RNG device tree configuration

Contents

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

Stable: 13.05.2020 - 08:43 / Revision: 13.05.2020 - 08:56

A quality version of this page, approved on *13 May 2020*, was based off this revision.

Contents

1 Article purpose

This article explains how to configure the **RNG** internal peripheral when it is assigned to the Linux®OS. In that case, it is controlled by the Hardware random framework.

The configuration is performed using the device tree mechanism that provides a hardware description of the RNG peripheral, used by the STM32 RNG Linux driver.

If the peripheral is assigned to another execution context, refer to How to assign an internal peripheral to a runtime context article for guidelines on peripheral assignment and configuration.

2 DT bindings documentation

The *RNG* is represented by the *STM32 RNG device tree bindings*[1]

3 DT configuration

This hardware description is a combination of the **STM32 microprocessor** device tree files (*.dtsi* extension) and **board** device tree files (*.dts* extension). See the Device tree for an explanation of the device tree file split.

STM32CubeMX can be used to generate the board device tree. Refer to How to configure the DT using STM32CubeMX for more details.

3.1 DT configuration (STM32 level)

The RNG node is declared in stm32mp151.dtsi[2]. It describes the hardware register address, clock and reset.

```
 rng1: rng@54003000 {                                          Comments
         compatible = "st,stm32-rng";
         reg = <0x54003000 0x400>;                              --> Register location
and length
         clocks = <&scmi0_clk CK_SCMI0_RNG1>;
         resets = <&scmi0_reset RST_SCMI0_RNG1>;
         status = "disabled";
 };
```

Warning

This device tree part is related to STM32 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

This part is used to enable the RNG used on a board which is done by setting the **status** property to **okay**.

A clock-error-detect property is available depending the clock choosen for entropy. It can be enabled to manage the clock detection.

3.3 DT configuration examples

```
&rng1 {
        status = "okay";
        clock-error-detect;
};
```

4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.
The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

Please refer to the following links for additional information:

- Device tree bindings
- STM32MP151 device tree

Operating System

Random Number Generator

Device Tree

Stable: 05.11.2021 - 11:08 / Revision: 05.11.2021 - 11:05

Contents

1 Article purpose

This article explains how to configure the **RNG** internal peripheral when it is assigned to the Linux®OS. In that case, it is controlled by the Hardware random framework.

The configuration is performed using the device tree mechanism that provides a hardware description of the RNG peripheral, used by the STM32 RNG Linux driver.

If the peripheral is assigned to another execution context, refer to How to assign an internal peripheral to a runtime context article for guidelines on peripheral assignment and configuration.

2 DT bindings documentation

The *RNG* is represented by the *STM32 RNG device tree bindings*[1]

3 DT configuration

This hardware description is a combination of the **STM32 microprocessor** device tree files (*.dtsi* extension) and **board** device tree files (*.dts* extension). See the Device tree for an explanation of the device tree file split.

STM32CubeMX can be used to generate the board device tree. Refer to How to configure the DT using STM32CubeMX for more details.

3.1 DT configuration (STM32 level)

The RNG node is declared in stm32mp151.dtsi[2]. It describes the hardware register address, clock and reset.

```
 rng1: rng@54003000 {                                          Comments
         compatible = "st,stm32-rng";
         reg = <0x54003000 0x400>;                              --> Register location
and length
         clocks = <&scmi0_clk CK_SCMI0_RNG1>;
         resets = <&scmi0_reset RST_SCMI0_RNG1>;
         status = "disabled";
 };
```

Warning

This device tree part is related to STM32 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

This part is used to enable the RNG used on a board which is done by setting the **status** property to **okay**.

A clock-error-detect property is available depending the clock choosen for entropy. It can be enabled to manage the clock detection.

3.3 DT configuration examples

```
&rng1 {
        status = "okay";
        clock-error-detect;
};
```

4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.
The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

Please refer to the following links for additional information:

- Device tree bindings
- STM32MP151 device tree

Operating System

Random Number Generator

Device Tree

Stable: 17.02.2021 - 19:55 / Revision: 17.02.2021 - 19:55

Contents

1 Article purpose

This article explains how to configure the **RNG** internal peripheral when it is assigned to the Linux®OS. In that case, it is controlled by the Hardware random framework.

The configuration is performed using the device tree mechanism that provides a hardware description of the RNG peripheral, used by the STM32 RNG Linux driver.

If the peripheral is assigned to another execution context, refer to How to assign an internal peripheral to a runtime context article for guidelines on peripheral assignment and configuration.

2 DT bindings documentation

The *RNG* is represented by the *STM32 RNG device tree bindings*[1]

3 DT configuration

This hardware description is a combination of the **STM32 microprocessor** device tree files (*.dtsi* extension) and **board** device tree files (*.dts* extension). See the Device tree for an explanation of the device tree file split.

STM32CubeMX can be used to generate the board device tree. Refer to How to configure the DT using STM32CubeMX for more details.

3.1 DT configuration (STM32 level)

The RNG node is declared in stm32mp151.dtsi[2]. It describes the hardware register address, clock and reset.

```
 rng1: rng@54003000 {                                          Comments
         compatible = "st,stm32-rng";
         reg = <0x54003000 0x400>;                              --> Register location
and length
         clocks = <&scmi0_clk CK_SCMI0_RNG1>;
         resets = <&scmi0_reset RST_SCMI0_RNG1>;
         status = "disabled";
 };
```

Warning

This device tree part is related to STM32 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

This part is used to enable the RNG used on a board which is done by setting the **status** property to **okay**.

A clock-error-detect property is available depending the clock choosen for entropy. It can be enabled to manage the clock detection.

