How to use STPMIC1L for a wall adapter powered application on STM32MP13x lines MPUs

Introduction

This application note applies to the STM32MP13x MPU devices as detailed in the table below. The devices are referred to as STM32MP13x in the rest of the document. It is powered by the STPMIC1Lx power management IC, which is fully featured to supply a core chipset (STM32MP13x, DDR memory, and a flash memory).

This document provides an example of a hardware reference design based on an STM32MP13x device. The latter is powered by a 5 V power supply external source via the STPMIC1LAPQR power management IC, suitable for peripheral I/O voltages of 3.3 V.

This document is intended for product architects and designers who require information about power management and STPMIC1Lx settings, and it focuses on:

- Reference design block diagram
- Power distribution topology
- Power on/off and low-power management
- User reset and crash recovery management
- Safety management and PMIC tuning

Table 1. Applicable products

Reference	Applicable products
STM32MP13x	STM32MP131A, STM32MP131C, STM32MP133A, STM32MP133C, STM32MP135A, STM32MP135C
STPMIC1Lx	STPMIC1LAPQR

1 General information

This document applies to STM32MP13x Arm®-based MPUs and STPMIC1Lx power management IC.

Note: Arm® is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

2 Overview

This application note describes the interaction between the STM32MP13x and the STPMIC1LAPQR including the management of the following peripherals:

- DC input power source from main power supply: 5 V typical (4.1 V to 5.5 V)
- DDR3L memory
- Peripheral I/O interface voltage (V_{DDIO}) at 3.3 V powered by the STPMIC1LAPQR
- SD card as boot device

Not covered in this application note:

- Other DDR type (lpDDR3)
- Peripheral interface with I/O voltage (V_{DD}) of 1.8 V

In this document, MPU terminology refers to the STM32MP13x. The PMIC terminology refers to the STPMIC1Lx generic device. The STPMIC1LA is used to highlight specific behaviors predefined in the STPMIC1LAPQR NVM.

2.1 Reference documents

Table 2. Reference documents

-	Reference	Title
STMicroelectronics document [1]		
[1]	DS14839	STPMIC1L highly integrated power management IC for microprocessor units
[2]	RM0475	STM32MP13xx advanced Arm®-based 32-bit MPUs
[3]	AN5474	Getting started with STM32MP13x lines hardware development
[4]	DS13874	STM32MP135A STM32MP135D
[5]	Wiki page DRR config	https://wiki.st.com/stm32mpu/wiki/DDRCTRL_and_DDRPHYC_device_tree_configuration

1. *Refer to www.st.com.*

3 Glossary

Table 3. **Glossary**

Term	Meaning
BUCK	Step down SMPS regulator
LDO	Low drop out linear regulator
MPU	Microprocessor unit
NVM	Non-volatile memory
PMIC	Power management integrated circuit
SMPS	Switching mode power supply
SW	Software

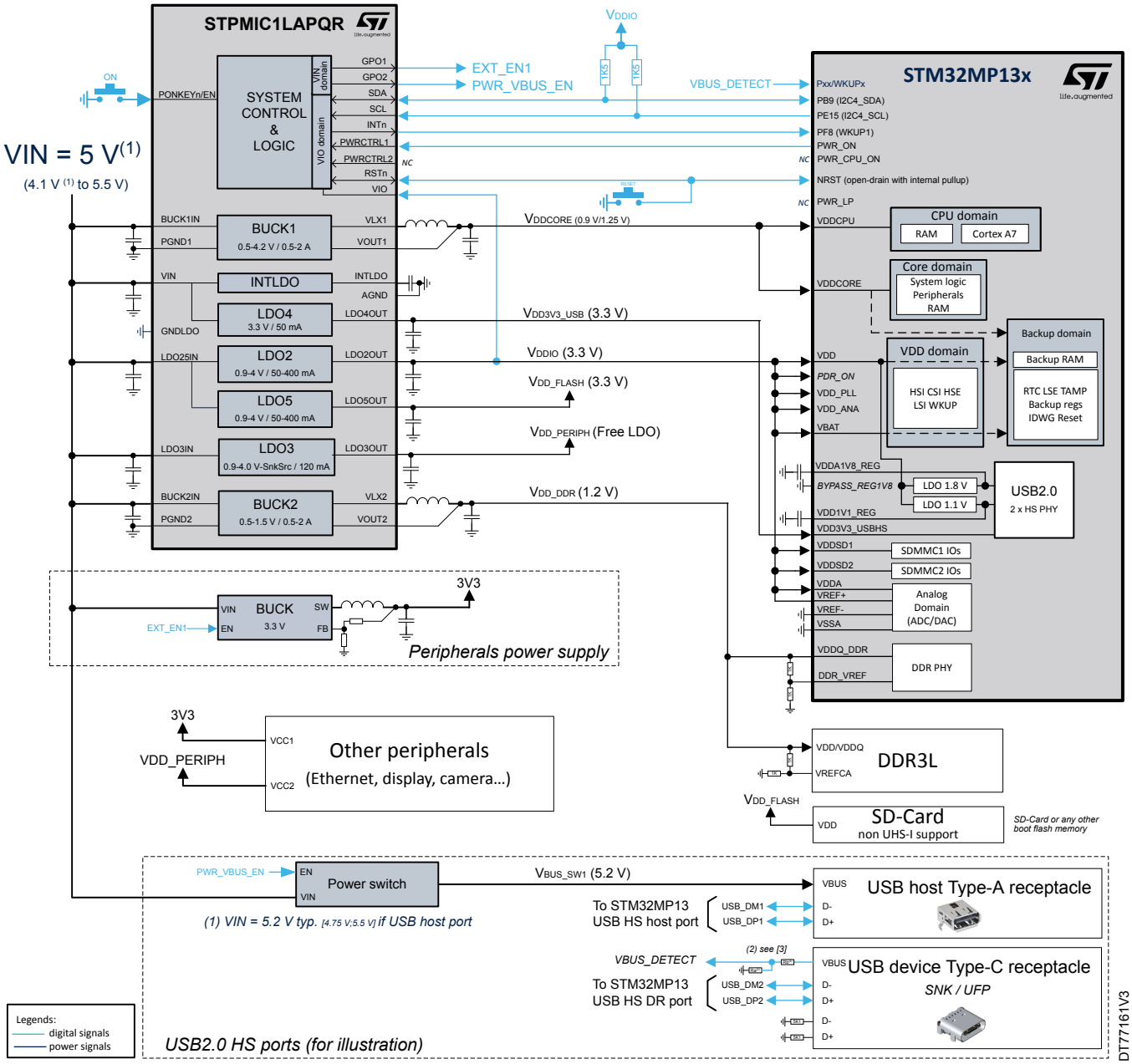
4 5 V power supply application reference design

The reference design shown in Figure 1 targets an application powered by a main supply of 5 V (5.2 V if USB host port). The STM32MP13 core and CPU voltages, the DDR3L DRAM, and the SD card boot flash memory are powered from the STPMIC1LA. For illustration, the application peripherals such as USB ports, SD card, or Ethernet PHY are powered from a discrete regulator or a power switch.

Note: The SD card boot flash memory is used as an illustration. It could be replaced by any supported flash memory such as eMMC, NAND flash memory, or serial flash memory (see [3]).

The main peripheral interfaces operate with an I/O voltage of 3.3 V.

Figure 1. STM32MP13x and STPMIC1LA with DDR3L, SD card, and external SMPS to supply peripherals



Note: The following elements are not shown in the diagram:

- *STM32MP13x decoupling scheme (see [3]).*
- *STPMIC1LA discrete components value (see [1]).*
- *VIN source and related protection, such as ESD, EMI filtering, and overvoltage protection.*

4.1 Power distribution

The STPMIC1Lx integrates the regulators that supply:

- The STM32MP13x power domains
- The application DRAM and flash memory

4.1.1 V_{DDCPU} power domain (900 mV/1.25 V)

V_{DDCPU} supplies the MPU Arm® Cortex®-A7 CPU digital power domain.

With STPMIC1LA, the V_{DDCPU} is merged with the V_{DDCORE} at PCB level and is powered from the BUCK1 step-down SMPS (see Section 4.1.2). Accordingly, the CPU frequency is limited to nominal value (up to 650 MHz) as overdrive voltage is not allowed on the V_{DDCORE} domain.

4.1.2 V_{DDCORE} power domain (900 mV/1.25 V)

V_{DDCORE} is the main MP13 digital power domain.

V_{DDCORE} supplies all core domains including the system logic, internal RAM, peripherals, and the backup domain.

V_{DDCORE} is merged with V_{DDCPU} at PCB level and powered from the PMIC BUCK1 step-down SMPS which has high efficiency and has excellent load transient response across operating conditions.

At power-up, the V_{DDCORE} is enabled automatically by the PMIC at 1.22 V, then must be set by software at nominal voltage (1.25 V) during MPU initialization. (See Section 5.2.1)

V_{DDCORE} is enabled in:

- Run mode

V_{DDCORE} is lowered to 900 mV in:

- Low-power LPLV-Stop mode

V_{DDCORE} is disabled in:

- Low-power Standby mode
- V_{BAT} mode and OFF mode

In low-power mode, the PWR_ON output of the MPU manages the PMIC V_{DDCORE} regulator. The PWR_ON output is connected to the PWRCTRL1 input of the PMIC. V_{DDCORE} also supplies the backup domain (see document [4]) in Run, Stop, and LPLV-Stop modes.

4.1.3 V_{DDIO} power domain (3.3 V)

V_{DDIO} is the power supply for the following independent MPU domains:

- V_{DD}
- V_{DDSD1}
- V_{DDSD2}
- V_{DD_PLL}
- V_{DD_ANA}
- V_{DDA}/V_{REF+}
- V_{BAT}

The MPU V_{DD}, V_{DD_ANA}, and V_{DD_PLL} domains must be connected together. They supply the MPU I/Os, the system analog such as oscillators (HSE, HSI), and PLLs.

The MPU V_{DDA} supplies the ADC and the voltage reference buffer (VREFBUF) to generate the V_{REF+} reference voltage of the ADC. The ADC performance is directly impacted by the noise level from the V_{REF+} source, but also by the V_{DDA} source noise level (due to the V_{DDA} power supply rejection ratio).

Note: If V_{DDA} is powered from the V_{DDIO} power domain, a low pass filter with low DC impedance might be inserted in between the V_{DDIO} power source and V_{DDA} depending on the required ADC performance.

Note: V_{REF+} must be connected to the V_{DDIO} power source only if limited ADC performance is expected.

The MPU V_{BAT} supplies the retention domain which includes the backup RAM, RTC, LSE, tamper, backup registers, watchdog, and reset blocks. In this application, the MPU V_{BAT} is powered from V_{DDIO} domain as no backup battery is present. VBAT may be supplied from a backup battery if the application requires keeping the backup domain powered when the main power supply source of the application (VIN) is removed.

VDDIO also supplies the PMIC VIO domain which embeds the I^2C interface, the PWRCTRLx, the RSTn, and the INTn pins. (See Section 4.2.)

V_{DDIO} is powered by the PMIC LDO2 linear regulator which has a very low quiescent current to reduce power consumption during low-power mode.

At power-up, the V_{DDIO} is enabled automatically by the PMIC at 3.3 V. The LDO2 is the first regulator switched on at power-up. (See Section 5.2.1.)

V_{DDIO} is ON in all modes except in OFF mode or V_{BAT} mode, when the main power source (VIN) of the application is removed.

4.1.4 V_{DD3V3_USB} power domain

V_{DD3V3_USB} power domain supplies the USB2.0 HS PHY (V_{DD33_USBHS}) of the MPU.

V_{DD3V3_USB} is powered from the dedicated PMIC LDO4 having a fixed output voltage at 3.3 V.

At power-up, the V_{DD3V3_USB} regulator is automatically enabled at 3.3 V by the PMIC. (See Section 5.2.1.)

V_{DD3V3_USB} can be kept enabled in Run, and LPLV-Stop modes if a USB peripheral is connected to the application.

