



# STM32MP15 backup registers



## Contents

1. STM32MP15 backup registers .....	3
2. TAMP internal peripheral .....	9
3. STM32MPU Embedded Software architecture overview .....	10
4. STM32MP15 ROM code overview .....	10
5. STM32MPU Embedded Software distribution .....	10
6. U-Boot overview .....	10
7. Linux remoteproc framework overview .....	10
8. Boot chains overview .....	10
9. TF-A overview .....	11
10. DDRCTRL and DDRPHYC internal peripherals .....	11
11. Power overview .....	11
12. RETRAM internal memory .....	11
13. RCC internal peripheral .....	11



# STM32MP15 backup registers

Stable: 12.02.2019 - 08:18 / Revision: 06.12.2018 - 11:20

Template:ArticleMainWriter Template:ArticleApprovedVersion

## Contents

1 Article purpose .....	3
2 Overview .....	3
3 Backup registers usage .....	3
<b>3.1 At boot time .....</b>	<b>4</b>
<b>3.2 At runtime .....</b>	<b>4</b>
4 Memory mapping .....	5
5 References .....	9

## 1 Article purpose

The purpose of this article is to explain how the TAMP backup registers are used by STM32MPU Embedded Software.

## 2 Overview

The STM32MP15 embeds 32 backup registers of 32 bits. A programmable border allows to split those backup registers into a secure and a non-secure group.

By default, the ROM code defines the 10 first backup registers as secure, but this secure/non-secure border can be changed later on from the secure world.

## 3 Backup registers usage

This paragraph explains for which purpose some backup registers are used by the ROM code and STM32MPU Embedded Software distribution.

Then, the next chapter shows the backup register mapping used to fulfill those needs.



Template:DarkBlue

## 3.1 At boot time

be.ST

### ST internal information

**Non-secure backup registers** not yet developed usage:

- by U-Boot or Linux remoteproc to store the Cortex<sup>®</sup>-M4 firmware **integrity check value**. This firmware can be loaded in U-Boot or in Linux<sup>®</sup>, depending on the boot chain configuration.

- **Non-secure backup registers** are used:
  - during a cold boot:
    - by U-Boot to initialize the **boot counter**, that should be reset later on by the application.
  - after a reset:
    - by U-Boot to get an eventual **forced boot mode** that was set before reset. This can be useful to set U-Boot in programmer mode after a reboot, for instance. Note that this **forced boot mode** is not interpreted by the ROM code.
    - by U-Boot to increment the **boot counter** and perform given actions if a predefined number of successive boots is reached, due to cyclic resets before the application is alive (and clears the counter).
- **Secure backup registers** are used:
  - to tell to the FSBL (TF-A or U-Boot SPL) how to behave:
    - on cold boot, the ROM code sets the **magic number** to 0x0: this value tells to the FSBL that a complete DDR initialization is needed before jumping to the SSBL (U-Boot).
    - on wakeup from Standby with DDR in self-refresh **low power mode**, if the **magic number** == 0xCA7FACE0 then the FSBL performs a partial DDR initialization to exit Self-Refresh then it branches the Arm<sup>®</sup> Cortex<sup>®</sup>-A7 core 0 non-secure execution to the given **branch address** (in Linux<sup>®</sup> kernel, that was set during secure context saving before the Standby **low power mode** entering).
  - by Linux<sup>®</sup> kernel on Arm<sup>®</sup> Cortex<sup>®</sup>-A7 core 0 (via a PSCI secure service) to tell to the ROM code how to start Arm<sup>®</sup> Cortex<sup>®</sup>-A7 core 1 (and enable the SMP mode): when Arm<sup>®</sup> Cortex<sup>®</sup>-A7 core 1 non-secure sees the **magic number** == 0xCA7FACE1 then it jumps to the given **branch address**.
  - by the ROM code during wakeup from Standby **low power mode** to recover the Cortex<sup>®</sup>-M4 firmware **integrity check value** and compare it to the one computed on RETRAM before starting the Cortex<sup>®</sup>-M4 again.

Notice: the ROM code knows if Cortex<sup>®</sup>-A7 and/or Cortex<sup>®</sup>-M4 have to be restarted after Standby thanks to RCC\_MP\_BOOTCR register, so the backup registers are not used here.

## 3.2 At runtime

- Non secure backup registers
  - own the **boot counter** and should be reset by the application after a successful startup.
  - are used to store Cortex<sup>®</sup>-M4 retention firmware **integrity check value** before going to Standby mode, if the Cortex<sup>®</sup>-M4 needs to be started on wakeup from Standby mode by the ROM code.



- Secure backup registers
  - are used by [secure services](#) to store:
    - Arm® Cortex®-A7 core 0 **branch address** that are used by the ROM code on wakeup from Standby mode.
    - Arm® Cortex®-M4 **security perimeter** that is restored by the ROM code before starting the Cortex®-M4 on wakeup from Standby.

## 4 Memory mapping

The table below shows the backup register mapping used by [STM32MPU Embedded Software](#).

The TAMP backup register base address is 0x5C00A100, corresponding to TAMP\_BKP0R.

