



RNG internal peripheral



Contents

1. RNG internal peripheral	3
2. STM32MP15 resources	6
3. ETZPC internal peripheral	10
4. OP-TEE overview	13
5. Hardware random overview	17
6. STM32CubeMP1 architecture	20
7. STM32CubeMX	24
8. STM32MPU Embedded Software architecture overview	27
9. How to assign an internal peripheral to a runtime context	31



RNG internal peripheral

Stable: 11.02.2019 - 11:21 / Revision: 18.01.2019 - 16:23

A quality version of this page, accepted on 11 February 2019, was based off this revision.

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

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

1 Article purpose

The purpose of this article is to:

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

2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.



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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.

3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm[®] Cortex[®]-A7 secure core to be controlled in OP-TEE with [OP-TEE RNG driver](#)

or

- the Arm[®] Cortex[®]-A7 non-secure core to be controlled in Linux[®] by the [Linux hardware random framework](#)

or

- the Arm[®] Cortex[®]-M4 to be controlled in STM32Cube MPU Package by [STM32Cube RNG driver](#)

Chapter [#Peripheral assignment](#) exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks	Comment
main Cortex -A7	Cor tex -A7 no	Cortex-M4	

Do	Peri	Software frameworks			Comment
main (OP-TEE)	non-secure (Linux)	(STM32Cube)			
		OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the **STM32CubeMX** tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

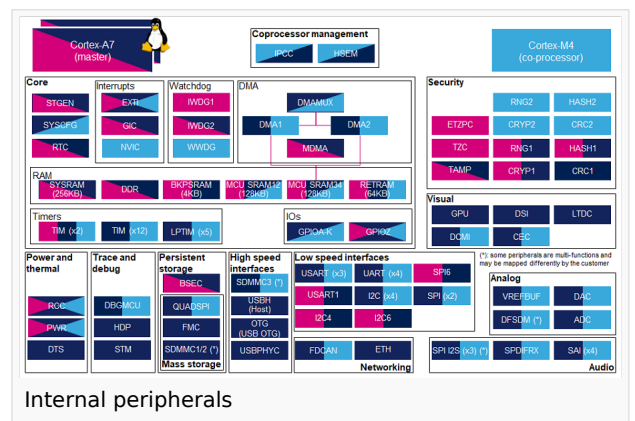
3.2.4 Peripheral assignment

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.



Do	Peri	Runtime allocation			Comment
main (OP-TEE)	non-secure (Linux)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)		



Do ma in	Per iph era l T E E	Runtime allocation				Comme nt
S e c u r i t y	R N G	RNG1				Assig nment (singl e choic e)
		RNG2				

4 How to go further

Not applicable.

5 References

Random Number Generator

Open Portable Trusted Execution Environment

Microprocessor Unit

RNG internal peripheral

Stable: 21.02.2020 - 08:59 / Revision: 14.02.2020 - 10:13

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	7
2 Peripheral overview	7
2.1 Features	7
2.2 Security support	7
3 Peripheral usage and associated software	7
3.1 Boot time	7



3.2 Runtime	8
3.2.1 Overview	8
3.2.2 Software frameworks	8
3.2.3 Peripheral configuration	8
3.2.4 Peripheral assignment	8
4 How to go further	9
5 References	10

1 Article purpose

The purpose of this article is to:

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

2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.

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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.



3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE with OP-TEE RNG driver

or

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the Linux hardware random framework

or

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube RNG driver

Chapter #Peripheral assignment exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai Cor tex -A7 sec ure (O P- T E E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
Se cu rity	R N G	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

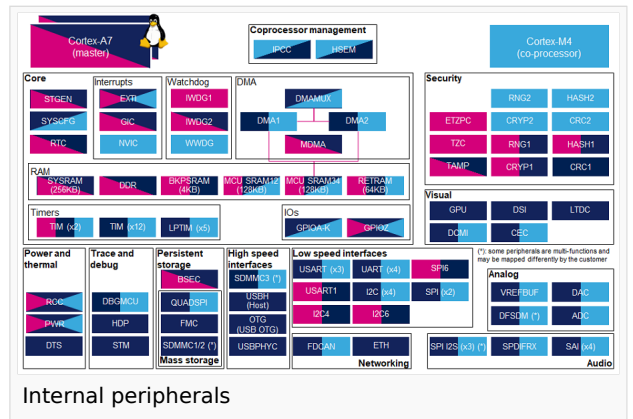
3.2.4 Peripheral assignment

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		Runtime allocation				Comment
Security	Peripheral Cortex-A7 non-secure (OPTEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
		RNG1				Assignment (single choice)
		RNG2				

4 How to go further

Not applicable.



