UM12433

KITPF09FRDMPGM Programming board

Rev. 1.0 — 20 November 2025 User manual

Document information

Information	Content
Keywords	PF09 NXP GUI, SBC, Body and Comfort, PF09, NXP GUI, IMX9x, ISO 26262
Abstract	The KITPF09FRDMPGM programming board user guide is intended for the engineers involved in the evaluation, design, implementation, and validation of PF09 fail-safe system basis chips.

1 Introduction

The KITPF09FRDMPGM programming board user manual is intended for engineers involved in the evaluation and design of PF09 PMIC for high-performance applications.

The KITPF09FRDMPGM programming board supports the development on the PF09 family of devices. This document covers connecting the hardware, installing the NXP GUI software, configuring the environment, and using the kit. It includes guidance on how to use the registers, try OTP configurations, and burn the part.

The KITPF09FRDMPGM programming board is a socketed board supporting PF09 PMIC with 5 SMPS and three LDOs. In this document, "PF09" refers to these supported devices.

The devices can be placed and removed from the board using the socket.

2 Finding kit resources and information on the NXP website

NXP Semiconductors provides online resources for this evaluation board and its supported device(s) on http://www.nxp.com.

The information page for KITPF09FRDMPGM program board is at https://www.nxp.com/products/PF09. The information page provides overview information, documentation, software, and tools, parametric, ordering information.

2.1 Collaborate in the NXP community

The NXP community is for sharing ideas and tips, asking and answering technical questions, and receiving input on just about any embedded design topic.

The NXP community is at http://community.nxp.com.

3 Getting ready

Working with the KITPF09FRDMPGM requires the kit contents, more hardware, and a Windows PC workstation with installed software.

3.1 Kit contents

The KITPF09FRDMPGM kit contains the following items:

- Assembled and tested evaluation board with preprogrammed KL25Z microcontroller in an antistatic bag
- 3.0 ft USB-STD A to USB-B-micro cable
- Jumpers mounted on board

3.2 Additional hardware

In addition to the kit contents, the following hardware is necessary and is beneficial when working with this kit:

- Power supply capable of providing 5 V

3.3 Minimum system requirements

This evaluation board requires a Windows PC workstation.

- USB-enabled computer with Windows 7 or Windows 10
- FTDI USB serial port driver (for FT230X basic UART device)

3.4 Software

Installing software is necessary to work with KITPF09FRDMPGM Program board.

All listed software is available on the information page of the evaluation board at 9-Channel PMIC for High-Performance Applications, ASIL D and SIL-2.

- NXP GUI for automotive family installation package - latest version

4 Getting to know the hardware

The KITPF09FRDMPGM kit provides a platform for supporting designs based on the PF09 Family of NXP, not for evaluation. All PF09 Family features can be configured and programmed using NXP GUI software by accessing the registers in **Read** and **Write** mode. Output voltages, digital signals (I^2C, RSTB) are accessible through hook connectors.

4.1 Kit overview

The hardware of the kit consists of the KITPF09FRDMPGM program board embedded with a KL25Z microcontroller, and the USB cable required to connect the board to the PC.

The KITPF09FRDMPGM program board features a socket that allows the user to fuse an individual PF09 Family using the device's two time programmable (OTP) feature without extra tools. The board also contains LEDs and test points that provide a means of limited monitoring performance in real time. An emulation mode allows the user to test as many configurations as needed before programming the part.

The KL25Z is connected on the bottom side of the KITPF09FRDMPGM programming board. The role of the KL25Z is to manage I^2C communication between the KITPF09FRDMPGM program board and the GUI installed on the PC. The KL25Z draws power either from the USB cable connected to the PC or from the battery supply (when not connected to the GUI).

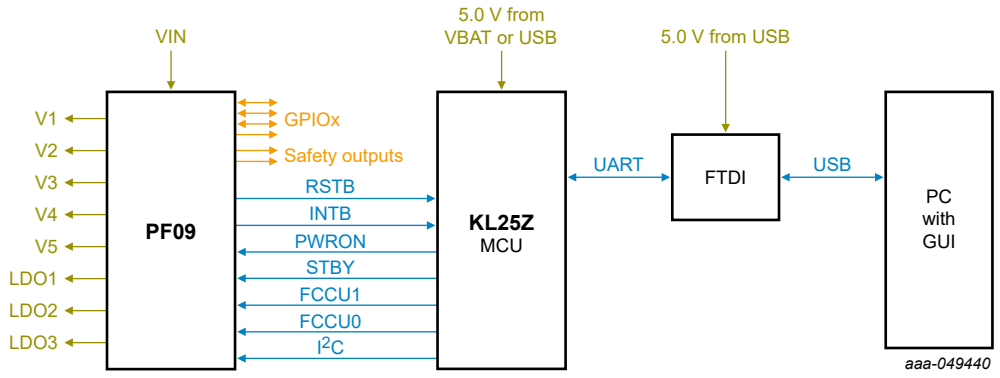


Figure 1. Overview diagram of communication between PF09 and NXP GUI

Note: *This document uses the PF09 as a default configuration. Specificities for PF are explained in the flow if necessary. Otherwise, the features are common to PF09.*

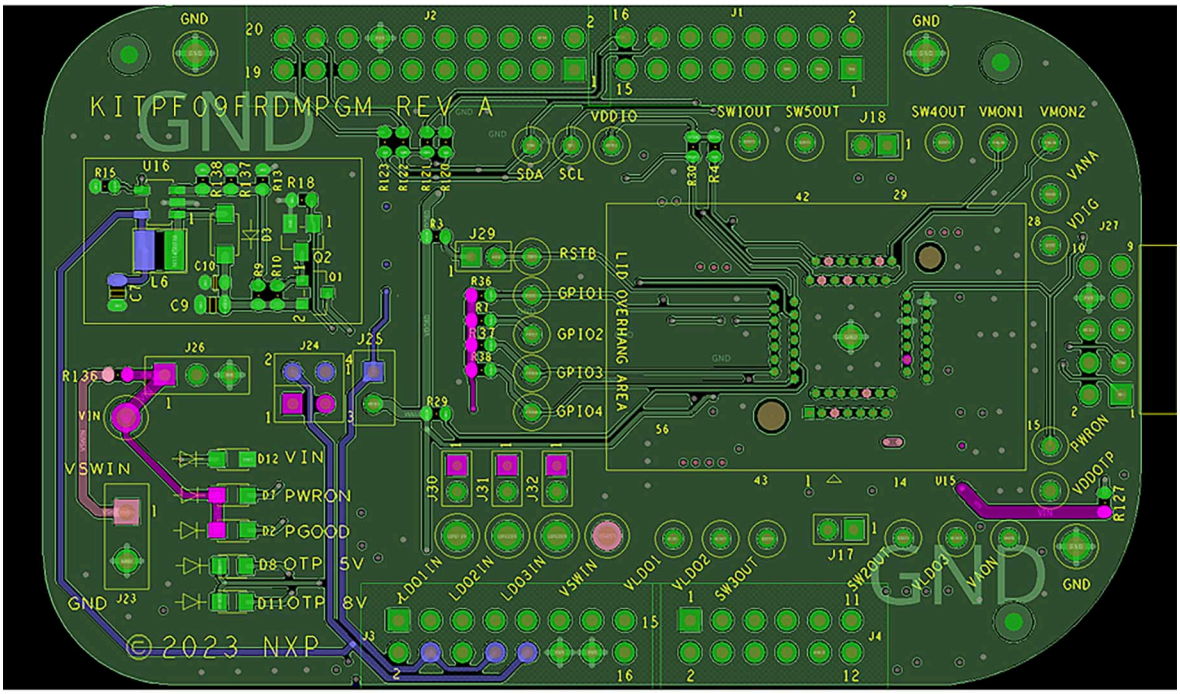
4.2 Board features

- Header connectors for I/O configuration
- Selectable power input for programming
- Access pin connectors for accessing inputs and outputs
- Indicator LEDs green to indicate PGOOD, PWRON, and VIN

- RED LED to indicate that the OTP 8 V burning voltage is set and BLUE LED to indicate that the OTP 5 V voltage is set
- Advanced system monitoring via the KL25Z MCU
- Embedded USB to I^2C protocol for easy connection to the software GUI through the KL25Z MCU

4.3 KITPF09FRDMPGM featured components

Figure 2 shows the placement of connectors, configuration headers, and LEDs signaling. For information on the default jumper and switch configuration for PF09, see Section 7.

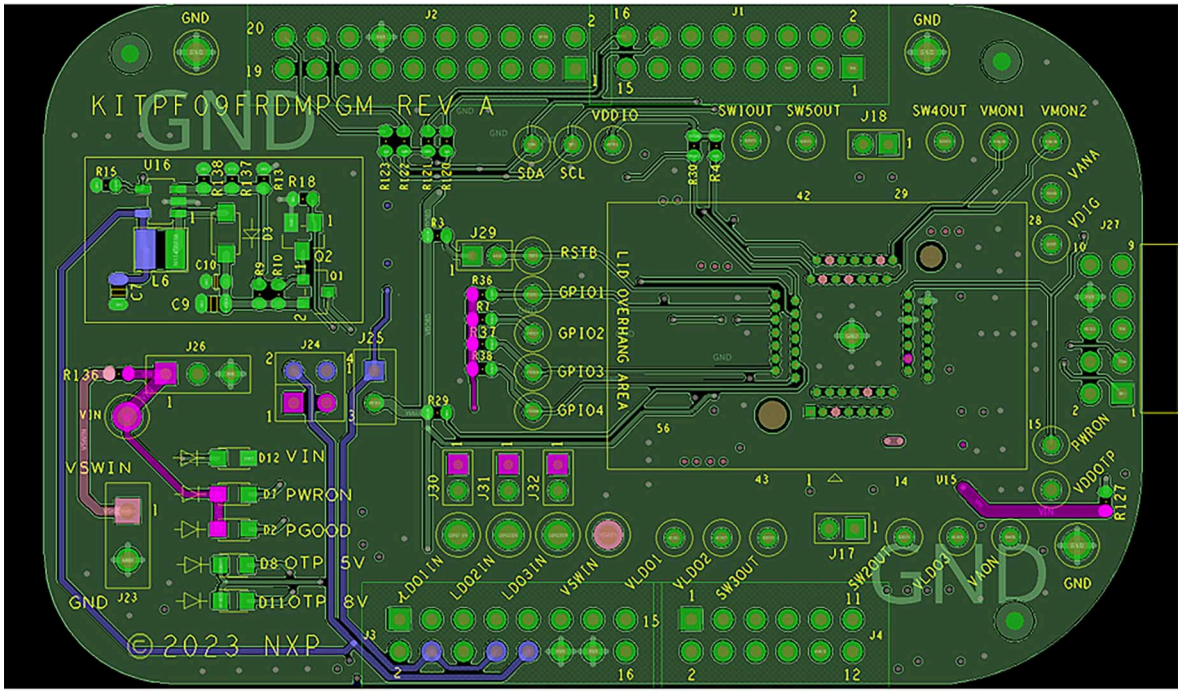


aaa-063850

Figure 2. Location of the KITPF09FRDMPGM featured components

4.4 KITPF09FRDMPGM program board

This section describes the KITPF09FRDMPGM program board by explaining the features associated with each jumper and switch, giving information on hardware configuration to enable these features, and providing advice on LED signaling and test points available on the board.



aaa-063850

Figure 3. Location of the KITPF09FRDMPGM featured components

The default debug configuration enables the board to be fully controlled by the KL25Z MCU (via I^2C) and the GUI. Figure 3 shows the default jumper and switch positions for PF09. See Table 1 below for PF09 default jumpers and switches configuration.

Table 1. Configuration jumpers

Connector	Pin count	Line	Connection
J1/J2/J3/J4	---	Freedom board headers	---
J17	1	SW2OUT	Open
	2	SW3OUT	Open
J18	1	SW4OUT	Open
	2	SW5OUT	Open
J29	1	Pullup for RSTB	**Jumper connected**
	2	RSTB	
J23	1	VIN	Open
	2	GND	Open
J24	1	VIN	Open
	2	USB_PWR	
	3	VIN	Open
	4	3V3_MCU	
J25	1	VDDIO	**Jumper connected**
	2	3V3_MCU	
J26	1	VIN	Open

Table 1. Configuration jumpers*...continued*

Connector	Pin count	Line	Connection
	2	PWRON	Open
	3	GND	Open
J27	1	SCL	Open
	2	PWRON	Open
	3	SDA	Open
	4	VDDOTP	Open
	5	IO1	Open
	6	VDDIO	Open
	7	R127 0 ohm to VIN	Open
	8	GND	Open
	9	Open	Open
	10	Open	Open

Table 2. Access point functions

Function	Description	Color
LDO1IN	Monitoring point for the LDO1 input voltage	Red
LDO2IN	Monitoring point for the LDO2 input voltage	Red
LDO3IN	Monitoring point for the LDO3 input voltage	Red
VSWIN	Switchers input voltage	Red
VIN	External Input voltage	Red
VLDO1	Monitoring point for the LDO1 output	White
VLDO2	Monitoring point for the LDO2 output	White
VLDO3	Monitoring point for the LDO3 output	White
GND	Board grounds direct connection -4 easy access points	White
VAON	Monitoring point for the VAON output	White
SW1 Output	Monitoring point for the SW1 output	White
SW2 Output	Monitoring point for the SW2 output	White
SW3 Output	Monitoring point for the SW3 output	White
SW4 Output	Monitoring point for the SW4 output	White
SW5 Output	Monitoring point for the SW5 output	White
GPIO1	GPIO1 access point – bidirectional pin	White
GPIO2	GPIO2 access point – bidirectional pin	White
GPIO3	GPIO3 access point – bidirectional pin	White
GPIO4	GPIO4 access point – bidirectional pin	White
VMON1	Monitoring point for the external monitor VMON1	White
VMON2	Monitoring point for the external monitor VMON2	White

Table 2. Access point functions*...continued*

Function	Description	Color
VDIG	VDIG monitoring point	White
VANA	VANA monitoring point	White
PWRON	Power on input pin access point	White
VDDIO	VDDIO Monitoring point	White
RSTB	System reset signal output monitoring point	White
VDDOTP	VDDOTP Monitoring point	White
I2C-SDA	I^2C SDA access point – bidirectional pin	White
I2C-SCL	I^2C SCL access point – bidirectional pin	White

Note: *Test-points are routed directly to the pins.*

4.4.1 Voltage selection and configuration

The input voltage supply can come from a power via hook connector VIN (external power). Follow Table 1. The use of an external power supply is recommended as the USB capability may be limited.

Input range is from 3.3 V to 5 V and matching to VIN in OTP configuration.

VDDOTP requires 8 V for PMIC programing (VDDOTP_BST), voltage provided in the program board.

4.4.2 Output voltages

All buck switchers and LDOs can be configured at different output voltage, up to 3.3 V. LDOs can also be configured as a simple load switch. Output voltage uses discrete values and not all values are available. This is done via OTP configuration or I^2C communication.

Table 3. Output voltage

Function	Output voltage range
SW2 output	From 0.3 V to 3.3 V
SW3 output	From 0.3 V to 3.3 V
SW4 output	From 0.3 V to 3.3 V
SW5 output	From 0.3 V to 3.3 V
SW1 output	From 0.5 V to 3.3 V
VLDO1	From 0.75 V to 3.3 V
VLDO2	From 0.65 V to 3.3 V
VLDO3	From 0.65 V to 3.3 V

4.4.3 MCU communication I^2C selection

Table 4. MCU communication I^2C selection

Signal name	Description
SDA	I^2C is selected as the MCU communication protocol.
SCL	I^2C is selected as the MCU communication protocol.

4.4.4 Debug and OTP configuration

During normal system operation, the DBGOFF (Debug Off), Section 7.3.1, state is a transitory state between a power on event and the power up sequence. It serves as the gating state to ensure that the PMIC is ready to start a power up sequence, and allow synchronization of two or more PMICs providing full power architecture to a complex system.

VDDIO supply is required, in programmer board, VDDIO is fixed, so the device is able to communicate through the I^2C bus during the DBGOFF state, allowing the PMIC to operate in development/debugging modes. In such scenarios, the DBGOFF state becomes a full biased OFF state to allow full access to the functional and OTP registers to enable OTP emulation and/or OTP fuse programming.

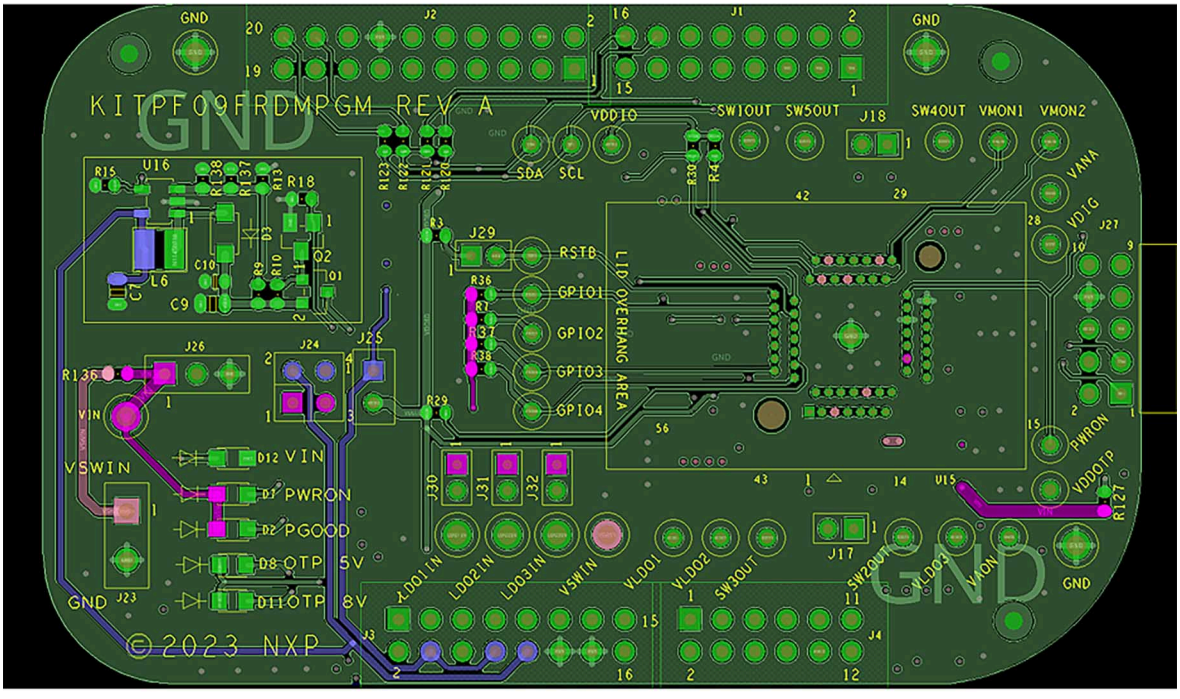
OTP mode is active when the DEBUG pin is set to 8 V. Once in OTP mode, specific keys are necessary to enter test mode, see Section 7.3.2, or OTP programming process.

4.4.5 LED signaling

LED signaling displays the state of the following signals:

- Signals: PGOOD, PWRON
- Power supply: VIN
- OTP control: VDDOTP - 5 V, VDDOTP - 8 V

The VIN LED is permanently on. OTP control LEDs are either 5 V or 8 V status control signals. Consider them with current consumption measurements for better analysis.



aaa-063850

Figure 4. Location of the KITPF09FRDMPGM featured components

4.4.6 Schematic layout and bill of materials

The board layout and bill of materials for the KITPF09FRDMPGM Program board are available at https://www.nxp.com/products/PF09.

5 Installing and configuring software tools

Note: *This section will use the PF09 as an example to explain software installation, configuration, and use. Specific explanations for PF09 are added on the side if necessary. Otherwise, the method is general to PF09.*

5.1 Flashing KL25Z MCU GUI firmware

The KITPF09FRDMPGM is delivered with the KL25Z firmware already flashed.

5.2 Installing the PF09 NXP GUI software package

To install the PF09 NXP GUI, first download or obtain the NXP GUI package, then follow the instructions below:

1. Open the NXP GUI package. Unzip and open the 1-NXP_GUI_Setup folder. Current version: v3.1.499). Same installation procedure.