V_{DD3V3_USB} must be disabled in Standby and OFF mode: when V_{DDCORE} is OFF.

At runtime, V_{DD3V3_USB} is controlled by the MPU software through an I²C command to the PMIC. In low-power mode, PWR_ON output of the MPU can manage the PMIC V_{DD3V3_USB} regulator via the PWRCTRL1 input of the PMIC.

4.1.5 DDR power domain (V_{DD_DDR})

V_{DD_DDR} is dedicated to DDR3L volatile memory IC power supply (V_{DD} and V_{DDQ}) and the MPU DDR interface (V_{DDQ_DDR}).

V_{DD_DDR} (1.35 V) is powered from the PMIC BUCK2 step-down SMPS.

Note: V_{TT_DDR} is an optional supply to the DDR3L IC bus termination resistor network that is not required for point-to-point topology (single DDR3L IC as illustrated in Figure 1) and recommended for fly-by topology (several DDR3L ICs). When required, V_{TT_DDR} (0.675 V) is powered from the PMIC LDO3 multipurpose LDO and must be set in sink-source mode. It provides voltage equal to BUCK2 output voltage / 2 (V_{OUT2} / 2). In that case, the PMIC LDO3 is dedicated to power supply DDR3L bus termination resistors network.

Note: When the LDO3 is used in sink-source mode (V_{TT_DDR}), it must be supplied from BUCK2 output (LDO3IN = V_{DD_DDR}).

Note: Resistor voltage dividers of 1K/1K are required to supply respectively the DDR3L IC V_{REFCA} and MPU DDR_V_{REF} reference voltages using V_{DD_DDR} as a voltage source for the resistor voltage dividers.

At power-up, V_{DD_DDR} regulator is not automatically started by the PMIC. It is powered up by software by sending I²C commands to PMIC (see Section 5.2.1 and [5] for more details).

V_{DD_DDR} is enabled in:

- Run mode
- Stop mode
- Low power LPLV-Stop mode
- Standby mode suspend to RAM (see Section 5.3.3: Standby mode (DDR3L in self-refresh))

V_{DD_DDR} is disabled in:

- Standby mode suspend to flash (see Section 5.3.4: Standby mode (DDR3L OFF))

4.1.6 V_{DD_FLASH} power domain (3.3 V)

V_{DD_FLASH} power domain aims to supply the boot flash memory of the MPU (handled by the MPU boot ROM). As illustrated in Figure 1, the boot flash memory is an SD card. However, any supported flash memory such as eMMC, NAND flash memory, or serial flash memory can replace the SD card (see [3]).

V_{DD_FLASH} is powered from the PMIC LDO5 linear regulator.

At power-up, the V_{DD_FLASH} is automatically enabled by the PMIC at 3.3 V. (See Section 5.2.1.)

The V_{DD_FLASH} regulator is controlled by software at runtime by sending an I2C command to PMIC. Additionally, it is controlled by MPU PWR_ON signal (via PMIC PWRCTRL1) to switch OFF V_{DD_FLASH} when entering a low-power mode and to switch ON V_{DD_FLASH} on low-power mode exit. This is specifically suitable for Standby mode as MPU boot ROM requires access to the boot flash memory on standby mode exit.

4.1.7 3V3 peripherals power supply (3.3 V)

As illustrated in Figure 1, a 3V3 power domain is used to power supply application peripherals around the MPU (Ethernet PHY, display, camera…).

3V3 is powered from a discrete regulator, such as a general purpose 3.3 V step-down SMPS. A PMIC GPOx can be used to control it:

- STPMIC1LA GPO1: automatically set by PMIC during power-up sequence
- STPMIC1LA GPO2: keep to 0 by PMIC during the power-up sequence

See GPO1 and GPO2 for more details.

4.2 Control signals between STPMIC1LA and STM32MP13x

This section outlines the default behavior of the STPMIC1LA and the method of communication with the STM32MP13x. (See [1] for more details.)

4.2.1 STPMIC1LA default behavior with STM32MP13x

The PMIC NVM settings are configured to boot the MPU from flash memory, such as an SD card, or to boot the MPU from the USB interface. In production, booting from the USB interface is suitable for flashing and then executing the production software and/or the final application software.

The default NVM configuration is available in [1] and summarized in the following table:

Table 4. Default STPMIC1LA NVM configuration

Regulator	Name	Rank	Default output voltage	Default configuration
BUCK1	V_{DDCORE}	RANK2	1.22 V	ON
BUCK2	V_{DD_DDR}	RANK0	N/A	OFF
LDO2	V_{DDIO}	RANK1	3.3 V	ON
LDO3	*Free*	RANK0	N/A	OFF
LDO4	V_{DD3V3_USB}	RANK5	3.3 V	ON
LDO5	V_{DD_FLASH}	RANK4	3.3 V	ON
GPO1	EXT_EN1	RANK3	N/A	ON
GPO2	EXT_EN2	RANK0	N/A	OFF

The STPMIC1L regulators startup is spread over five ranks. This is to comply with the MPU power-up sequence constraints and to avoid current peaks on the main power supply. The voltage value of each regulator is defined to fit with the MPU optimum voltage requirements.

Safety management

The STPMIC1LA safety management is set by default in NVM to restart systematically the application. When a hard fault is detected, the PMIC performs a power cycle: reset assertion, then power-down sequence, then power-up sequence, and reset deassertion to restart the application. (See Section 6 for more details.)

PMIC tuning (optional)

The STPMIC1LA NVM can be reprogrammed by the customer to tune regulator settings, and the safety management behavior to fit with the MPU based application requirements. This action can be done with the ST tools STM32CubeProgrammer.

4.2.2 STPMIC1L digital control interface

The PMIC integrates:

- An I²C slave interface,
- Three digital input control pins (PONKEYn/EN, PWRCTRL1, PWRCTRL2),
- A digital output interrupt pin (INTn),
- A bidirectional digital reset pin (RSTn),
- Two general-purpose outputs (GPO1/2).

I²C interface

The PMIC is controlled by the MPU via the I²C interface to:

- Enable/disable, set the voltage, and operating mode of the regulators.
- Set regulators external control for low-power mechanisms (PWRCTRL1/2).
- Set the interrupt controller or read interrupt status.
- Set the protection (watchdog, overcurrent, undervoltage) or read protection status.
- Tune the PMIC NVM default configuration for end-product (power-up sequence, safety management).

PONKEYn/EN pin

The PONKEYn/EN pin can be muxed either with the PONKEYn or EN digital input feature.

For STPMIC1LA, this pin is muxed as PONKEYn by default. The PMIC NVM can be reprogrammed to change this setting to the EN pin feature using the PKEY_EN_CFG bit in NVM_MAIN_CTRL_SHR3 register:

- When the PKEY_EN_CFG is set to 0, the PONKEYn/EN digital input acts as PONKEYn (default).
- When the PKEY_EN_CFG is set to 1, the PONKEYn/EN digital input acts as EN.

PONKEYn pin description

The STPMIC1L PONKEYn pin is a digital active low input signal with a built-in pull-up resistor. It is usually connected to a user push-button allowing the following operations:

- Turn on the PMIC (from PMIC OFF state).
- Wake up the application from a low-power mode (typically from Standby mode) by generating an interrupt on signal falling or rising edge.
- Force a switch-off or a power cycling condition with a long press. This duration is programmable as described in Turn-off conditions.

Note: The usage of a user push-button connection to PONKEYn is optional as the STPMIC1LA is automatically turned on when the application is powered. (See Section 4.2.1.)

EN pin description

The STPMIC1L EN (enable) pin is a digital input signal with a programmable polarity and a programmable pull-up or pull-down resistor. The EN feature powers ON the PMIC when the pin is active (for example: EN = 1) or powers OFF the PMIC when the pin is inactive (for example: EN = 0).

This feature is mainly targeted for USB bus-powered application, where the EN pin of the PMIC is controlled from a USB UCSI power delivery controller to power ON/OFF the application. Alternatively, this new feature enables usage of several PMIC working together into the same application.

RSTn pin

The PMIC RSTn pin must be connected to the MPU NRST pin. Additionally, it can be connected to a “RESET” user push-button. This pin has a built-in pull-up resistor. Therefore, no additional discrete pull-up resistor is needed on this signal. Nevertheless, a 10 nF capacitor to GND must be placed as close as possible to the MPU NRST pin. This is specifically required to avoid EMI/ESD coupling as there is no debounce circuitry in the MPU or the PMIC.

The PMIC RSTn pin is a digital active low bidirectional signal with a built-in pull-up resistor:

- When PMIC asserts RSTn (such as during the power-up or the power-down sequence), it drives the NRST signal low (open drain). The MPU is forced into a system reset until the PMIC releases the RSTn.
- When the MPU asserts an NRST signal such as an MPU watchdog event, or by pressing the “RESET” button, the PMIC immediately asserts its RSTn pin and performs a noninterruptible power cycle. The PMIC performs a power-down sequence followed by a power-up sequence and releases the RSTn.

At the end of the power-cycle sequence, PMIC waits for the NRST signal to go high before rearming the reset detection mechanism to avoid infinite loop reset.

INTn signal

The PMIC INTn pin is a digital output (open drain) active low interrupt line connected to the MPU PF8 (wake-up) input pin. This pin has a built-in pull-up resistor. Therefore, no additional discrete pull-up resistor is needed on this signal.

PF8 has both interrupt and wake-up capabilites:

- To manage interrupt from the PMIC when the MPU operates in either Run or low-power mode (except Standby mode).
- To wake up the MPU when it operates in Standby mode.

PWRCTRL1, PWRCTRL2

The PMIC has two power control digital input signals that can be connected to dedicated MPU control signals.

Each PMIC regulator can be set to be controlled from a single PWRCTRL signal. Thus, typically to switch ON/OFF or to change the output voltage of a regulator depending on the PWRCTRL signal state. Alternatively, a PWRCTRL signal can be set to reset the regulator at the value defined in PMIC NVM (see document [1] for more details).

As illustrated in Figure 1 application, the MPU PWR_ON output pin controls the PMIC PWRCTRL1 input pin. The MPU PWR_ON pin has an internal mux with either PWR_ON or PWR_LP signal:

- PWR_ON pin is mux to PWR_LP signal to manage LPLV-Stop mode.
- PWR_ON pin is mux to PWR_ON signal (default) to manage Standby mode.

(See Section 5.1.2 for more details about PMIC PWRCTRL settings.)

GPO1 and GPO2

The PMIC has two general-purpose push-pull (internally referenced to VIN voltage) outputs. Those two outputs are targeted to control discrete external regulators.

For example, to control regulators supplying application peripherals or discrete power switch as illustrated in Figure 1 application. It can also be used as general-purpose GPOs to control some peripherals or to control another PMIC in slave mode (for example, an STPMIC1L GPOx connected to another STPMIC1L EN input pin).

GPO1 and GPO2 are driven by the PMIC with the same registers set as any other PMIC regulator. Accordingly, they are seen as PMIC regulators by the MPU PMIC software driver (see Table 4 for default value of GPO1 and GPO2 at PMIC power-up).

5 Power management

5.1 Operating modes

The application can switch to different operating modes depending on the system activity. The MPU manages the operating modes, which in turn control the power management. The operating modes are described in the table below according to the application illustrated in Figure 1.

Table 5. Operating modes

Operating mode	PMIC state	PWR_ON	Description	Notes
Run	POWER_ON	1	V_{DDIO} power on V_{DDCORE} power on System clock on Peripherals power on/off (via GPO1/2) DDR3L active	-
LPLV-Stop	POWER_ON	0	V_{DDIO} power on V_{DDCORE} power on at lower voltage System clocks off Peripherals power on/off (via GPO1/2) DDR3L self-refresh	MPU PWR_ON pin is internally muxed to PWR_LP signal
Standby	POWER_ON	0	V_{DDIO} power on V_{DDCORE} power off System clocks off Peripherals power off (via GPO1/2) DDR3L self-refresh or off	MPU PWR_ON pin is internally muxed to PWR_ON signal
V_{BAT}	NO_SUPPLY	-	Backup domain powered from backup battery if present	-
OFF	OFF	-	All regulators power off Backup domain powered from backup battery if present	-

5.1.1 Application turn-on/turn-off conditions

The PMIC autonomously manages the power-up and the power-down sequence when respectively a turn-on or a turn-off condition occurs.

