



RTC internal peripheral

RTC internal peripheral



Contents

1. RTC internal peripheral	3
2. GIC internal peripheral	7
3. How to assign an internal peripheral to a runtime context	11
4. NVIC internal peripheral	15
5. Power overview	19
6. RCC internal peripheral	23
7. RTC device tree configuration	27
8. RTC overview	31
9. STGEN internal peripheral	35
10. STM32CubeMX	39
11. STM32MP15 resources	43
12. STM32MPU Embedded Software architecture overview	47
13. TAMP internal peripheral	51



A quality version of this page, approved on *19 February 2020*, was based off this revision.

Contents

1 Article purpose	4
2 Peripheral overview	5
2.1 Features	5
2.2 Security support	5
3 Peripheral usage and associated software	6
3.1 Boot time	6
3.2 Runtime	6
3.2.1 Overview	6
3.2.2 Software frameworks	6
3.2.3 Peripheral configuration	6
3.2.4 Peripheral assignment	6



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

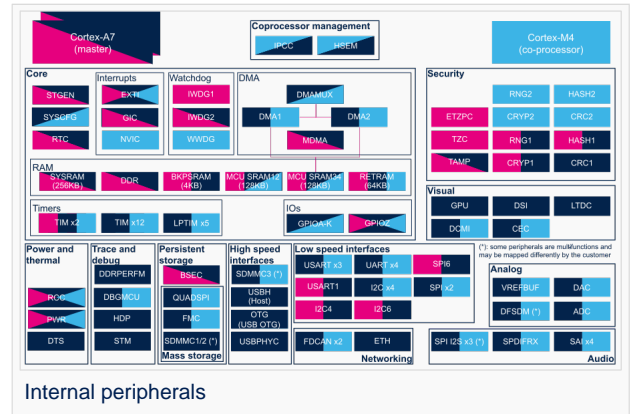
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 26.03.2021 - 13:17 / Revision: 18.03.2021 - 14:47

Contents

1 Article purpose	8
2 Peripheral overview	9
2.1 Features	9
2.2 Security support	9
3 Peripheral usage and associated software	10
3.1 Boot time	10
3.2 Runtime	10
3.2.1 Overview	10
3.2.2 Software frameworks	10
3.2.3 Peripheral configuration	10
3.2.4 Peripheral assignment	10



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

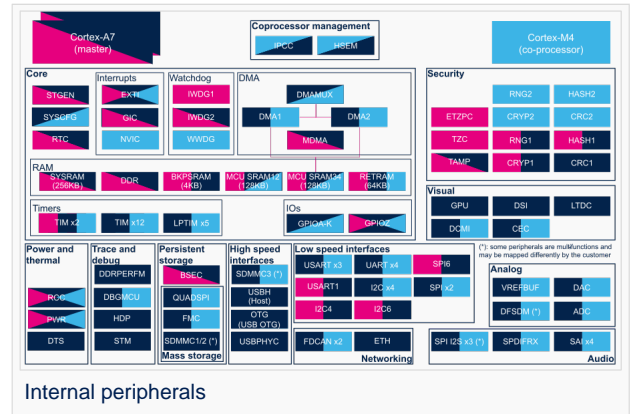
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 08.03.2021 - 16:13 / Revision: 16.02.2021 - 17:11

Contents

1 Article purpose	12
2 Peripheral overview	13
2.1 Features	13
2.2 Security support	13
3 Peripheral usage and associated software	14
3.1 Boot time	14
3.2 Runtime	14
3.2.1 Overview	14
3.2.2 Software frameworks	14
3.2.3 Peripheral configuration	14
3.2.4 Peripheral assignment	14