5 References

Random Number Generator
Open Portable Trusted Execution Environment
Microprocessor Unit

RNG internal peripheral

Stable: **Not stable** / Revision: 12.02.2020 - 16:43

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	10
2 Peripheral overview	11
2.1 Features	11
2.2 Security support	11
3 Peripheral usage and associated software	11
3.1 Boot time	11
3.2 Runtime	11
3.2.1 Overview	11
3.2.2 Software frameworks	11
3.2.3 Peripheral configuration	12
3.2.4 Peripheral assignment	12
4 How to go further	13
5 References	13

1 Article purpose

The purpose of this article is to:

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



2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.

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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.

3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm[®] Cortex[®]-A7 secure core to be controlled in OP-TEE with OP-TEE RNG driver

or

- the Arm[®] Cortex[®]-A7 non-secure core to be controlled in Linux[®] by the Linux hardware random framework

or

- the Arm[®] Cortex[®]-M4 to be controlled in STM32Cube MPU Package by STM32Cube RNG driver

Chapter #Peripheral assignment exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks	Comment
mai Cor tex	Cor tex		

Do	Peri	Software frameworks			Comment
Security (OP-TEE)	-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the **STM32CubeMX** tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

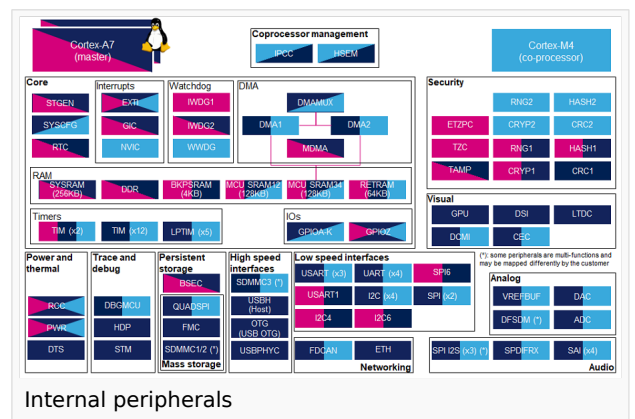
3.2.4 Peripheral assignment

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.



Do	Peri	Runtime allocation		Comment
main or insecure	Cortex-A7	Cortex-A7 non-secure	Cortex-M4	



Do ma ine	Per iph era O P- T E E	Runtime allocation			Comme nt
		(Linux)	(STM32Cube)		
S e c u r i t y	R N G	RNG1			Assig nment (singl e choic e)
		RNG2			

4 How to go further

Not applicable.

5 References

Random Number Generator

Open Portable Trusted Execution Environment

Microprocessor Unit

RNG internal peripheral

Stable: 12.03.2020 - 12:15 / Revision: 14.10.2019 - 14:35

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	14
2 Peripheral overview	14
2.1 Features	14
2.2 Security support	14
3 Peripheral usage and associated software	14
3.1 Boot time	14



3.2 Runtime	15
3.2.1 Overview	15
3.2.2 Software frameworks	15
3.2.3 Peripheral configuration	15
3.2.4 Peripheral assignment	15
4 How to go further	16
5 References	17

1 Article purpose

The purpose of this article is to:

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

2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.

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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.



