



---

## Hardware random overview



---

## Contents

---

1. Hardware random overview .....	3
2. How to control a RNG in userspace .....	13
3. Menuconfig or how to configure kernel .....	23
4. RNG device tree configuration .....	33
5. STM32CubeMX .....	43



A quality version of this page, approved on 17 February 2021, was based off this revision.

This article gives information about the Linux<sup>®</sup> hardware random framework.

## Contents

1 Article Purpose .....	4
2 Framework purpose .....	5
3 System overview .....	6
3.1 Component description .....	6
3.2 API description .....	7
4 Configuration .....	8
4.1 Kernel configuration .....	8
4.2 Device tree configuration .....	8
5 How to use the framework .....	9
5.1 How to use from char device .....	9
5.2 How to use from sysfs .....	9
6 How to trace and debug the framework .....	10
7 Source code location .....	11
8 To go further .....	12
9 References .....	13



---

## 1 Article Purpose

---

This article gives information about the hardware random (HWRNG) framework.



---

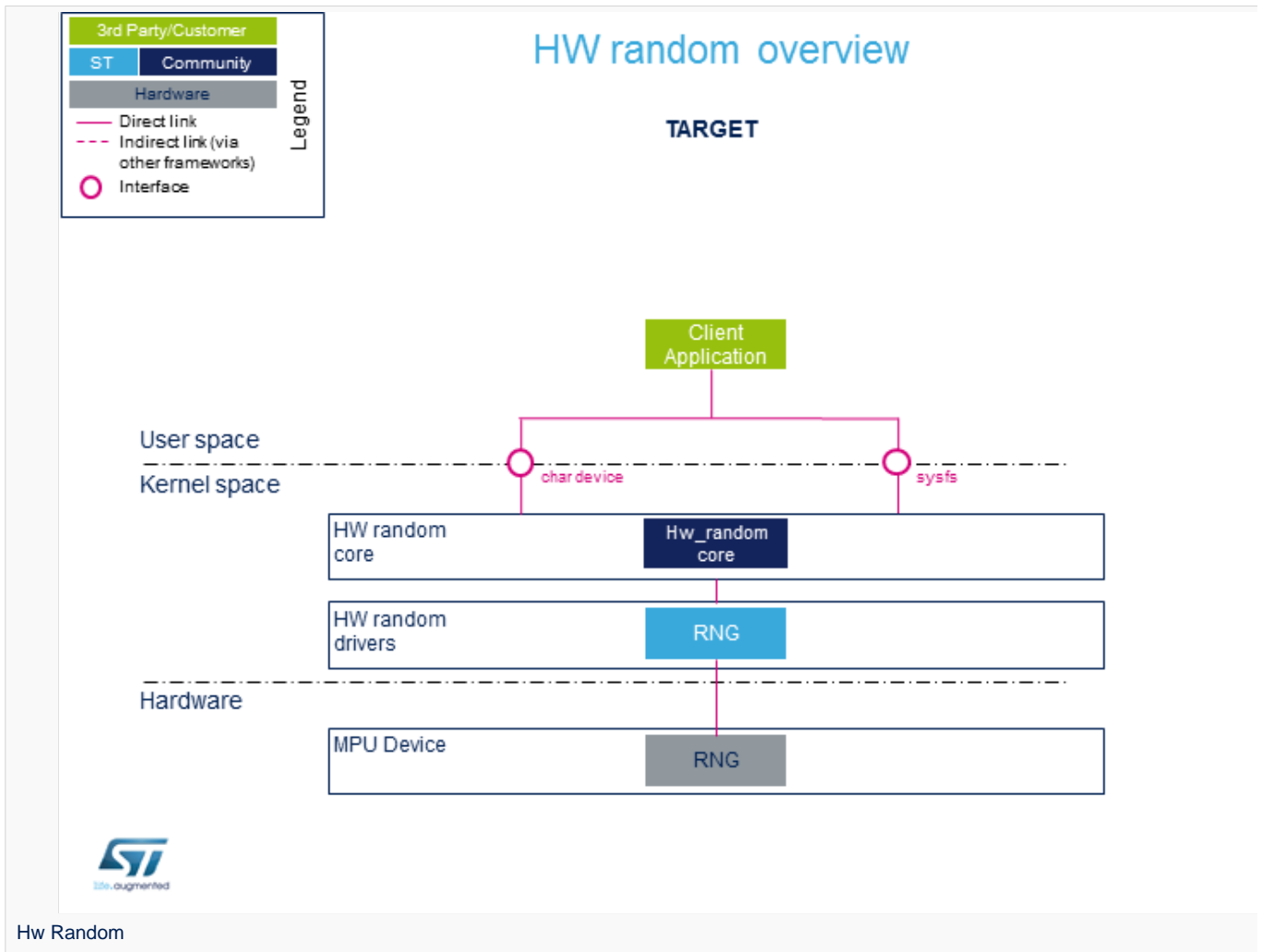
## 2 Framework purpose

---

The Hardware random framework is integrated in the kernel. It provides access to RNG peripherals and focuses on supporting the hardware number generator.

### 3 System overview

The HW random framework allows retrieving random numbers in userland.



#### 3.1 Component description

- **HW random core** (Kernel space)

Generic interface in kernel space. This layer is in charge of creating the character device (char device) and sysfs to access hw\_random.

- **RNG** (Kernel space)

Hardware random Linux<sup>®</sup> drivers handling the HW blocks.

- **RNG** (Hardware)

HW blocks handling the RNG peripheral.



### 3.2 API description

The Hardware random framework uses char device API<sup>[1]</sup> ioctl operations. For additional information, refer to:

- sysfs interface.
- Kernel Documentation directory<sup>[2]</sup>



## 4 Configuration

### 4.1 Kernel configuration

The Hardware random support is activated by default in ST deliveries. No specific configuration is required apart from enabling or disabling peripheral support using Linux<sup>®</sup> Menuconfig tool. Refer to [Menuconfig](#) or [how to configure kernel](#) and select:

```
[*] Device Drivers --->
  [*] Character devices --->
    [*] Hardware Random Number Generator Core support --->
      [*] STMicroelectronics STM32 random number generator
```

### 4.2 Device tree configuration

DT configuration can be done thanks to the [STM32CubeMX](#).

A detailed device tree configuration is described in [RNG device tree configuration](#).