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

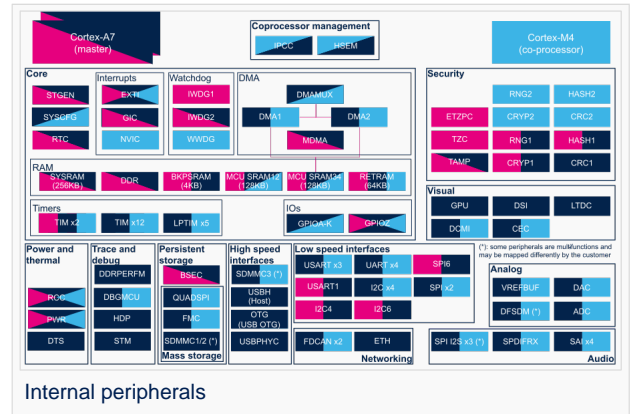
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 25.03.2021 - 13:50 / Revision: 18.03.2021 - 17:29

Contents

1 Article purpose	16
2 Peripheral overview	17
2.1 Features	17
2.2 Security support	17
3 Peripheral usage and associated software	18
3.1 Boot time	18
3.2 Runtime	18
3.2.1 Overview	18
3.2.2 Software frameworks	18
3.2.3 Peripheral configuration	18
3.2.4 Peripheral assignment	18



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

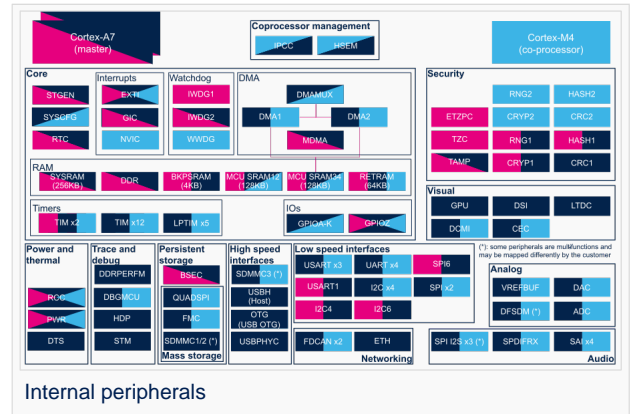
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 01.12.2020 - 10:35 / Revision: 01.12.2020 - 09:20

Contents

1 Article purpose	20
2 Peripheral overview	21
2.1 Features	21
2.2 Security support	21
3 Peripheral usage and associated software	22
3.1 Boot time	22
3.2 Runtime	22
3.2.1 Overview	22
3.2.2 Software frameworks	22
3.2.3 Peripheral configuration	22
3.2.4 Peripheral assignment	22



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

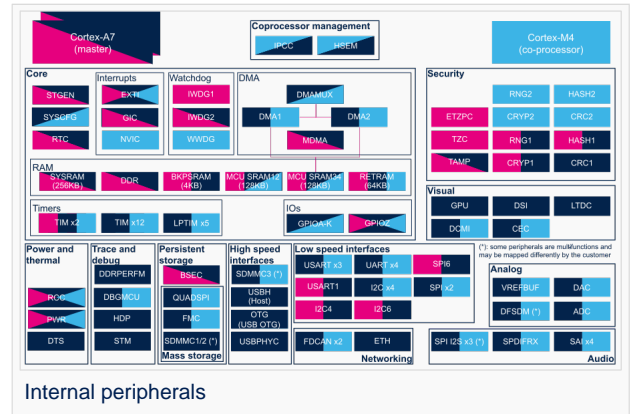
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 25.09.2020 - 09:10 / Revision: 25.09.2020 - 09:09

Contents

1 Article purpose	24
2 Peripheral overview	25
2.1 Features	25
2.2 Security support	25
3 Peripheral usage and associated software	26
3.1 Boot time	26
3.2 Runtime	26
3.2.1 Overview	26
3.2.2 Software frameworks	26
3.2.3 Peripheral configuration	26
3.2.4 Peripheral assignment	26



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

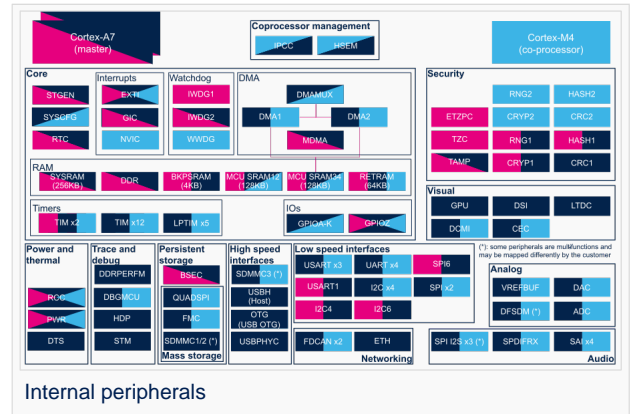
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 09.09.2021 - 14:07 / Revision: 09.09.2021 - 14:07