TAMP register	Sec urity	ROM / software register name	Comment
TAMP_BKP31R	No n-se cure		
TAMP_BKP30R	No n-se cure		
TAMP_BKP29R	No n-se cure		
TAMP_BKP28R	No n-se cure		
TAMP_BKP27R	No n-se cure	BACKUP_M4_WAKEUP_ARE A_HASH	SHA-256 integrity check value computed on <a href="#">RETRAM</a> by <a href="#">Linux remoteproc</a> during the coprocessor firmware loading and checked by the ROM code on wakeup from Standby before starting the coprocessor



STM32MP15 backup registers

TAM P regist er	Sec urity	ROM / software register name	Comment
R	re		
TA MP _BK P26 R	No n- se cu re		
TA MP _BK P25 R	No n- se cu re		
TA MP _BK P24 R	No n- se cu re		
TA MP _BK P23 R	No n- se cu re	BACKUP_M4_ WAKEUP_ARE A_LENGTH	Amount of bytes hashed in <a href="#">RETRAM</a> to compute the integrity check value
TA MP _BK P22 R	No n- se cu re	BACKUP_M4_ WAKEUP_ARE A_START	Start address in <a href="#">RETRAM</a> from where the integrity check value has to be computed
TA MP _BK P21 R	No n- se cu re	BACKUP_BOO T_COUNTER	Boot counter
TA MP _BK P20 R	No n- se cu re	BACKUP_BOO T_MODE <sup>[1]</sup>	Boot mode context information
TA	No		



STM32MP15 backup registers

TAM P regist er	Sec urity	ROM / software register name	Comment
MP _BK P19 R	n- se cu re		
TA MP _BK P18 R	No n- se cu re		
TA MP _BK P17 R	No n- se cu re		(Reserved for future use)
TA MP _BK P16 R	No n- se cu re		(Reserved for future use)
TA MP _BK P15 R	No n- se cu re		(Reserved for future use)
TA MP _BK P14 R	No n- se cu re		(Reserved for future use)
TA MP _BK P13 R	No n- se cu re		(Reserved for future use)
TA MP _BK	No n- se		



STM32MP15 backup registers

TAMP register	Security	ROM / software register name	Comment
P12R	cu re		(Reserved for future use)
TAMP11R	No n- se cu re		(Reserved for future use)
TAMP10R	No n- se cu re		(Reserved for future use)
TAMP9R	Se cu re		(Reserved for future use)
TAMP8R	Se cu re		(Reserved for future use)
TAMP7R	Se cu re		(Reserved for future use)
TAMP6R	Se cu re		(Reserved for future use)
TAMP5R	Se cu re	BACKUP_BRANCH_ADDRESSES <sup>[1]</sup>	CPU0 or CPU1 branch address
TAMP4R	Se cu re	BACKUP_MAGIC_NUMBER <sup>[1]</sup>	CPU0 or CPU1 boot magic number





TAMP register	Security	ROM / software register name	Comment
TAMP_BKP3R	Secure	BACKUP_M4_SECURITY_PERIMETER_EXTI3	Value of AEIC TZENR3
TAMP_BKP2R	Secure	BACKUP_M4_SECURITY_PERIMETER_EXTI2	Value of AEIC TZENR2
TAMP_BKP1R	Secure	BACKUP_M4_SECURITY_PERIMETER_EXTI1	Value of AEIC TZENR1
TAMP_BKP0R	Secure	BACKUP_WAKEUP_SEC	Wakeup parameters

## 5 References

- [1.01.11.2 arch/arm/mach-stm32mp/include/mach/stm32.h](#)

Das U-Boot -- the Universal Boot Loader (see [U-Boot\\_overview](#))

First Stage Boot Loader

Second Stage Boot Loader

Doubledata rate (memory domain)

Power State Coordination Interface

symetric multiprocessing

Tamper

Secure Hash Algorithm

## Permission error

*Stable: 24.09.2019 - 13:57 / Revision: 19.09.2019 - 13:53*

You do not have permission to read this page, for the following reason:



The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 15.10.2019 - 11:55 / Revision: 15.10.2019 - 11:55*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 29.01.2020 - 16:12 / Revision: 29.01.2020 - 16:12*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 09.10.2019 - 09:37 / Revision: 26.09.2019 - 10:05*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 23.01.2020 - 13:52 / Revision: 23.01.2020 - 13:46*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 03.06.2020 - 09:44 / Revision: 03.06.2020 - 09:44*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 22.01.2020 - 16:05 / Revision: 22.01.2020 - 10:03*



You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 30.01.2020 - 13:42 / Revision: 30.01.2020 - 13:40*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 09.03.2020 - 15:20 / Revision: 06.03.2020 - 14:43*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 21.02.2019 - 14:20 / Revision: 20.02.2019 - 15:21*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 04.02.2020 - 15:59 / Revision: 04.02.2020 - 15:50*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer

## Permission error

---

*Stable: 04.02.2020 - 15:40 / Revision: 04.02.2020 - 15:27*

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST\_editors, ST\_readers, Selected\_editors, sysop, reviewer