## 5 How to use the framework

The framework provides external interfaces from userland : [How to control RNG](#).

### 5.1 How to use from char device

The community tool for using Hardware random framework is `rng_tools`<sup>[3]</sup> which provides a complete set of utilities related to random number generators:

- **rngd**: runs a background daemon that opens `/dev/hwrng` file (default) to connect and retrieve random numbers.
- **rngtest**: runs different tests that check the entropy and verify the compliance regarding FIPS 140-2 standard.

### 5.2 How to use from sysfs

Available devices compatible with Hardware framework can be listed using `sysfs` commands:

```
Board $> cat /sys/class/misc/hw_random/rng_available  
stm32-rng
```

The selected device is shown here:

```
Board $> cat /sys/class/misc/hw_random/rng_current  
stm32-rng
```

To select a different device:

```
Board $> echo "stm32-rng"> /sys/class/misc/hw_random/rng_current
```



---

## 6 How to trace and debug the framework

---

Light information on the framework can be accessed by using `sysfs`.

By default, the framework does not provide any specific debug output or dynamic debugging tool.



---

## 7 Source code location

---

Hardware random drivers and framework are available here<sup>[4]</sup>.



---

## 8 To go further

---

Code examples are directly available from [rng-tools<sup>\[3\]</sup>](#) github.



## 9 References

- <https://bootlin.com/doc/legacy/accessing-hardware/accessing-hardware.pdf>
- Documentation/admin-guide/hw\_random.rst
- 3.03.1 Rng\_tools source code
- drivers/char/hw\_random , Hw\_random sources

Linux<sup>®</sup> is a registered trademark of Linus Torvalds.

Random Number Generator

System File System (See <https://en.wikipedia.org/wiki/Sysfs> for more details)

Application programming interface

Device Tree

Stable: 03.02.2020 - 08:42 / Revision: 03.02.2020 - 08:27

This article gives information about the Linux<sup>®</sup> hardware random framework.

### Contents

1 Article Purpose .....	14
2 Framework purpose .....	15
3 System overview .....	16
3.1 Component description .....	16
3.2 API description .....	17
4 Configuration .....	18
4.1 Kernel configuration .....	18
4.2 Device tree configuration .....	18
5 How to use the framework .....	19
5.1 How to use from char device .....	19
5.2 How to use from sysfs .....	19
6 How to trace and debug the framework .....	20
7 Source code location .....	21
8 To go further .....	22
9 References .....	23



---

## 1 Article Purpose

---

This article gives information about the hardware random (HWRNG) framework.



