RNG internal peripheral

Contents

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

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
Cortex-A7	Cortex-A7 no	Cortex-M4	

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

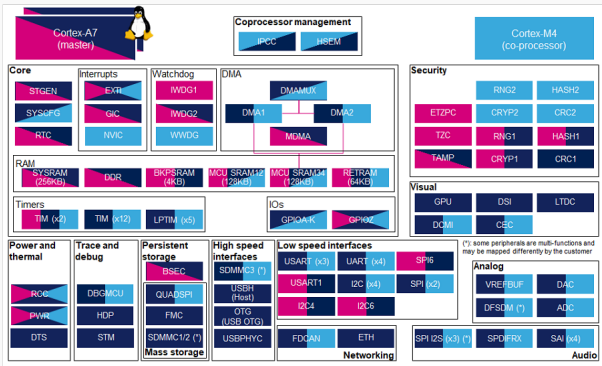
3.2.3 Peripheral configuration

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.



Internal peripherals

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 possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	

Domain	Peripheral	Runtime allocation				Comment
	OP-TEE)					
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: 21.02.2020 - 08:59 / Revision: 14.02.2020 - 10:13

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

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 n Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Security	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

3.2.3 Peripheral configuration

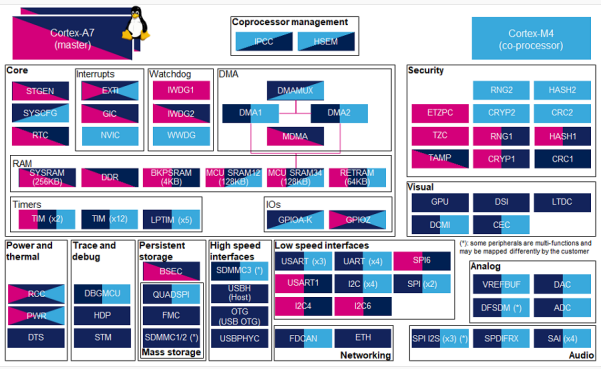
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in 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 possiblities 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)			
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: **Not stable** / Revision: 12.02.2020 - 16:43*

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

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		

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

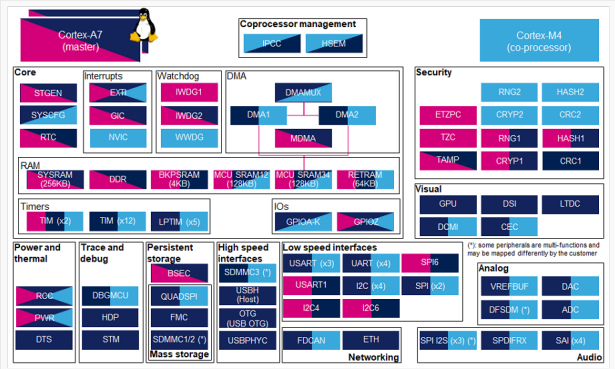
3.2.3 Peripheral configuration

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.



Internal peripherals

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 possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Inst a	Cortex-A7 secure	Cortex-A7 non-secure	Cortex-M4	

Domain	Peripheral (OP-TEE)	Runtime allocation (Linux)	Runtime allocation (STM32Cube)			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: 12.03.2020 - 12:15 / Revision: 14.10.2019 - 14:35

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

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 n Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Security	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

3.2.3 Peripheral configuration

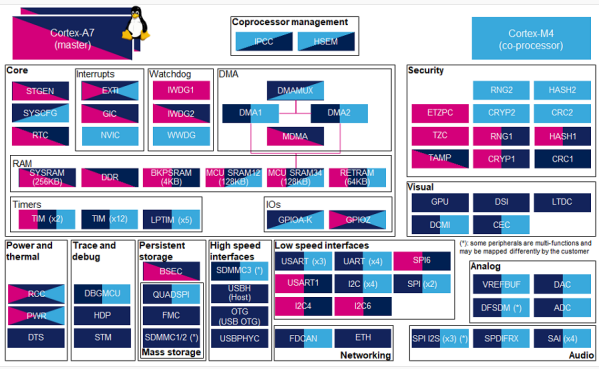
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in 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 possiblities 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)			
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: 12.02.2020 - 16:46 / Revision: 12.02.2020 - 16:44

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

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		

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

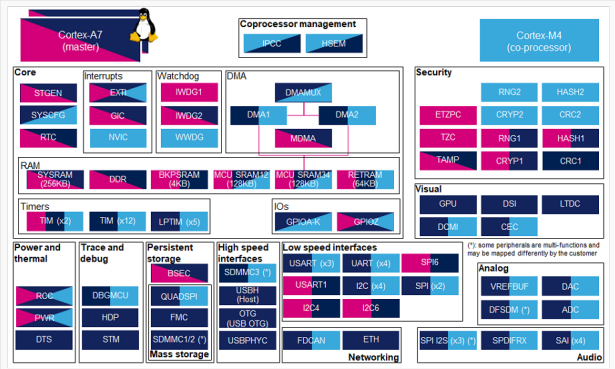
3.2.3 Peripheral configuration

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.



