MASK

Table 5-20. MASK Settings on Register 0x1E

Register Address	Bit	
	Bit#	**Field Name**
0x25	7	MASK_INT_FOR_PB
	6-5	MASK_EFFECT
	4	MASK_INT_FOR_RV
	3	SENSOR_0_WARM_MASK
	2	SENSOR_1_WARM_MASK
	1	SENSOR_2_WARM_MASK

5.10 Exporting an NVM Configuration File

To save the configuration of the registers, the NVM settings can be exported as a single file. The TPS65214-GUI currently supports three file types: .csv, .json, and .txt file formats. Figure 5-9 shows how to export the NVM settings using the File tab. The file format must be selected on "Save File Type(s)" before using the "Save Register Map".

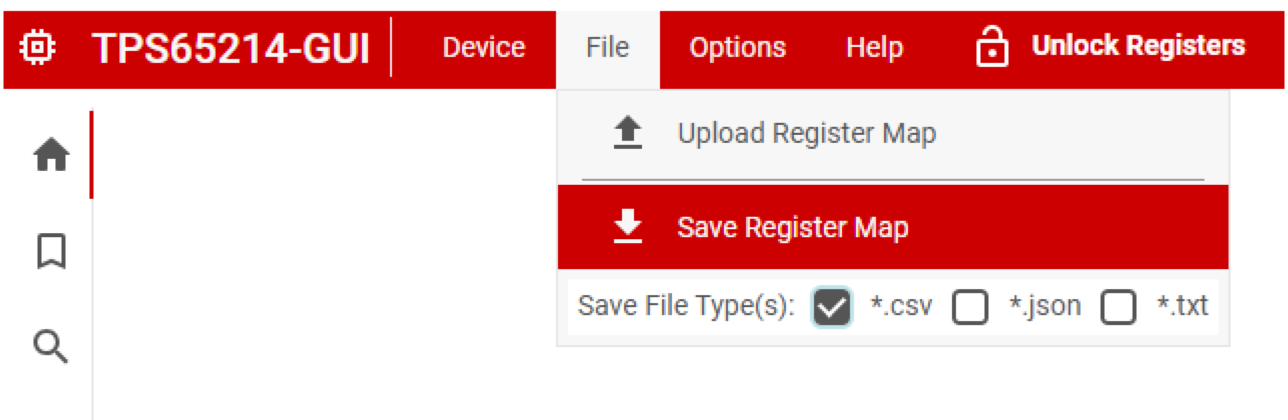

Figure 5-9. Export NVM Settings Using TPS65214-GUI

5.11 Loading a NVM Configuration File to PMIC

The diagram shown in Figure 5-10 describes the process to load a pre-configured NVM file (.csv, .json, or .txt format) into the PMIC NVM. Customers may reuse their pre-configured files to re-program the PMICs on their production line or work with a third-party programming service.

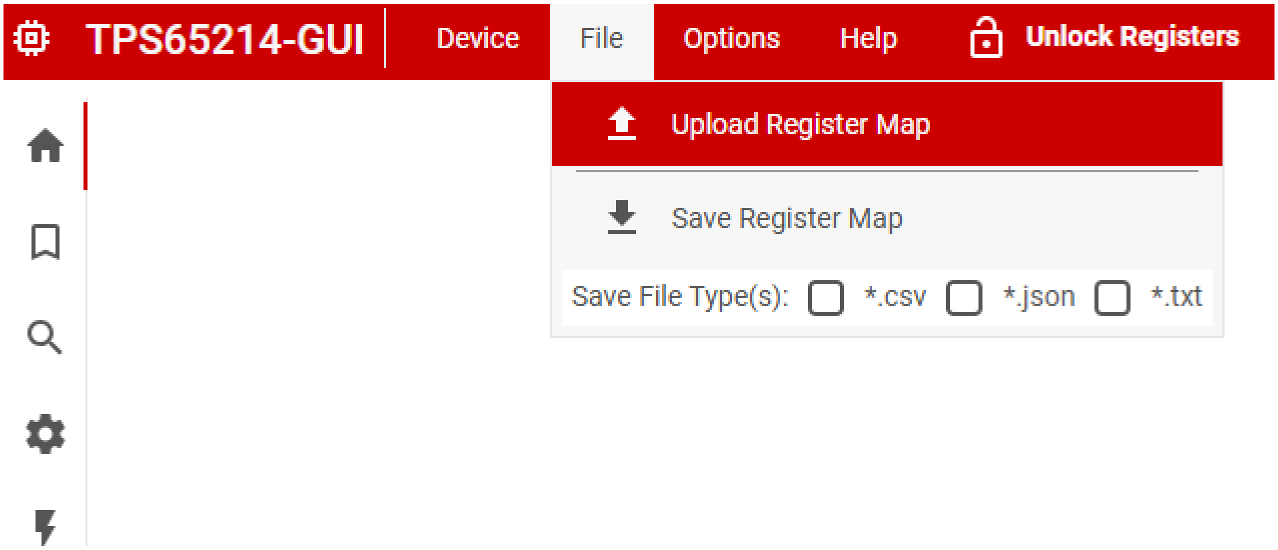

Figure 5-10. Loading NVM Configuration File

6 NVM Programming

Once the register map is fully updated and configured to the desired settings, the new values can be saved into the NVM by applying 8V to the MODE/STBY pin and writing 0x0A to address 0x34.