---

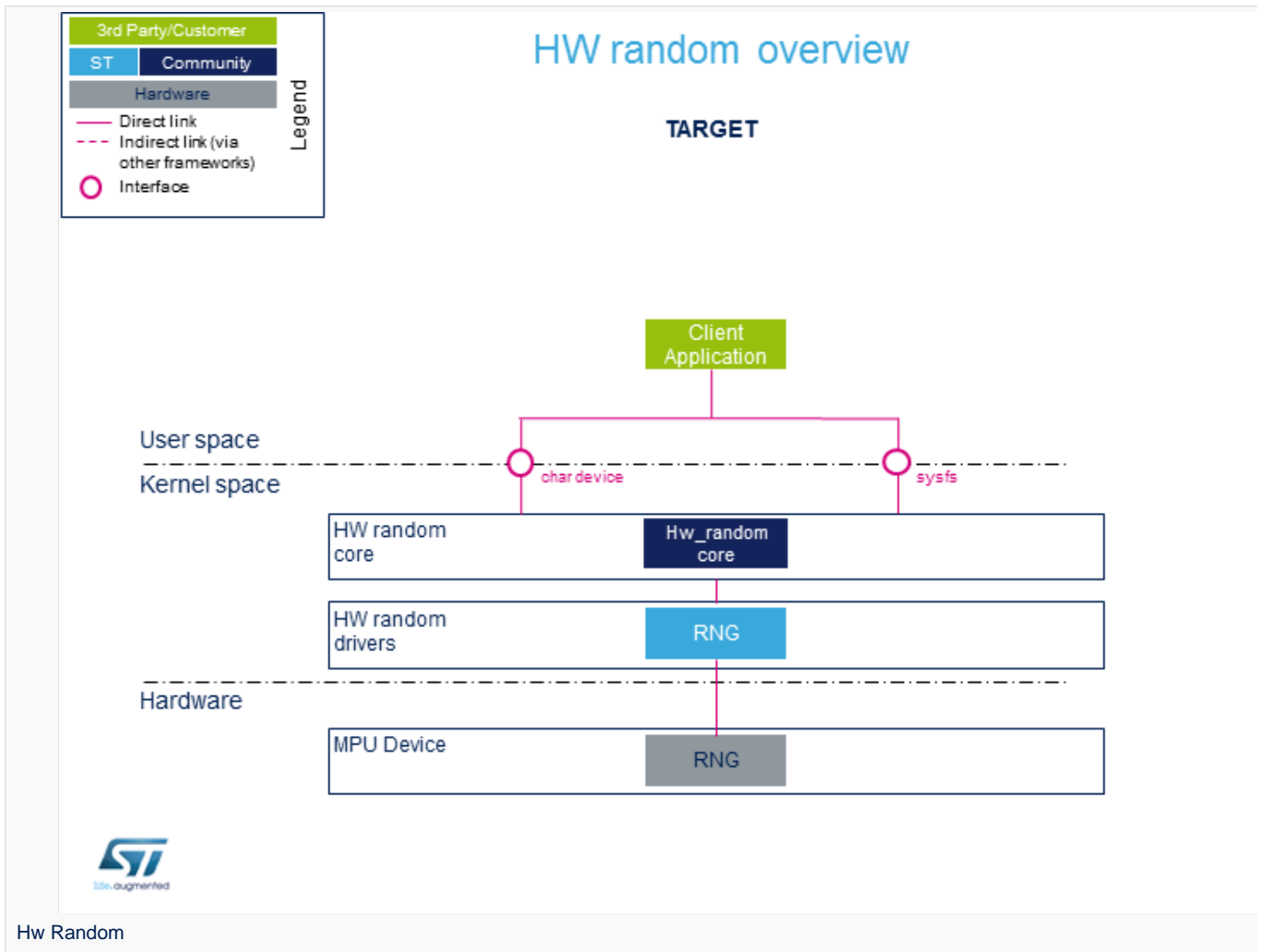
## 2 Framework purpose

---

The Hardware random framework is integrated in the kernel. It provides access to RNG peripherals and focuses on supporting the hardware number generator.

### 3 System overview

The HW random framework allows retrieving random numbers in userland.



#### 3.1 Component description

- **HW random core** (Kernel space)

Generic interface in kernel space. This layer is in charge of creating the character device (char device) and sysfs to access hw\_random.