3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE with OP-TEE RNG driver

or

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the Linux hardware random framework

or

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube RNG driver

Chapter #Peripheral assignment exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai Cor tex -A7 sec ure (O P- T E E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
Se cu rity	R N G	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

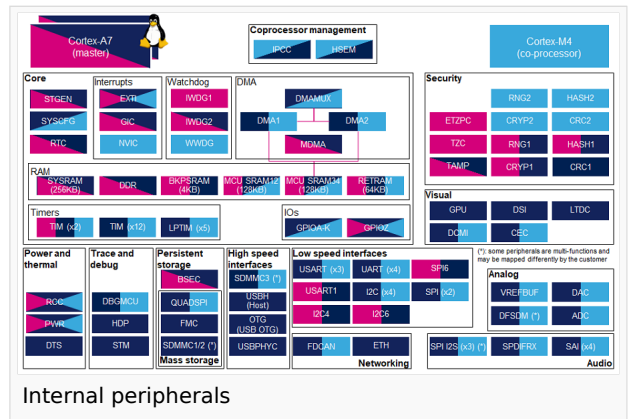
3.2.4 Peripheral assignment

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

Do		Runtime allocation				Comme
ma						nt
in						
Insta nanc e	Per iph era l C o r t e x - A 7 s e c u r e (O P T E E)	Cortex-A7 non-secure (Linux)		Cortex-M4 (STM32Cube)		
		RNG				Assig nment (singl e choic e)
Sec ur ity	RNG	RNG1				
		RNG2				

4 How to go further

Not applicable.



5 References

Random Number Generator
Open Portable Trusted Execution Environment
Microprocessor Unit

RNG internal peripheral

Stable: 12.02.2020 - 16:46 / Revision: 12.02.2020 - 16:44

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	17
2 Peripheral overview	18
2.1 Features	18
2.2 Security support	18
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	19
3.2.4 Peripheral assignment	19
4 How to go further	20
5 References	20

1 Article purpose

The purpose of this article is to:

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



2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.

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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.

3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm[®] Cortex[®]-A7 secure core to be controlled in OP-TEE with OP-TEE RNG driver

or

- the Arm[®] Cortex[®]-A7 non-secure core to be controlled in Linux[®] by the Linux hardware random framework

or

- the Arm[®] Cortex[®]-M4 to be controlled in STM32Cube MPU Package by STM32Cube RNG driver

Chapter #Peripheral assignment exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks	Comment
mai Cor tex	Cor tex		

Do	Peri	Software frameworks			Comment
Security (OP-TEE)	-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the **STM32CubeMX** tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

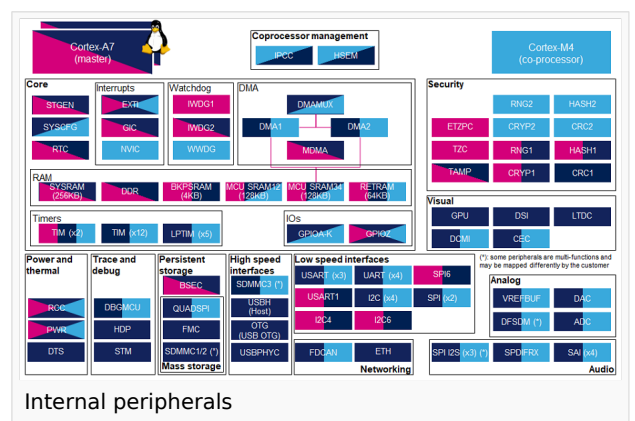
3.2.4 Peripheral assignment

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.



Do	Peri	Runtime allocation		Comment
main or insecure	Cortex-A7	Cortex-A7 non-secure	Cortex-M4	



Do ma in e	Per iph era O P- T E E	Runtime allocation			Comme nt
		(Linux)	(STM32Cube)		
S e c u r i t y	R N G	RNG1			Assig nment (singl e choic e)
		RNG2			

4 How to go further

Not applicable.

5 References

Random Number Generator

Open Portable Trusted Execution Environment

Microprocessor Unit

RNG internal peripheral

Stable: 21.02.2020 - 08:39 / Revision: 04.02.2020 - 15:22

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	21
2 Peripheral overview	21
2.1 Features	21
2.2 Security support	21
3 Peripheral usage and associated software	21
3.1 Boot time	21



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
4 How to go further	23
5 References	24

1 Article purpose

The purpose of this article is to:

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

2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.

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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.