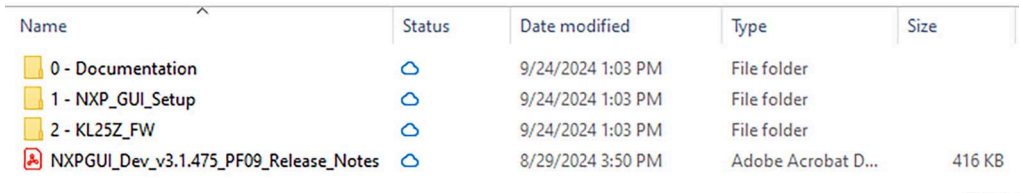


Figure 5. GUI folder list

2. Double click **NXP_GUI-version-Setup.exe** to launch the application and follow the instructions.

Figure 6. NXP_GUI-version-Setup.exe

3. Click **Next** when the initial application setup window appears.
4. On the License Agreement window, read the license information and click **I Agree**.

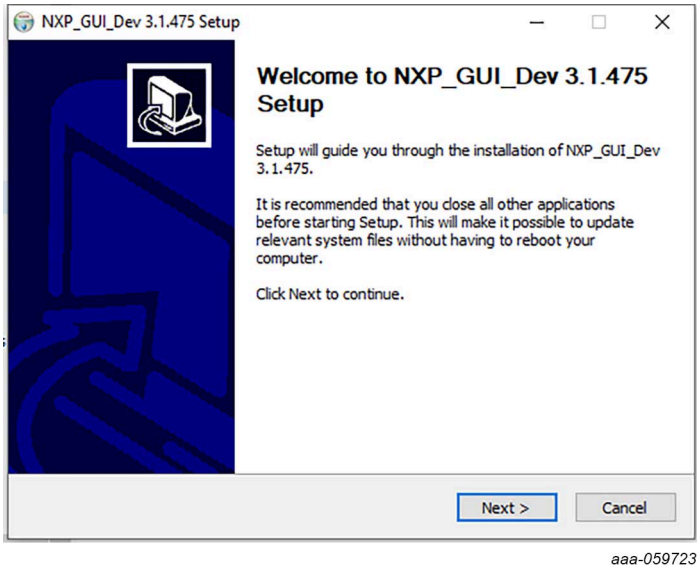
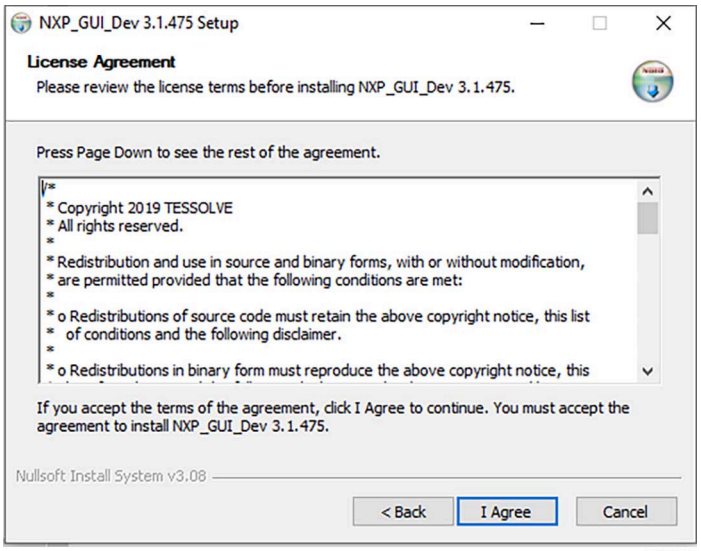


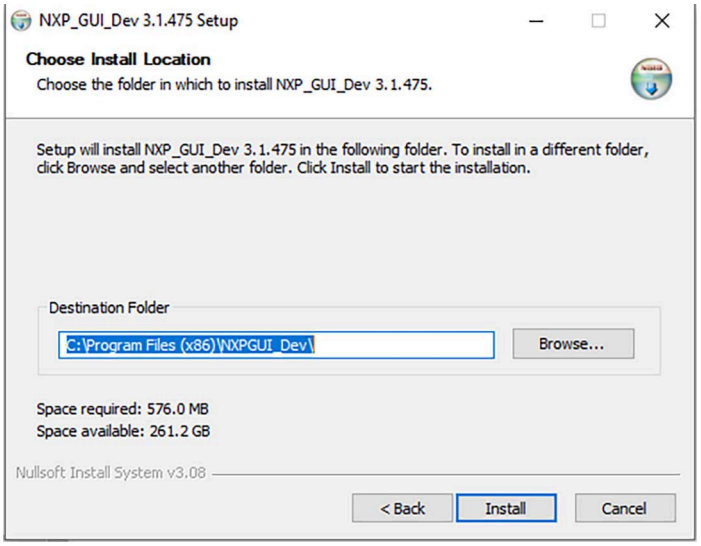
Figure 7. GUI setup window



aaa-059724

Figure 8. GUI licence agreement

5. In the Choose Components window, select the GUI components to install, then click **Next**.



aaa-059726

Figure 9. GUI choose install location

6. In the "choose install location window", choose the folder to install the GUI.

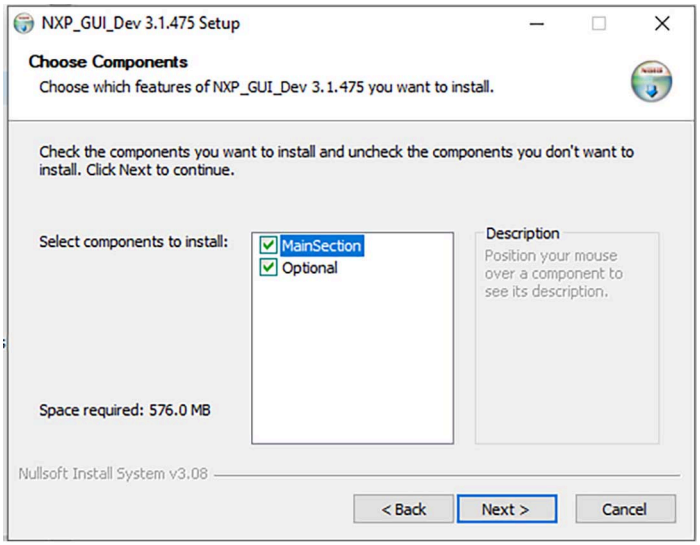


Figure 10. GUI choose components to install

7. In the Completing Setup window, select the following options:

- Run NXP_GUI
- Show read me

Then click **Finish** to complete the installation.

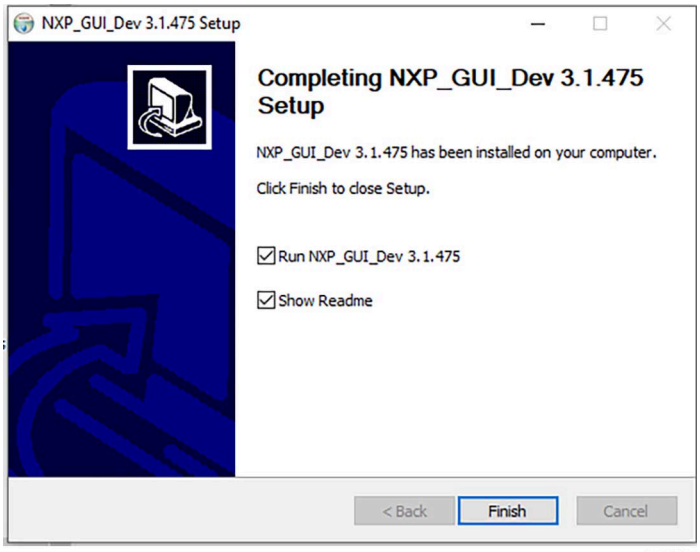


Figure 11. Complete setup

5.3 Launching the PF09 NXP GUI

When the KITPF09FRDMPGM kit is set up and the GUI is installed, follow the steps below to launch the GUI:

1. Click the Windows icon (bottom-left corner) and locate NXPGUI in the Windows all apps bar, then click the NXPGUI icon to launch the GUI.

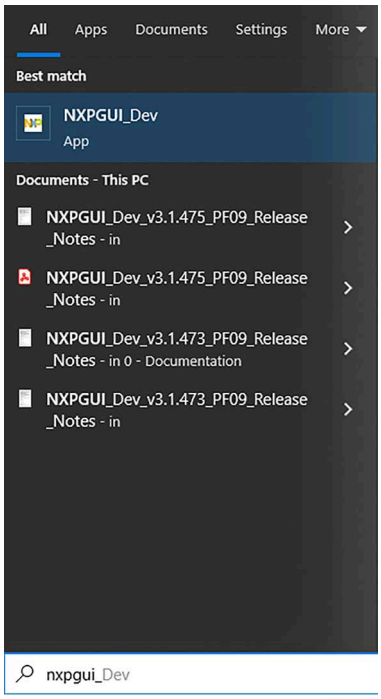


Figure 12. Launch the GUI

2. When the GUI opens, the first window to appear is the Kit Selection window. In the Kit Selection window, select the settings shown below. When finished selecting the settings, click **OK**

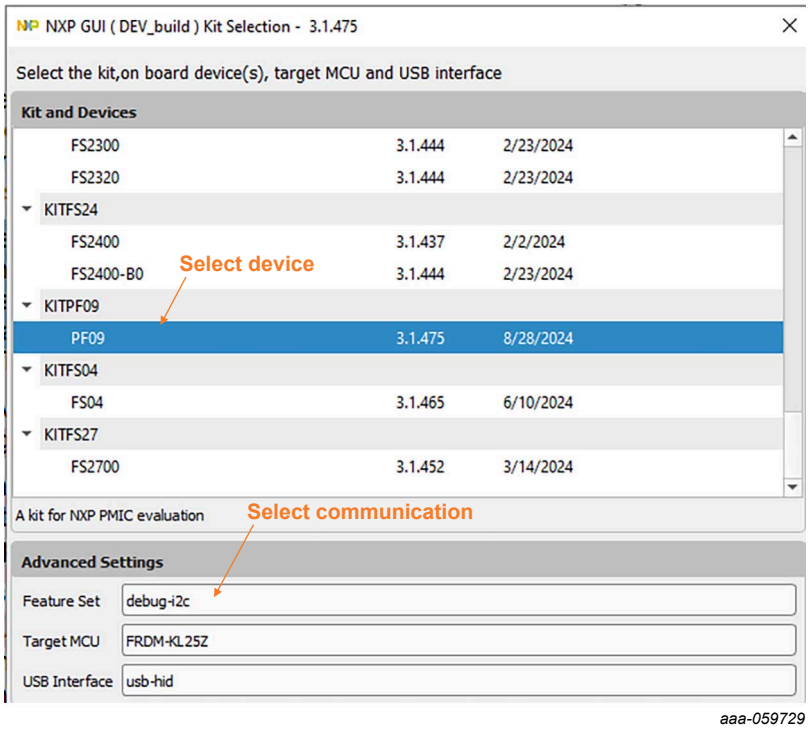


Figure 13. GUI settings

Note: *For PF09, select the corresponding device name.*

For the PF09 part the communication protocol is I^2C.

To avoid the Kit selection window on every launch, check the box **Use this configuration and do not ask again**. The Kit selection window can be enabled/disabled through the File main menu item once the GUI is launched by checking/ unchecking the **Do not display GUI Kit selection at Start box**.

6 PF09 NXP GUI

This section gives guidance on use of the PF09 NXP GUI.

6.1 NXP GUI framework window

The framework window consists of the following sections:

- **Connection toolbar:** Used to start communication with the device, enter or exit Test/OTP modes, fix I^2C frequency and I^2C main address, enable watchdog.
- **Framework settings:** Manages file import/export and framework configuration.
- **Window log:** Reports USB and Device communication events.
- **USB and device status:** Indicates if USB or Device is connected or disconnected, shows communication protocol (I^2C), shows firmware and GUI version, displays current device mode.
- **Tools access bar:** Provides quick access to the PF09 evaluation tools and features.

Note: *Power tool is unavailable*

- **Tab content:** Shows the content of each tool or tab. There may be several tabs, boxes, or windows inside.

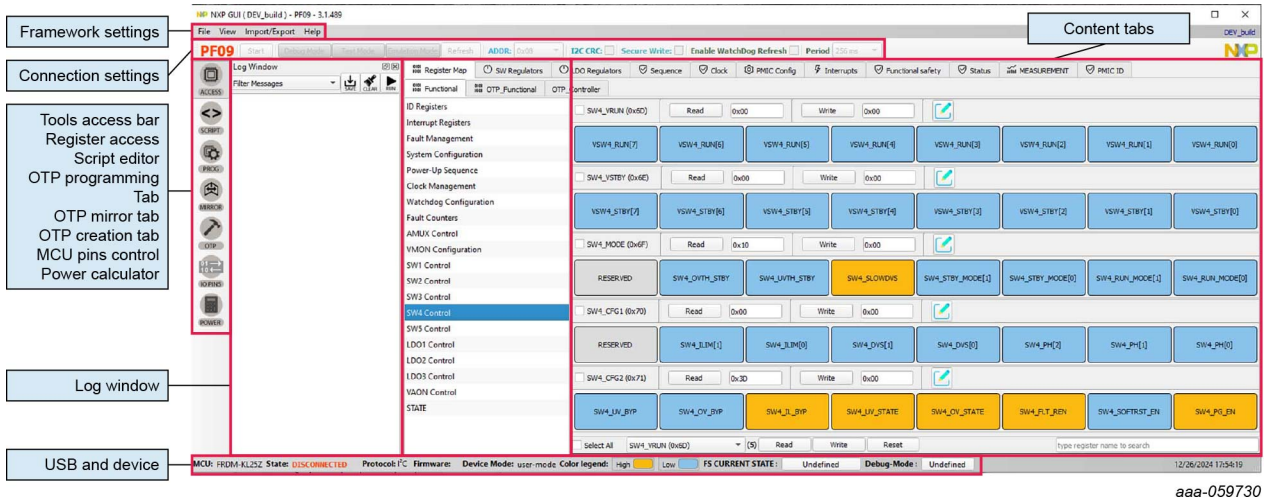


Figure 14. GUI framework

If the Log window, Tools access bar, or connection toolbar do not appear on the screen, right-click on the framework settings bar. This action displays three selection boxes (checked if display is active):

- Log window
- Tab corresponds to the tools access bar
- Connection toolbar

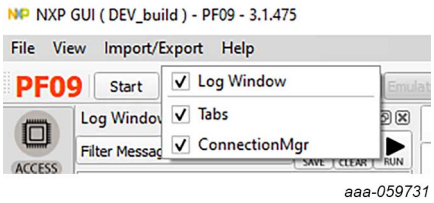


Figure 15. Framework settings bar

Note: *Right click on the Framework or Connection Toolbar.*

6.2 Framework settings bar

The framework settings section appears at the top left corner of the Framework window. It consists of four items:

- **File**
- **View**
- **Import/export**
- **Help**

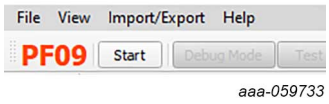


Figure 16. Framework settings bars

6.2.1 File menu item

To choose to load/save a configuration and to choose to display or not the GUI Kit selection at **Start** or **Exit** the application.

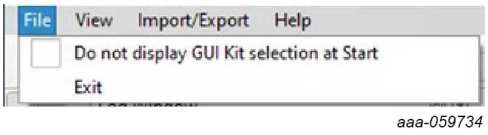


Figure 17. File menu

- **Do not display GUI Kit selection at Start**: Check this box to always open the NXP GUI as the current GUI version. Uncheck this box to display the product selection box at NXP GUI startup.
- **Exit**: Exits the NXP GUI application.

6.2.2 View menu item

The view menu contains the following options:

- **Display**: To enable or disable the Connection toolbar (enabled by default).

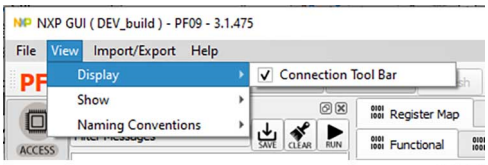


Figure 18. View menu - display

- **Show**: Allows the user to access various sections of the GUI and display the Log window.

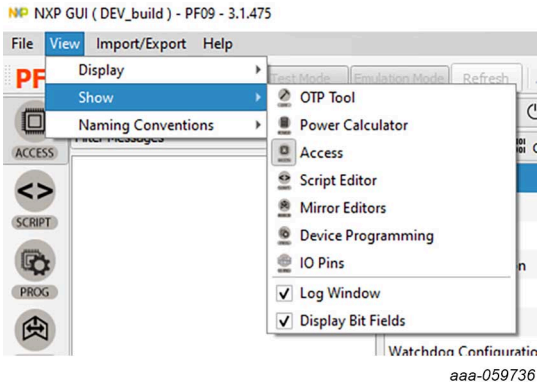


Figure 19. View menu - show

- **Naming conventions**: Allows the user to select **Friendly** or **Register** name display for the OTP tool. Option enabled only when the OTP tool is active.

The naming convention options are:

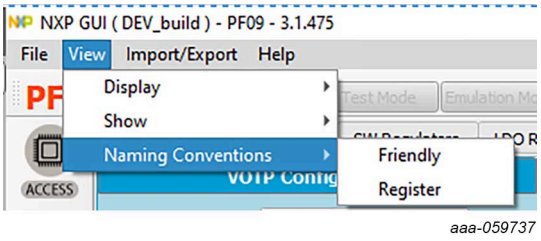


Figure 20. View menu - naming conventions

- **Friendly**: Register names are displayed as user-friendly names in the OTP tool.
- **Register**: Register names are displayed by their technical name in the OTP tool. Example: SW1 power up sequence ⇔ OTP_SW1_SEQ.

6.2.3 Import/export menu item

The **import/export** menu item allows the user to manage the files needed for Mirror emulation, for the PROG tool, and for GUI configuration. This menu item is only active when the OTP tool has been selected. See Section 6.5.2 for details.

- **TBB:** Exports the OTP configuration into a TBB script file that can be used to load the Mirror registers using:

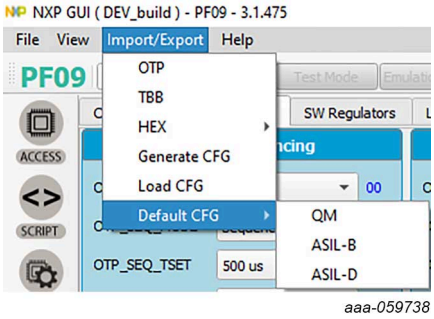


Figure 21. Import/export menu

SCRIPT tool. The same file can be used by the PROG tool to burn OTP fuses.

- **HEX:** Outputs the OTP configuration in Intel-Hex (I-HEX) or Simple Hex (S-HEX) script file format.
- **Save CFG:** To save the current configuration as a CFG file.
- **Load CFG:** To load a previously saved configuration from a CFG file.
- **Default CFG:** To load a predefined OTP configuration in QM or ASIL B to use as a starting point.

6.2.4 Help menu item

The **Help** menu button contains links to additional information, displays GUI and firmware version numbers, and a glossary for acronyms.

- **Documentation:** Lists online NXP documentation related the PF09 GUI.
- **About:** Displays the version number of the GUI currently installed.

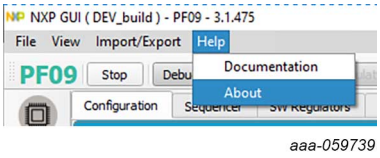


Figure 22. Help menu

6.3 Connection toolbar

The Connection toolbar menu is located directly below the Framework settings menu in the top left corner of the Framework window.

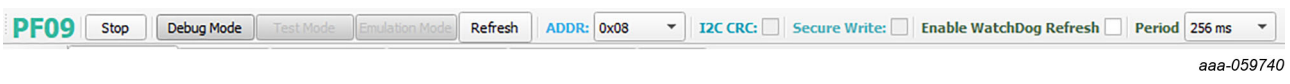


Figure 23. Connection toolbar

Note: *The Connection toolbar is not displayed if not selected in the Frameworks setting>View>Display menu item.*

The Connection toolbar allows the user to start or stop communication with the device, enter or exit Test mode and OTP mode, fix I^2C frequency (depending on chosen MCU communication protocol at start) and I^2C main address, enable watchdog.