- **RNG** (Kernel space)

Hardware random Linux<sup>®</sup> drivers handling the HW blocks.

- **RNG** (Hardware)

HW blocks handling the RNG peripheral.





### 3.2 API description

The Hardware random framework uses char device API<sup>[1]</sup> ioctl operations. For additional information, refer to:

- sysfs interface.
- Kernel Documentation directory<sup>[2]</sup>



---

## 4 Configuration

---

### 4.1 Kernel configuration

The Hardware random support is activated by default in ST deliveries. No specific configuration is required apart from enabling or disabling peripheral support using Linux<sup>®</sup> Menuconfig tool. Refer to [Menuconfig](#) or [how to configure kernel](#) and select:

```
[*] Device Drivers --->
  [*] Character devices --->
    [*] Hardware Random Number Generator Core support --->
      [*] STMicroelectronics STM32 random number generator
```

### 4.2 Device tree configuration

DT configuration can be done thanks to the [STM32CubeMX](#).

A detailed device tree configuration is described in [RNG device tree configuration](#).



## 5 How to use the framework

The framework provides external interfaces from userland : [How to control RNG](#).

### 5.1 How to use from char device

The community tool for using Hardware random framework is [rng\\_tools<sup>\[3\]</sup>](#) which provides a complete set of utilities related to random number generators:

- **rngd**: runs a background daemon that opens `/dev/hwrng` file (default) to connect and retrieve random numbers.
- **rngtest**: runs different tests that check the entropy and verify the compliance regarding FIPS 140-2 standard.

### 5.2 How to use from sysfs

Available devices compatible with Hardware framework can be listed using sysfs commands:

```
Board $> cat /sys/class/misc/hw_random/rng_available  
stm32-rng
```

The selected device is shown here:

```
Board $> cat /sys/class/misc/hw_random/rng_current  
stm32-rng
```

To select a different device:

```
Board $> echo "stm32-rng"> /sys/class/misc/hw_random/rng_current
```



---

## 6 How to trace and debug the framework

---

Light information on the framework can be accessed by using `sysfs`.

By default, the framework does not provide any specific debug output or dynamic debugging tool.



---

## 7 Source code location

---

Hardware random drivers and framework are available here<sup>[4]</sup>.



---

## 8 To go further

---

Code examples are directly available from [rng-tools<sup>\[3\]</sup>](#) github.



## 9 References

- <https://bootlin.com/doc/legacy/accessing-hardware/accessing-hardware.pdf>
- Documentation/admin-guide/hw\_random.rst
- 3.03.1 Rng\_tools source code
- drivers/char/hw\_random , Hw\_random sources

Linux<sup>®</sup> is a registered trademark of Linus Torvalds.

Random Number Generator

System File System (See <https://en.wikipedia.org/wiki/Sysfs> for more details)

Application programming interface

Device Tree

Stable: 31.03.2021 - 08:47 / Revision: 26.03.2021 - 08:44

This article gives information about the Linux<sup>®</sup> hardware random framework.

### Contents

1 Article Purpose .....	24
2 Framework purpose .....	25
3 System overview .....	26
3.1 Component description .....	26
3.2 API description .....	27
4 Configuration .....	28
4.1 Kernel configuration .....	28
4.2 Device tree configuration .....	28
5 How to use the framework .....	29
5.1 How to use from char device .....	29
5.2 How to use from sysfs .....	29
6 How to trace and debug the framework .....	30
7 Source code location .....	31
8 To go further .....	32
9 References .....	33



---

## 1 Article Purpose

---

This article gives information about the hardware random (HWRNG) framework.





