

RNG internal peripheral

Stable: 24.09.2019 - 14:51 / Revision: 24.09.2019 - 14:50

Contents

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

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 mai n	Peri phe ral	Software frameworks			Comment
		Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Sec urity	RNG	OP-TEE RNG driver	Linux hardware random framework	STM32Cube RNG driver	

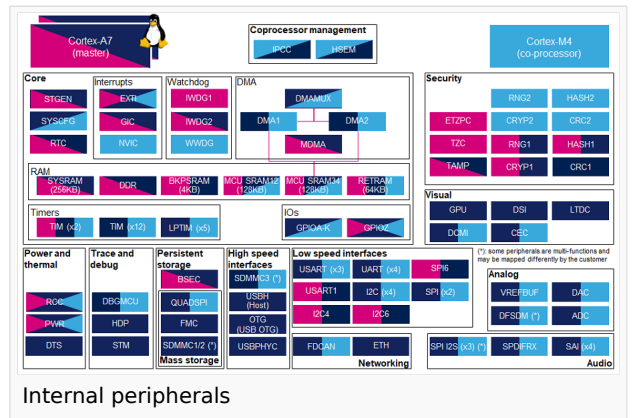
3.2.3 Peripheral configuration

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 ma in	Pe ri ph er al	Runtime allocation			Comme nt
		Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	
Se cur ity	RN G	RNG1	<input type="checkbox"/>	<input type="checkbox"/>	Assignm ent (single choice)
		RNG2			

4 How to go further

Not applicable.

5 References

Random Number Generator

Open Portable Trusted Execution Environment

Microprocessor Unit