The PMIC automatically powers up when the application is powered from a valid power source. When VIN rises above the PMIC V_{INOK_rise} internal threshold (see [1] for more details), it triggers a PMIC turn-on condition as the “AUTO turn-on” bit is set by default in the STPMIC1LA NVM.

Turn-on conditions

When the application is in OFF mode (PMIC in OFF state with V_{IN} present), a turn-on condition is required to power up the PMIC, and then run the application. Similarly, if the application needs to go into power-off mode, a turn-off condition is required to power-down the PMIC.

If the PMIC is in OFF state, it is powered up by one of the three triggers described in the table below:

Table 6. STPMIC1L turn-on conditions

Condition	Trigger	Description
AUTO turn-on	Internal	The PMIC starts automatically when the VIN voltage rises above the V_{INOK_rise} threshold. This feature is set by default in the STPMIC1LA (see “AUTO turn-ON” section in [1] for more details)
PONKEYn user button pressed	External	PONKEYn/EN pin falling edge (PONKEYn feature set in PMIC NVM: default)
EN pin asserted	External	PONKEYn/EN pin asserted (EN feature set in PMIC NVM)

After a turn-on condition, the PMIC carries out a transitional power-up sequence as described in Section 5.2.

Turn-off conditions

A turn-off condition leads the PMIC to perform a transitional power-down sequence. It begins by asserting the RSTn (MPU NRST), then turns off the regulators in the reverse order of the power-up sequence. Once the PMIC ends the power-down sequence, it goes into one of the following states:

- OFF state
- Automatic restart (power cycle)
- FAIL_SAFE_LOCK state (see [1] for more details)

This depends on whether the source that has triggered the turn-off condition is a common switch-off or hard fault. For a more detailed explanation about hard faults, see Section 6: Safety management.

The turn-off conditions are described in the table below:

Table 7. PMIC turn-off conditions

Condition	Hard fault	Description
EN	NO	EN deasserted (PKEY_EN_CFG bit set in PMIC NVM) *Note: The PMIC goes in a transitional power-down state. It then goes and stays in OFF state as long as EN is deasserted.*
Software switch OFF	NO	I²C commands “SWOFF” sent by the MPU to the PMIC *Note: The PMIC goes in a transitional power-down state. It then goes and stays in OFF state until a turn-on condition is met. If the restart request bit (RREQ_EN) is set with the SWOFF bit, the PMIC restarts automatically.*
Overcurrent	YES	Overcurrent or short-circuit on predefined regulators (See Section 6.2.1)
VIN undervoltage	YES	VIN voltage falls below the PMIC V_{INOK_fall} threshold *Note: If restart conditions are met, the PMIC waits for the VIN voltage to rise above the VINOK_rise threshold before powering up.* (See Section 6.2.2)
Thermal shutdown	YES	PMIC temperature above T_{SHDN_Rise} threshold *Note: If restart conditions are met, the PMIC waits for the temperature to decrease below T_{SHDN_Fall} before powering up.* (See Section 6.2.3)
Watchdog	YES	PMIC watchdog timer elapsed *Note: The watchdog timer is not enabled by default.* (See Section 6.2.4)

Condition	Hard fault	Description
PONKEYn user button long press	YES	PONKEYn signal is asserted for 10 s (See Section 6.2.5)

Note: The PMIC restarts automatically after a turn-off condition is triggered by a hard fault source (behavior programed by default in the STPMIC1LA NVM).

5.1.2 STPMIC1L power control management (PWRCTRLx)

PMIC PWRCTRL1/2 input signals are dedicated to managing MPU power modes. These signals must be configurated before the MPU enters low-power mode, to ensure expected regulators or GPOx behavior when MPU enters and exits low-power mode.

The PMIC PWRCTRL1/2 signals are independently driven from MPU (typically PWR_ON) to control some PMIC regulators (or GPOs) behavior by setting the appropriate registers as described in [1]. As such, it is possible to define:

- The control source selection of the regulator (PWRCTRL1 or 2).
- The polarity of the respective PWRCTRLx is used to determine whether the signals are active low or active high.

A PMIC PWRCTRL signal aims typically to switch between two control registers: xxxx_MAIN_CR and xxxx_ALT_CR, a PMIC regulator, or a GPO. xxxx corresponds to the targeted regulator or GPO.

For example, when a PWRCTRL signal is driven by MPU from high to low state, the PMIC internally switches from a main (xxxx_MAIN_CR) control register content to an alternate (xxxx_ALT_CR) control register content, and vice versa.

5.1.3 STPMIC1L mask-reset option

If the application needs to have one or several PMIC regulators to be kept enabled while the PMIC performs a reset sequence, the MPU bootloader software must program the PMIC mask reset option by setting:

- The PMIC BUCKS_MRST_CR register to targeted BUCK converters
- The PMIC LDOS_MRST_CR register to targeted LDO
- The PMIC GPOS_MRST_CR register to targeted GPO

A reset sequence is triggered after the PMIC RSTn signal is asserted by the MPU or the user reset push button. Refer to [1] for more details on the PMIC mask_reset option.

This is typically the case for the LDO2 powering the MPU V_{DD} power domains. The power cycle on V_{DD} must be masked by setting the PMIC register: LDOS_MRST_CR [1] = 1.

This prevents losing the content in:

- The MPU backup RAM
- The MPU backup register content
- The JTAG debug interface (included in the OTP controller)

Note: These settings must be programmed by the MPU software bootloader via an I²C command to the PMIC after each application power-up. This is necessary because the content of BUCKS_MRST_CR, LDOS_MRST_CR and GPOS_MRST_CR is reset at the end of a PMIC reset cycle.

5.2 Application power-up/power-down sequence

The power-up sequence is the transition managed by the STPMIC1LA between application power-off and application Run mode and reciprocally for application power-down mode. The application power-up and power-down sequence shown in Figure 2 is based on the reference designed in Figure 1.

5.2.1 Power-up triggered by main supply (VIN) plugin and power-down by software shutdown

When the application is connected to an external power supply, the PMIC powers up automatically when VIN rises above PMIC V_{INOK_rise} threshold. When power-up ends, the PMIC releases the RSTn signal.

Note: The STPMIC1LA has the AUTO_TURN_ON bit and PONKEYn feature enabled by default in PMIC NVM. In this section, PMIC terminology refers to STPMIC1LA.

Once the RSTn signal is released, the MPU boots (including the DDR3L initialization by MPU software) and the MPU reaches system run mode.

When the MPU software sends the "SWOFF" command to PMIC, a turn-off condition occurs, the PMIC enters power-down, and then goes into the OFF mode.

The above sequence is detailed below and illustrated in Figure 2:

1. The application has no power.
2. A power supply is connected to the application: VIN voltage rises.
3. Once VIN voltage is above PMIC V_{INPOR_Rise} (2.3 V typ.) threshold:
   a. The PMIC goes to INIT&LOAD transitional state to preload its NVM contents and checks its integrity. If the PMIC NVM integrity is valid, the PMIC initializes, launches, and executes the VIN_DLY = 10 ms.
   b. Once VIN_DLY elapses, the PMIC states machine goes in the CHECK&LOAD transitional state (as the AUTO_TURN_ON bit is set in PMIC NVM: see Turn-on conditions).

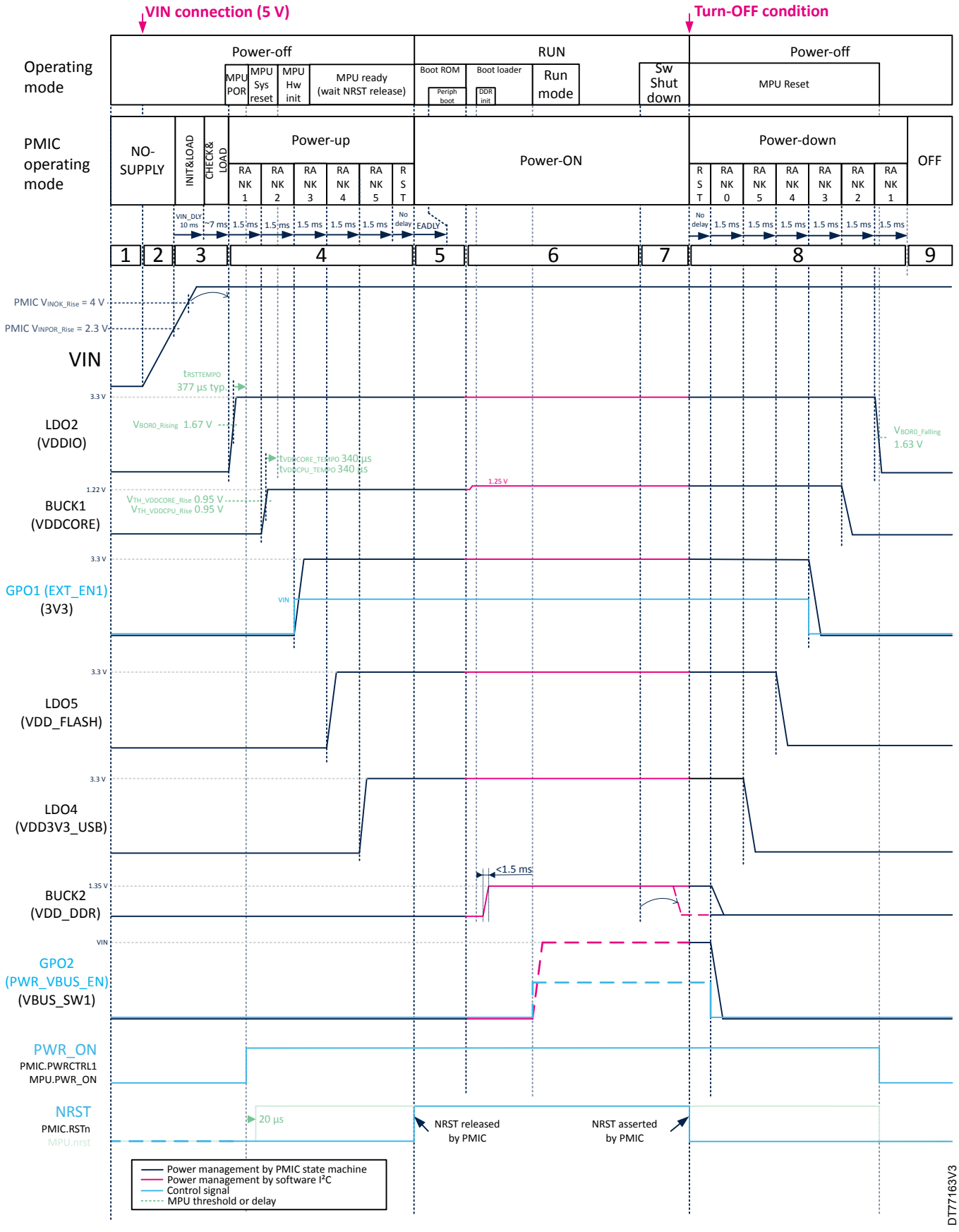
Note: The VIN_DLY timer is a passive delay used to wait for VIN voltage stabilization. It is set to 10 ms by default in PMIC NVM. This delay can be changed by reprograming the NVM.

Note: The PMIC CHECK&LOAD duration is about 7 ms typ.