---

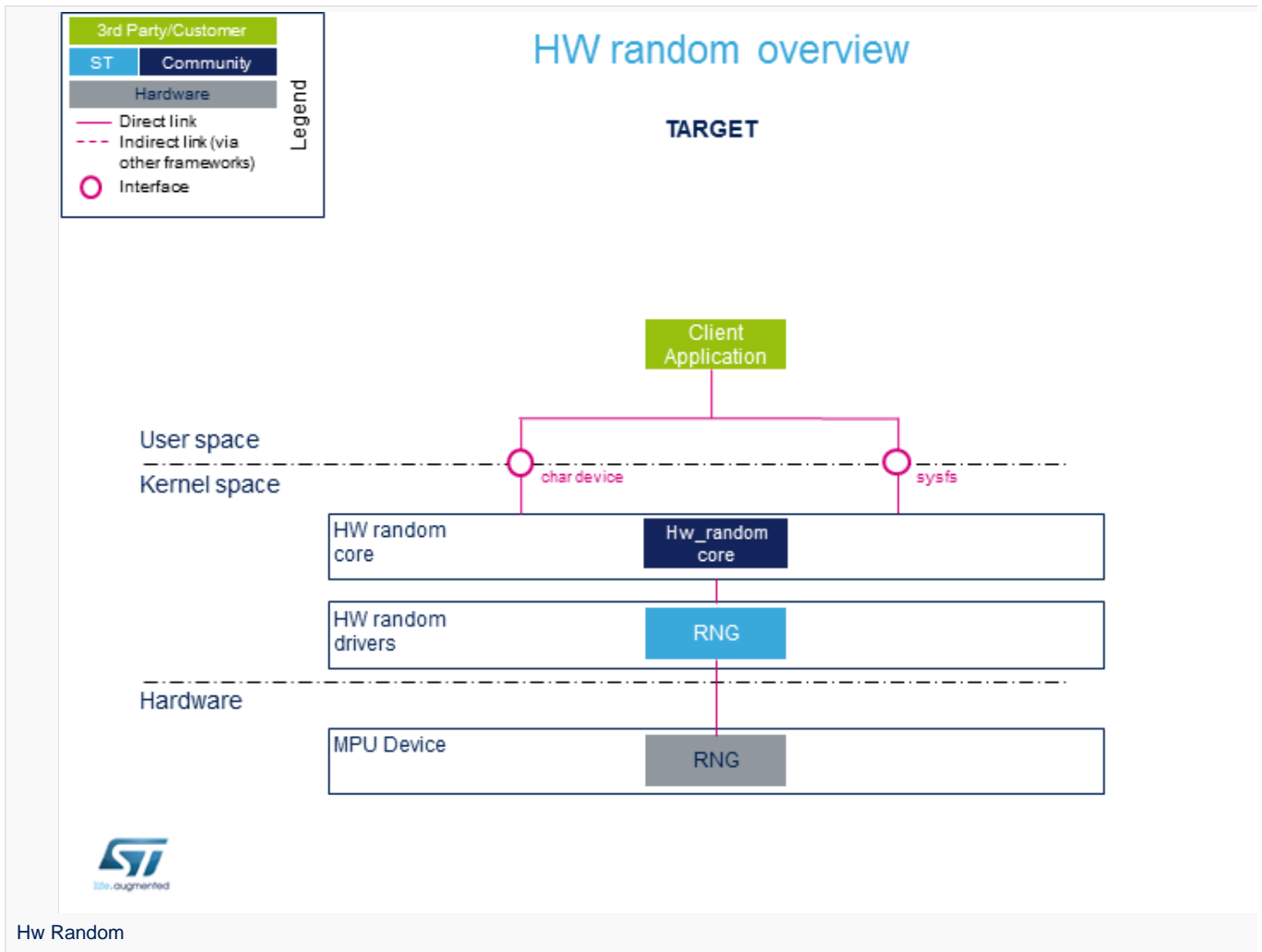
## 2 Framework purpose

---

The Hardware random framework is integrated in the kernel. It provides access to RNG peripherals and focuses on supporting the hardware number generator.

### 3 System overview

The HW random framework allows retrieving random numbers in userland.



#### 3.1 Component description

- **HW random core** (Kernel space)

Generic interface in kernel space. This layer is in charge of creating the character device (char device) and sysfs to access hw\_random.

- **RNG** (Kernel space)

Hardware random Linux<sup>®</sup> drivers handling the HW blocks.

- **RNG** (Hardware)

HW blocks handling the RNG peripheral.



## 3.2 API description

The Hardware random framework uses char device API<sup>[1]</sup> ioctl operations. For additional information, refer to:

- sysfs interface.
- Kernel Documentation directory<sup>[2]</sup>



---

## 4 Configuration

---

### 4.1 Kernel configuration

The Hardware random support is activated by default in ST deliveries. No specific configuration is required apart from enabling or disabling peripheral support using Linux<sup>®</sup> Menuconfig tool. Refer to [Menuconfig](#) or [how to configure kernel](#) and select:

```
[*] Device Drivers --->
  [*] Character devices --->
    [*] Hardware Random Number Generator Core support --->
      [*] STMicroelectronics STM32 random number generator
```

### 4.2 Device tree configuration

DT configuration can be done thanks to the [STM32CubeMX](#).