The Connection toolbar consists of the following items:

- **Start/stop**: To open or close communication with the device.
- **Mode**: To enter or exit Test mode, and exit OTP mode.

- **Main ADD**: To assign the I^2C address to the device.
- **Enable watchdog refresh**: To enable/disable theWatchdog refresh.
- **Period**: To set the Watchdog period.

6.3.1 Device connection

1. The device connection boxes appear first in the Connection toolbar menu.

aaa-059741

Figure 24. Device connection – Start/stop

2. When the KL25Z MCU is not connected through the USB port, the State indicator in the USB and Status bar shows **Undefined**, the PF09 header text appears red, and the **Start** button is not available.

aaa-059742

Figure 25. Not detected

3. After the USB cable is connected, the State indicator displays **Disconnected** and the **Start** button becomes available.

aaa-059742

Figure 26. Disconnected

4. Click **Start** to start communication with the PF09.
5. At this point, the State indicator displays **Connected** and PF09 header text changes from red to green.

aaa-059743

Figure 27. Connected

6. Usually, once connected, the next step is to load an OTP configuration and write it in the *mirror registers*.

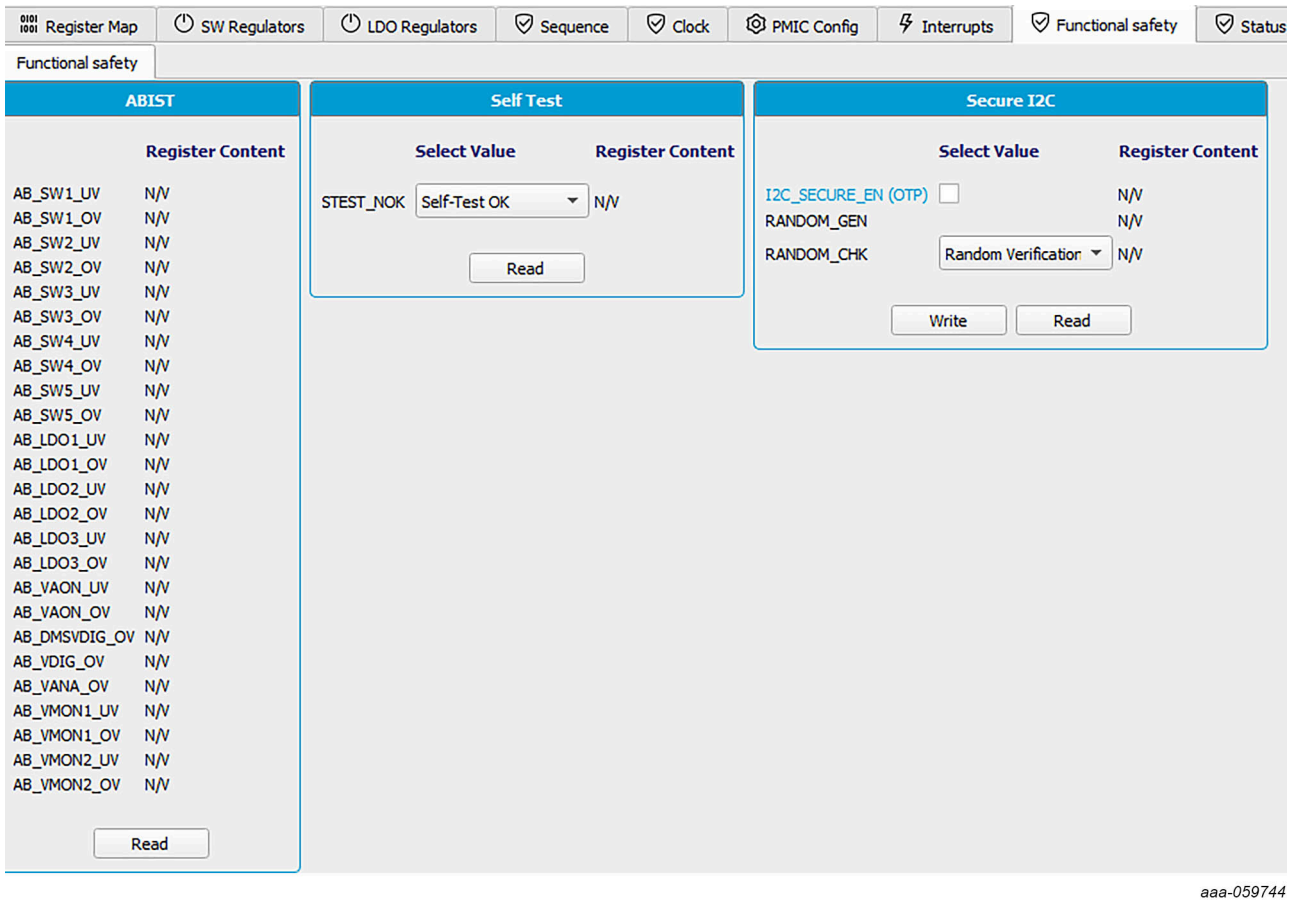


Figure 28. Watchdog enablement and configuration

6.3.2 I²C communication configuration

The PF09 supports both I²C communication protocol. I²C configuration settings for MCU side appear in the *connection toolbar*, allowing to choose the protocol applicable parameters.

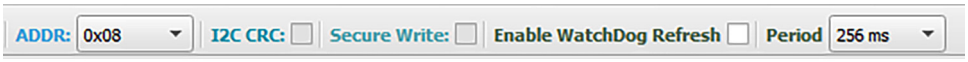


Figure 29. Communication configuration

6.3.3 Watchdog management

The PF09 provides watchdog monitoring with keys as a functional safety feature. Watchdog monitoring requests synchronize actions from the MCU. The watchdog feature can be disabled in QM version only. The watchdog is always enabled for an ASIL B part.

6.3.4 Watchdog enablement and configuration

On the PF09 side, when the part is QM, watchdog disablement is possible by OTP. Watchdog monitoring default configuration is done via I²C via the ACCESS tool.

6.3.5 Watchdog configuration on MCU side

On the KL25Z MCU side, actions are configured with the watchdog management boxes in the Connection toolbar menu.

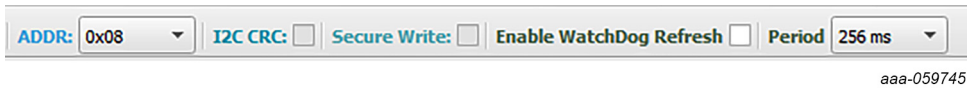


Figure 30. Watchdog management

The mechanism is fully operative when both PMIC and MCU actions are enabled and have matching configurations.

The steps for watchdog enablement are described by Figure 31.

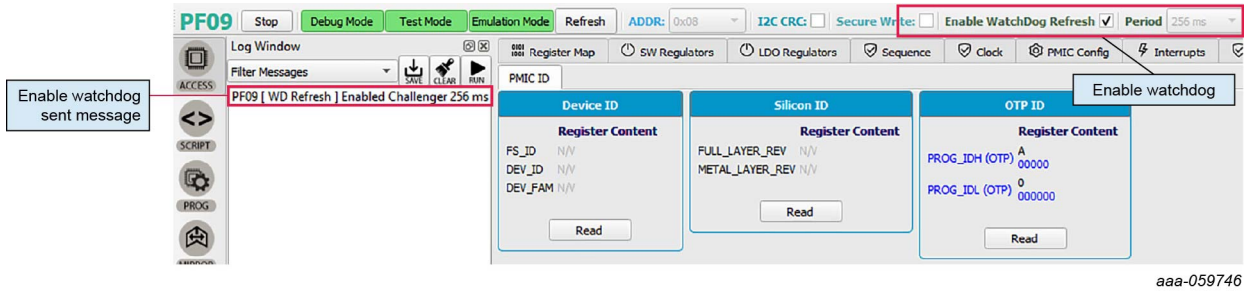


Figure 31. Enable watchdog refresh unchecked

1. Select *watchdog period* via the *period* selection box in the *connection* toolbar.
2. Enable *watchdog refresh* by checking the corresponding box to enable watchdog monitoring on the MCU side.
3. A message is displayed in the *log window* with the selected period and type values.

If the *period selection* box is unavailable, verify that *enable watchdog refresh* is unchecked.

6.4 USB and device status bar

The *USB and device* status bar is at the bottom of the Framework window. This toolbar gives information on the GUI and firmware version, on the USB connection status, and on PF09 active modes.

Figure 32. USB and device status bar

6.5 Tools access bar

The *tools access* bar appears in a vertical row along the left side of the *Framework* window. The bar provides access to tools that implement various GUI functions. The *tools access* bar consists of nine items:

Table 5. Tool access bar

Figure 33. Tool access bar

ACCESS	Provides access to I^2C registers via Register map or thematic tabs
SCRIPT	To create, open, save, and run scripts
PROG	To manage the OTP fuse-burning process
MIRROR	Provides access to Mirror registers
OTP	To create and save OTP configuration files, with customer and program details
IO PINS	To read and set MCU I/O pins
POWER	To compute power dissipation of the device in a given application

6.5.1 POWER

The POWER tool is used to compute the power dissipation of the device in a given application.

Figure 34 shows how to use the POWER tool. Input values are selected on the interface from keyboard inputs or selection boxes. Results are shown in the green boxes. Power dissipation graphs can be displayed from the interface using System or IC POWER dissipation buttons. *Load and save POWER configuration* buttons are used to resume the power dissipation calculation later, by saving and loading a text file.

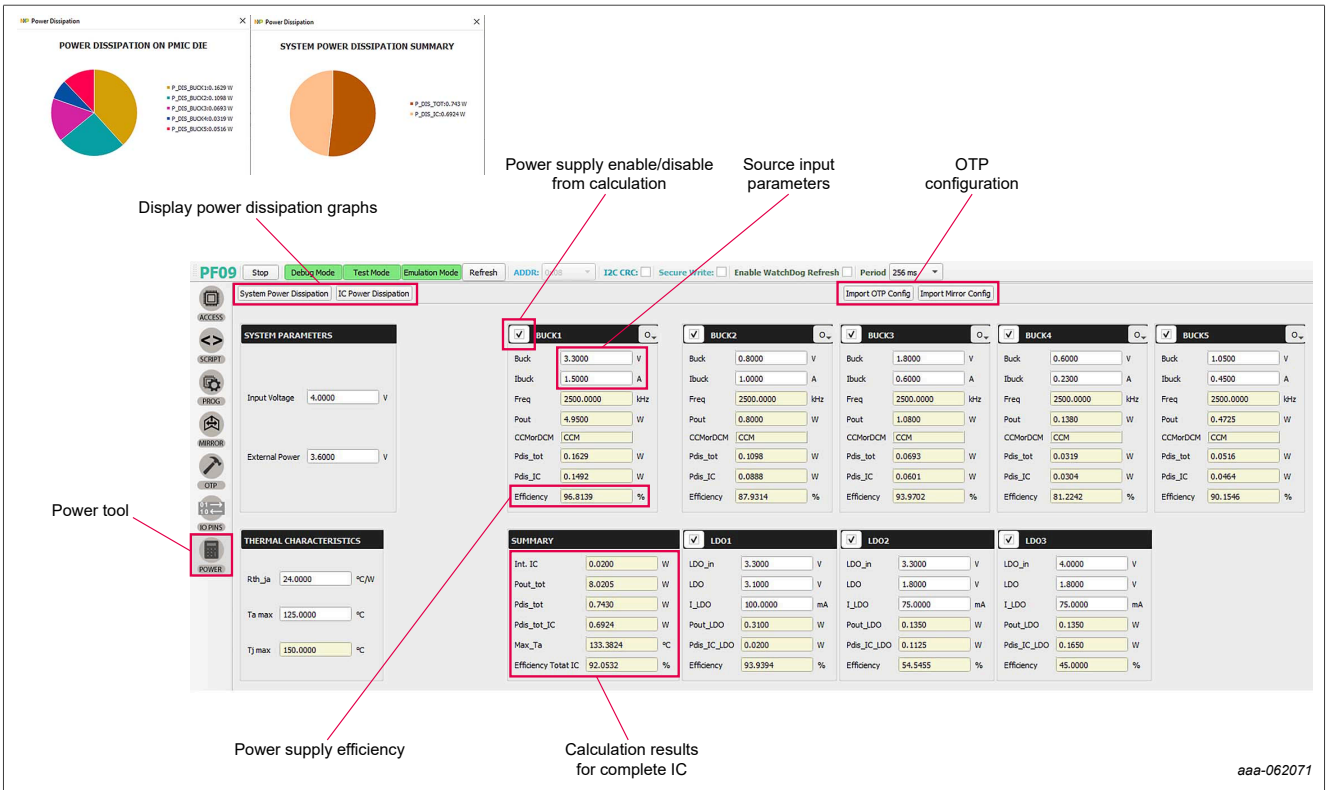


Figure 34. POWER Tool

6.5.2 OTP

The OTP tool is used to enter an OTP configuration. The OTP configuration can be saved as a CFG file or exported as a TBB file:

- The CFG file is used by the GUI to log all of the configuration information. The CFG file can be used to save an OTP configuration or load *mirror registers* with the MIRROR tool in *debug mode*.
- The TBB file can be created with a .txt extension. The TBB file can be used to load the *mirror registers* with the SCRIPT tool in *debug mode* or to burn OTP fuses using the PROG tool.

In the OTP tool, the displayed parameters differ depending on selected device type (ASIL B or QM) in Program Details box. The OTP configuration is not addressed here. For information on OTP configuration contents, refer to the PF09. For information on OTP and *mirror registers*, see Section 7.6.

The OTP tool's main panel is divided into two windows:

- OTP parameters setting
- OTP details

6.5.2.1 OTP parameters setting window

The OTP parameters setting section is organized into five tabs.

Note: *The Functional Safety window differs depending on the selected safety level. Windows are the same for QM and ASIL B parts for all other tabs. Safety level can be selected in the Program Details box of Section 6.5.2.2.*

6.5.2.2 System configuration tab

The *system configuration* tab provides a means of setting miscellaneous PF09 system configuration parameters, including clock, I/Os, and power-up sequence configuration.

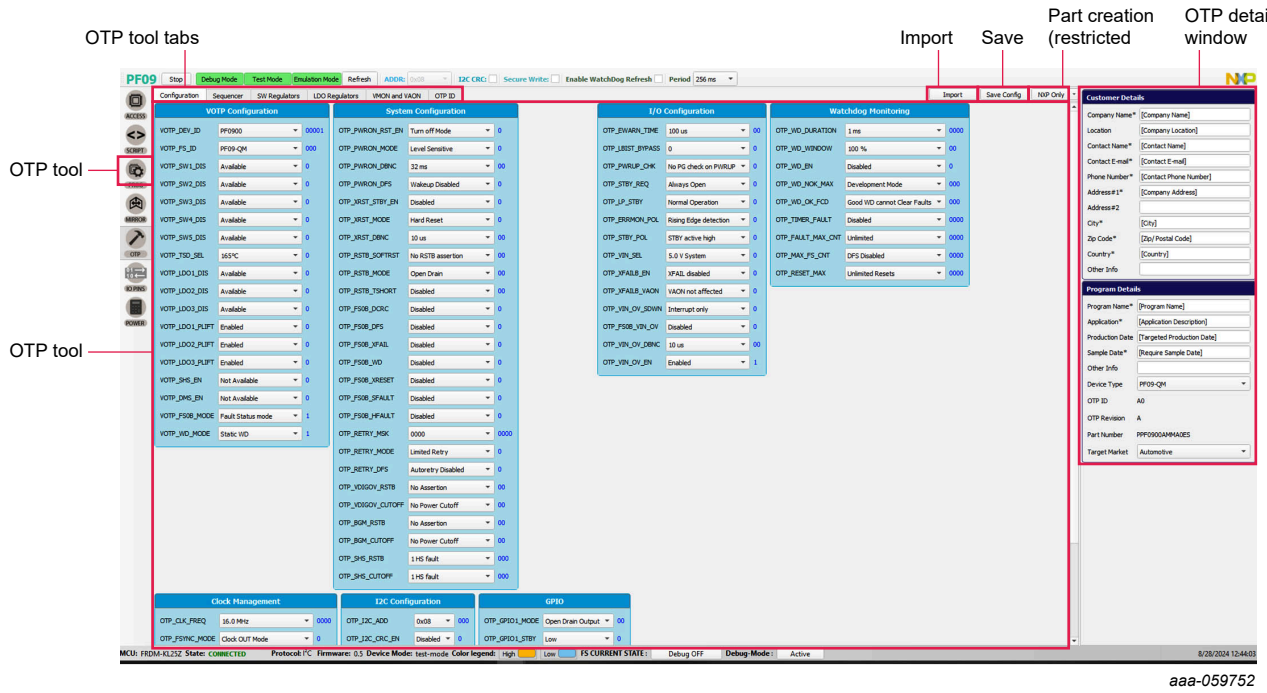


Figure 35. OTP tool main panel

6.5.2.3 System sequencer tab

The tab displays the set power-up sequence as graph in the *sequence diagram* box.

- **Power Sequencing box:** Sequence timing parameters definition.
- **Power-Up Sequence box:** Sequence definition.

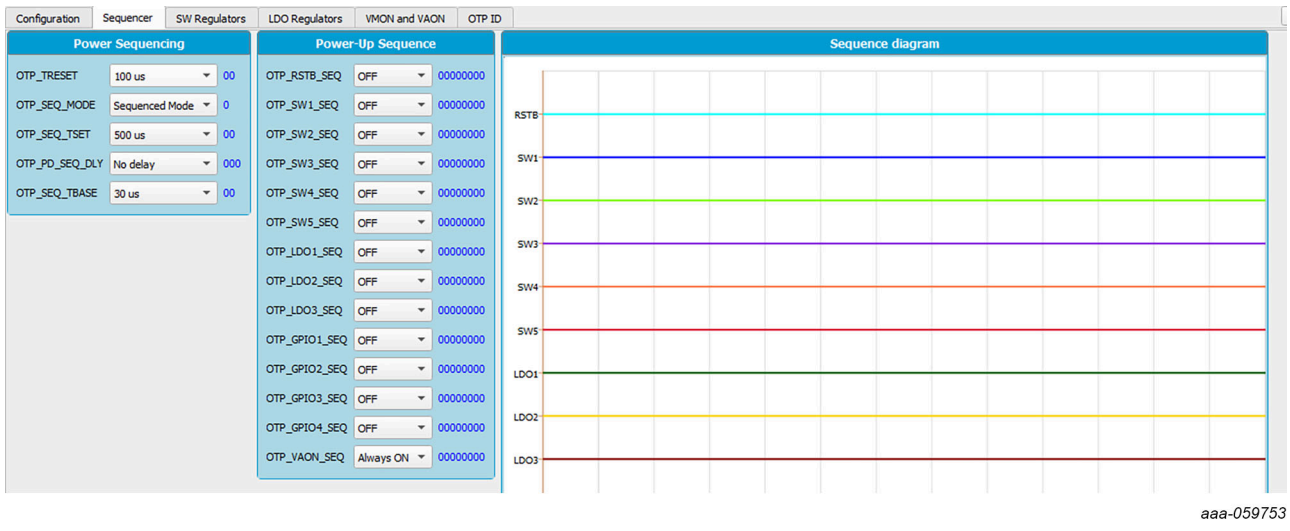
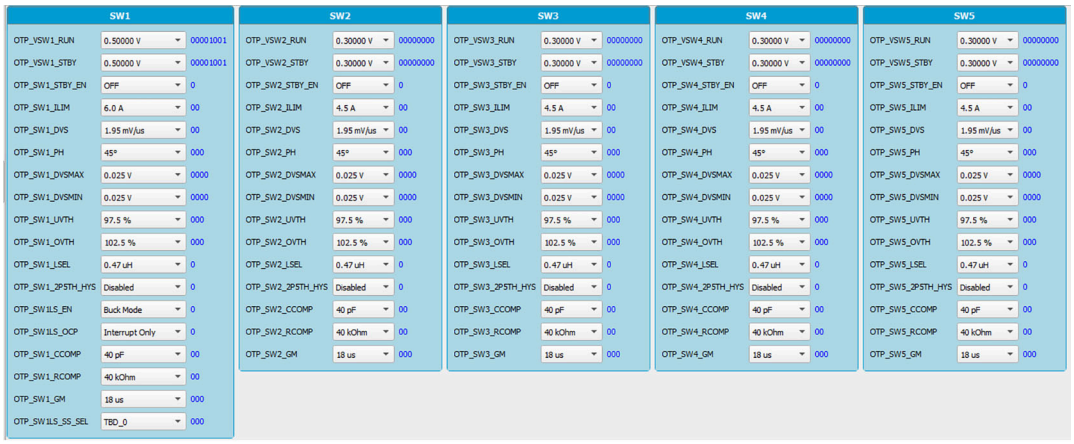