NVM values can only be successfully changed when the MODE/STBY pin is set to a voltage equal or greater than $V_{MODE/STBY\_PROG}$. The TPS65214EVM-SKT makes this easier by featuring an on-board boost converter that can be shorted to MODE/STBY with a switch and jumper.

The I2C pins must also be pulled up to an external 3.3V supply.

Note

Writing 0x0A (CUST_PROG_CMD) to register address 0x34 commits the current register settings to the device NVM, becoming the new power-up defaults. Once the NVM is reprogrammed, the DIY_NVM_PROGRAM_CMD_ISSUED bit is set, and the device ignores any subsequent CUST_PROG_CMD commands to prevent inadvertent changes to the NVM.

Note

Customer programmable registers correspond to addresses 0x2 to 0x27. Only bits marked with (X) in the reset column of the register map have NVM programmable default settings. All other bits keep the factory settings listed in the register map. Additionally, NVM programming is unidirectional: any configurable bit can only be written from '0' to '1'.

Figure 6-1 shows the button that saves the register settings into the NVM when using the programming tab of the TPS65214-GUI.Table 6-1 shows the register field to be written when referring

to the register map.

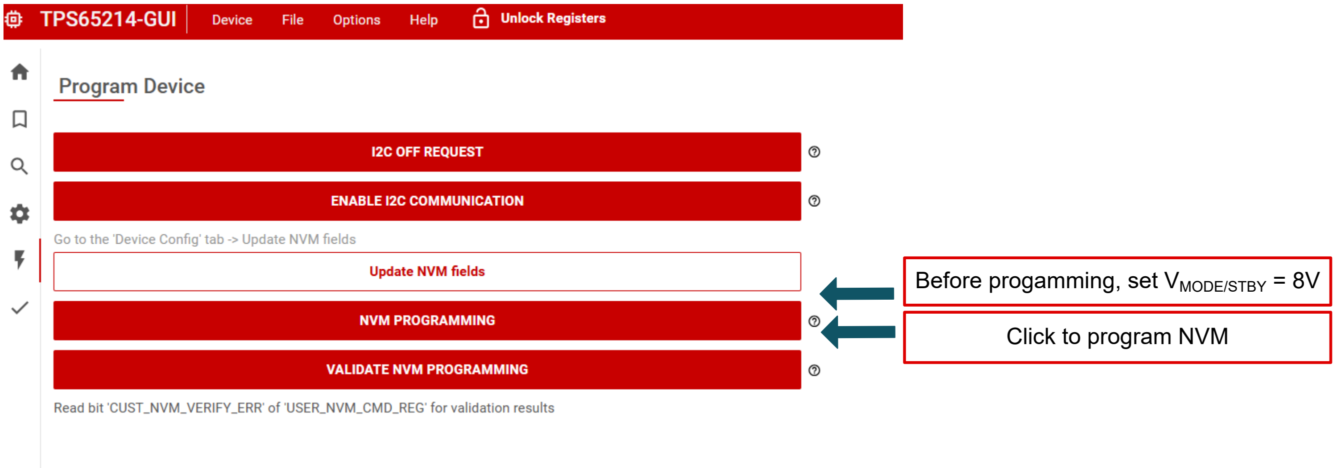

Figure 6-1. NVM Programming Using TPS65214-GUI

Table 6-1. I2C Write to Save Register Settings into NVM

Register Address	Bit		Data
	Bit#	Field Name	
0x34	3-0	USER_NVM_CMD	0x0A

7 Non-NVM Registers

The PMIC register map contains NVM and non-NVM bits. Register addresses 0x00 to 0x27 contains the NVM bits which are backed up by EEPROM. These register settings can be changed by I2C and default values can be re-programmed as described in this programming guide. The reset value for each of the NVM bits may be marked as "X" in the data sheet register map as the bit value can be re-programmed and may be unique for each orderable part number.

Non-NVM bits are located in register addresses 0x28 to 0x37. These registers settings can be changed by I2C but the default values cannot be re-programmed. Register settings for non-NVM bits go back to their default values after a power cycle and every time the PMIC enters Initialize state. The default value for non-NVM bits can be found in the data sheet register map, under "Reset" column.

Default values for the blank user-programmable part TPS6521405VAFR part are set to 0h except for the fields listed in the following section, Section 8.

8 TPS6521405 default settings

The user-programmable TPS6521405VAFR variant comes with all Bucks, LDOs and GPIOs disabled by default and the corresponding output voltage registers set to the lowest values. All register settings are configured as 0h except the fields listed in the tables below.

Table 8-1. TPS6521405VAFR NVM registers NOT configured as 0h

Register Address	Field Name	Value	Description
0x00	TI_DEV_ID	0x44	TPS6521405 identifier
0x01	NVM_ID	0x05	TPS6521405 identifier
0x26	I2C_ADDRESS_REG	0x30	I2C address upon reprogramming set to 0x30 by default

Table 8-2. TPS6521405VAFR non-NVM registers NOT configured as 0h

Register Address	Field Name	Value	Description
0x28	MANUFACTURING_VER	X	Silicon Revision version. 0x01 = Revision 1, 0x02 = Revision 2, ...
0x2A	DISCHARGE_CONFIG	0x37	Discharge enabled

A References

1. Texas Instruments, TPS6521405 data sheet
2. Texas Instruments, TPS65214EVM-SKT user's guide