A detailed device tree configuration is described in [RNG device tree configuration](#).



## 5 How to use the framework

The framework provides external interfaces from userland : [How to control RNG](#).

### 5.1 How to use from char device

The community tool for using Hardware random framework is `rng_tools`<sup>[3]</sup> which provides a complete set of utilities related to random number generators:

- **rngd**: runs a background daemon that opens `/dev/hwrng` file (default) to connect and retrieve random numbers.
- **rngtest**: runs different tests that check the entropy and verify the compliance regarding FIPS 140-2 standard.

### 5.2 How to use from sysfs

Available devices compatible with Hardware framework can be listed using `sysfs` commands:

```
Board $> cat /sys/class/misc/hw_random/rng_available  
stm32-rng
```

The selected device is shown here:

```
Board $> cat /sys/class/misc/hw_random/rng_current  
stm32-rng
```

To select a different device:

```
Board $> echo "stm32-rng"> /sys/class/misc/hw_random/rng_current
```



---

## 6 How to trace and debug the framework

---

Light information on the framework can be accessed by using `sysfs`.

By default, the framework does not provide any specific debug output or dynamic debugging tool.



---

## 7 Source code location

---

Hardware random drivers and framework are available here<sup>[4]</sup>.



---

## 8 To go further

---

Code examples are directly available from [rng-tools<sup>\[3\]</sup>](#) github.





## 9 References

- <https://bootlin.com/doc/legacy/accessing-hardware/accessing-hardware.pdf>
- Documentation/admin-guide/hw\_random.rst
- 3.03.1 Rng\_tools source code
- drivers/char/hw\_random , Hw\_random sources

Linux<sup>®</sup> is a registered trademark of Linus Torvalds.

Random Number Generator

System File System (See <https://en.wikipedia.org/wiki/Sysfs> for more details)

Application programming interface

Device Tree

Stable: 13.05.2020 - 08:40 / Revision: 13.05.2020 - 08:39

This article gives information about the Linux<sup>®</sup> hardware random framework.

### Contents

1 Article Purpose .....	34
2 Framework purpose .....	35
3 System overview .....	36
3.1 Component description .....	36
3.2 API description .....	37
4 Configuration .....	38
4.1 Kernel configuration .....	38
4.2 Device tree configuration .....	38
5 How to use the framework .....	39
5.1 How to use from char device .....	39
5.2 How to use from sysfs .....	39
6 How to trace and debug the framework .....	40
7 Source code location .....	41
8 To go further .....	42
9 References .....	43



---

## 1 Article Purpose

---

This article gives information about the hardware random (HWRNG) framework.