3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE with OP-TEE RNG driver

or

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the Linux hardware random framework

or

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube RNG driver

Chapter #Peripheral assignment exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai Cor tex -A7 sec ure (O P- T E E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
Se cu rity	R N G	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

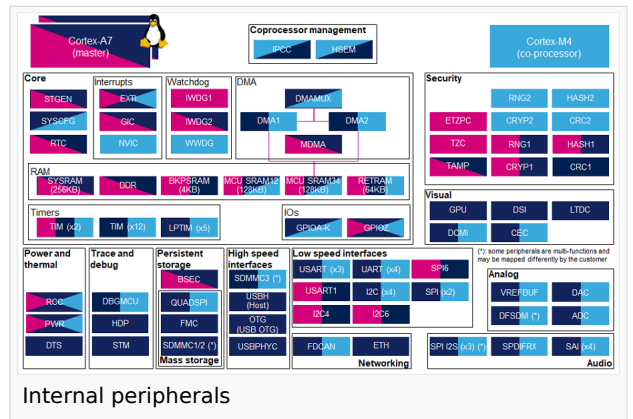
3.2.4 Peripheral assignment

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
Security	RNG	RNG1				Assignment (single choice)
		RNG2				

4 How to go further

Not applicable.



5 References

Random Number Generator

Open Portable Trusted Execution Environment

Microprocessor Unit

RNG internal peripheral

Stable: 31.01.2020 - 13:04 / Revision: 31.01.2020 - 13:02

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	24
2 Peripheral overview	25
2.1 Features	25
2.2 Security support	25
3 Peripheral usage and associated software	25
3.1 Boot time	25
3.2 Runtime	25
3.2.1 Overview	25
3.2.2 Software frameworks	25
3.2.3 Peripheral configuration	26
3.2.4 Peripheral assignment	26
4 How to go further	27
5 References	27

1 Article purpose

The purpose of this article is to:

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



2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.

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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.

3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm[®] Cortex[®]-A7 secure core to be controlled in OP-TEE with OP-TEE RNG driver

or

- the Arm[®] Cortex[®]-A7 non-secure core to be controlled in Linux[®] by the Linux hardware random framework

or

- the Arm[®] Cortex[®]-M4 to be controlled in STM32Cube MPU Package by STM32Cube RNG driver

Chapter #Peripheral assignment exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks	Comment
mai Cor tex	Cor tex		

Do	Peri	Software frameworks			Comment
Security (OP-TEE)	-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the **STM32CubeMX** tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

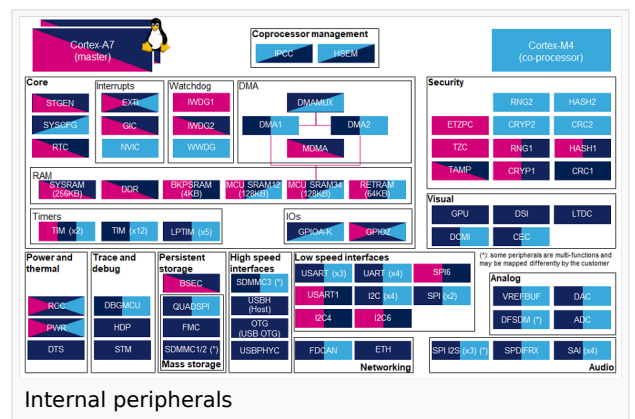
3.2.4 Peripheral assignment

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.



Do	Peri	Runtime allocation		Comment
main or insecure	Cortex-A7	Cortex-A7 non-secure	Cortex-M4	



Do ma ine	Per iph era O P- T E E	Runtime allocation			Comme nt
		(Linux)	(STM32Cube)		
S e c u r i t y	R N G	RNG1			Assig nment (singl e choic e)
		RNG2			

4 How to go further

Not applicable.

5 References

Random Number Generator

Open Portable Trusted Execution Environment

Microprocessor Unit

RNG internal peripheral

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

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	28
2 Peripheral overview	28
2.1 Features	28
2.2 Security support	28
3 Peripheral usage and associated software	28
3.1 Boot time	28



3.2 Runtime	29
3.2.1 Overview	29
3.2.2 Software frameworks	29
3.2.3 Peripheral configuration	29
3.2.4 Peripheral assignment	29
4 How to go further	30
5 References	31

1 Article purpose

The purpose of this article is to:

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

2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.

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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.