4. Once the CHECK&LOAD state ends and VIN voltage is above the PMIC V_{INOK_rise} (4 V typ.) threshold, the PMIC starts a power-up sequence. The PMIC regulators follow the power-up sequence predefined in PMIC NVM:
   a. The PMIC asserts the RSTn signal.
   b. RANK1 (1.5 ms): The LDO2 (V_{DDIO}) is enabled at 3.3 V. Once V_{DDIO} voltage is above MPU VBOR0 rising threshold (1.67 V typ.), an MPU $t_{RSTTEMPO}$ (377 µs typ.) delay is started. Once $t_{RSTTEMPO}$ elapses, the MPU PWR_ON signal goes high, and the MPU enters in system reset.
   c. RANK2 (1.5 ms): The BUCK1 (V_{DDCORE}) is enabled at 1.22 V. Once V_{DDCORE} voltage is above MPU $V_{TH\_VDDCORE}$ rising threshold (950 mV typ.), an MPU $t_{VDDCORE\_TEMPO}$ (340 µs typ.) delay is started to wait for V_{DDCORE} voltage to reach the minimum operating voltage. Once $t_{VDDCORE\_TEMPO}$ elapses, the MPU starts the HSI oscillators, then performs internal hardware initialization. The MPU then waits for NRST to release.
   d. RANK3 (1.5 ms): The GPO1 (EXT_EN1) is enabled. The 3V3 discrete SMPS regulator is enabled and the 3V3 voltage rises.
   e. RANK4 (1.5 ms): LDO5 (V_{DD_FLASH}) is enabled at 3.3 V.
   f. RANK5 (1.5 ms): LDO4 (V_{DD3V3_USB}) is enabled.
   g. Once RANK5 is ended, the PMIC releases the RSTn that releases MPU NRST.
5. Once NRST is released, the MPU enters Run mode:
   a. The CPU starts to execute the boot ROM: EADLY timer starts (refer to EADLY timer for more information).
   b. Once EADLY elapses, the boot ROM reads, verifies, and executes the bootloader from the external flash memory (for example: the SD card as illustrated in Figure 1).
6. The bootloader software performs initializations, then loads and executes the application software:
   a. Set PMIC BUCK1 (V_{DDCORE}) at 1.25 V (V_{DDCORE} and V_{DDCPU} nominal voltage).
   b. Enable DDR regulators: enable BUCK2 (V_{DD_DDR}) at 1.35 V. Then, set a 1.5 ms timer to wait for DDR voltages stabilization.
   c. Once the 1.5 ms timer elapses, the bootloader software initializes the DDR controller and DDR memory IC.
   d. The bootloader loads the application software into DDR3L and executes it.
   e. The system runs.
7. When a shutdown request occurs:
   a. The software prepares to power-off properly.
   b. The software shuts down the DDR3L: disable BUCK2 and LDO3 (recommended but not mandatory).
   c. The software sends the "SWOFF" command to PMIC to power-down.

8. The PMIC performs a power-down sequence:
   a. The PMIC asserts the RSTn, asserting the MPU NRST signal.
   b. RANK0 (1.5 ms): The BUCK2 (V_{DD_DDR}), and GPO2 are disabled.
   c. RANK5 (1.5 ms): The LDO4 (V_{DD3V3_USB}) is disabled.
   d. RANK4 (1.5 ms): The LDO5 (V_{DD_FLASH}) is disabled.
   e. RANK3 (1.5 ms): GPO1 (EXT_EN1) is disabled.
   f. RANK2 (1.5 ms): BUCK1 (V_{DDCORE}) is disabled.
   g. RANK1 (1.5 ms): LDO2 (V_{DDIO}) is disabled. Once V_{DDIO} voltage is below MPU VBOR0 falling threshold (1.63 V typ.), the MPU PWR_ON signal goes low, and the MPU enters in POR.
9. The PMIC is in OFF mode: the application is powered off.

Figure 2. Power-up and power-down sequence MPU with PMIC



5.2.2 Power-down triggered by PMIC hard fault (safety management)

When the PMIC detects a hard fault (see Section 6.2), it triggers a turn-off condition followed by a power-down sequence similar to step 8 in Figure 2. Once the power-down sequence ends, the PMIC can either restart (similar to step 4 in Figure 2) or go in FAIL_SAFE_LOCK state depending on safety management settings (see Section 6.1).

By default, the STPMIC1LA always restarts after a turn-off condition triggered by a hard fault.

5.2.3 Power-down triggered by main supply removal (VIN)

The application in Figure 1 is powered off by a power removal (VIN).

Once the VIN supply is below V_{INOK_fall}, the PMIC asserts an NRST and then enters power-down as shown in Figure 2 step 8.

Limitation: when the main power is removed, the VIN voltage drops very quickly to the V_{INOK_fall} value, in less than a few milliseconds (depending on system activity). Only then the power-down sequence starts. As soon as the PMIC asserts an NRST, system activity is immediately stopped and power consumption drops, slowing the VIN drop. Nevertheless, VIN may drop below the PMIC $V_{IN\_POR\_fall}$ threshold before the power-down sequence ends. In this case, the PMIC regulator pull-down discharge resistors are no longer controlled by the PMIC. The output regulator decoupling capacitors discharge in an uncontrolled way. A bulk decoupling capacitor (a few hundred µF) may be inserted on the VIN path to limit VIN drop speed.

Note: If the application main supply is immediately inserted after VIN drops below PMIC $V_{IN\_POR\_fall}$ threshold, the PMIC goes in INIT&LOAD state then in CHECK&LOAD state (see Figure 2 step 3) where the regulator pull-down discharge resistors are enabled. Thanks to this, output regulator decoupling capacitors are properly discharged before going into the power-up step. (See Figure 2 step 4.)

5.3 Low-power mode management

The MPU supports several low-power modes to reduce power consumption. These are described in Section 5.1: Operating modes. The modes supported by the application and their advantages/disadvantages are presented in the table below:

Table 8. Low-power mode supported by the application

Power mode	Advantages	Disadvantages
Stop	Very fast recovery from Stop to Run mode	Low-power consumption gain
LPLV-Stop	V_{DDCORE} voltage is lowered	Few EXTI wake-up sources are available to exit this mode To exit this mode, time is necessary to restore the lowered supply to its nominal values
Standby (DDR in self-refresh)	Very-low power consumption All MPU power domains are powered off except V_{DDIO} DDR is maintained in self-refresh (suspend to RAM)	Few EXTI wake-up sources are available to exit this mode Longer exit recovery duration than LPLV-Stop
Standby (DDR OFF)	Lowest power consumption All MPU power domains are powered off except V_{DDIO} DDR is powered off (suspend to FLASH)	Few EXTI wake-up sources are available to exit this mode Longer exit recovery duration than Standby (DDR in self-refresh)

The MPU manages the low-power modes. As described in PWRCTRL1, PWRCTRL2, the power control signals are connected as defined in the table below:

Table 9. Power control signals

MPU output	PMIC input
PWR_ON	PWRCTRL1

The PWR_ON signal is controlled from the MPU state machine. That is because in low-power mode, no software is running and the PMIC regulators cannot be controlled by any I²C command from software (see Table 5. Operating modes).

Before entering in low-power mode, the MPU software must prepare the PMIC to enter any of these power modes by setting:

- PMIC xxxx_MAIN_CR registers: settings for run mode behavior
- PMIC xxxx_ALT_CR registers: settings for the targeted low-power mode behavior

Note: xxxx corresponds to the targeted regulator or GPO.

The MPU software must also set some internal delays used in low-power modes:

- LPLV-Stop
- Standby mode

The delays are described in the following section.

5.3.1 MPU internal timer for low-power mode management

EADLY timer

The EADLY timer is a programmable timer used to wait for the external flash memory to be ready before the boot ROM performs read access to it (SD card, eMMC, FMC-NAND, OCTOSPI). This ensures that the boot ROM can reliably read the boot software from the boot flash memory. The EADLY timer duration is set to 5 ms by default after a system reset. It is recommended to keep this default value.

POPL timer

POPL timer is a programmable timer used to force the MPU into standby mode for a minimum duration. When entering in standby mode, the PWR_ON signal goes low for minimum POPL duration forcing V_{DDCORE} power supply voltages to drop before standby mode exit. This is to guaranty MPU core, CPU, and flash memory domains to restart properly if a wake-up event occurs just after MPU enters standby mode.

The software sets the POPL timer prior to Standby mode entry.

Recommended values: PWR_CR3[POPL] = 3 ms minimum for Section 5.3.3 and Section 5.3.4.

Note: POPL timer may be set to a higher value if some application peripherals require their power supplies to be fully discharged to restart properly. In that case, the POPL duration should be set according to the duration required to drop the peripheral power supply voltages. The latter are supplied by discrete regulators such as the 3.3 V supply illustrated in Figure 1.

PWRLP_TEMPO timer

The PWRLP_TEMPO is a delay between the time when the system exits an LPLV-Stop low-power modes and the moment where it is allowed to enable the PLLs. It is then able to provide a clock to the CPU and enters Run mode. This delay is linked to the core domain (V_{DDCORE}) and should be set in the RCC_PWRLPDLYCR [PWRLP_DLY [21:0]] register bitfield prior to entering low-power mode.

The PWRLP_DLY timer should be set by software before entering low-power mode. Recommended values are proposed Section 5.3.2 depending on the targeted low-power mode.

MRD timer

The MRD is a programmable timer defining the minimum pulse duration of the NRST system reset signal. This timer is useful when the supply is provided by a discrete power component, so it can be set to 0 with a PMIC.

$t_{VDDCORE\_TEMPO}$ delay

The $t_{VDDCORE\_TEMPO}$ is an internal and fixed delay started to wait for V_{DDCORE} to reach the Run mode operating supply level when the V_{DDCORE} is enabled and reaches the threshold $V_{TH\_VDDCORE}$. As long as V_{DDCORE} is below $V_{TH\_VDDCORE}$, the core domain is in reset. This delay is used at power-up or when the system exits from Standby low-power mode.

$t_{SEL\_VDDCORETEMPO}$ delay

The $t_{SEL\_VDDCORETEMPO}$ is an internal and fixed delay started to wait for V_{DDCORE} to reach the Run mode operating supply level when the system exits the LPLV-Stop mode.

5.3.2 LPLV-Stop mode

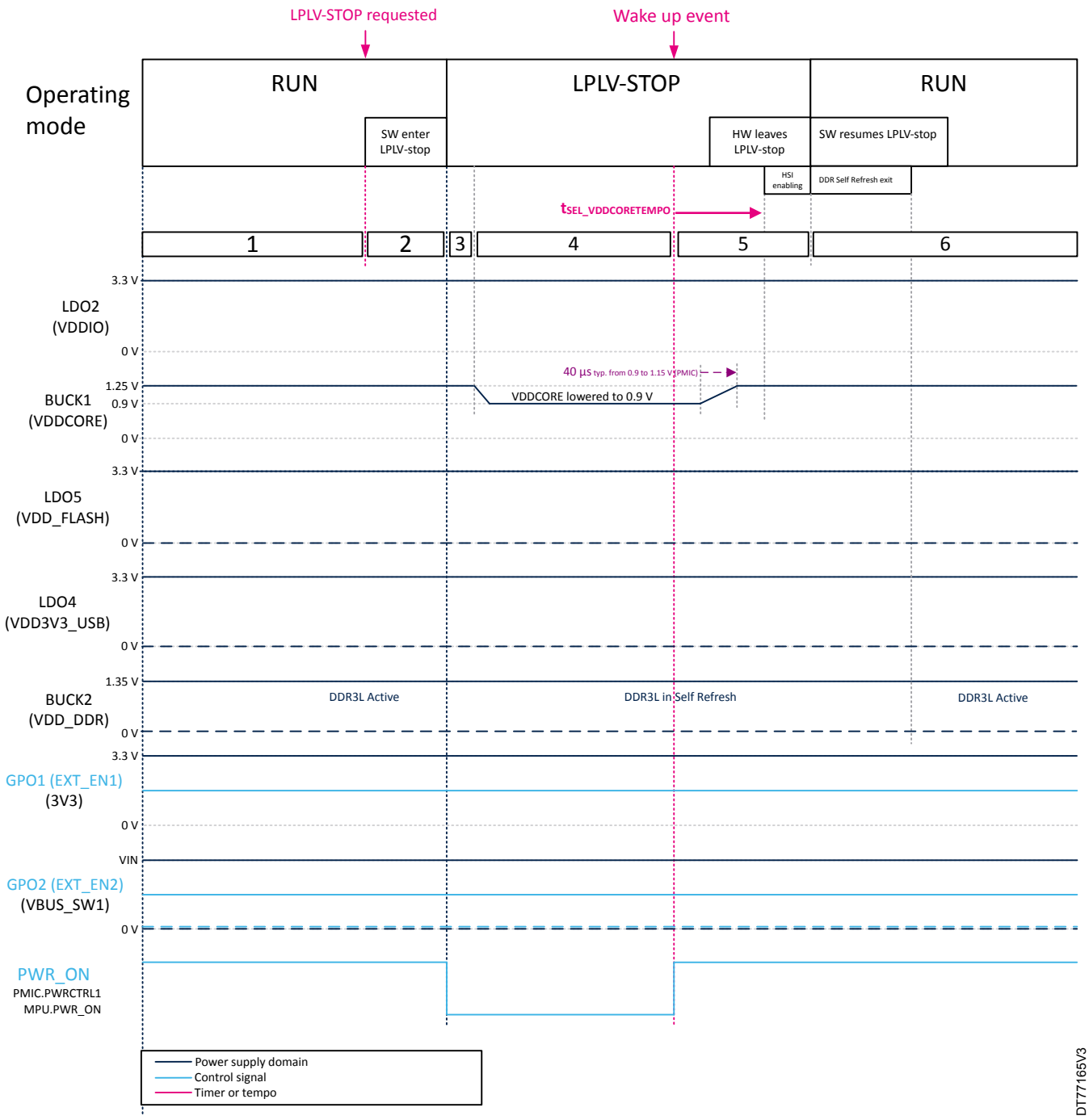
The LPLV-Stop1 mode is described below and is shown in Figure 3 based on the implementation shown in Figure 1.