---

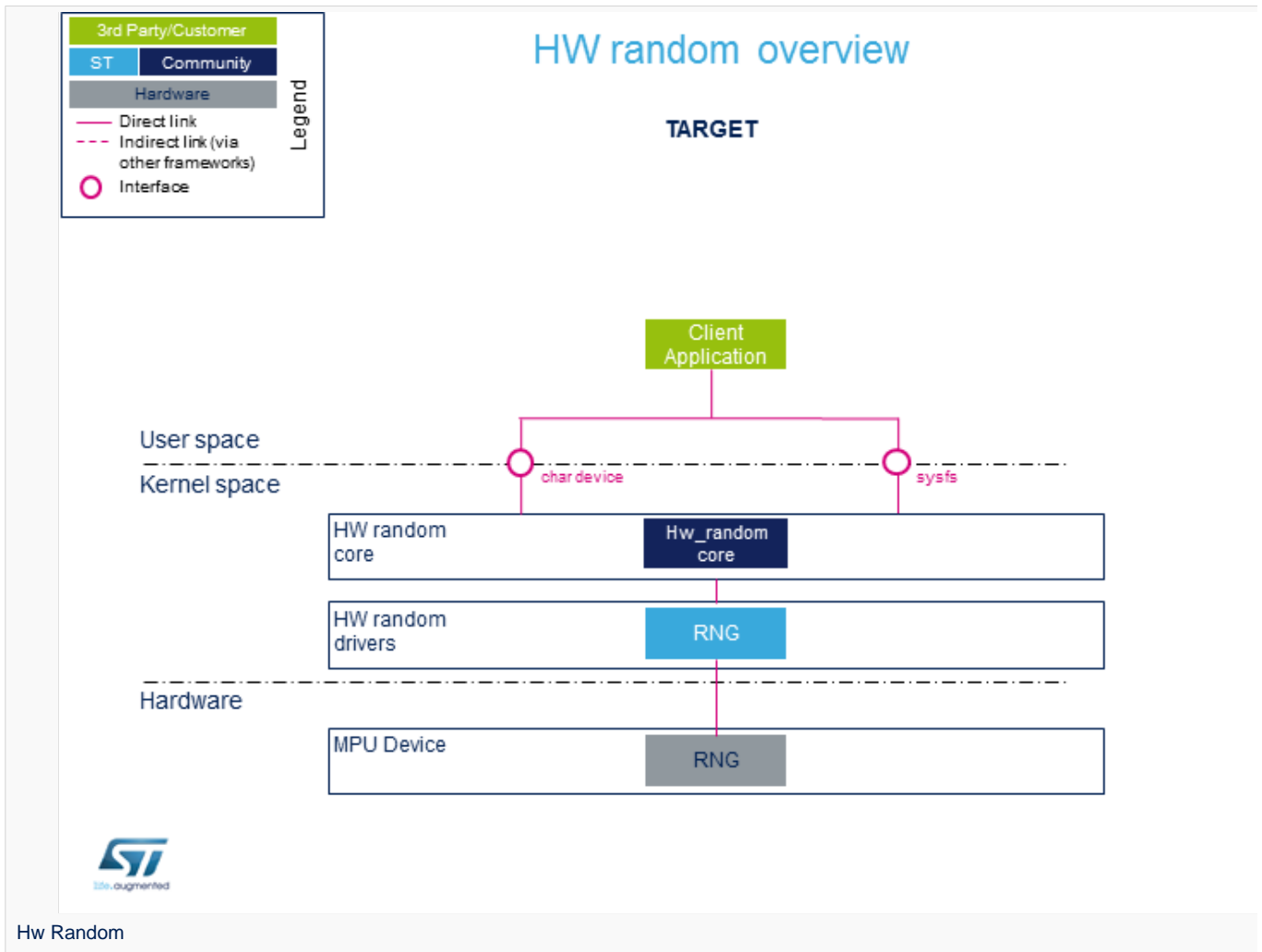
## 2 Framework purpose

---

The Hardware random framework is integrated in the kernel. It provides access to RNG peripherals and focuses on supporting the hardware number generator.

### 3 System overview

The HW random framework allows retrieving random numbers in userland.



#### 3.1 Component description

- **HW random core** (Kernel space)

Generic interface in kernel space. This layer is in charge of creating the character device (char device) and sysfs to access hw\_random.

- **RNG** (Kernel space)

Hardware random Linux<sup>®</sup> drivers handling the HW blocks.

- **RNG** (Hardware)

HW blocks handling the RNG peripheral.



## 3.2 API description

The Hardware random framework uses char device API<sup>[1]</sup> ioctl operations. For additional information, refer to:

- sysfs interface.
- Kernel Documentation directory<sup>[2]</sup>



## 4 Configuration

### 4.1 Kernel configuration

The Hardware random support is activated by default in ST deliveries. No specific configuration is required apart from enabling or disabling peripheral support using Linux<sup>®</sup> Menuconfig tool. Refer to [Menuconfig](#) or [how to configure kernel](#) and select:

```
[*] Device Drivers --->
  [*] Character devices --->
    [*] Hardware Random Number Generator Core support --->
      [*] STMicroelectronics STM32 random number generator
```

### 4.2 Device tree configuration

DT configuration can be done thanks to the [STM32CubeMX](#).

A detailed device tree configuration is described in [RNG device tree configuration](#).



## 5 How to use the framework

The framework provides external interfaces from userland : [How to control RNG](#).

### 5.1 How to use from char device

The community tool for using Hardware random framework is `rng_tools`<sup>[3]</sup> which provides a complete set of utilities related to random number generators:

- **rngd**: runs a background daemon that opens `/dev/hwrng` file (default) to connect and retrieve random numbers.
- **rngtest**: runs different tests that check the entropy and verify the compliance regarding FIPS 140-2 standard.

### 5.2 How to use from sysfs

Available devices compatible with Hardware framework can be listed using sysfs commands:

```
Board $> cat /sys/class/misc/hw_random/rng_available  
stm32-rng
```

The selected device is shown here:

```
Board $> cat /sys/class/misc/hw_random/rng_current  
stm32-rng
```

To select a different device:

```
Board $> echo "stm32-rng"> /sys/class/misc/hw_random/rng_current
```



---

## 6 How to trace and debug the framework

---

Light information on the framework can be accessed by using `sysfs`.