Figure 36. Power sequence configuration

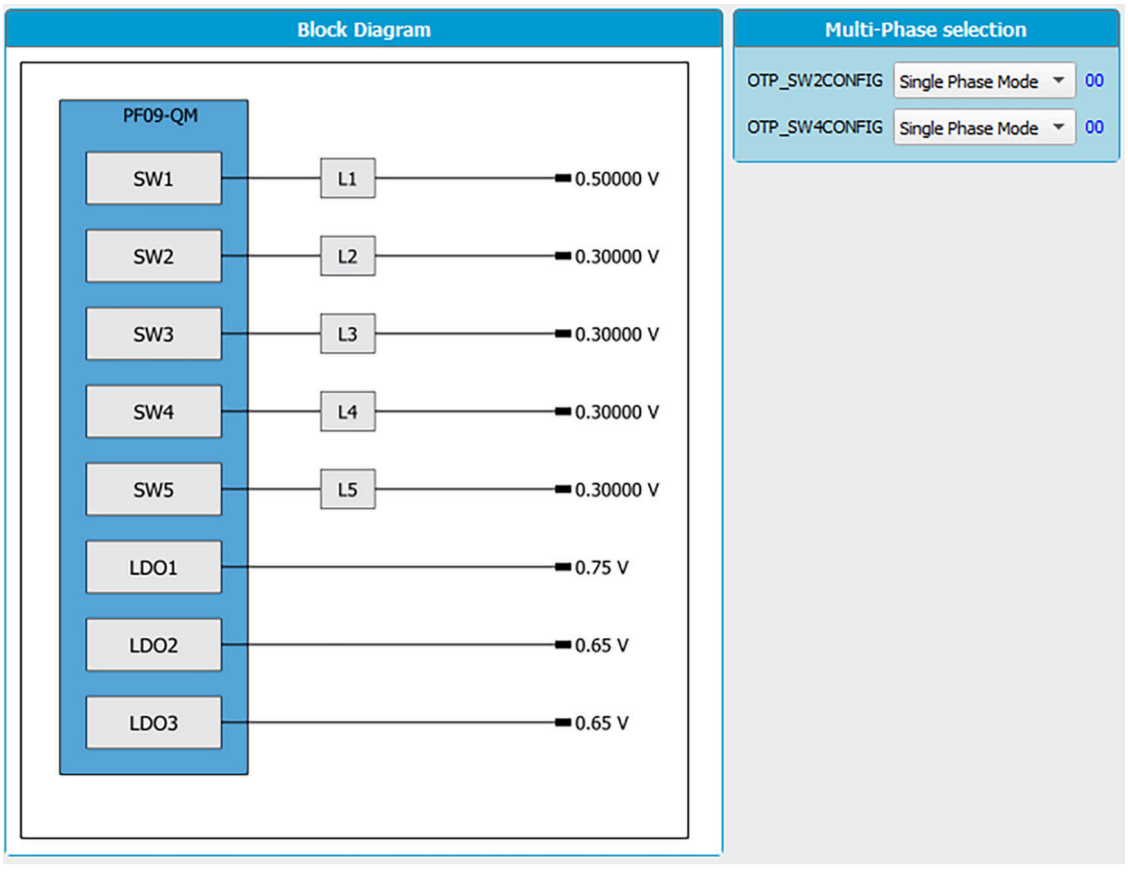
6.5.2.4 SW regulators tab

The *SW regulators* tab allows the user to set OTP parameters for device outputs. This tab displays a simplified diagram of the selected configuration as a visual crosscheck.



aaa-059754

Figure 37. SW Regulator tab



aaa-059755

Figure 38. SW Regulator tab

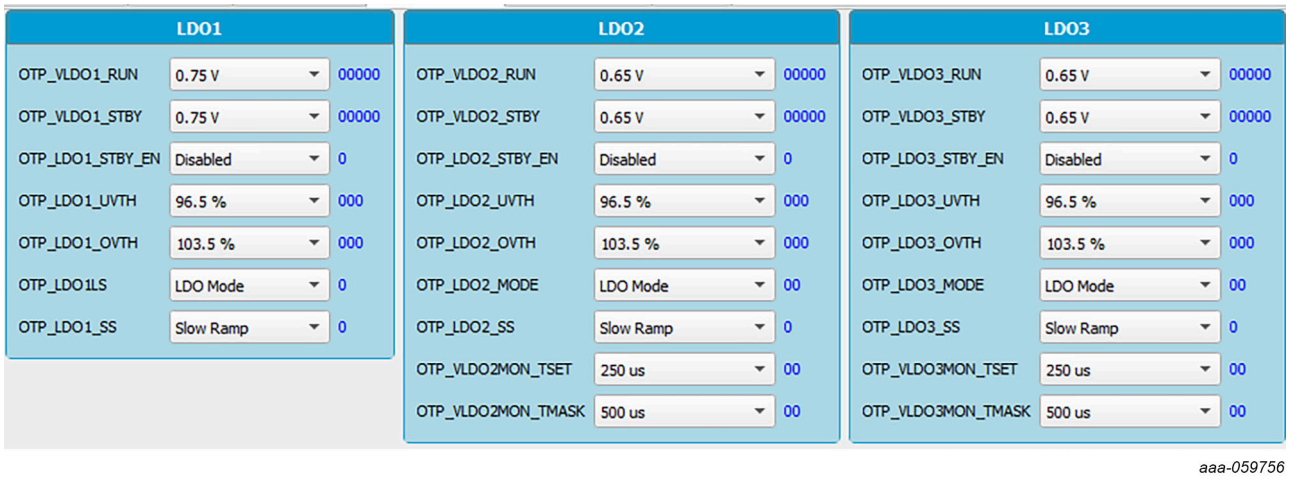


Figure 39. LDO regulators tab

6.5.2.5 VMON and VAON tab

The VMON and VAON tab allows the user to set OTP VAON-related parameters, such as under/over-voltage parameters, fault behavior and timer.

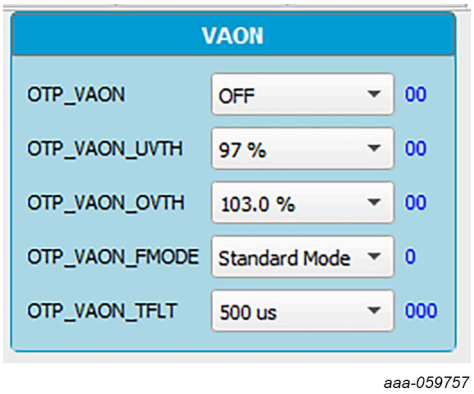


Figure 40. VMON and VAON tab

6.5.2.6 Program ID tab

The Program ID tab displays the OTP ID. Only NXP users can create an OTP ID.

Figure 41. Program ID tab

6.5.2.7 OTP Details window

The OTP Details window collects and stores information about the customer and the OTP version. All the information entered in this section will be part of the CFG and TBB files.

This section is organized into two boxes:

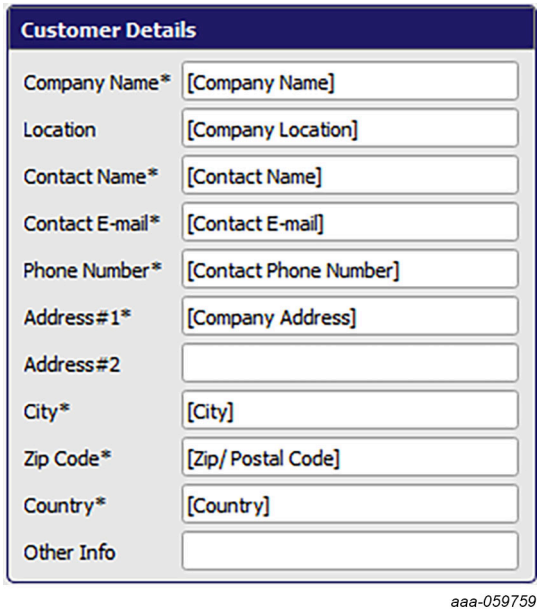


Figure 42. Customer details

Note: *Collects and displays customer contact information.*

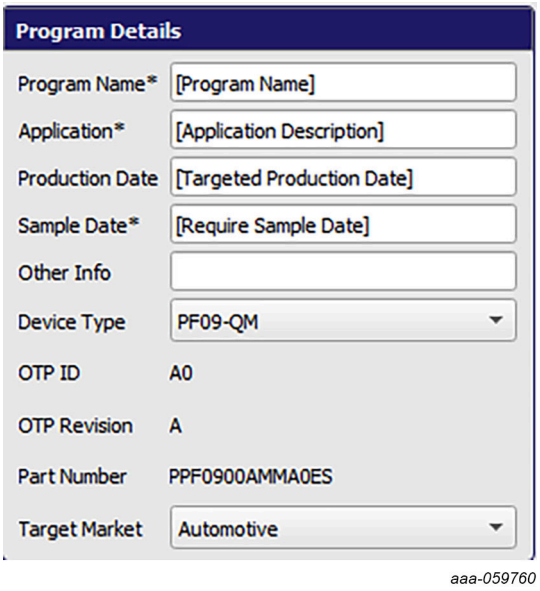


Figure 43. OTP program details

Note: *In addition to comments related to the application, this window allows the user to: Select device type (see **Functional Safety** tab paragraph for details), display the OTP ID (set by NXP only), display the build part number (set by NXP only), select target market, either automotive or industrial*

6.5.3 PROG

The PROG tool provides access to device programming configuration and tools for the OTP burning process.

Note: *Use this tab cautiously, as a device can be programmed just twice.*

The programming tool is available only in Test mode. The programming screen consists of three sections:

- Device programming configuration window
- OTP mode window
- Fuse box status window

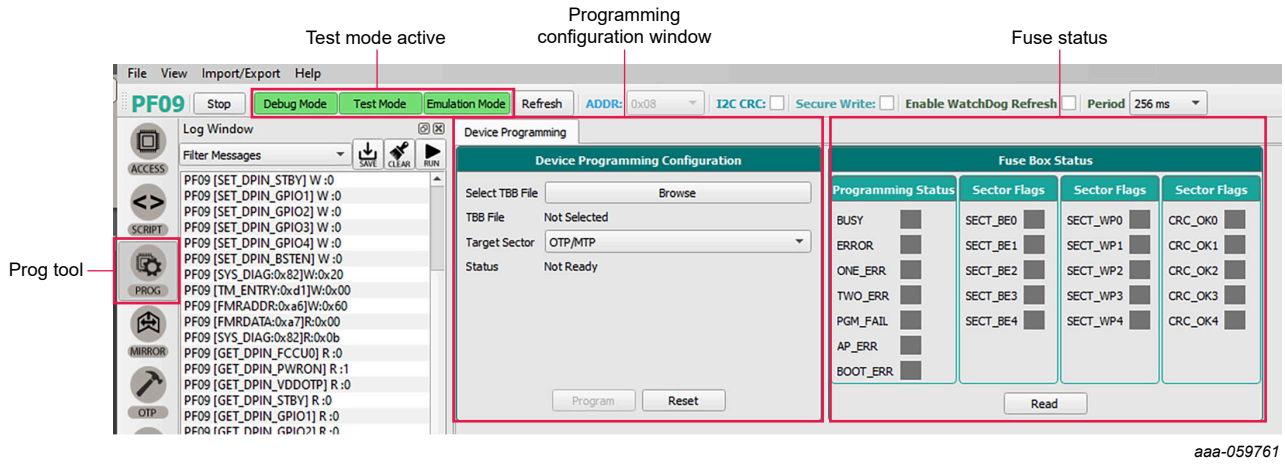


Figure 44. PROG tool

6.5.3.1 Device programming configuration window

From this window, the user can program an OTP operation from a saved TBB file. To program the part by fusing OTP, see Section 6.5.3.

6.5.3.2 OTP mode window

OTP mode window allows the user to:

- Exit OTP mode by sending an I^2C frame (by checking OTP mode exit box).
- Verify that the device is in OTP mode (condition: OTP 8 V applied to DBG pin at start-up).

6.5.3.3 Fuse Box Status window

The Fuse Box Status window shows the OTP fuse status. In the indicator boxes, blue indicates the section has not been programmed yet, orange indicates the section has been fused already.

As a reference, an empty part has the same Fuse Box status as shown in Figure 90.

- **OTP** is customer OTP configuration.
- **MTP** is a duplicate of Sector 1 for second time programming. See Section 7.6.

6.5.4 SCRIPT

The Script editor allows the user to create script sequences or to send existing sequences to the device. Commands include reading/writing individually to a register, to a digital pin, or to an analog pin. This tool allows the emulation of an OTP configuration.

The Script editor window consists of four sections:

- **Log window**
- **Script commands** (Window between Log Window and Script Commands Window)
- **Script window and its Script bar** (Under Script Command Window and Script Results Window)
- **Script results window**

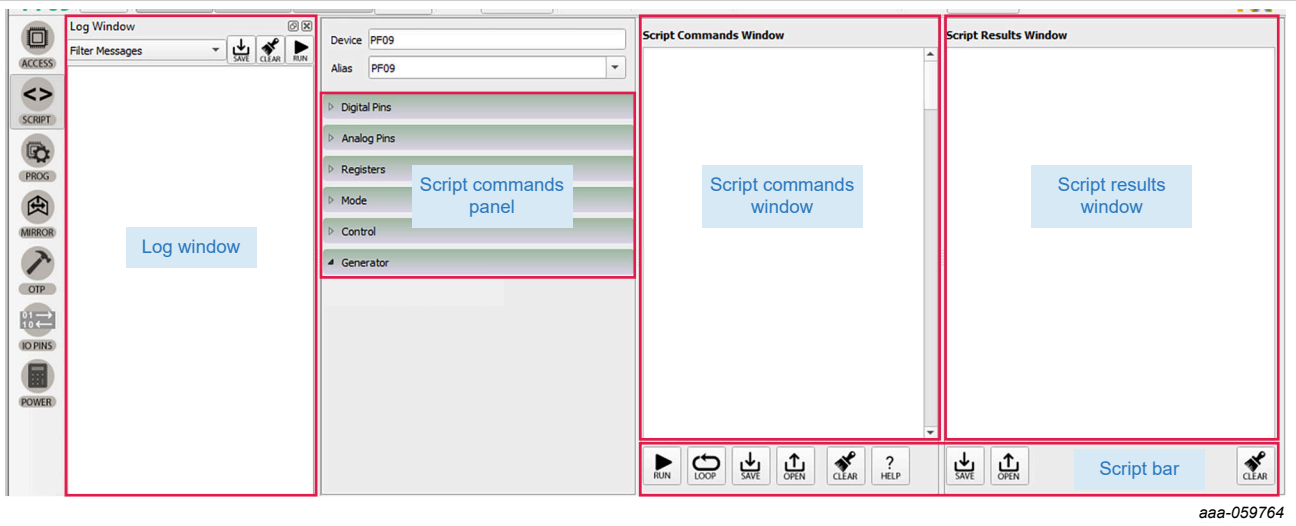


Figure 45. SCRIPT tool

6.5.4.1 Log window

The Log window lists events as they occur in real time when the script is executing. The Filter Messages box allows the user to limit log messages to certain events (Register Read, Register Write, Pin Read, Pin Write).

The Log window menu bar also contains buttons for saving the log contents to a file, clearing the log, or running the script in the Script Command window.

The Log window shows the CRC for each sent or received frame.

Note: *The Log window content can be saved to a file, then opened as a Script to run it. See Section 7.9.*

6.5.4.2 Script commands panel

The Script commands panel allows the user to enter commands into the Script command window by clicking the appropriate command. Facilitated command entry ensures error-free syntax in the command. The commands are organized into functional categories. Opening a panel tab and selecting a specific pin or register makes the associated command appear in the Script command window.

Figure 46. SCRIPT tool

- **Digital pins:** To enter a script command to read or write the value of the selected digital pin.
- **Analog pins:** To enter a script command to read the value of the selected analog pin.
- **Registers:** To enter a script command to read or write functional and safety registers.
- **Mode:** To enter a script command setting the desired mode.
- **Control:** To enter a script command to pause script execution (execution halts when the Pause command is encountered and a pop-up window appears, prompting the user to continue execution), to delay script execution (default is 300 ms, editable to any ms value in the Script Commands Window), or to exit script execution.
- **Generator:** To clear the content of the Script Command Window and enters a pre-prepared script sequence.

An INIT script example has been integrated to the Generator tab. The example provides a generic step-by-step procedure to initialize the device with a default configuration and transition to Normal applicative mode.

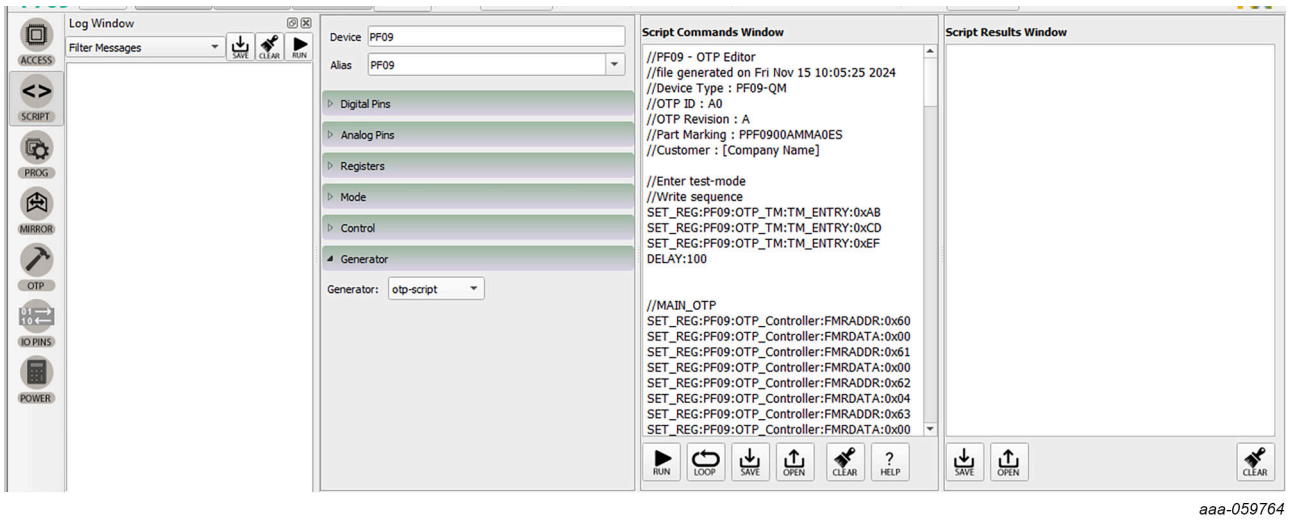


Figure 47. Generate an OTP – mirror registers writing script example (OTP-Script)

Note: ***HELP*** *button in the Script bar gives information on Commands.*

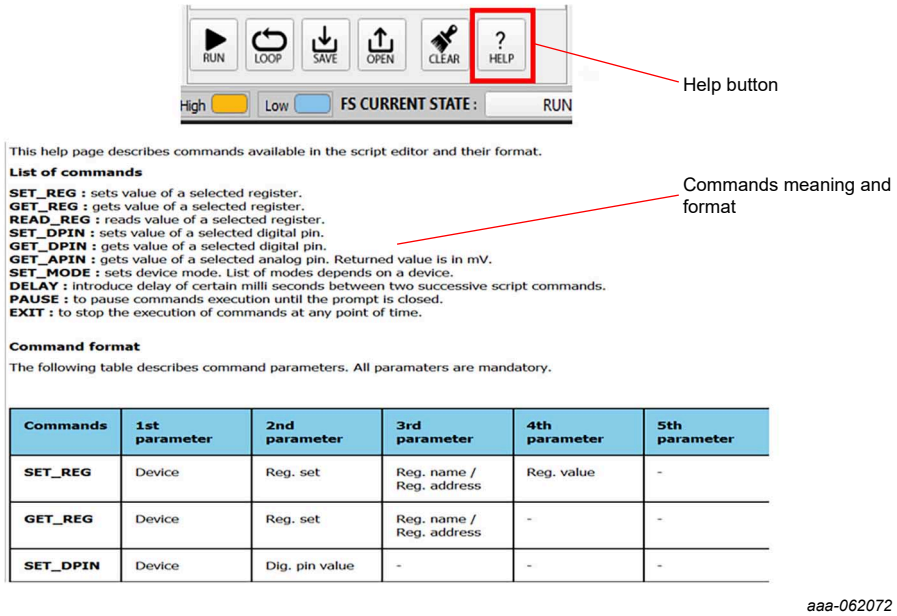


Figure 48. Script command panel help

All menu items work in a similar way. The example below shows a typical process using the Registers menu tab.