1. The application is powered up and operates in Run mode. PWR_ON is in high state. In this application, the PWR_LP signal is multiplexed on the PWR_ON pin (PWR_CR1[LPCFG] bit is enabled).
2. When the LPLV-Stop mode is requested, the software prepares to enter LPLV-Stop mode:
   a. The MPU performs internal settings such as:
      i. Disabling PWRLP_TEMPO (set in RCC_PWRLPDLYCR [PWRLP_DLY[21:0]] = 0).
      ii. Stopping the appropriate clocks.
      iii. Setting the DDR3L to self-refresh.
   b. The MPU performs the PMIC settings (see Table 10).
   c. The MPU PWR_CR1[LPDS] and PWR_CR1[LVDS] bit are enabled. The regulator enters into LPLV-Stop low-power mode and V_{DDCORE} is then lowered to LPLV-Stop mode value.
3. Once the system is in LPLV-Stop:
   a. The MPU PWR_ON output is deasserted (PMIC PWRCTRL1 signal goes low).
4. The PMIC regulators, which are affected to the PWR_ON, take the configuration set in the xxxx_ALT_CR registers (see Table 10) after ~ 20 µs (internal PMIC delay):
   a. The V_{DDCORE} regulator output decreases to retention voltage (from 1.25 V to 0.9 V).
5. On a wake-up event, the MPU leaves the LPLV-Stop:
   a. The MPU PWR_ON output signal is asserted (the PMIC PWRCTRL1 signal goes high), and the MPU starts the $t_{SEL\_VDDCORETEMPO}$ internal delay (234 µs) to wait for the V_{DDCORE} to switch from retention voltage (0.9 V) to minimum operating voltage (1.15 V).
   b. The PMIC regulators which are affected to the PWR_ON take the configuration set in the xxxx_MAIN_CR registers (see Table 10) after ~ 20 µs (internal PMIC delay):
      i. The V_{DDCORE} regulator switches from retention voltage (0.9 V) to minimal nominal operating voltage (1.15 V) in 40 µs typical and converges to nominal operating voltage (1.25 V). Accordingly, V_{DDCORE} voltage is ready before the $t_{SEL\_VDDCORETEMPO}$ MPU delay elapses.
   c. Once the $t_{SEL\_VDDCORETEMPO}$ internal delay elapses, clocks are enabled.
6. Once the clocks are stable, the MPU goes immediately in Run mode (as the PWRLP_TEMPO is bypassed) and the software resumes LPLV-Stop: DDR3L exits from self-refresh.

Table 10. PMIC configuration for LPLV-Stop mode

Regulator	PWRCTRLx affectation	Register xxxx_MAIN_CR		Register xxxx_ALT_CR	
		Configuration	VOUT	Configuration	VOUT
LDO2 (V_{DDIO})	-	ON	3.3	-	-
BUCK1 (V_{DDCORE})	PWR_ON	ON	1.25	ON	0.9
LDO5 (V_{DD_FLASH})	PWR_ON	ON/OFF	3.3	ON/OFF	3.3
LDO4 (V_{DD3V3_USB})	-	ON/OFF	N/A	-	N/A
BUCK2 (V_{DD_DDR})	PWR_ON	ON	1.35	ON	1.35
LDO3 (free)	-	-	-	-	-
GPO1 (3V3)	PWR_ON	ON	N/A	ON	N/A
GPO2 (V_{BUS_SW1})	-	ON/OFF	N/A	-	N/A

Figure 3. **LPLV-Stop sequence**



5.3.3 Standby mode (DDR3L in self-refresh)

The Standby mode is used when a very-low power consumption is required. Most of the PMIC regulators are switched off. The content of MPU registers and memories are lost except for the backup domains (V_{DDIO} is kept enabled). The DDR3L is set in self-refresh (V_{DD_DDR} is kept enabled) to maintain the system in "suspend to RAM."

The Standby mode with DDR3L in self-refresh is described in the Figure 4 based on the implementation shown in Figure 1.

1. The application is powered up and operates in Run mode. PWR_ON is in a high state.

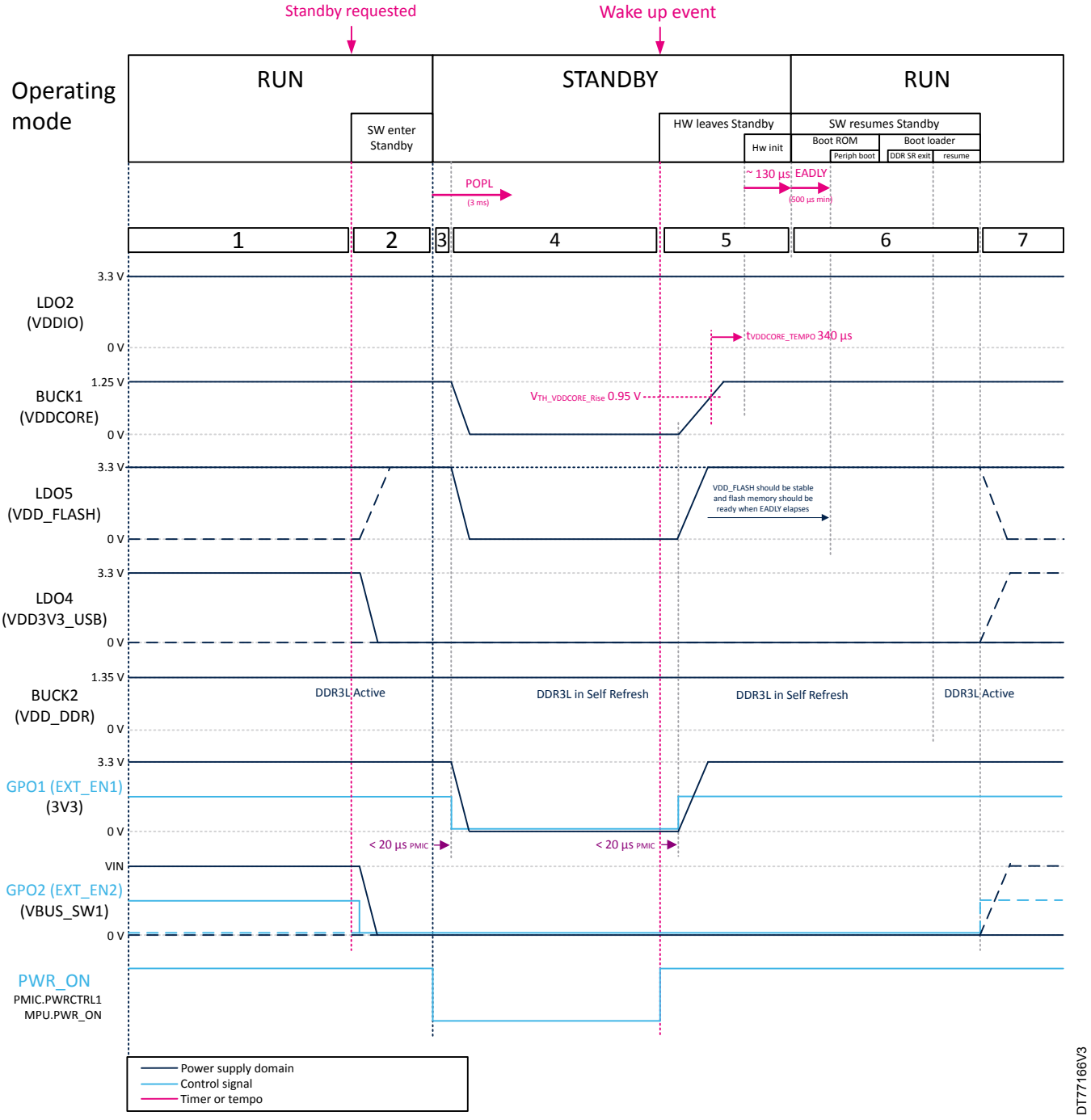
2. When the Standby mode is requested, the software prepares to enter Standby mode:
   a. The MPU performs internal settings such as:
      i. Setting the POPL timer (see POPL timer), to define a minimum pulse duration of PWR_ON ensuring V_{DDCORE} voltage full discharge before restarting.
      ii. Set the EADLY timer (see EADLY timer).
      iii. Stopping some clocks.
      iv. Setting DDR3L to self-refresh.
   b. The MPU configures the PMIC as described in Table 11.
      i. If turned off, the V_{DD_FLASH} regulator turns ON.
      ii. If turned on, the V_{DD3V3_USB} regulator turns OFF.
      iii. If turned on, the V_{BUS_SW1} power switch (controlled by PMIC GPO2) turns OFF.
   c. The PWR_MPUCR[PDDS] bit is enabled (Standby mode is allowed).
3. Once the system is in Standby mode:
   a. The MPU PWR_ON signal is deasserted (PMIC PWRCTRL1 signal goes low).
   b. The POPL timer is started to keep the application in Standby for a minimum POPL timer duration.
4. The PMIC regulators affected to PWR_ON take the configuration set in the xxxx_ALT_CR registers (see Table 11) after ~ 20 µs (internal PMIC delay):
   a. The V_{DDCORE} regulator turns OFF.
   b. The V_{DD_FLASH} regulator turns OFF.
   c. The 3V3 regulator (controlled by PMIC GPO1) turns OFF.
5. On a wake-up event, the MPU leaves Standby mode (MPU waits for the POPL timer to elapse before leaving Standby mode):
   a. The MPU PWR_ON signal is asserted (PMIC PWRCTRL1 signal goes high).
   b. The PMIC regulators affected to the PWR_ON take the configuration set in the xxxx_MAIN_CR registers (see Table 11) after ~ 20 µs (internal PMIC delay):
      i. The V_{DDCORE} regulator turns ON.
      ii. The V_{DD_FLASH} regulator turns ON.
      iii. The 3V3 regulator (controlled by PMIC GPO1) turns ON.
   c. Once the V_{DDCORE} voltage reaches the $V_{TH\_VDDCORE}$ threshold, an MPU $t_{VDDCORE\_TEMPO}$ (340 µs typ.) delay is started to wait for V_{DDCORE} voltage to reach the minimum operating voltage. Once $t_{VDDCORE\_TEMPO}$ elapses, the MPU performs internal hardware initialization.
6. Once the MPU ends internal hardware initialization, the MPU enters Run mode:
   a. The CPU starts to execute the boot ROM: EADLY timer is started (refer to EADLY timer for more information).
   b. Once EADLY elapses, the boot ROM reads, verifies, and executes the bootloader from the external flash memory (for example: the SD card as illustrated in Figure 1).
   c. The software detects an "exit from Standby mode": it exits DDR from self-refresh and resumes the application software.
7. The system runs the application software:
   a. Peripherals switched OFF before entering Standby mode (such as USB) can be resumed.