By default, the framework does not provide any specific debug output or dynamic debugging tool.





---

## 7 Source code location

---

Hardware random drivers and framework are available here<sup>[4]</sup>.



---

## 8 To go further

---

Code examples are directly available from [rng-tools<sup>\[3\]</sup>](#) github.



## 9 References

- <https://bootlin.com/doc/legacy/accessing-hardware/accessing-hardware.pdf>
- Documentation/admin-guide/hw\_random.rst
- 3.03.1 Rng\_tools source code
- drivers/char/hw\_random , Hw\_random sources

Linux<sup>®</sup> is a registered trademark of Linus Torvalds.

Random Number Generator

System File System (See <https://en.wikipedia.org/wiki/Sysfs> for more details)

Application programming interface

Device Tree

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

This article gives information about the Linux<sup>®</sup> hardware random framework.

### Contents

1 Article Purpose .....	44
2 Framework purpose .....	45
3 System overview .....	46
3.1 Component description .....	46
3.2 API description .....	47
4 Configuration .....	48
4.1 Kernel configuration .....	48
4.2 Device tree configuration .....	48
5 How to use the framework .....	49
5.1 How to use from char device .....	49
5.2 How to use from sysfs .....	49
6 How to trace and debug the framework .....	50
7 Source code location .....	51
8 To go further .....	52
9 References .....	53



---

## 1 Article Purpose

---

This article gives information about the hardware random (HWRNG) framework.



---

## 2 Framework purpose

---

The Hardware random framework is integrated in the kernel. It provides access to RNG peripherals and focuses on supporting the hardware number generator.





## 3.2 API description

The Hardware random framework uses char device API<sup>[1]</sup> ioctl operations. For additional information, refer to:

- sysfs interface.
- Kernel Documentation directory<sup>[2]</sup>



---

## 4 Configuration

---

### 4.1 Kernel configuration

The Hardware random support is activated by default in ST deliveries. No specific configuration is required apart from enabling or disabling peripheral support using Linux<sup>®</sup> Menuconfig tool. Refer to [Menuconfig](#) or [how to configure kernel](#) and select:

```
[*] Device Drivers --->
  [*] Character devices --->
    [*] Hardware Random Number Generator Core support --->
      [*] STMicroelectronics STM32 random number generator
```

### 4.2 Device tree configuration

DT configuration can be done thanks to the [STM32CubeMX](#).

A detailed device tree configuration is described in [RNG device tree configuration](#).





## 5 How to use the framework

The framework provides external interfaces from userland : [How to control RNG](#).

### 5.1 How to use from char device

The community tool for using Hardware random framework is [rng\\_tools<sup>\[3\]</sup>](#) which provides a complete set of utilities related to random number generators:

- **rngd**: runs a background daemon that opens `/dev/hwrng` file (default) to connect and retrieve random numbers.
- **rngtest**: runs different tests that check the entropy and verify the compliance regarding FIPS 140-2 standard.

### 5.2 How to use from sysfs

Available devices compatible with Hardware framework can be listed using sysfs commands:

```
Board $> cat /sys/class/misc/hw_random/rng_available  
stm32-rng
```

The selected device is shown here:

```
Board $> cat /sys/class/misc/hw_random/rng_current  
stm32-rng
```

To select a different device:

```
Board $> echo "stm32-rng"> /sys/class/misc/hw_random/rng_current
```



---

## 6 How to trace and debug the framework

---

Light information on the framework can be accessed by using `sysfs`.

By default, the framework does not provide any specific debug output or dynamic debugging tool.



---

## 7 Source code location

---

Hardware random drivers and framework are available here<sup>[4]</sup>.



---

## 8 To go further

---

Code examples are directly available from [rng-tools<sup>\[3\]</sup>](#) github.



---

## 9 References

---

- <https://bootlin.com/doc/legacy/accessing-hardware/accessing-hardware.pdf>
- Documentation/admin-guide/hw\_random.rst
- 3.03.1 Rng\_tools source code
- drivers/char/hw\_random , Hw\_random sources

Linux<sup>®</sup> is a registered trademark of Linus Torvalds.

Random Number Generator

System File System (See <https://en.wikipedia.org/wiki/Sysfs> for more details)

Application programming interface

Device Tree