3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE with OP-TEE RNG driver

or

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the Linux hardware random framework

or

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube RNG driver

Chapter #Peripheral assignment exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai Cor tex -A7 sec ure (O P- T E E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
Se cu rity	R N G	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

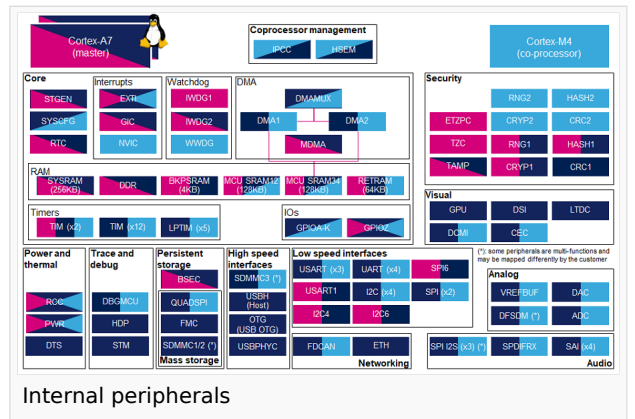
3.2.4 Peripheral assignment

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		Runtime allocation				Comment
Security	Peripheral Cortex-A7 non-secure (OPTIONE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
		RNG1				Assignment (single choice)
		RNG2				

4 How to go further

Not applicable.



5 References

Random Number Generator

Open Portable Trusted Execution Environment

Microprocessor Unit

RNG internal peripheral

Stable: **Not stable** / Revision: 30.01.2020 - 08:45

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	31
2 Peripheral overview	32
2.1 Features	32
2.2 Security support	32
3 Peripheral usage and associated software	32
3.1 Boot time	32
3.2 Runtime	32
3.2.1 Overview	32
3.2.2 Software frameworks	32
3.2.3 Peripheral configuration	33
3.2.4 Peripheral assignment	33
4 How to go further	34
5 References	34

1 Article purpose

The purpose of this article is to:

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



2 Peripheral overview

The **RNG** peripheral is used to provide 32-bit random numbers.

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

RNG1 is a **secure** peripheral (under ETZPC control).

RNG2 is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

RNG instances are not used as boot devices.

3.2 Runtime

3.2.1 Overview

RNG instances can be allocated to:

- the Arm[®] Cortex[®]-A7 secure core to be controlled in OP-TEE with OP-TEE RNG driver

or

- the Arm[®] Cortex[®]-A7 non-secure core to be controlled in Linux[®] by the Linux hardware random framework

or

- the Arm[®] Cortex[®]-M4 to be controlled in STM32Cube MPU Package by STM32Cube RNG driver

Chapter #Peripheral assignment exposes which instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks	Comment
mai Cor tex	Cor tex		

Do	Peri	Software frameworks			Comment
Security (OP-TEE)	-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

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 the **STM32CubeMX** tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

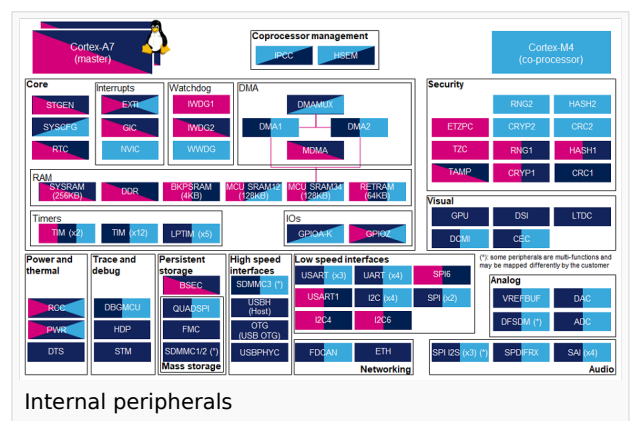
3.2.4 Peripheral assignment

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.



Do	Peri	Runtime allocation		Comment
main context - A7 insecure	Cortex-A7 non-secure	Cortex-M4		
		Cortex-A7 non-secure	Cortex-M4	



Do ma in e	Per iph era l P- T E E	Runtime allocation				Comme nt
		(Linux)	(STM32Cube)			
S e c u r i t y	R N G	RNG1				Assig nment (singl e choic e)
		RNG2				

4 How to go further

Not applicable.

5 References

Random Number Generator

Open Portable Trusted Execution Environment

Microprocessor Unit