Table 11. PMIC configuration for Standby DDR in self-refresh

Regulator	PWRCTRLx affectation	Register xxxx_MAIN_CR		Register xxxx_ALT_CR	
		Configuration	VOUT	Configuration	VOUT
LDO2 (V_{DDIO})	-	ON	3.3	-	-
BUCK1 (V_{DDCORE})	PWR_ON	ON	1.25	OFF	-
LDO5 (V_{DD_FLASH})	PWR_ON	ON	3.3	OFF	-
LDO4 (V_{DD3V3_USB})	-	OFF	N/A	-	N/A
BUCK2 (V_{DD_DDR})	PWR_ON	ON	1.35	ON	1.35

Regulator	PWRCTRLx affectation	Register xxxx_MAIN_CR		Register xxxx_ALT_CR	
		Configuration	VOUT	Configuration	VOUT
LDO3 (free)	-	-	-	-	-
GPO1 (3V3)	PWR_ON	ON	N/A	OFF	N/A
GPO2 (V_{BUS_SW1})	-	OFF	N/A	-	N/A

Figure 4. Standby (DDR in self-refresh) sequence



5.3.4 Standby mode (DDR3L OFF)

The Standby mode is used when a very-low power consumption is required. Most of the PMIC regulators are switched off. The content of MPU registers and memories are lost except for the backup domain (V_{DDIO} is kept enabled). The DDR3L is powered off (V_{DD_DDR} is disabled), so the system is in "suspend to flash."

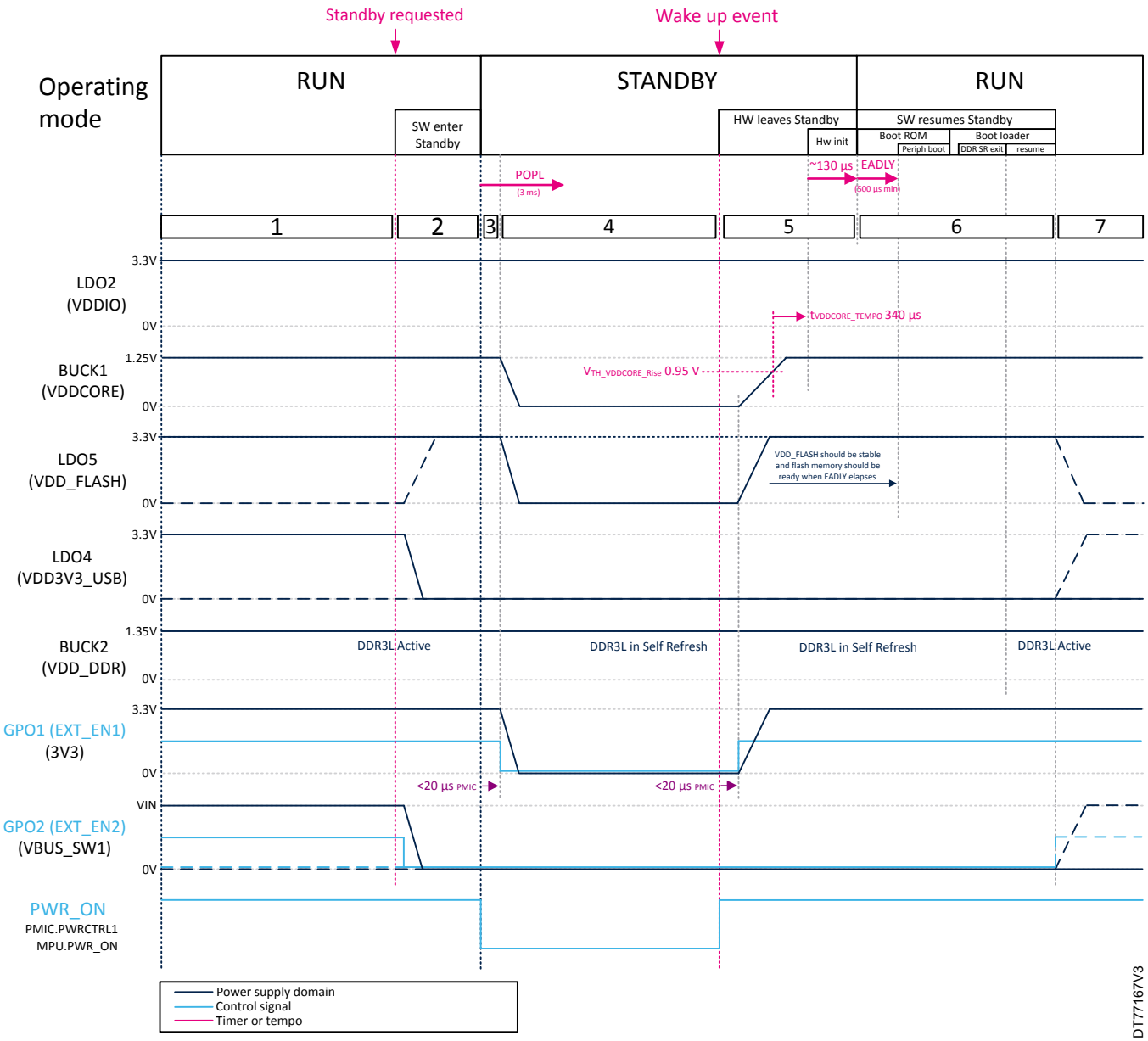
This section focuses on Standby mode with DDR3L OFF. This mode is described below and is shown in Figure 5 based on the implementation shown in Figure 1.

1. The application is powered up and operates in Run mode. PWR_ON is in high state.
2. When the Standby mode is requested, the software prepares to enter Standby mode:
   a. The MPU performs internal settings such as:
      i. Setting the POPL timer (see POPL timer), to define a minimum pulse duration of PWR_ON ensuring V_{DDCORE} voltage full discharge before restarting.
      ii. Setting the EADLY timer (see EADLY timer).
      iii. Stopping some clocks.
      iv. Disabling DDR controller.
   b. The MPU performs the STPMIC1L as described in Table 12.
      i. If turned off, the V_{DD_FLASH} regulator turns ON.
      ii. If turned on, the V_{DD3V3_USB} regulator turns OFF.
      iii. If turned on, the V_{BUS_SW1} power switch (controlled by PMIC GPO2) turns OFF.
   c. The PWR_MPUCR[PDDS] bit is enabled (Standby mode is allowed).
3. Once the system is in Standby mode:
   a. The MPU PWR_ON signal is deasserted (PMIC PWRCTRL1 signal goes low).
   b. The POPL timer is started to keep the application in Standby for a minimum POPL timer duration.
4. The PMIC regulators affected to PWR_ON take the configuration set in the xxxx_ALT_CR registers (see Table 12) after ~ 20 µs (internal PMIC delay):
   a. The V_{DDCORE} regulator turns OFF.
   b. The V_{DD_FLASH} regulator turns OFF.
   c. The V_{DD_DDR} regulator turns OFF.
   d. The 3V3 regulator (controlled by PMIC GPO1) turns OFF.
5. On a wake-up event, the MPU leaves the Standby mode (MPU waits for the POPL timer to elapse before leaving Standby mode):
   a. The MPU PWR_ON signal is asserted (PMIC PWRCTRL1 signal goes high).
   b. The PMIC regulators which are affected to the PWR_ON, take the configuration set in the xxxx_MAIN_CR registers (see Table 12) after ~ 20 µs (internal PMIC delay):
      i. The V_{DDCORE} regulator turns ON.
      ii. The V_{DD_FLASH} regulator turns ON.
      iii. The V_{DD_DDR} regulator turns ON.
      iv. The 3V3 regulator (controlled by PMIC GPO1) turns ON.
   c. Once V_{DDCORE} voltage reaches the $V_{TH\_VDDCORE}$ threshold, an MPU $t_{VDDCORE\_TEMPO}$ (340 µs typ.) delay is started to wait for V_{DDCORE} voltage to reach the minimum operating voltage. Once $t_{VDDCORE\_TEMPO}$ elapses, the MPU performs internal hardware initialization.
6. Once the MPU ends internal hardware initialization, the MPU enters Run mode:
   a. The CPU starts to execute the boot ROM: EADLY timer is started (refer to EADLY timer for more information).
   b. Once EADLY elapses, the boot ROM reads, verifies, and executes the bootloader from the external flash memory (for example: the SD card as illustrated in Figure 1).
   c. The bootloader detects an "exit from Standby mode":
      i. It initializes DDR controller and DDR3L memory.
      ii. It resumes the application software from flash memory to DDR3L.
7. The system runs the application software:
   a. Peripherals switched OFF before entering Standby mode (such as USB) can be resumed.

Table 12. **PMIC configuration Stanby DDR OFF**

Regulator	PWRCTRLx affectation	Register xxxx_MAIN_CR		Register xxxx_ALT_CR	
		Configuration	VOUT	Configuration	VOUT
LDO2 (V_{DDIO})	-	ON	3.3	-	-
BUCK1 (V_{DDCORE})	PWR_ON	ON	1.25	OFF	-
LDO5 (V_{DD_FLASH})	PWR_ON	ON	3.3	OFF	-
LDO4 (V_{DD3V3_USB})	-	OFF	N/A	-	N/A
BUCK2 (V_{DD_DDR})	PWR_ON	ON	1.35	OFF	-
LDO3 (free)	-	-	-	-	-
GPO1 (3V3)	PWR_ON	ON	N/A	OFF	N/A
GPO2 (V_{BUS_SW1})	-	OFF	N/A	-	N/A

Figure 5. **Standby (DDR OFF) sequence**



5.4 Crash recovery/user reset/hard fault management

An MPU crash, a user reset, or a PMIC hard fault are managed similarly by a power cycling sequence handled by PMIC.

An MPU crash (iwdg1_out_rst, iwdg2_out_rst) or MPU system reset or a user reset (push button) generates a reset pulse on the NRST signal to PMIC. As introduced in RSTn pin, the MPU, and the PMIC both have interconnected bidirectional reset pins (see Figure 1 signal NRST). The reset pulse triggers the PMIC to produce an immediate power cycle sequence. This power cycling ensures a correct reset and restart of the peripherals following a global application reset (NRST).

A default STPMIC1LA hard fault (see Table 7 and Section 5.2.2) is managed by a power cycling; like a system reset. This behavior programmed into the STPMIC1LA NVM may be adjusted depending on expected safety management behavior (see Section 6.1).