- How to use the script tool: Click the Register tab brings up the parameters panel shown in Figure 49. A Write operation to the CTRL1 register in the functional register group has been selected. The value 0x00 is selected as the value to be written to the register. Hit **Enter** with the cursor in the value field enters the well- formed command in the Script commands window.

To apply the commands on the Script Commands window, click **RUN** in the Script bar. The sent commands are displayed in the Log window and command results are displayed in the Script results window.

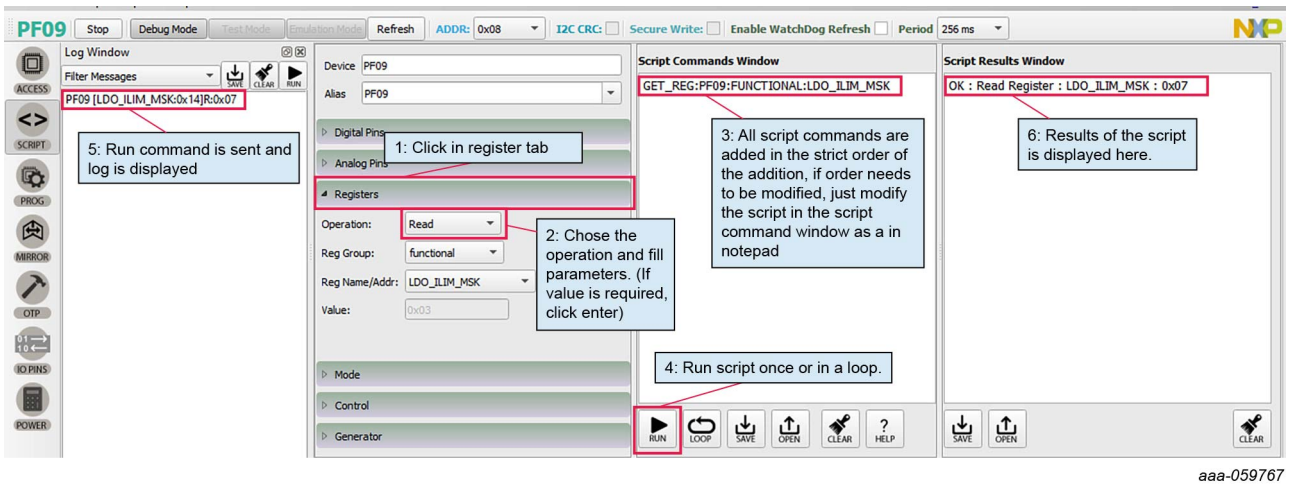


Figure 49. Using the SCRIPT tool

6.5.4.3 Script commands window

The Script Commands window is the area where existing script files can be loaded in Test mode only and where script commands are entered, edited, and executed.

Another use of the Script tool is to replay a Log commands sequence, for example to run a routine. This specific use is detailed in Section 7.9.

The Script bar at the bottom of the window contains the following six buttons:

- **RUN:** Initiates execution of the script sequence in the Script Command window.
- **LOOP:** Executes the script as a loop. Select **LOOP**, then click **RUN**.
- **SAVE:** Saves the content of the Script Command window as a .txt file that can be reloaded.
- **OPEN:** Clears the current content and loads a previously saved script into the Script Command window.

Note: *The loaded file must have a .txt extension.*

- **CLEAR:** Clears the current content of the Script command window.
- **HELP:** Shows a list of all script editor commands with their formats.

aaa-059768

Figure 50. Script commands window

6.5.4.4 Script results window

Script results window displays the results of an executed script. The menu bar at the bottom of the window contains the following three buttons:

- **SAVE:** Saves the content of the Script Results window as .txt file that can be subsequently reloaded.
- **OPEN:** Clears the current content and loads a previously saved result file into the Script Results window.
- **CLEAR:** Clears the current content of the Script Results windows.

aaa-059769

Figure 51. Script results window

6.5.5 MIRROR

The MIRROR tool provides access to all Mirror registers. Mirror registers are an emulation of OTP registers. Mirror registers can be read/written multiple times, whereas OTP registers can be burned just twice. In case of a power-on reset, the Mirror registers are reset to the default OTP configuration (empty if OTP sectors are not burned). Mirror registers can be read and written in Test mode only. See Section 7.3.2.

Note: *The MIRROR tool display tabs are similar to the corresponding OTP tool tab. To avoid confusion, the tab header background colors are different. The MIRROR tool tab header is purple. OTP tool tab headers are blue.*

The MIRROR tool is available only in Test mode. Test mode loads require keys in order to have full access to Mirror registers. This tool allows the user to:

- Read and write Mirror register values.
- Load a CFG file into the Mirror registers and debug the configuration. For further details on CFG file preparation and parameters configuration, see Section 6.5.2.

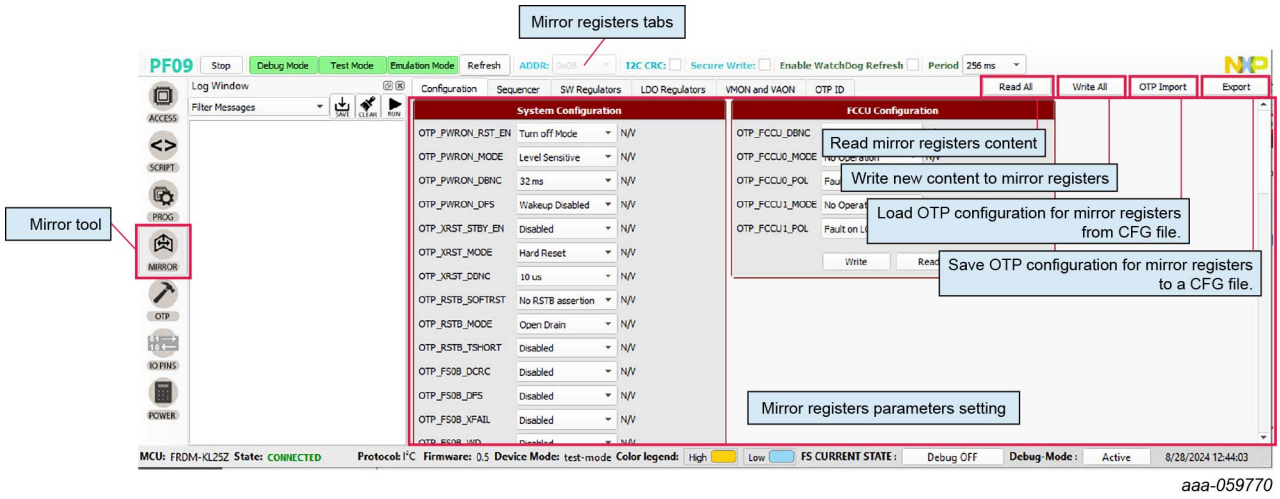


Figure 52. MIRROR tool

6.5.6 ACCESS

The ACCESS tool is the central tool for an evaluation session. This tool gives access to GUI functions that configure, monitor, and control the PF09 device during the evaluation session.

The ACCESS tool provides access all I^2C registers, displayed either:

- In a Register Map format with direct access to register bits.
- In thematic tabs with a graphical and more readable view.

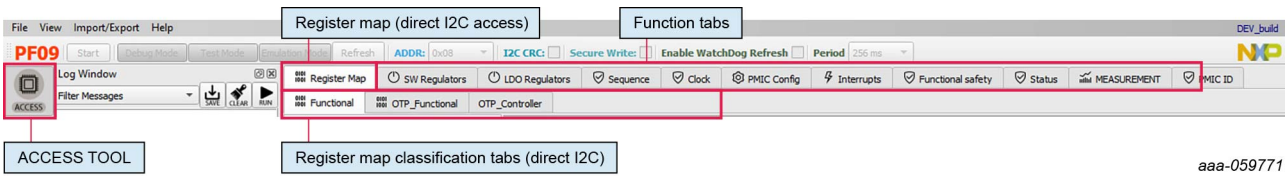


Figure 53. ACCESS tool

The tab content contains 11 tabs:

- **Register map**
- **SW regulators**
- **LDO regulators**
- **Sequence**
- **Clock**
- **PMIC config**
- **Interrupts**
- **Functional**
- **Safety**
- **Status**
- **Measurement**
- **PMIC ID**

6.5.6.1 Register map

The Register Map tab allows the user to read or write the PF09 I^2C registers, bit per bit.

Note: *I^2C registers are responsible for PF09 flexible device configuration, in opposition to OTP configuration, which is permanent once fuses are burned. The two levels of configuration work together and should be defined accordingly.*

The Register Map is composed of three subtabs:

- **ID registers**
- **Interrupt registers**
- **Fault management**
- **System configuration**
- **Clock management**
- **Watchdog configuration**
- **Fault counters**
- **VMON configuration**
- **SWx control**
- **LDOx control**
- **VAON control**
- **State**

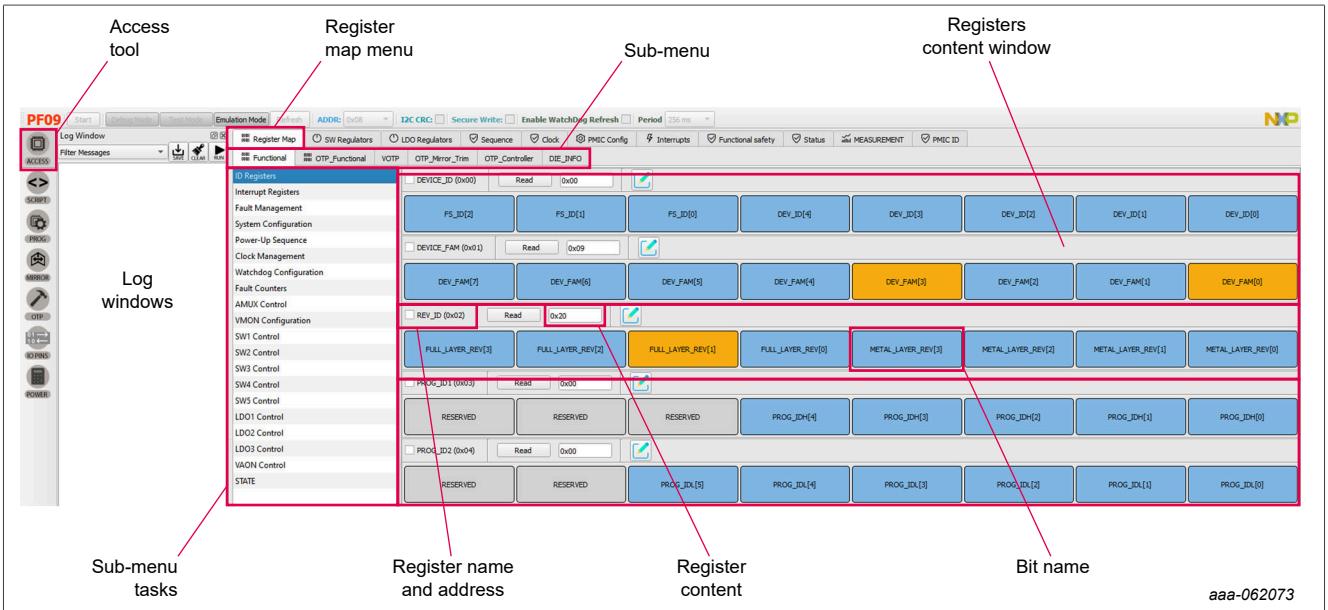


Figure 54. Register map menu

6.5.6.2 Modifying one bit content by clicking on the bit name

In the Registers Content window, the user can access the PF09 I^2C registers. Two types of registers exist:

- Read-only registers (for example, Status) appear with a ‘Read button’ only,
- Read/Write registers (for example, System configuration, Regulators Control, etc.) appear with both a ‘Read’ and a ‘Write’ button.

The user can click a bit name to change its value, when the bit is writable.

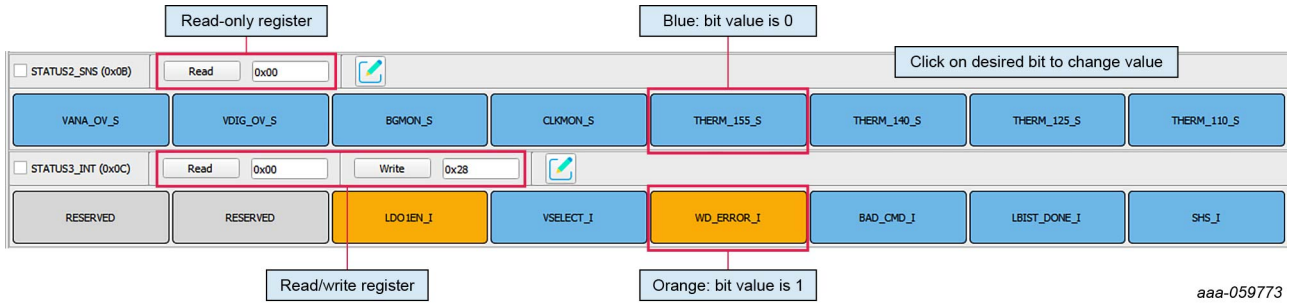


Figure 55. Register content window

6.5.6.3 Accessing one register content using Read and Write buttons

In the Registers Content window, the user can read and/or write the PF09 I^2C registers using the Read and Write buttons.

To **read** the content of one register:

- Click **Read** next to the corresponding register name,
- The current register content is displayed as a hexadecimal value next to the Read button. To **write** the content of one register:
- Enter the desired register value as a hexadecimal value in the box next to the Write button.
- Click **Write** next to the corresponding register name.
- As a crosscheck, the current register value can be accessed by clicking **Read**.

Note: *Click Read to get the register value. Click Write to change the register value.*

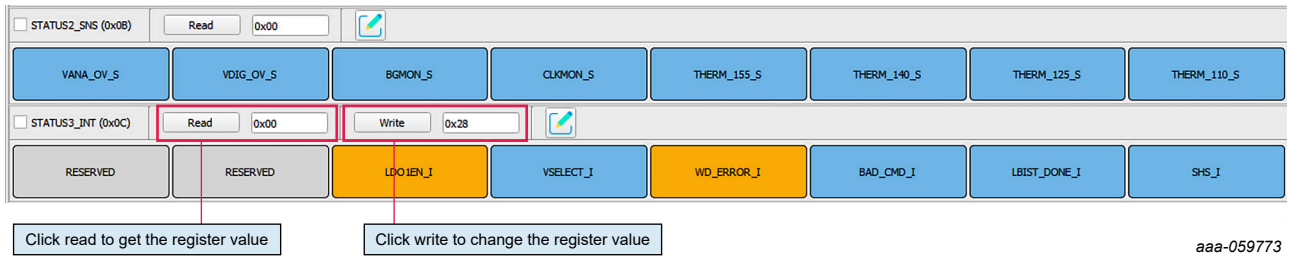


Figure 56. Accessing one register content using Read and Write buttons

6.5.6.4 Accessing one register content using Bit-Map Dialog

In the Registers Content window, the user can read and/or write the PF09 I2C registers using the Bit-Map Dialog window.

Clicking the **Green Pencil** next to the corresponding register name opens the Bit-Map Dialog window and shows the name and description of all of the register's bitfields.

The Bit-Map Dialog window:

- Allows the user to change bit values from selection boxes on the left side
- Displays the current register content on the right side.

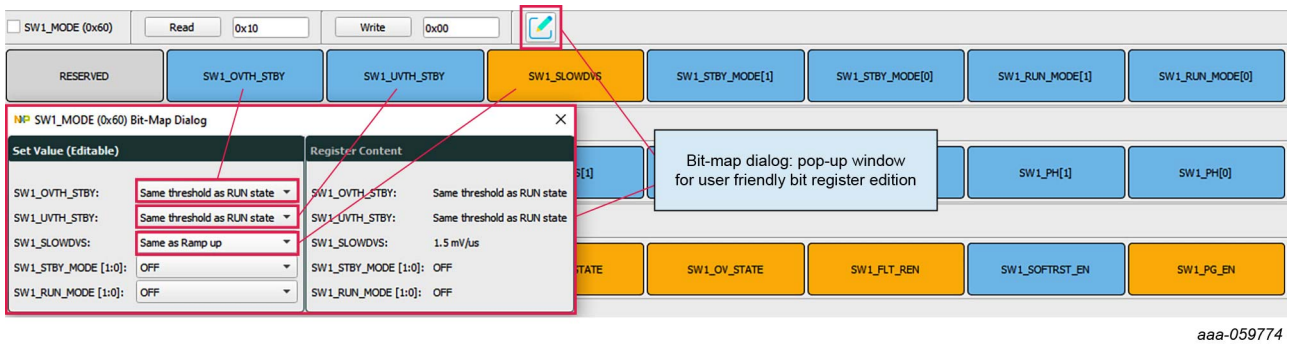


Figure 57. Accessing one register content using Bit-Map Dialog

6.5.6.5 Modifying multiple registers using the lower Read/Write/Reset bar

In the Registers Content window, the user can read, write, and/or reset multiple PF09 I^2C registers at a time using the lower Read/Write/Reset bar.

- Checkboxes allow the user to select the registers to be read or written.
- Selecting the Select All checkbox allows the user to simultaneously act on all the registers in the Register subtab.
- Read, Write, and Reset buttons allow the user to act on the previously selected registers. The Reset button switches all the bits in the register to 0.

Navigating in the Registers content in the Register subtab can be facilitated using the selection box in the bar.

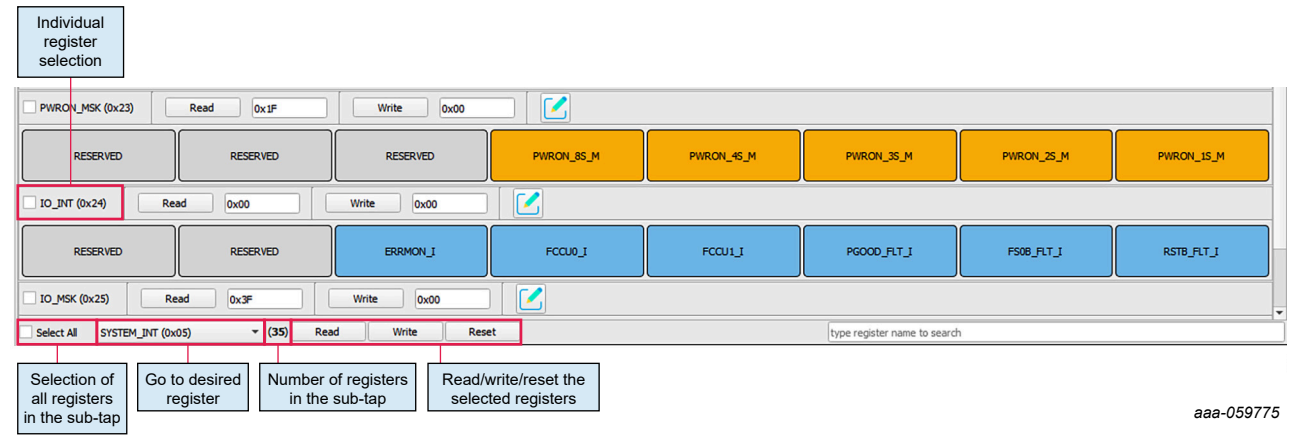


Figure 58. Read/Write/Reset bar

6.5.6.6 SW Regulators

The SW Regulators Tabs allows the user to read or write the parameters related to the Switchers configuration.