Internal peripherals

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 possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure	Cortex-A7 non-secure	Cortex-M4	

Domain	Peripheral (OP-TEE)	Runtime allocation (Linux)	Runtime allocation (STM32Cube)			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: 21.02.2020 - 08:39 / Revision: 04.02.2020 - 15:22

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

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 n Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Security	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

3.2.3 Peripheral configuration

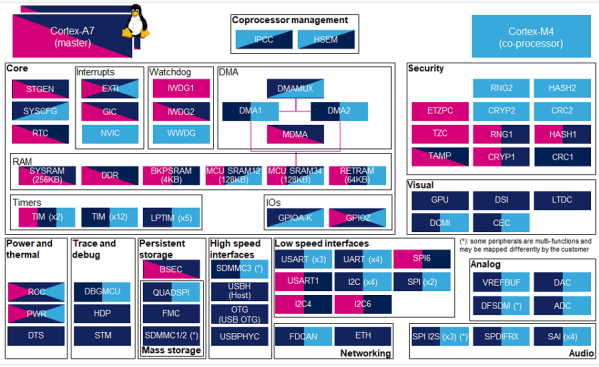
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

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		

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

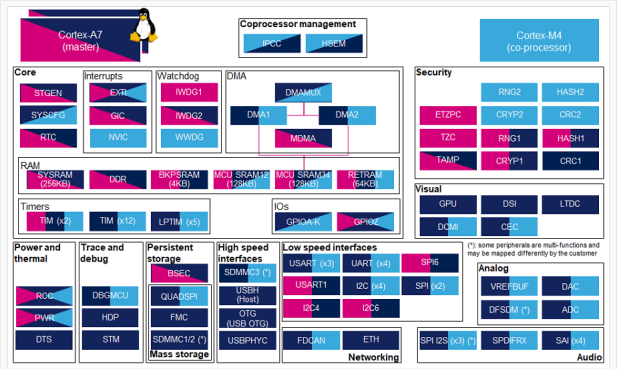
3.2.3 Peripheral configuration

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.



Internal peripherals

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 possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Insta	Cortex-A7 secure	Cortex-A7 non-secure	Cortex-M4	

Domain	Peripheral (OP-TEE)	Runtime allocation (Linux)	Runtime allocation (STM32Cube)			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: 15.10.2019 - 11:55 / Revision: 15.10.2019 - 11:55

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

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 n Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Security	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

3.2.3 Peripheral configuration

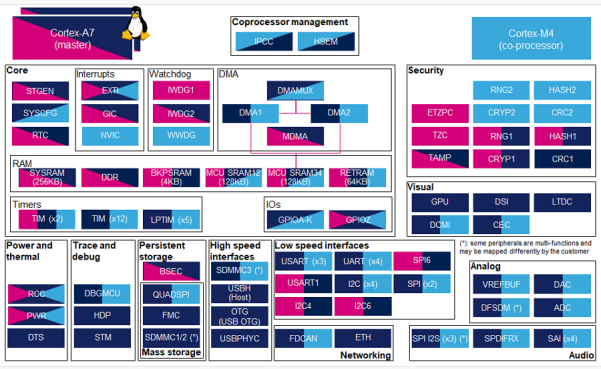
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in 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 possiblities 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)			
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: **Not stable** / Revision: 30.01.2020 - 08:45*

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

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	Cortex		

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

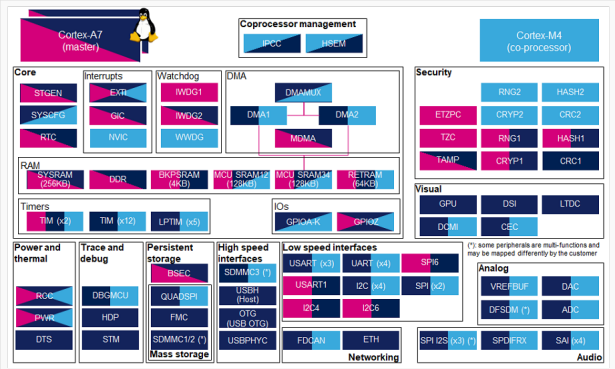
3.2.3 Peripheral configuration

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.



Internal peripherals

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 possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Insta	Cortex-A7 secure	Cortex-A7 non-secure	Cortex-M4	

Domain	Peripheral (OP-TEE)	Runtime allocation (Linux)	Runtime allocation (STM32Cube)			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