Contents

1 Article purpose	28
2 Peripheral overview	29
2.1 Features	29
2.2 Security support	29
3 Peripheral usage and associated software	30
3.1 Boot time	30
3.2 Runtime	30
3.2.1 Overview	30
3.2.2 Software frameworks	30
3.2.3 Peripheral configuration	30
3.2.4 Peripheral assignment	30



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

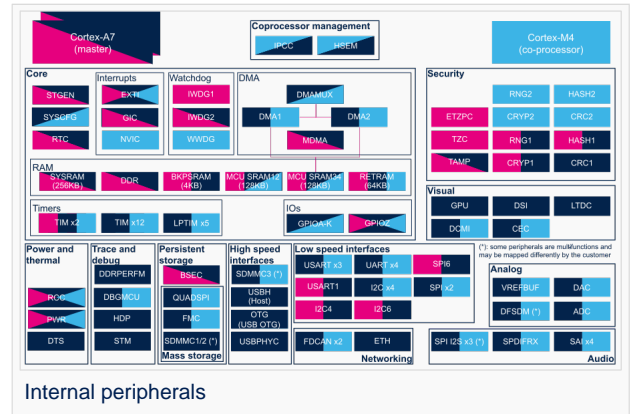
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 14.09.2021 - 14:27 / Revision: 14.09.2021 - 14:25

Contents

1 Article purpose	32
2 Peripheral overview	33
2.1 Features	33
2.2 Security support	33
3 Peripheral usage and associated software	34
3.1 Boot time	34
3.2 Runtime	34
3.2.1 Overview	34
3.2.2 Software frameworks	34
3.2.3 Peripheral configuration	34
3.2.4 Peripheral assignment	34



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

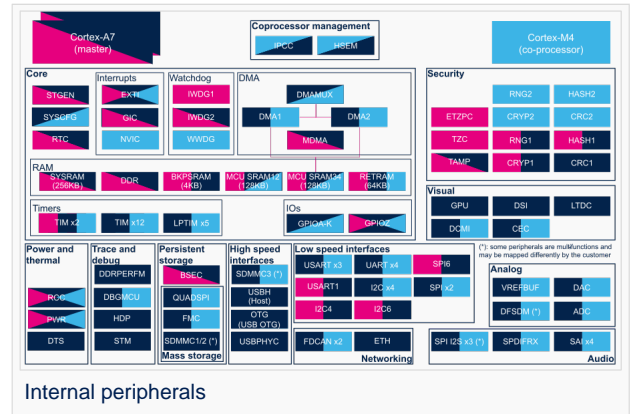
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 25.09.2020 - 09:42 / Revision: 25.09.2020 - 09:36

Contents

1 Article purpose	36
2 Peripheral overview	37
2.1 Features	37
2.2 Security support	37
3 Peripheral usage and associated software	38
3.1 Boot time	38
3.2 Runtime	38
3.2.1 Overview	38
3.2.2 Software frameworks	38
3.2.3 Peripheral configuration	38
3.2.4 Peripheral assignment	38



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

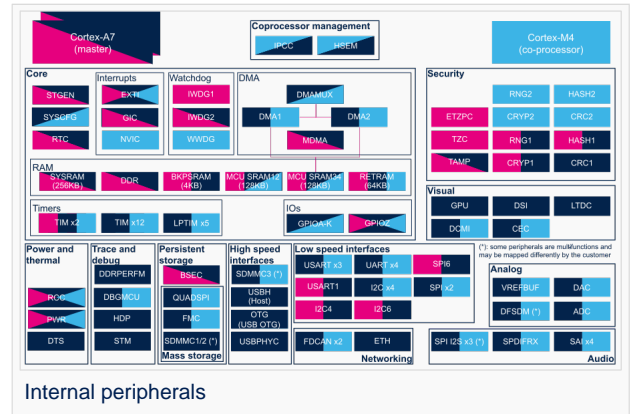
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 23.09.2020 - 13:22 / Revision: 12.06.2020 - 13:25