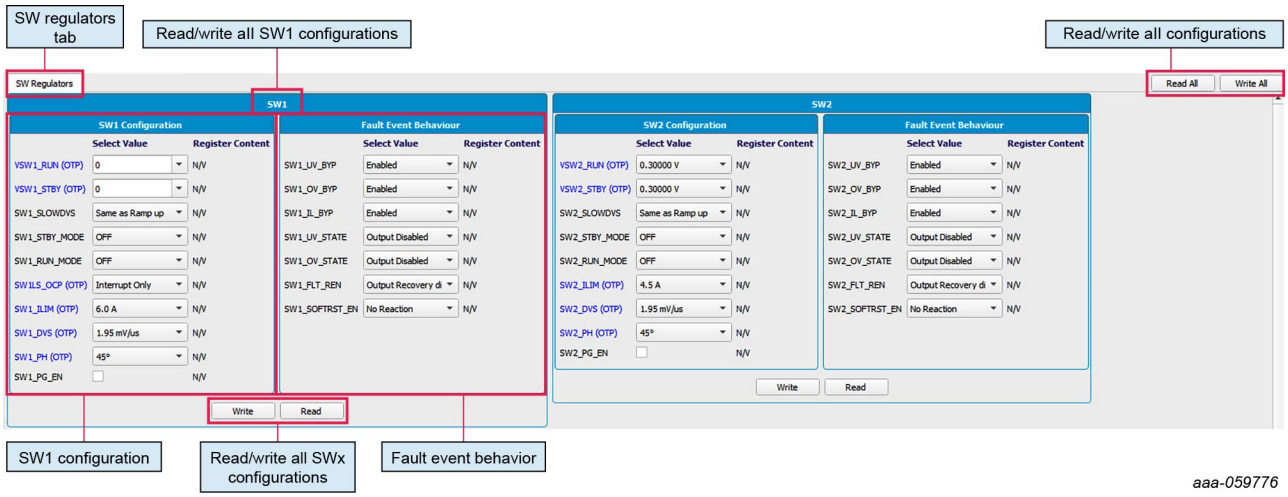


Figure 59. SW Regulators tab

- **SWx configuration:** To choose the switcher x configuration for Run, stand-by, etc.
- **Fault event behavior:** Configuration related to the Switcher Fault behavior.

6.5.6.7 LDO Regulators

The LDO Regulators Tabs allows the user to read or write the parameters related to the Switchers configuration.

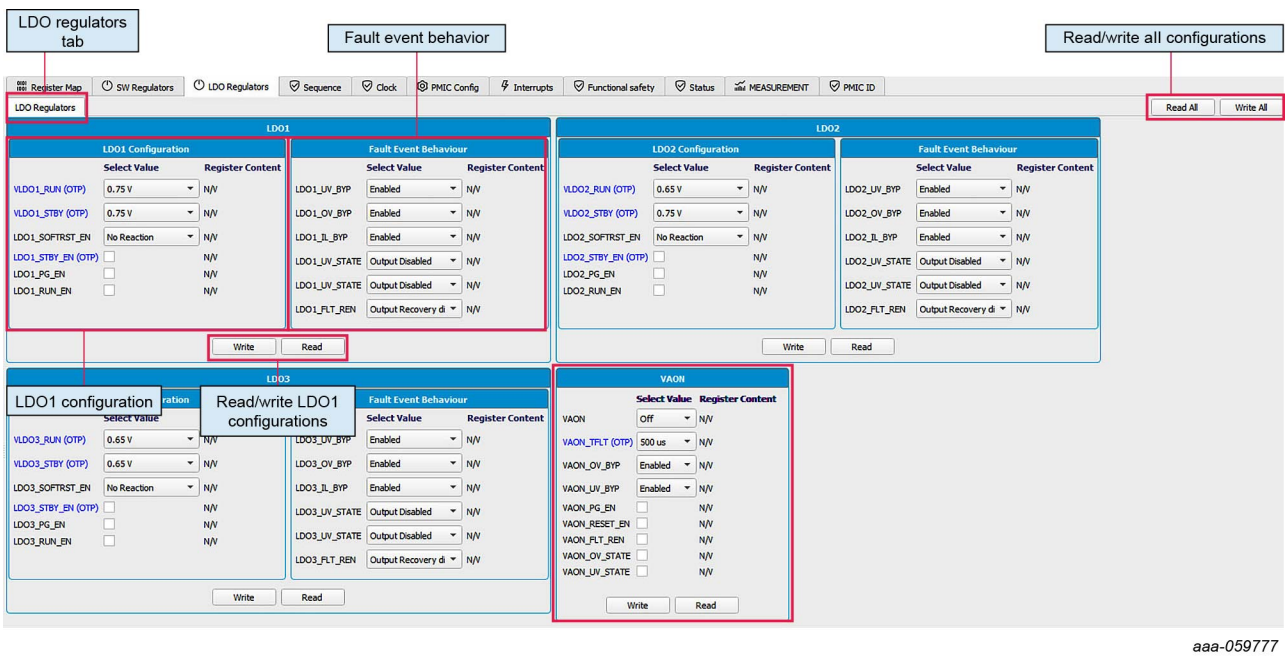


Figure 60. LDO Regulators tab

6.5.6.8 Sequence

The Sequence Tab provides all configuration required for sequencing the device.

- **Power Sequencing window:** Timing parameters

- **Power Up Slot window:** Power and monitoring sequence

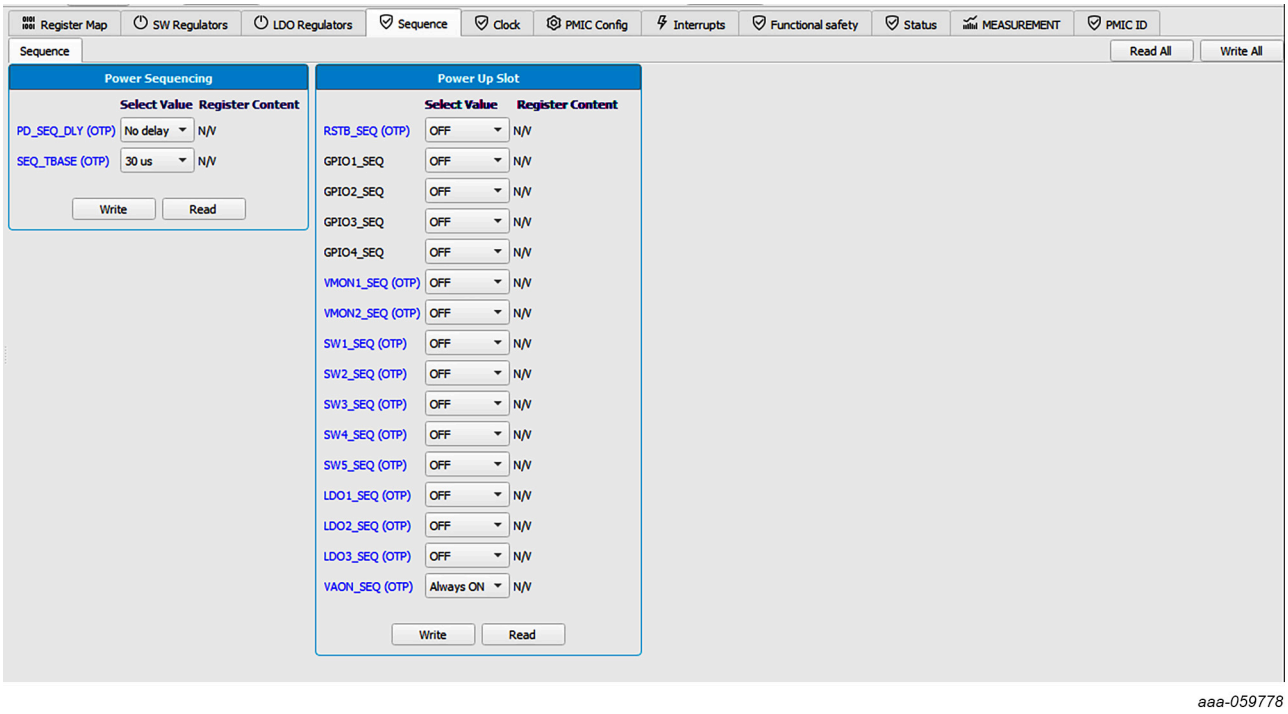


Figure 61. Sequence tab

6.5.6.9 Clock tab

The Sequence Tab provides all configuration required for external synchronization or synchronization when using more than once device.

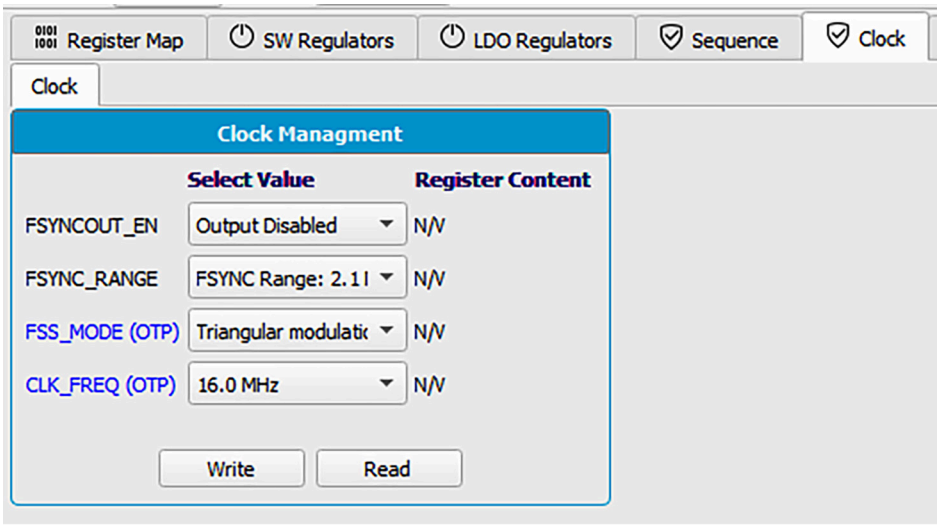
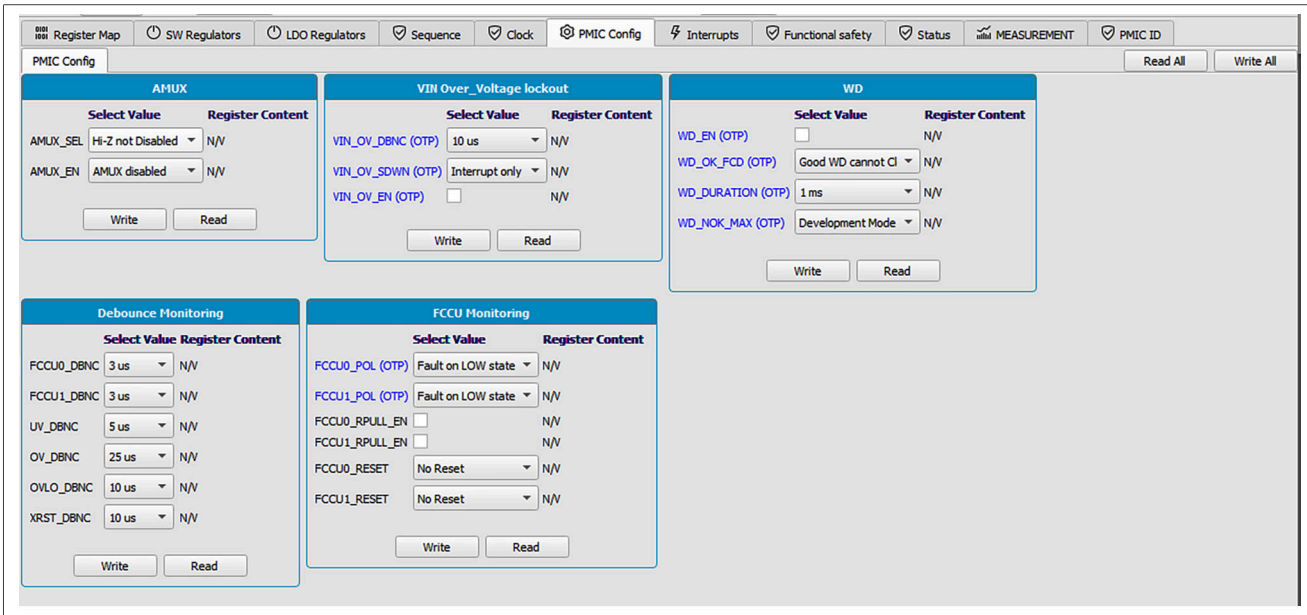


Figure 62. Clock tab

6.5.6.10 PMIC configuration tab

The Sequence tab provides configuration for the Watchdog, VIN Overvoltage, multiplexed functionality and input monitoring(FCCUx).

- **AMUX window:** AMUX
- **VIN Over_Volatage lockout window:** VIN MON
- **WD window:** Watchdog
- **Debounce monitoring and FCCU monitoring windows:** FCCUx

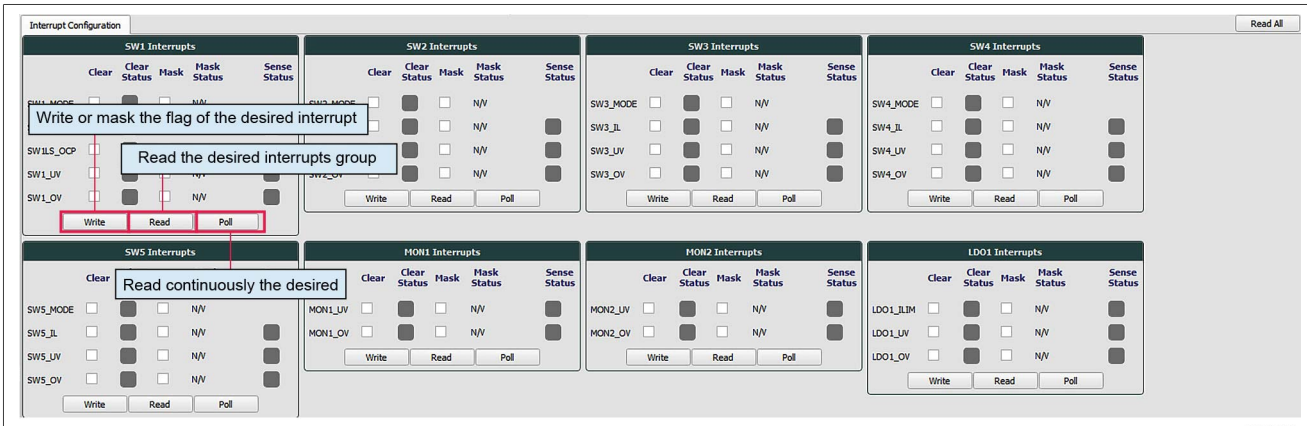


aaa-059780

Figure 63. PMIC tab

6.5.6.11 Interrupts

The Interrupts tab allows the user to read All Flags related to Interruption, both masked and unmasked.

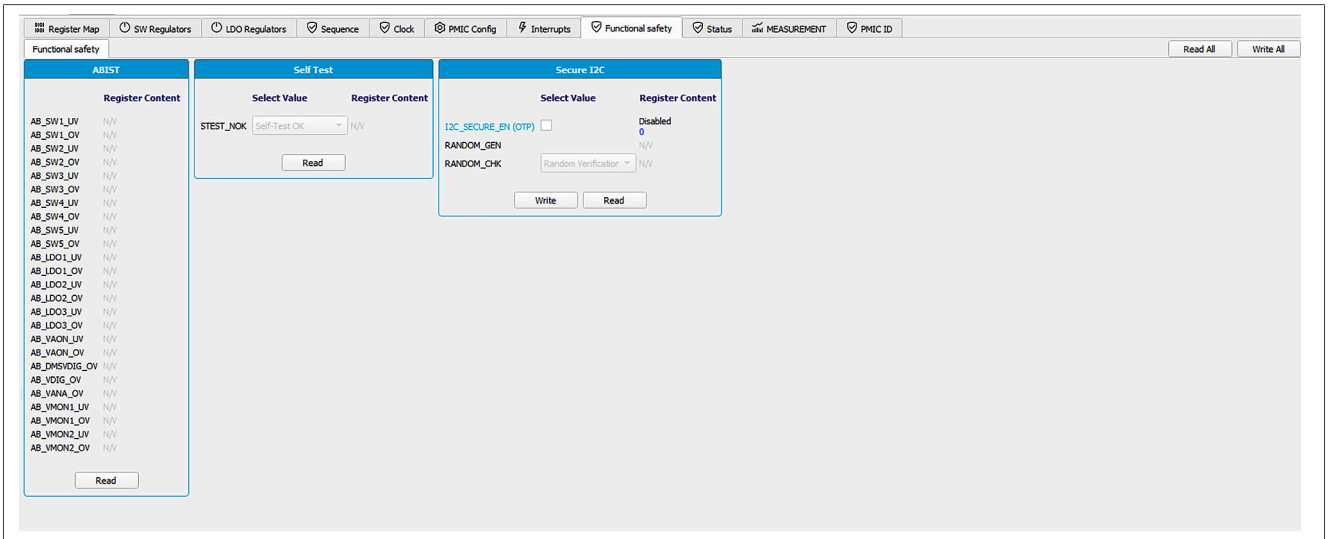


aaa-059781

Figure 64. Interrupts tab

6.5.6.12 Functional safety

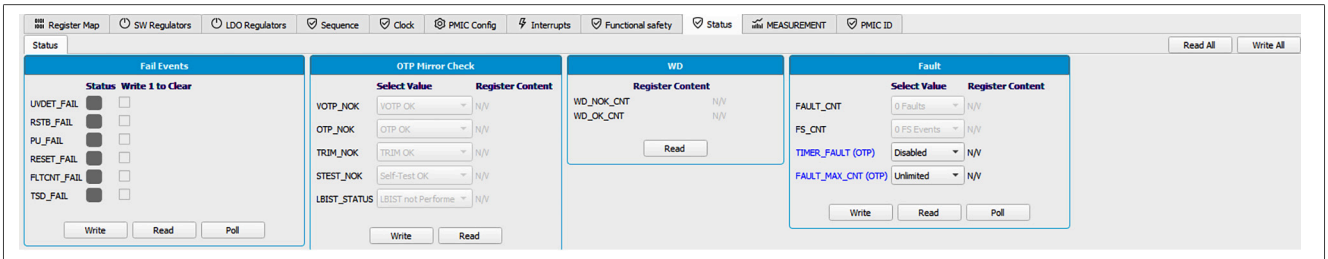
The Functional Safety tab allows the user to carry out the safety diagnosis by reading the register content. The Functional Safety tab allows the user to read the ABIST Flags and observe the ABIST status.



aaa-059782

Figure 65. Functional safety tab

6.5.6.13 Status tab



aaa-059783

Figure 66. Status tab

6.5.6.14 Measurement Tab

The Measurement tab allows the user to measure voltage levels and temperature values of multiple key locations from onboard sensors. The Measurement feature uses one ADC to successively measure multiple points. The Measurement feature must be enabled beforehand.

- **AMUX measurement summary:** Displays the measurements of voltage levels and temperature values from sensors placed at key locations. Click **Read** to obtain or refresh the measurements.
- **Temperature measurements graph:** Displays the measurements of the Temperature sensors inside the PMIC in function of time. Click **Clear** to clean the graph or **Poll** for performing the continuous measurements.
- **Voltage measurement graph:** Displays the selected voltage measurement values as a function of time in a graph. Click **Clear** to clean the graph or **Poll** for performing the continuous measurements.

