TPS65219 Programming Instructions

Texas Instruments

Power Management IC

June - 2022

TPS65219 Programming Instructions | Considerations

Things to consider when re-programing the DIY NVM on a prototype board with PMIC+Processor/SoC:

- DIY PMIC should be in Initialize state during NVM programming. Some register settings (i,.e BUCKS_CONFIG, LDOx_LSW_CONFIG) should not be re-programmed in Active state when the rails are enabled.
- For prototype boards, if one of the PMIC rails is used to supply VIO (i.e. pull-up supply for I2C) it wont be available in Initialize state because all the PMIC rails will be OFF. This will require an external I2C master (i.e USB2ANY) and an external 1.8V or 3.3V VIO to enable the I2C communication.
- After applying the external VIO-voltage, the EN_OSC_DIY command must be send immediately (within ~10seconds). Since regulators are disabled, their active discharge are enabled and will dissipate the power from the external VIO-supply. Discharge is disabled after the EN_OSC_DIY command is received. See step#4 in the programming instructions for information on how the "EN_OSC_DIY" command is executed.
- Reprogramming the frequency mode (BUCK_FF_ENABLE) is not allowed in any of the PMIC states.

Note: This preliminary short datasheet contains more details than we would normally include prior to APL, as *they are subject to change without notice*. We will not keep a version-history of this interim release.

TPS65219 Programming Instructions | Programming from Initialize State

1. **Connect input supply/VSYS**
   – Apply 3.3V or higher to VSYS, PVIN_Bx (also to PVIN_LDOx if they are supplied by the same pre-reg as VSYS).

2. **Send PMIC to Initialize State (only needed if PMIC was in the Active State)**
   - Send the PMIC to Initialize state by sending an OFF request through I2C. This requires writing a 0x01 to bit0 (I2C_OFF_REQ) on register 0x29 (MFP_CTRL Register).

     → [I2C command: write 0x01 to register 0x29]

3. **Apply external 3.3V to VIO**
   - External pull-up supply for I2C pins (SDA/SCL)

4. **Enable OSC (EN_OSC_DIY)**
   - Once the PMIC is in Initialize state and all the rails are disabled, the oscillator needs to be enabled for I2C communication. This requires writing 0x9 to bits 3-0 (USER_NVM_CMD) on register 0x34 (USER_NVM_CMD_REG Register).

     → [I2C command: write 0x09 to register 0x34]

5. **Update all registers with the desired values**
   - Example#1: if changing Buck2 output voltage to 3.3V, the I2C command would be wiring 0xB3 to register 0x09
   - Example#2: if changing the LDO4 output voltage to 1.8V, the I2C command would be writing 0x98 to register 0x04

Note: This preliminary short datasheet contains more details than we would normally include prior to APL, as *they are subject to change without notice*. We will not keep a version-history of this interim release.

TPS65219 Programming Instructions | Programming from Initialize State

6. **Save the new values register values into the EEPROM**
   - This will require writing 0xA to bits 3-0 (USER_NVM_CMD) on register 0x34 (USER_NVM_CMD_REG).

     → [I2C command: write 0x0A to register 0x34]

7. **Optional: Validate the EEPROM changes**
   - Compare the new EEPROM content with the register settings for the customer programmable registers. This will require writing 0x07 to bits 3-0 (USER_NVM_CMD) on register 0x34 (USER_NVM_CMD_REG).

     → [I2C command: write 0x07 to register 0x34]

8. **Disable OSC**
   - This turns OFF the OSC and disconnect the external 3.3V VIO. The same I2C command also activates the active discharge. This requires writing 0x06 to bits 3-0 (USER_NVM_CMD) on register 0x34 (USER_NVM_CMD_REG).

     → [I2C command: write 0x06 to register 0x34]

9. **Recommended to perform a power cycle and confirm the new settings**
   - After power cycle, once the device is in active state, check the new settings (i.e. new output voltages) to confirm they because the new power-up defaults.
   - Read bits 4-3 (STATE) on register 0x35 (POWER_UP_STATUS_REG) to confirm PMIC is in Active state. (Active state correspond to 0x11 settings on bit 4-3)
   - Read Interrupt register (INT_SOURCE/address: 0x2B) to confirm there are no interrupts.

Note: This preliminary short datasheet contains more details than we would normally include prior to APL, as *they are subject to change without notice*. We will not keep a version-history of this interim release.

Important notice and disclaimer

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), 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, or other requirements.

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

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