Contents

1 Article purpose	40
2 Peripheral overview	41
2.1 Features	41
2.2 Security support	41
3 Peripheral usage and associated software	42
3.1 Boot time	42
3.2 Runtime	42
3.2.1 Overview	42
3.2.2 Software frameworks	42
3.2.3 Peripheral configuration	42
3.2.4 Peripheral assignment	42



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

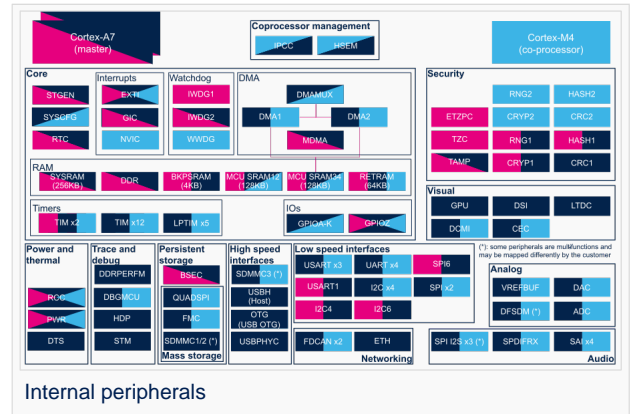
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 20.07.2021 - 09:02 / Revision: 11.03.2021 - 08:07

Contents

1 Article purpose	44
2 Peripheral overview	45
2.1 Features	45
2.2 Security support	45
3 Peripheral usage and associated software	46
3.1 Boot time	46
3.2 Runtime	46
3.2.1 Overview	46
3.2.2 Software frameworks	46
3.2.3 Peripheral configuration	46
3.2.4 Peripheral assignment	46



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

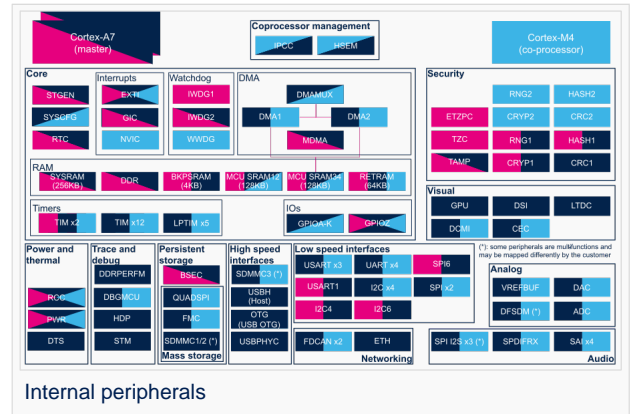
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 26.03.2021 - 11:32 / Revision: 12.03.2021 - 11:07

Contents

1 Article purpose	48
2 Peripheral overview	49
2.1 Features	49
2.2 Security support	49
3 Peripheral usage and associated software	50
3.1 Boot time	50
3.2 Runtime	50
3.2.1 Overview	50
3.2.2 Software frameworks	50
3.2.3 Peripheral configuration	50
3.2.4 Peripheral assignment	50



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

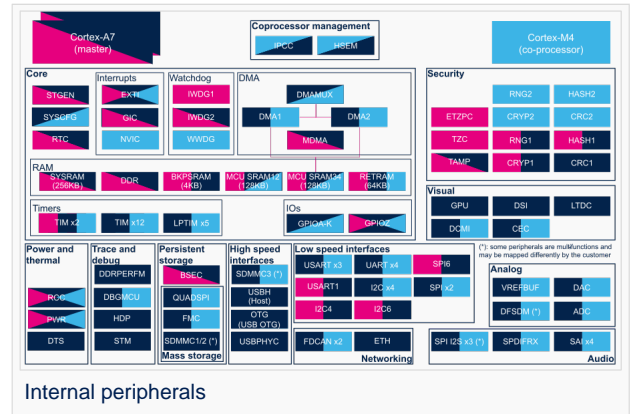
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment

Stable: 15.09.2021 - 06:37 / Revision: 15.09.2021 - 06:35

Contents

1 Article purpose	52
2 Peripheral overview	53
2.1 Features	53
2.2 Security support	53
3 Peripheral usage and associated software	54
3.1 Boot time	54
3.2 Runtime	54
3.2.1 Overview	54
3.2.2 Software frameworks	54
3.2.3 Peripheral configuration	54
3.2.4 Peripheral assignment	54



1 Article purpose

The purpose of this article is to

- briefly introduce the RTC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the RTC peripheral.



2 Peripheral overview

The **RTC** peripheral is used to provide the date and clock to the application. It supports programmable alarms and wake up capabilities.

2.1 Features

Refer to [STM32MP15 reference manuals](#) for the complete list of features, and to the software components, introduced below, to know which features are really implemented.

2.2 Security support

The **RTC** peripheral is a **secure** peripheral.



3 Peripheral usage and associated software

3.1 Boot time

By default after a backup domain power-on reset (performed at boot time), all RTC registers can be read or written in both secure and non-secure modes.

In OpenSTLinux distribution, the first stage bootloader (FSBL, running in secure mode) keeps this default configuration, leaving full control to Linux® at runtime.

The **RTC** peripheral is able to generate two interrupts:

- A secure interrupt, connected to the Arm®Cortex®-A7 GIC, not used in OpenSTLinux distribution.
- A non-secure interrupt, connected both to Arm®Cortex®-A7 GIC and Cortex-M4 NVIC: this interrupt is used on Linux® and by default in OpenSTLinux distribution.

The **RTC** peripheral is part of the backup domain which reset and clock are controlled via the **RCC** by the first stage bootloader (FSBL, running in secure mode) at boot time.

The RTC reset occurs when the backup domain is reset. To avoid clearing the **TAMP** register contents, this is only done on cold boot, not on wake up.

3.2 Runtime

3.2.1 Overview

The **RTC** peripheral can be allocated to the Arm®Cortex®-A7 non-secure core to be used under Linux® with RTC framework.

3.2.2 Software frameworks

Domain	Peripheral	Software components	Comment
OP-TEE	Linux	STM32Cube	
Core	RTC		Linux RTC framework

3.2.3 Peripheral configuration

The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via **STM32CubeMX** tool for all internal peripherals, and then manually completed (especially for external peripherals) according to the information given in the corresponding software framework article.

For Linux kernel configuration, please refer to **RTC device tree configuration**.

3.2.4 Peripheral assignment

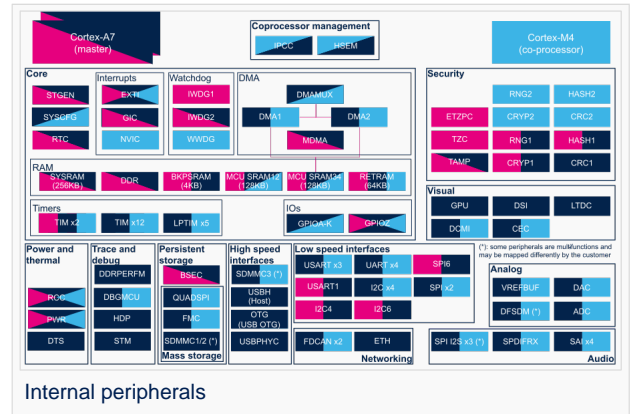
Check boxes illustrate the possible peripheral allocations supported by **STM32 MPU Embedded Software**:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.

The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possibilities might be described in STM32MP15 reference manuals



Internal peripherals

Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core	RTC	RTC		RTC is mandatory to resynchronize ST GEN after exiting low-power modes.

Real Time Clock

First Stage Boot Loader

Linux® is a registered trademark of Linus Torvalds.

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



Cortex®

Open Portable Trusted Execution Environment