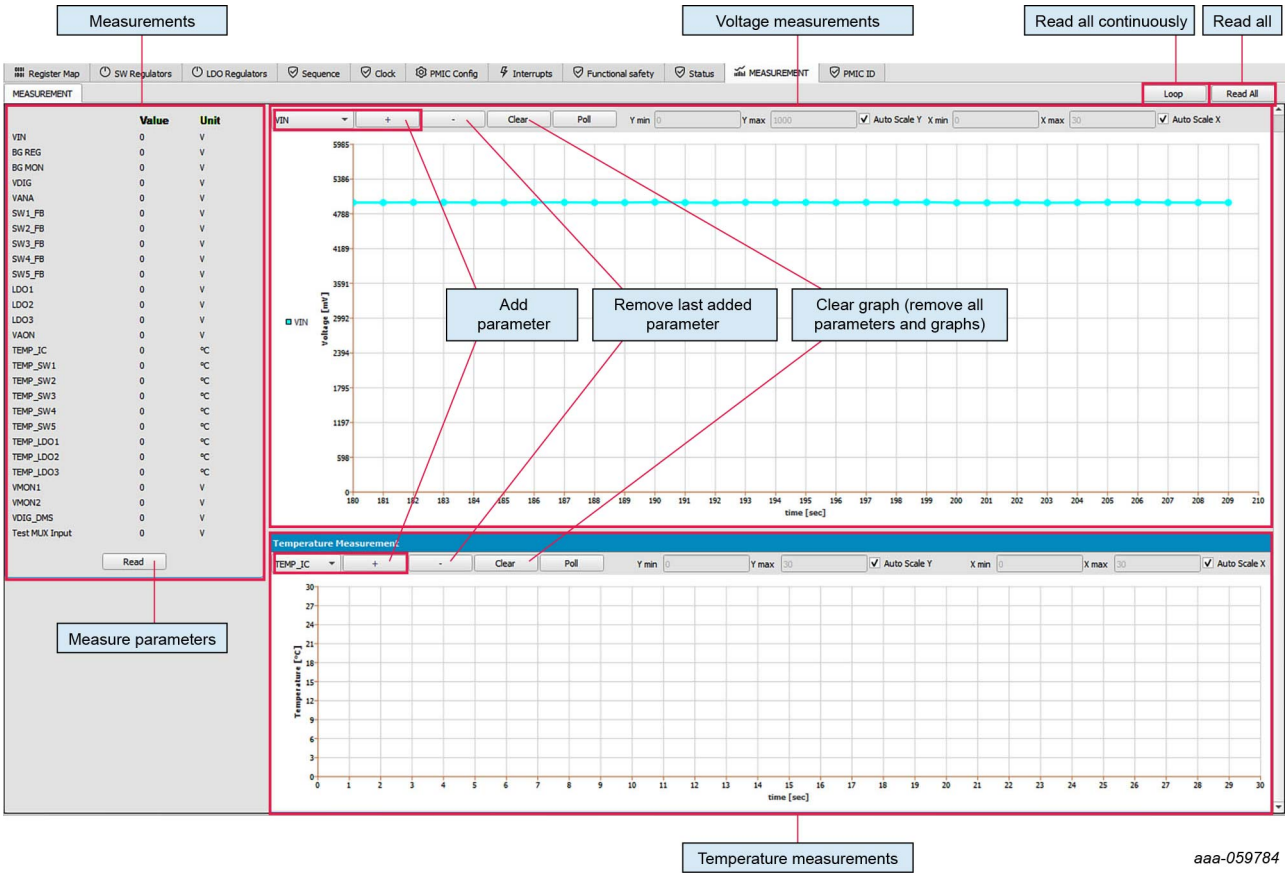


Figure 67. Measurement tab

6.5.7 PMIC ID

Provide general device Information.

- **Device ID:** Shows the safety level information
- **Silicon ID:** Provides the silicon revision.
- **OTP ID:** Identifies the device configuration.

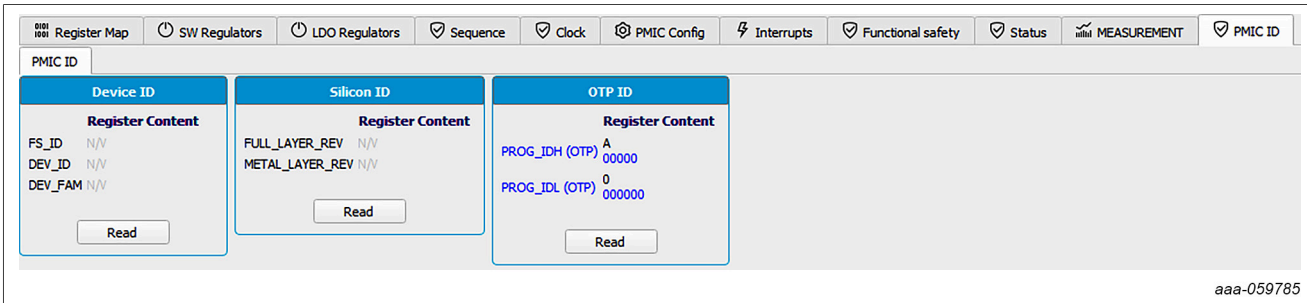


Figure 68. PMIC ID

6.5.8 I/O PINS

The I/O PINS tool provides a means of reading INTB, FCCU1, PGOOD, RSTB, GPIO1, GPIO2 and BSTEN, and Setting FCCU0, PWRON, VDDOTP, STBY, GPIO1, GPIO2, GPIO3, GPIO4 and BSTEN.

Note: *The configuration of GPIOx pins must be done by OTP. Depending on the chosen functionality for GPIOx pins, the hardware must be configured as explained in Section 4.4.*

The IO PINS panel consists of three sections:

- **Log window:** Maintains a running log of events initiated during the current session. A drop-down menu in the upper left allows the log to be filtered by register read, register write, pin read, and pin write. Buttons in the upper right allow the Log window contents to be saved, cleared, or run.
- **PF09 Pins Toggling window:** To use the PINS tool, it is mandatory to select the PF09 I/Os settings as inputs in the OTP/Mirror tabs.
- **PF09 Pins Setting window:** To poll the listed Pins as outputs pins during a selected time duration or read the pins.

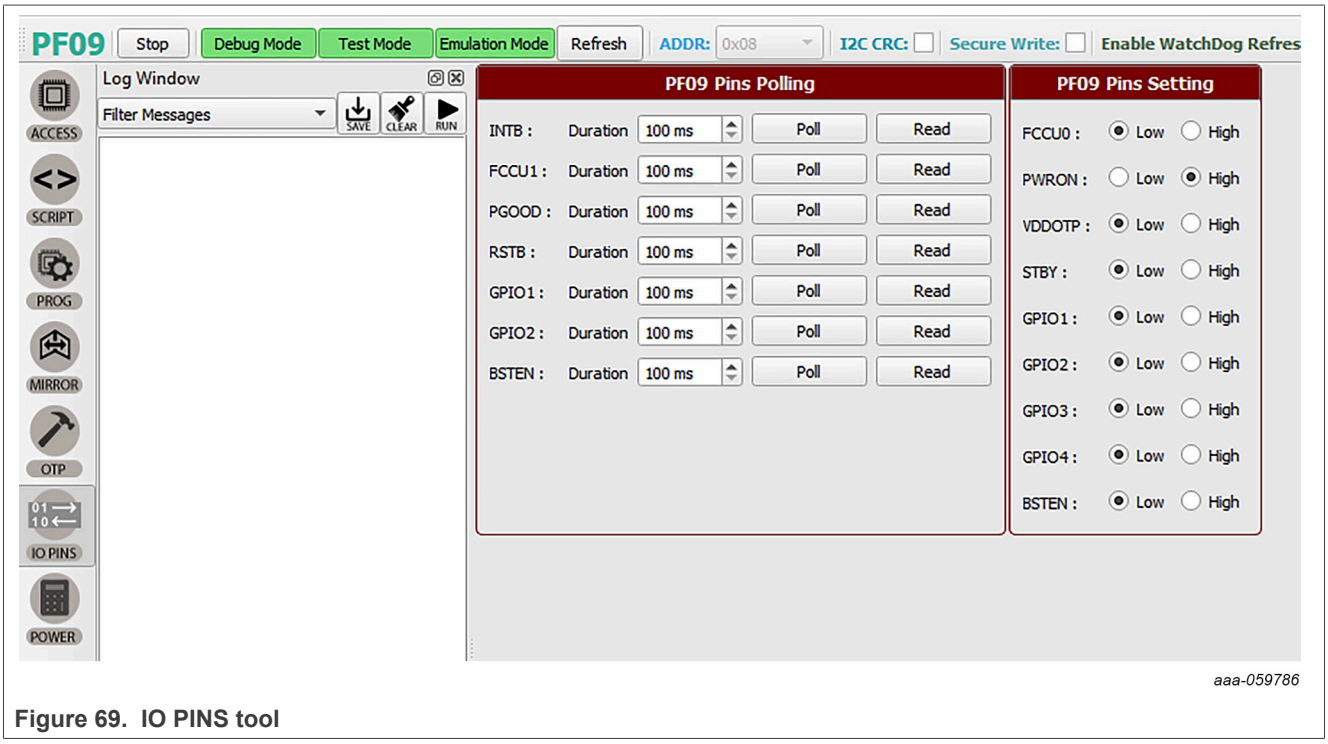


Figure 69. IO PINS tool

7 Setting up and running the KITPF09FRDMPGM

This section gives guidance on how to set up and run the KITPF09FRDMPGM Program board and GUI.

The device has a high level of flexibility thanks to the parameter configuration available by using the OTP registers. The user should learn about OTP and Mirror registers before operating with the device.

7.1 Setting up the KITPF09FRDMPGM

The procedure for setting up the KITPF09FRDMPGM program board is as follows:

1. Make sure that the board has the jumpers configured in their default positions. The default debug configuration enables the board to be fully controlled by the KL25Z MCU (via I^2C) and the GUI. Section 4.4 shows the default jumpers and switches configuration for PF09 Family.
2. Connect the power supply to VIN (hook connector). **The power supply should be set to a nominal value of 5 V**.

3. Make sure the USB cable between the board and the PC is securely connected. This connection is critical because the USB port serves as a communication channel between the PC and the KL25Z MCU onboard, and also provides voltages and references to some onboard circuits.

7.2 Connecting the KITPF09FRDMPGM to the GUI

The procedure for connecting the KITPF09FRDMPGM to the GUI is as follows:

1. To launch the NXP GUI application. See Section 5.3.
2. In the USB and Device Status bar at the bottom of the GUI window, the State message should display DISCONNECTED when the USB cable is plugged in, but communication has not yet been established between the KL25Z MCU and the PF09 SBC.

aaa-059787

Figure 70. USB cable plugged in, communication not established yet/USB cable not plugged in.

3. To establish communication between the KL25Z MCU and the PF09, and allow the GUI to take control, click **Start** in the Connection toolbar in the top-left corner of the GUI window. Once the communication is established, the State message becomes "CONNECTED".

aaa-059788

Figure 71. USB cable plugged in, communication established

Note: *If the Start button does not light on and stays grey:*

4. Turn off and on the power supply.
5. Reset the KL25Z MCU via the button in the KL25Z board.
6. Verify (disconnect and connect the USB cable from the Computer) that Windows is recognizing and installing the USB drivers.

7.3 Operation modes

The KITPF09FRDMPGM provides three distinct operation modes with direct impact on device functionalities. Understanding these modes helps the user interact with the GUI properly.

The voltage level on PF09 DEBUG pin is one condition for entering a given operation mode. Table 6 gives the hardware configuration conditions to enter Normal, Debug, or OTP/Test mode.

Table 6. Mode entry hardware conditions

Configuration						
Board signal	**Signal name**	**Signal type**	**Run mode**	**Debug mode [1]**	**Test mode [2]**	**Emulation mode [2]**
FCCU0	SET_DPIN_ FCCU0	PFO9	W:0	W:0	W:0	W:0
PWRON	SET_DPIN_ PWRON	PFO9	W:1	➔ W:0 ➔ W:1	W:1	W:1
VDDOTP	SET_DPIN_ VDDOTP	PFO9	W:0	W:1	W:1	W:1
STBY	SET_DPIN_ STBY	PFO9	W:0	W:0	W:0	W:
Boost enable	SET_DPIN_ BSTEN	Free board	W:0	W:0	W:0	W:0

1. PWRON off, VDDOTP Activation, and PWRON on.
2. No changes in the control lines.

7.3.1 Run mode

The Normal mode operation is used for final product test in the automotive environment. Normal mode entry at power-on reset.

When Normal mode is active, the device operates with all the functionalities enabled by the loaded OTP configuration.

7.3.1.1 Run mode activation

There are two ways to activate Normal mode:

- With the GUI using mode selection in Connection Toolbar (recommended)
- Without the GUI (hardware only)

7.3.1.2 With the GUI using mode selection in the Connection toolbar (recommended)

1. Set the EVB configuration. See Section 7.1.
2. Plug the USB cable between the PC and the board.
   - The Red LED D11 for OTP 8 V generation is ON. The Blue LED D8 for OTP 5 V generation is Off, and the Green LED D12 for VIN is OFF.
3. Apply power supply VBAT.
   - The Red LED D11 for OTP 8 V generation is OFF. The Blue LED D8 for OTP 5 V generation is Off, and the Green LED D12 for VIN is ON.
4. Open the GUI and start the connection.

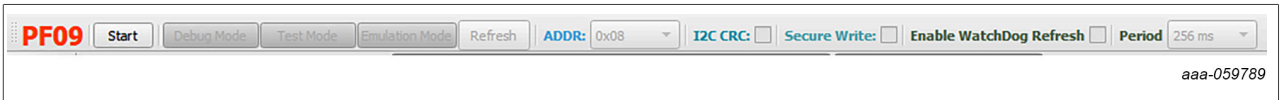


Figure 72. Open the GUI and start the connection.

Click to **Start.**
Start changes to **Stop** and **Debug mode** appears.

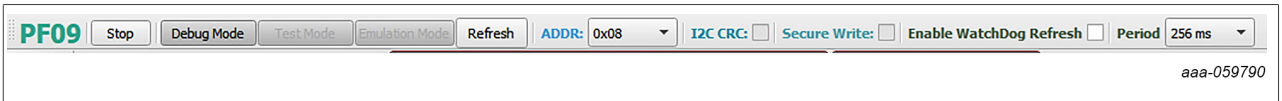


Figure 73. Start the GUI in Run mode.

5. Check that the GUI started in Run mode from the USB and device status bar:
   a. **FS CURRENT STATE:** RUN

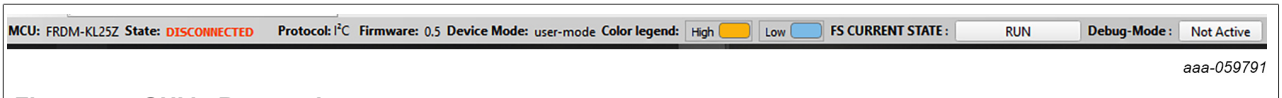


Figure 74. GUI in Run mode

6. The equipment is now set to Run mode.

7.3.2 Debug mode

7.3.2.1 Debug mode definition

The device starts in Debug mode when the hardware is configured accordingly. This mode requires the VDDOTP pin to be '1' to enter Debug mode automatically at start-up. This is done via GUI.

Debug mode works in parallel with Normal mode or Test mode. When Debug mode is active, the device runs with limited functionalities as some functions are disabled.

Note: *In Debug mode, OTP Registers cannot be modified.*

Debug mode is helpful during software development if the device has those functions enabled by OTP.

7.3.2.2 Debug mode activation

There are two ways to activate Debug mode:

7.3.2.3 With the GUI using mode selection in the Connection toolbar (recommended)

1. Set the default jumper configuration. See Section 7.1.
2. Plug the USB cable between the PC and the board.
   - The Red LED D11 for OTP 8 V generation is ON. The Blue LED D8 for OTP 5 V generation is Off, and the Green LED D12 for VIN is OFF.
3. Apply power supply VBAT.
   - The Red LED D11 for OTP 8 V generation is OFF. The Blue LED D8 for OTP 5 V generation is Off, and the Green LED D12 for VIN is ON.
4. Open the GUI and start the connection.
5. Click **Debug mode** when button is grey.
   - The button changes to green and **Test Mode** button appears.

Figure 75. Active Debug mode

6. Check that the GUI has detected Debug mode in the USB and device status bar.

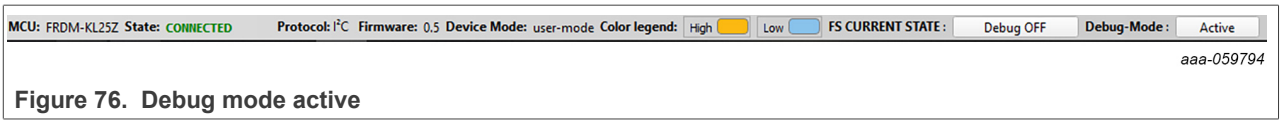


Figure 76. Debug mode active

7. The equipment is now set to Debug mode.

7.3.2.4 Debug mode deactivation

Once activated, Debug mode can be deactivated by Connection toolbar, removing power or by writing via I^2C the SYS_Diag(0x82) with 0x20, read and confirm a 0x04 value.

7.3.3 Test mode

7.3.3.1 Test mode definition

Test mode allows the user to write in the Mirror registers to configure or reconfigure the device for customer evaluation and to burn OTP fuses.

Note: *Mirror registers are an emulation of OTP registers. In a POR, the Mirror registers are reset to the default OTP configuration (empty if OTP not burned).*

Test mode requires **VDDOTP** = +8 V and valid Test mode keys sent by I^2C and to permanently fuse all the data stored in Mirror registers to the OTP sectors requires an extra key command, available in the PROG tool. The OTP fuse burning process is explained in Section 7.8.

When the Mirror registers configuration is done, the user can move back to Normal mode to power up the device with the given configuration.

7.3.3.2 Test mode activation: hardware and software

To start the device with a configuration loaded in the Mirror registers, follow the instructions below:

1. Set the default jumper configuration. See Section 7.1.
2. Plug the USB cable between the PC and the board.
   - The Red LED D11 for OTP 8 V generation is ON. The Blue LED D8 for OTP 5 V generation is Off, and the Green LED D12 for VIN is OFF.
3. Apply power supply VBAT.
   - The Red LED D11 for OTP 8 V generation is OFF. The Blue LED D8 for OTP 5 V generation is Off, and the Green LED D12 for VIN is ON.
4. Open the GUI and start the connection.
   - Click **Test Mode** button while grey to turn green.
   - **Emulation Mode** button activates after.

Figure 77. Activate Test mode

5. Check that the GUI has detected Debug mode in the USB and device status bar:

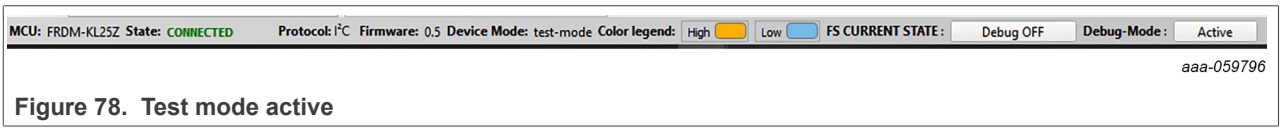


Figure 78. Test mode active

6. The equipment is now set to Test mode.

7.3.3.3 Test mode deactivation

Once activated, Test mode can be deactivated using the GUI by clicking on the green Test mode button from the Connection toolbar, removing power or by writing via I^2C the SYS_Diag(0x82) with 0x20, read and confirm a 0x04 value, this will set the device in normal mode.

7.3.3.4 Test mode operation

Operating in Test mode allows the user to create and test a preliminary version of a desired configuration in the Mirror registers, prior to submitting the configuration to the OTP fuse burning process.

There are two ways to load Mirror registers:

- With a TBB script via the SCRIPT tool. See Generate a TBB script.
- With the MIRROR tool. See OTP and Mirror registers.

These methods are functional with an empty or burned part. The procedure is explained in Section 7.7.

7.3.4 Emulation mode

7.3.4.1 Emulation mode definition

Emulation mode has the same capabilities as Test mode, plus diagnostic capabilities. Emulation mode requires **VDDOTP** = +8 V and valid Test mode keys sent by I^2C.

When the Mirror registers configuration is done, the user can move back to Normal mode to power up the device with the given configuration.

7.3.4.2 Test mode activation: hardware and software

To start the device with a configuration loaded in the Mirror registers, follow the instructions below:

1. Set the default jumper configuration. See Section 7.1.
2. Plug the USB cable between the PC and the board.
   - The Red LED D11 for OTP 8 V generation is ON, the Blue LED D8 for OTP 5 V generation is Off, and the Green LED D12 for VIN is OFF.
3. Apply power supply VBAT.
   - The Red LED D11 for OTP 8 V generation is OFF. The Blue LED D8 for OTP 5 V generation is Off, and the Green LED D12 for VIN is ON.
4. Open the GUI and start the connection.
   - Click the **Emulation Mode** button while grey to turn green.