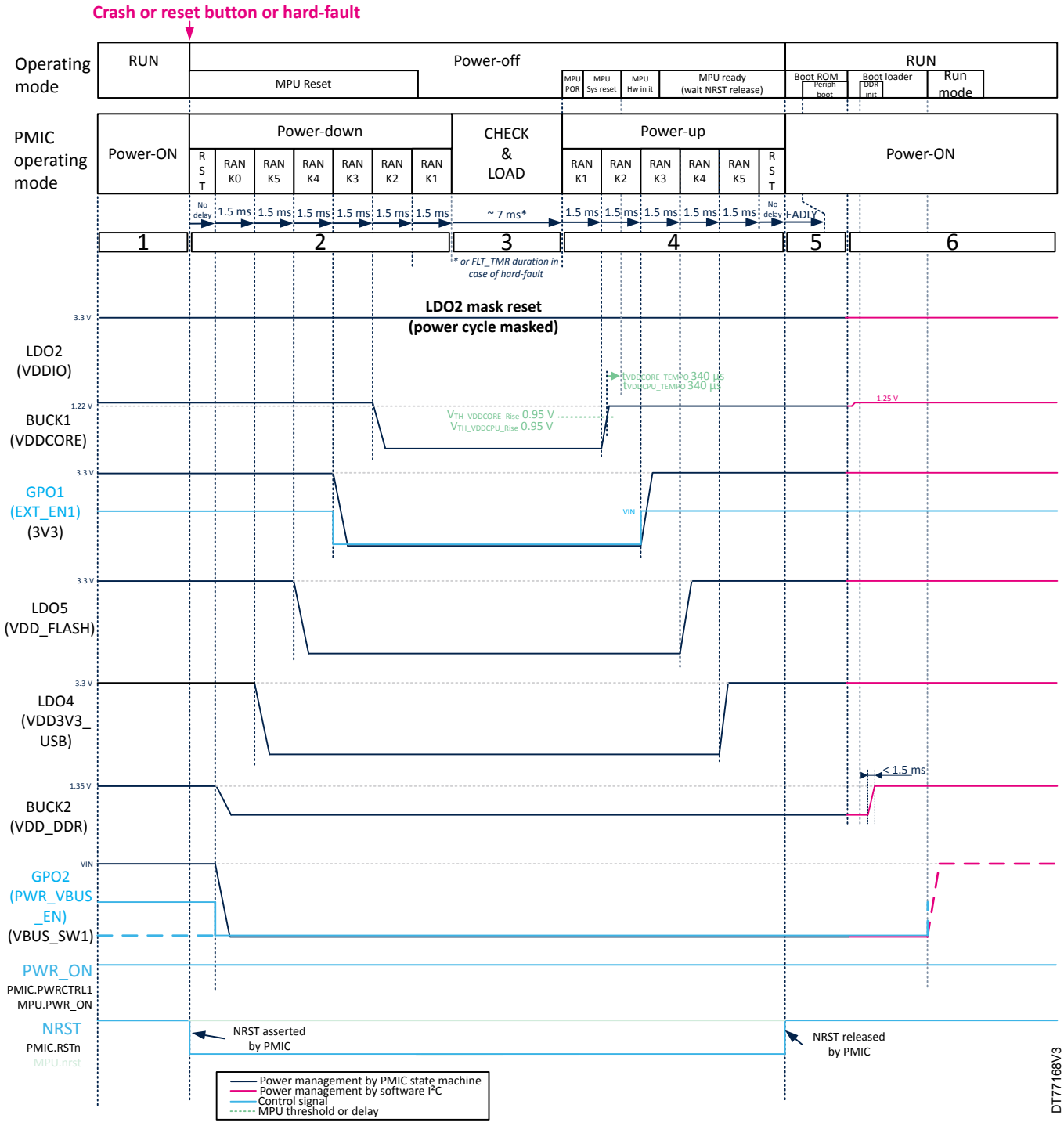
The sequence in Figure 6 illustrates an MPU crash recovery or user push-button reset or PMIC hard fault behavior according to the application shown in Figure 1. In this application, the mask-reset (see Section 5.1.3) is enable for the V_{DDIO} (PMIC LDO2) allowing to keep the LDO2 enabled during the power cycling sequence. Implicitly, the mask-reset on LDO2 has been set by software prior to the sequence; typically by the bootloader during application initialization.

1. The application is powered up and operates in Run mode. An MPU crash occurs or a user presses the reset button generating a pulse on the NRST signal, or the PMIC detects a hard fault.
2. Once the NRST signal is asserted or a hard fault is detected by the PMIC, the PMIC performs a power-down sequence:
   a. The PMIC asserts the RSTn, asserting the MPU NRST signal.
   b. RANK0 (1.5 ms): the BUCK2 (V_{DD_DDR}), and GPO2 are disabled.
   c. RANK5 (1.5 ms): the LDO4 (V_{DD3V3_USB}) is disabled.
   d. RANK4 (1.5 ms): the LDO5 (V_{DD_FLASH}) is disabled.
   e. RANK3 (1.5 ms): GPO1 (EXT_EN1) is disabled. The 3V3 discrete SMPS regulator is disabled.
   f. RANK2 (1.5 ms): BUCK1 (V_{DDCORE}) is disabled.
   g. RANK1 (1.5 ms): the LDO2 (V_{DDIO}) is kept enabled as mask-reset set on LDO2.
3. Once the power-down sequence ends, the PMIC goes in CHECK&LOAD state (see [1] for more details) to prepare for the power-up sequence.

Note: STPMIC1LA always restarts after a hard fault (default behavior).

4. Once the CHECK&LOAD ends (after ~ 7 ms for an NRST assertion or after FLT_TMR duration for a hard fault (see [1]) PMIC performs a power-up sequence:
   a. RANK1 (1.5 ms): the LDO2 (V_{DDIO}) is already enabled (mask-reset).
   b. RANK2 (1.5 ms): the BUCK1 (V_{DDCORE}) is enabled at 1.22 V. Once V_{DDCORE} voltage is above MPU $V_{TH\_VDDCORE}$ rising threshold (950 mV typ.), an MPU $t_{VDDCORE\_TEMPO}$ (340 µs typ.) delay is started. This is done to wait for V_{DDCORE} voltage to reach the minimum operating voltage. Once $t_{VDDCORE\_TEMPO}$ elapses, the MPU starts and the HSI oscillators then perform internal hardware initialization. The MPU then waits for NRST to release.
   c. RANK3 (1.5 ms): the GPO1 (EXT_EN1) is enabled. The 3V3 discrete SMPS regulator is enabled and the 3V3 voltage rises.
   d. RANK4 (1.5 ms): LDO5 (V_{DD_FLASH}) is enabled at 3.3 V.
   e. RANK5 (1.5 ms): LDO4 (V_{DD3V3_USB}) is enabled.
   f. Once RANK5 ends, the PMIC releases the RSTn that releases MPU NRST.
5. Once NRST is released, the MPU enters Run mode:
   a. The CPU starts to execute the boot ROM: EADLY timer is started (refer to EADLY timer for more information).
   b. Once EADLY elapses, the boot ROM reads, verifies, and executes the bootloader from the external flash memory (for example: the SD card as illustrated in Figure 1).
6. The bootloader software performs initializations then loads and executes the application software:
   a. Set PMIC BUCK1 (V_{DDCORE}) at 1.25 V (V_{DDCORE} and V_{DDCPU} nominal voltage).
   b. Enable DDR regulators: enable BUCK2 (V_{DD_DDR}) at 1.35 V then set a 1.5 ms timer to wait for DDR voltages stabilization.
   c. Once the 1.5 ms timer elapses, the bootloader software initializes the DDR controller and DDR memory IC.
   d. The bootloader loads the application software into DDR3L and executes it.
   e. The system runs.

Figure 6. Crash recovery sequence



6 Safety management

In this documentation, *safety management* is the concept of implementing mechanisms such as OCP (Over Current Protection), or watchdogs to maintain the system functional and robust. The objective is to protect the safety and integrity of the application against internal or external errors, or dysfunctions.

The safety management is provided by the MPU software and/or by the PMIC functionalities.

This section focuses on PMIC safety management functionalities. It is based on failure detection (hard fault) and related PMIC behavior (fail-safe management).

See [1] for more details.

6.1 PMIC fail-safe management

Each source of hard fault (see Section 6.2) has a dedicated independent fail-safe counter. This counter, named xxx_FLT_CNT (where xxx is the hard fault source), is incremented each time a hard fault event occurs, in addition to a turn-off condition.

The counter maximum fault iteration, xxx_FLT_CNT_MAX set in PMIC NVM, is used to define the maximum number of the hard fault iterations before the PMIC enters in FAIL_SAFE_LOCK state. By default, the STPMIC1LA maximal counter for each hard fault is infinite. It means that STPMIC1LA always restarts after a hard fault event occurs and never enters in FAIL_SAFE_LOCK state.

As long as the fail-safe counter xxx_FLT_CNT is below xxx_FLT_CNT_MAX, the PMIC carries out a power cycle each time a hard fault event occurs. A power cycle is defined by a power-down sequence, then waits for FLT_TMR (fault timer) before restarting, and then by a power-up sequence (see Section 5.4).

Once the number of hard fault iterations exceeds the xxx_FLT_CNT_MAX counter, the system is blocked in FAIL_SAFE_LOCK state. To exit this state, the PMIC must carry out a main supply removal or a PONKEYn long press (a special NVM setting is necessary).

The RST_FLT_CNT_TMR reset fault timer may be enabled by NVM to clear automatically all xxx_FLT_CNT fail-safe counters if no hard fault has occurred until RST_FLT_CNT_TMR elapses.

The following examples illustrate fail-safe management mechanisms.

Example 1 (STPMIC1LA behavior with a negative voltage glitch on VIN)

Initial condition: the STPMIC1LA NVM has the default value:

The VIN_FLT_CNT_MAX counter is configurated (NVM fail-safe shadow register NVM_FS_SHR1) at 1111 (infinite hard fault allowed).

Description:

A 5 V wall adaptor is plugged to the application main supply connector. The VIN rises above V_{INOK_rise} (4 V), the PMIC then powers up the application and the application initializes. A negative voltage glitch occurs on VIN (for example: due to bad contact at main supply connector) and the glitch voltage goes below V_{INOK_fall} (3.5 V). The PMIC triggers a VIN hard fault condition that triggers a power-off sequence and the dedicated fail-safe counter (VIN_FLT_CNT) is incremented. Once the power-off sequence ends, the PMIC evaluates the state transition and goes to the power-up sequence as VIN_FLT_CNT ≤ VIN_FLT_CNT_MAX. Once the power-up sequence ends, PMIC goes in power-on state and the application initializes and runs.

If several other negative glitches occur on the main supply input (VIN) below V_{INOK_fall}, the STPMIC1LA PMIC always restarts as VIN_FLT_CNT ≤ VIN_FLT_CNT_MAX is always true.

This behavior is identical for other hard fault sources: the STPMIC1LA always restarts.

Example 2 (PMIC behavior with negative voltage glitch on VIN and tuned fail-safe management in PMIC NVM)

Initial condition: The STPMIC1LA NVM has been tuned to adjust fail-safe management as follows:

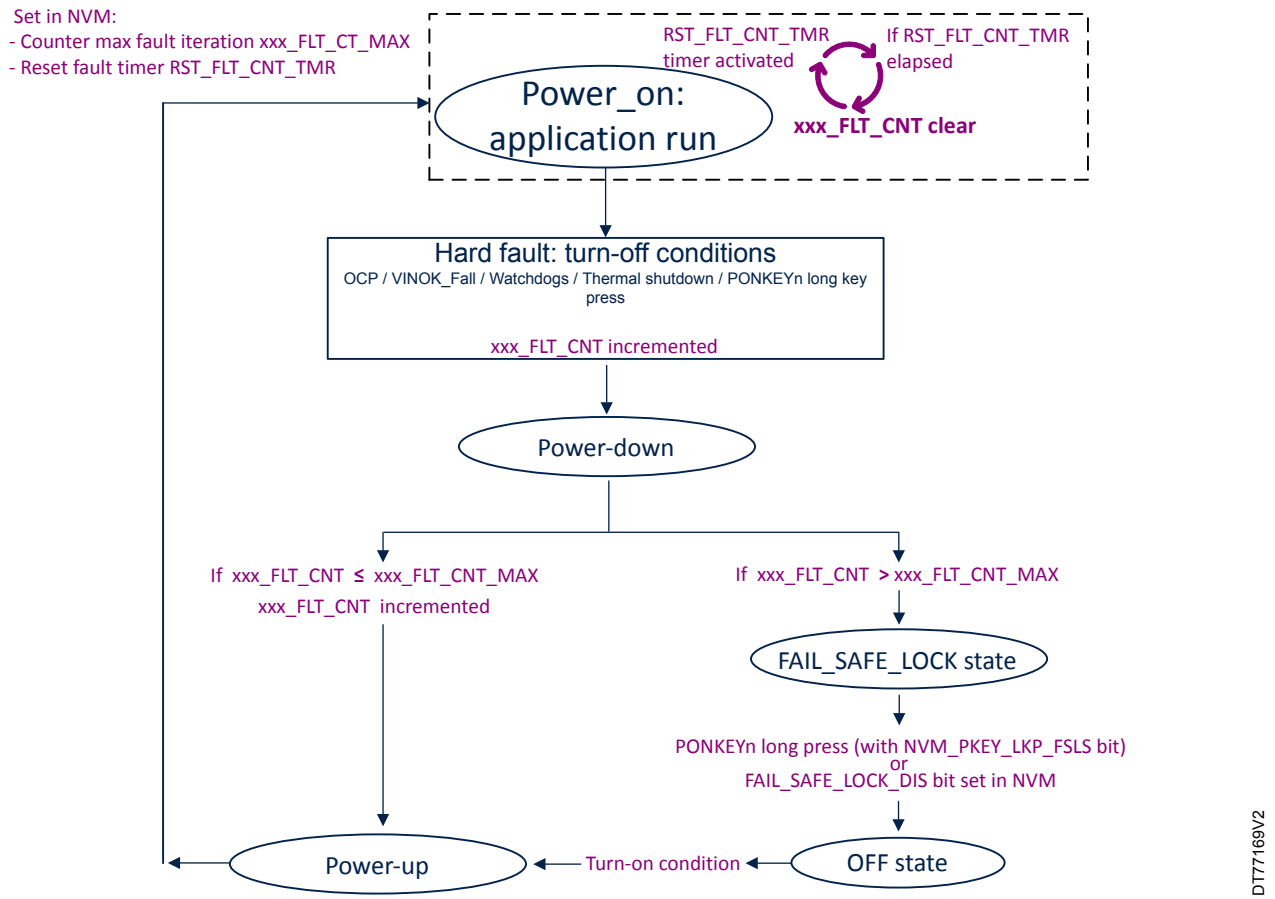
- VIN_FLT_CNT_MAX[3:0] = 0x0001 programmed in PMIC NVM_FS_SHR1 NVM: one hard fault on VIN is allowed.
- RST_FLT_CNT_TMR[1:0] = 0x10 programmed in PMIC NVM_FS_SHR2 NVM: all fault counters cleared if no hard fault is detected for six min.

Description:

A 5V wall adaptor is plugged to the application main supply connector. The VIN rises above V_{INOK_rise} (4 V), the PMIC then powers up the application and the application initializes. A negative voltage glitch occurs on VIN (for example: due to bad contact at main supply connector) and the glitch voltage goes below V_{INOK_fall} (3.5 V). The PMIC triggers a VIN hard fault condition that triggers a power off sequence and the dedicated fail-safe counter is incremented (VIN_FLT_CNT = 1). Once the power-off sequence ends, the PMIC evaluates the state transition and goes to the power-up sequence as VIN_FLT_CNT ≤ VIN_FLT_CNT_MAX condition is true. Once the power-up sequence ends, PMIC goes in power-on state and the application initializes and runs.

- If another negative voltage glitch on VIN occurs before the RST_FLT_CNT_TMR elapses (6 min): the PMIC triggers a VIN hard fault condition that triggers a power-off sequence. The dedicated fail-safe counter is incremented (VIN_FLT_CNT = 2). Once the power-off sequence ends, the PMIC evaluates state transition (VIN_FLT_CNT ≤ VIN_FLT_CNT_MAX condition is wrong), then the PMIC goes in FAIL_SAFE_LOCK_STATE and is locked in this state until next PMIC POR (main supply removal).
- If another negative voltage glitch on VIN occurs after the RST_FLT_CNT_TMR elapses (6 min): Once the RST_FLT_CNT_TMR elapses, all xxx_FLT_CNT are clear including the VIN_FLT_CNT. Then, if a new negative glitch on VIN occurs, the PMIC enters power-off and then power-up as VIN_FLT_CNT ≤ VIN_FLT_CNT_MAX condition is true.

Figure 7. PMIC fail-safe management mechanism



6.2 PMIC hard faults

PMIC has five hardware source of events considered as hard fault conditions:

- OCP: overcurrent protection, including short circuit
- VIN: undervoltage protection (VIN fall below V_{INOK_fall} threshold)
- TSHDN: thermal shutdown protection
- WDG: watchdog timer expiration
- PKEY: power-on key button long press

Each source of hard fault is managed in the same way:

- It triggers a turn-off condition (see Turn-off conditions) followed by a PMIC power-down sequence.
- Then, depending on fail-safe management settings, the PMIC can:
  - Restart automatically (power-up sequence). This is the default behavior of the STPMIC1LA.
  - Not restart automatically: PMIC is kept in FAIL_SAFE_LOCK or OFF state until an allowed turn-on condition is met.

Note: Implicitly, a hard fault can occur only when the PMIC is in a power-on state.

6.2.1 OCP overcurrent protection

All PMIC regulators implement two levels of protection against overcurrent or short circuit on their output.

Hiccup (level 0)

In case of overcurrent or short-circuit, each PMIC regulator (set in level 0) operates independently in Hiccup mode without impacting another power domain of the application. This level of protection is suitable for the non-critical power domain.

Once an OCP occurs, the regulator turns off for t_{HICCUP_DLY} timer (predefined in NVM), then restarts.

Accordingly, the regulator restarts infinitely until the overcurrent/short-circuit disappears.

OCP hard fault management (level 1)

In case of overcurrent or short-circuit, a PMIC regulator (set in level 1) triggers an OCP hard fault turn-off condition. Implicitly, the PMIC enters power-off, and then power-on (depending on fail-safe settings). This level of protection is suitable for critical power domain (such as MPU V_{DDCORE}, DDR3L regulators) where it is mandatory to restart the application completely in case of overcurrent or short-circuit.

The Section 6.3 illustrates PMIC OCP settings according to the application in Figure 1.

6.2.2 VIN undervoltage protection (VIN < V_{INOK_Fall})

The PMIC embeds an undervoltage protection to prevent MPU or peripherals from crashing in case PMIC regulators go out of regulation due to too low VIN input voltage.

Once the main VIN supply goes below V_{INOK_Fall} thresholds, even for a very short duration (such as a voltage glitch), a hard fault condition is generated: the VIN fail-safe counter (VIN_FLT_CNT) is incremented and the PMIC powers down.

If the VIN_FLT_CNT counter is higher than the VIN_FLT_CNT_MAX when power-down ends, the PMIC enters in FAIL_SAFE_LOCK state. Otherwise, the PMIC waits for FLT_TMR fault timer (t_{VINOK_Fall} = 100 ms), then the PMIC enters power-up and then goes to power-on. The application then initializes and runs.

6.2.3 TSHDN: thermal shutdown protection

The PMIC embeds a thermal protection to avoid overheating damage.

Two levels of thermal protection are available:

- First level, an interruption is sent to the MPU:
  Once the PMIC junction temperature goes higher than or below temperature thresholds (respectively T_{WRN_Rise} or T_{WRN_Fall}), the PMIC generates an interrupt to be caught and managed by the MPU.
- Second level, hard fault condition is generated:
  Once the PMIC junction temperature goes higher than T_{SHDN_Rise} temperature thresholds, a hard fault condition is generated: the thermal fail-safe counter (TSHDN_FLT_CNT) is incremented and the PMIC enters power-down.
  If the TSHDN_FLT_CNT counter is higher than the TSHDN_FLT_CNT_MAX when power-down ends, the PMIC enters in FAIL_SAFE_LOCK state. Otherwise, the PMIC waits for FLT_TMR fault timer (tS_{HDN_DLY} = 3 s) and PMIC junction temperature goes below t_{SHDN_Fall} threshold, then the PMIC enters power-up, and then goes to power-on. The application then initializes and runs.

6.2.4 WDG: watchdog timer expiration

The PMIC embeds a programmable watchdog that can be enabled at runtime by the software, or enabled by default at power-up (NVM setting):

- WDG_TMR_SET: watchdog timer duration value setting.

- WDG_TMR_CNT: watchdog timer down-counter.
- WDG_EN: watchdog enable bit.
- WDG_RST: watchdog clear bit. To be periodically set by the MPU software to reset the watchdog timer.

Note: A PMIC PWRCTRLx input can be affected to suspend the watchdog typically in low-power mode. When this PWRCTRL is asserted in low-power mode, the watchdog timer is suspended. When this PWRCTRL is desasserted, the watchdog timer is resumed.

If the MPU software fails to clear the PMIC watchdog timer (WDG_RST), the watchdog timer elapses.

Once the PMIC watchdog timer elapses, a hard fault condition is generated: the watchdog fault counter (WDG_FLT_CNT) is incremented and the PMIC enters power-down.

If the WDG_FLT_CNT counter is higher than the WDG_FLT_CNT_MAX when power-down ends, the PMIC enters in FAIL_SAFE_LOCK state. Otherwise, the PMIC enters power-up, then goes to power-on. The application then initializes and runs.

6.2.5 PKEY: power on key user button long press

A long press on the PONKEYn user button enables the PMIC hard fault condition to be triggered. It is similar to a system reset except that PMIC performs a power cycle in addition to asserting the reset signal.

The long press duration is set to 10 seconds by default with STPMIC1LA. PMIC NVM can be reprogrammed to adjust the duration by modifying NVM_PKEY_LKP_TMR or to disable this feature by setting NVM_PKEY_LKP_OFF bit.

Once the long key press PONKEYn timer elapses (10 s by default), a hard fault condition is generated: the PKEY fail-safe counter (PKEY_FLT_CNT) is incremented and the PMIC enters power-down.

If the PKEY_FLT_CNT counter is higher than the PKEY_FLT_CNT_MAX (infinite by default) when power-down ends, the PMIC enters in FAIL_SAFE_LOCK state. Otherwise, the PMIC enters power-up, and then goes to power-on. The application then initializes and runs.

6.3 OCP settings in the application

In the application illustrated in Figure 1, the two levels of protection are applied on the regulators as follows:

All regulators critical for the application are managed in OCP fail-safe (level 1), else in OCP Hiccup (level 0).

These settings can be modified by reprogramming the NVM_FS_OCP_SHRx registers in PMIC NVM according to Table 13.

Table 13. OCP management application

-	HICCUP (level 0)	Fail-safe (level 1)
BUCK1 (V_{DDCORE})	-	YES
BUCK2 (V_{DD_DDR})	-	YES
LDO2 (V_{DDIO})	-	YES
LDO3 (free)	YES	-
LDO4 (V_{DD3V3_USB})	YES	-
LDO5 (V_{DD_FLASH})	YES	-

Revision history

Table 14. **Document revision history**

Date	Version	Changes
21-Jul-2025	1	Initial release.
13-Nov-2025	2	Removed VTT_DDR DDR3L termination resistor network in the document. Updated Section 2: Overview. Updated Figure 1. STM32MP13x and STPMIC1LA with DDR3L, SD card, and external SMPS to supply peripherals. Updated introduction to Section 4: 5 V power supply application reference design. Updated Section 4.1: Power distribution. Updated Table 4. Default STPMIC1LA NVM configuration. Uodated Section 4.2.2: STPMIC1L digital control interface. Updated Table 5. Operating modes. Updated Section 5.2.1: Power-up triggered by main supply (VIN) plugin and power-down by software shutdown. Updated Figure 2. Power-up and power-down sequence MPU with PMIC. Updated Section 5.3: Low-power mode management. Updated Table 8. Low-power mode supported by the application. Updated Table 10. PMIC configuration for LPLV-Stop mode. Updated Figure 3. LPLV-Stop sequence. Updated Table 11. PMIC configuration for Standby DDR in self-refresh. Updated Figure 4. Standby (DDR in self-refresh) sequence. Updated Table 12. PMIC configuration Stanby DDR OFF. Updated Figure 5. Standby (DDR OFF) sequence. Updated Figure 6. Crash recovery sequence. Updated Table 13. OCP management application.

Contents

List of tables

List of figures