Figure 79. Active Emulation Mode

5. Check that the GUI has detected Emulation mode in the USB and device status bar:

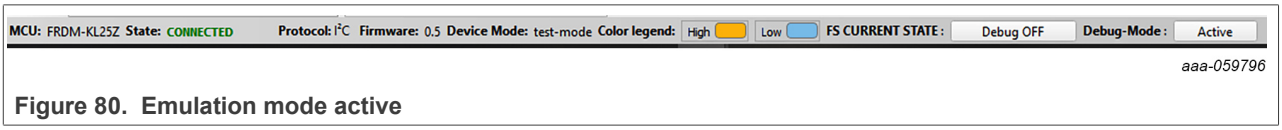


Figure 80. Emulation mode active

6. The equipment is now set to Emulation mode

7.3.4.3 Test mode deactivation

Once activated, Test mode can be deactivated using the GUI by clicking on the green Emulation mode button from the Connection Toolbar, removing power or by writing via I2C the SYS_Diag(0x82) with 0x20, read and confirm a 0x04 value, this will set the device in normal mode.

7.3.4.4 Test mode operation

Operating in Emulation mode allows the user to create and test a preliminary version of a desired configuration in the Mirror registers, prior to submitting the configuration to the OTP fuse burning process.

There are two ways to load Mirror registers:

- With a TBB script via the SCRIPT tool. See Generate a TBB script
- With the MIRROR tool. See OTP and Mirror registers

These methods are functional with an empty or burned part. The procedure is explained in section Section 7.7.

7.4 Generate a TBB script

A TBB file is needed to burn an OTP fuse or to emulate an OTP configuration by filling the Mirror registers through the Script tool.

To generate a TBB file, follow the procedure:

1. Go to the OTP tool from the Tool Access bar:
2. Configure the OTP parameters to fit the application's needs or import OTP configuration from a previously saved CFG file. The displayed parameters differ depending on the selected device type (ASIL B or QM) in the program Details box.
3. Enter Customer and program details in the window on the right side.
4. When done, export directly the OTP configuration to a TBB script by clicking **Export** in the Framework settings bar and choosing TBB:

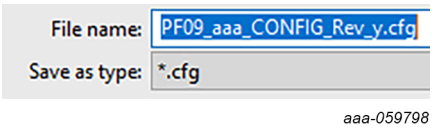


Figure 81. Save folder

5. Select the desired folder to save the TBB script:
   Note: *The script file must have a "PF09" prefix, with "aaa" being the part version and "y" the revision of the part version.*
6. TBB is now generated and saved.

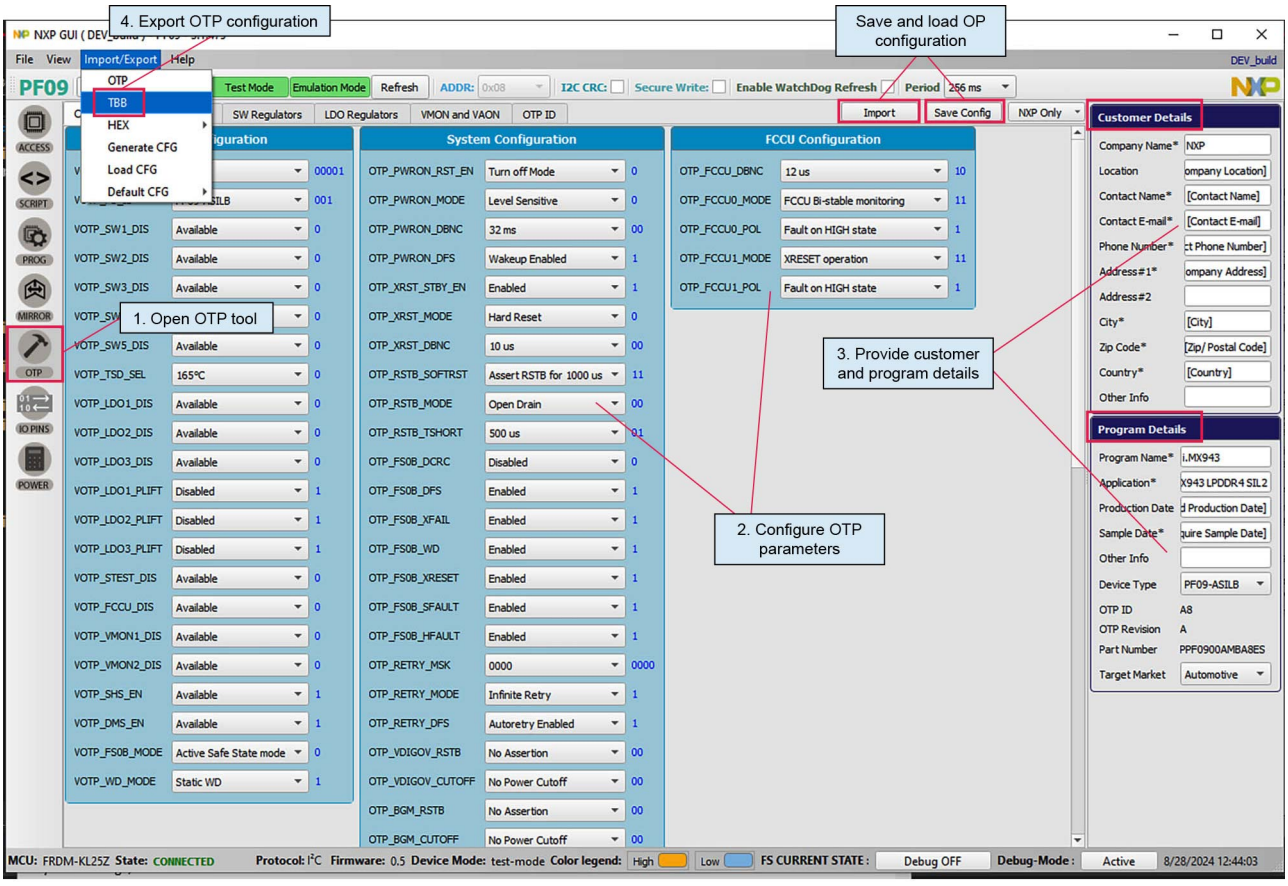
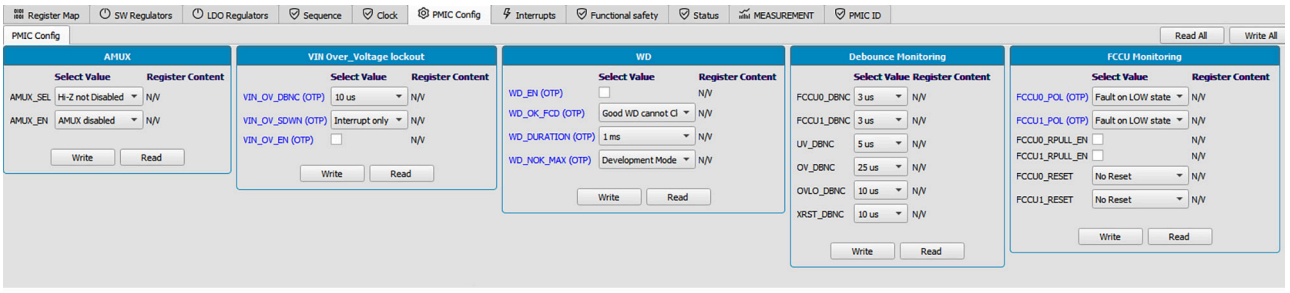


Figure 82. TBB script

7.5 First-start procedure: configure watchdog as Infinite Time Out

This procedure is to be used as a first-start and allows the user to enter system-level Normal mode execution with the watchdog configured with infinite timeout and window fully opened (equivalent to disabled).

1. Set up and connect the KITPF09FRDMPGM to the GUI using Section 7.1 and Section 7.2.
2. If the PF09 part is empty or if the configuration loaded from OTP should be modified:
   a. Enter test mode using Section 7.3.3.
   b. Load a configuration in the MIRROR tool, then write the configuration in the Mirror registers.
   c. Exit test mode from the Connection toolbar.
3. Go to ACCESS→ PMIC Config and click **Read all** to check that the device is in Normal, Debug, and INIT mode.
4. Go to ACCESS→ PMIC Config→ WD and configure WD_NOK_MAX(OTP) with Development mode. Figure 83 shows the location of the watchdog parameter to update in ACCESS tool:

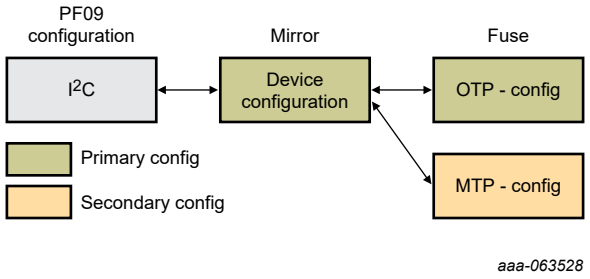


aaa-059800

Figure 83. ACCESS tool

7.6 OTP and Mirror registers

The device incorporates one OTP block for configuration. The OTP block is shared for main parameters and fail-safe parameters. Two sectors, OTP and MTP, are available so the device can be fused twice. The device configuration scheme is shown in Section 7.6.



aaa-063528

Figure 84. Device configuration block diagram

At device startup, the content of the last-programmed sector is loaded into the Mirror registers. The Mirror registers content is accessible from the NXP GUI. See Section 6.5.5. The NXP GUI manages the Mirror configuration, which facilitates access and content manipulation for OTP emulation. See Section 7.7.

The NXP GUI provides a tool to perform OTP programming, see Section 7.8. The first sector to be burned is S1, the second is S1bis. The NXP GUI automatically manages the second sector programming. It is not possible.

To revert to the previous sector. Once the user has reached MTP, it is not possible to burn the part again. However, OTP emulation mode is still available.

7.7 Operate OTP emulation by loading configuration in the Mirror registers

Mirror registers are an emulation of device registers (OTP/MTP). Mirror registers can be read/written multiple times, whereas device configuration registers can be burned twice.

Mirror registers can be read and written in Test mode only. See Section 7.3.

In a power-on reset, the Mirror registers are reset to the default OTP configuration (empty if OTP sectors not burned).

The MIRROR tool provides access to all Mirror registers with a similar disposition to the OTP tool.

7.7.1 Modify the Mirror registers with a TBB script

From a Test/Emulation mode active environment, the Mirror registers can be configured using the SCRIPT tool. See Section 6.5.4.

1. Open the configuration file:

Figure 85. Save folder

2. In the SCRIPT tool, click **OPEN** (fourth button) in the Script bar to load the TBB script file into the Script command window.

Figure 86. Script bar: Open

3. To execute the TBB script, click **RUN** (first button) in the bar at the bottom of the Script command window:

Figure 87. Script bar: Open

4. Deactivate Test mode to start the device with the given configuration and load the I^2C functional registers with the Mirror registers content.

As a crosscheck, open the ACCESS tool and check the fields in the Regulators tab, for example. If the operation is completed without problems, all fields should display the expected configuration.

7.7.2 Modify the Mirror registers with the MIRROR tool

To configure the Mirror registers, the MIRROR tool can be used to write/read directly into Mirror registers. See Section 6.5.5. This requires Test mode to be activated.

1. Open the MIRROR tool from the tool access bar:

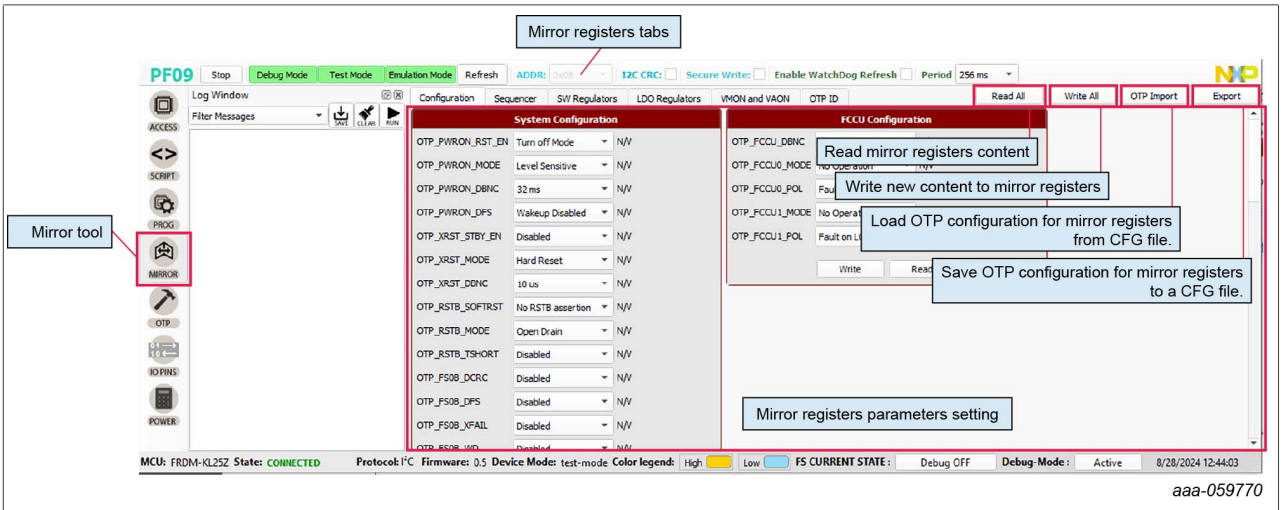


Figure 88. MIRROR tool

2. Configure Mirror registers to fit the application or load an existing configuration by importing a CFG file.
   Note: *It is possible to load a configuration from a previously made configuration save as a CFG file, using the Load CFG button in the top-right corner of the Mirror tool.*
3. Once done, click **Write All**. As a crosscheck, click **Read all**. If the operation is completed without problems, all fields should display the expected configuration.
4. Deactivate test mode to load the Mirror registers with the given configuration and start the device. I
   Note: *t is possible to save a current mirror registers configuration as a CFG file, using the Save CFG button in the top-right corner of the Mirror tool.*

7.8 Programming an OTP configuration

The PROG tool is used to permanently burn the PF09 fuses with the customer's OTP configuration from a TBB script file. See Section 6.5.3.

The TBB script can be generated from an OTP configuration. See Section 7.4.

Note: *The script file must have a "PF09" prefix, with "aaa" being the part version and "y" the revision of the part version.*

A PF09 device can be burned twice.

Requirement: Make sure that the socket contains a device that has not yet been burned before starting the burning process.

Follow this procedure to verify that the part is empty:

1. Set the default jumper configuration. See Section 7.1.
2. Plug the USB cable between the PC and the board.
3. Apply power supply VBAT.
4. Open the GUI and start the connection.
5. Switch to Test/Emulation mode.
6. Open the PROG tool from the tool access bar.
7. Make click on "Read" in the "Fuse Box status" section.
8. Check the flags in the Sector flags section of the Fuse box status window. See Section 6.5.3.

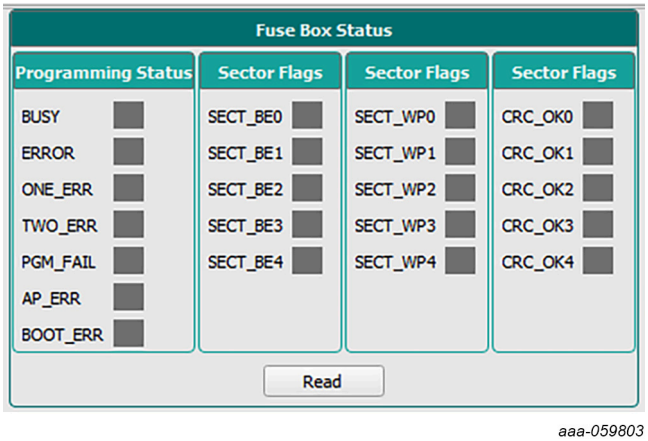


Figure 89. Sector flags status for an empty part

The device is empty and capable of being burned. It is not fused.

Figure 90 shows the Sector Flags status in the Programming interface for a VOTP section burned. It is VOTP fused.

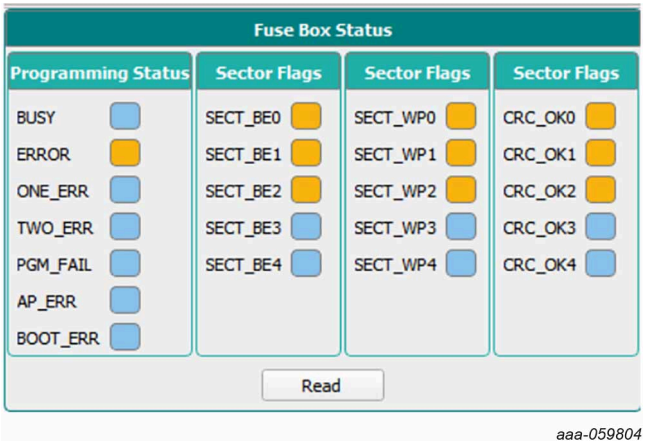


Figure 90. Sector flags status for VOTP programming

Figure 91 shows the Sector flags status in the Programming interface for a part burned once. It is OTP fused.

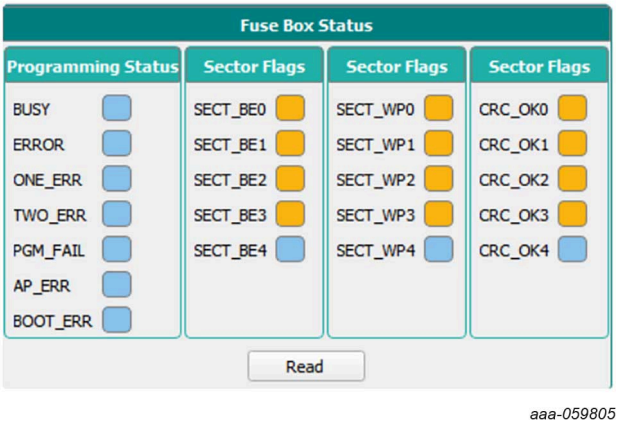


Figure 91. Sector flags status for OTP programming.

Figure 92 shows the Sector flags status in the Programming interface for a part burned twice. It is MTP fused.

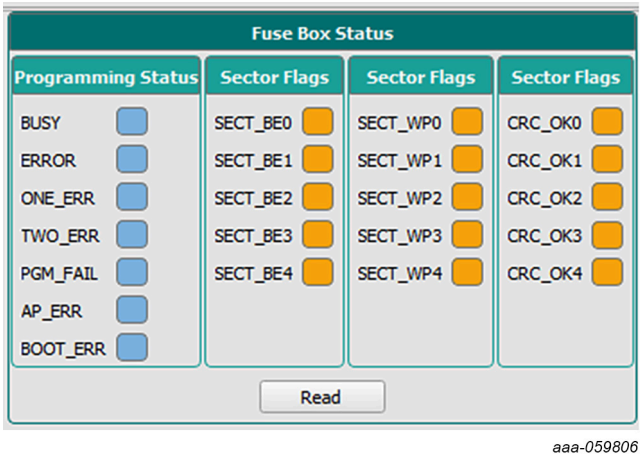


Figure 92. Sector flags status for MTP programming

Programming procedure:

1. Set the default jumper configuration. See Section 7.1.
2. Plug the USB cable between the PC and the board.
3. Apply power supply VBAT.
4. Open the GUI and start the connection.
5. Switch to Test/Emulation mode.
6. Open the PROG tool from the tool access bar.
7. Select TBB File
8. Click the **program** to start the burning process.

7.9 Save a routine from the Log window then Run it as a Script

The Log window shows the requests and answers transiting between the KL25Z MCU and the PF09 device. The commands are sent using I^2C.

Requests are sent to PF09 after a user action on the graphical interface. Answers are received by the MCU, then displayed to the user in the graphical interface. These exchanges are stored in the Log file.

When the user must operate the same routine multiple times on the device, for example to automate loading the Mirror registers, as shown in Figure 49, it is possible to record the actions from the Log file then run the routine later as a Script.

In the Script .txt file, it is possible to add:

- A delay between successive commands using DELAY: xx command with xx the delay in milliseconds.
- A pause in the script, for example to have time to change hardware configuration between two commands, using the PAUSE command.
- A comment in the script using // at line start.

8 Revision history

Document ID	Release date	Description
UM12433 v.1.0	20 November 2025	Initial release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Tables

Figures

Contents