3.3 DT configuration examples

```
&rng1 {
        status = "okay";
        clock-error-detect;
};
```

4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.
The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

Please refer to the following links for additional information:

- Device tree bindings
- STM32MP151 device tree

Linux® is a registered trademark of Linus Torvalds.

Operating System

Random Number Generator

Device Tree

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

Contents

1 Article purpose

This article explains how to configure the **RNG** internal peripheral when it is assigned to the Linux®OS. In that case, it is controlled by the Hardware random framework.

The configuration is performed using the device tree mechanism that provides a hardware description of the RNG peripheral, used by the STM32 RNG Linux driver.

If the peripheral is assigned to another execution context, refer to How to assign an internal peripheral to a runtime context article for guidelines on peripheral assignment and configuration.

2 DT bindings documentation

The *RNG* is represented by the *STM32 RNG device tree bindings*[1]

3 DT configuration

This hardware description is a combination of the **STM32 microprocessor** device tree files (*.dtsi* extension) and **board** device tree files (*.dts* extension). See the Device tree for an explanation of the device tree file split.

STM32CubeMX can be used to generate the board device tree. Refer to How to configure the DT using STM32CubeMX for more details.

3.1 DT configuration (STM32 level)

The RNG node is declared in stm32mp151.dtsi[2]. It describes the hardware register address, clock and reset.

```
 rng1: rng@54003000 {                                          Comments
         compatible = "st,stm32-rng";
         reg = <0x54003000 0x400>;                              --> Register location
and length
         clocks = <&scmi0_clk CK_SCMI0_RNG1>;
         resets = <&scmi0_reset RST_SCMI0_RNG1>;
         status = "disabled";
 };
```

Warning

This device tree part is related to STM32 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

This part is used to enable the RNG used on a board which is done by setting the **status** property to **okay**.

A clock-error-detect property is available depending the clock choosen for entropy. It can be enabled to manage the clock detection.

3.3 DT configuration examples

```
&rng1 {
        status = "okay";
        clock-error-detect;
};
```

4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.
The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

Please refer to the following links for additional information:

- Device tree bindings
- STM32MP151 device tree

Linux® is a registered trademark of Linus Torvalds.

Operating System

Random Number Generator

Device Tree

Stable: 12.02.2020 - 16:50 / Revision: 12.02.2020 - 16:49

Contents

1 Article purpose

This article explains how to configure the **RNG** internal peripheral when it is assigned to the Linux®OS. In that case, it is controlled by the Hardware random framework.

The configuration is performed using the device tree mechanism that provides a hardware description of the RNG peripheral, used by the STM32 RNG Linux driver.

If the peripheral is assigned to another execution context, refer to How to assign an internal peripheral to a runtime context article for guidelines on peripheral assignment and configuration.

2 DT bindings documentation

The *RNG* is represented by the *STM32 RNG device tree bindings*[1]

3 DT configuration

This hardware description is a combination of the **STM32 microprocessor** device tree files (*.dtsi* extension) and **board** device tree files (*.dts* extension). See the Device tree for an explanation of the device tree file split.

STM32CubeMX can be used to generate the board device tree. Refer to How to configure the DT using STM32CubeMX for more details.

3.1 DT configuration (STM32 level)

The RNG node is declared in stm32mp151.dtsi[2]. It describes the hardware register address, clock and reset.

```
 rng1: rng@54003000 {                                          Comments
         compatible = "st,stm32-rng";
         reg = <0x54003000 0x400>;                              --> Register location
and length
         clocks = <&scmi0_clk CK_SCMI0_RNG1>;
         resets = <&scmi0_reset RST_SCMI0_RNG1>;
         status = "disabled";
 };
```

Warning

This device tree part is related to STM32 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

This part is used to enable the RNG used on a board which is done by setting the **status** property to **okay**.

A clock-error-detect property is available depending the clock choosen for entropy. It can be enabled to manage the clock detection.

3.3 DT configuration examples

```
&rng1 {
        status = "okay";
        clock-error-detect;
};
```

4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

Please refer to the following links for additional information:

- Device tree bindings
- STM32MP151 device tree

Linux® is a registered trademark of Linus Torvalds.

Operating System

Random Number Generator

Device Tree

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

Contents

1 Article purpose

This article explains how to configure the **RNG** internal peripheral when it is assigned to the Linux®OS. In that case, it is controlled by the Hardware random framework.

The configuration is performed using the device tree mechanism that provides a hardware description of the RNG peripheral, used by the STM32 RNG Linux driver.

If the peripheral is assigned to another execution context, refer to How to assign an internal peripheral to a runtime context article for guidelines on peripheral assignment and configuration.

2 DT bindings documentation

The *RNG* is represented by the *STM32 RNG device tree bindings*[1]

3 DT configuration

This hardware description is a combination of the **STM32 microprocessor** device tree files (*.dtsi* extension) and **board** device tree files (*.dts* extension). See the Device tree for an explanation of the device tree file split.

STM32CubeMX can be used to generate the board device tree. Refer to How to configure the DT using STM32CubeMX for more details.

3.1 DT configuration (STM32 level)

The RNG node is declared in stm32mp151.dtsi[2]. It describes the hardware register address, clock and reset.

```
 rng1: rng@54003000 {                                          Comments
         compatible = "st,stm32-rng";
         reg = <0x54003000 0x400>;                              --> Register location
and length
         clocks = <&scmi0_clk CK_SCMI0_RNG1>;
         resets = <&scmi0_reset RST_SCMI0_RNG1>;
         status = "disabled";
 };
```

Warning

This device tree part is related to STM32 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

This part is used to enable the RNG used on a board which is done by setting the **status** property to **okay**.

A clock-error-detect property is available depending the clock choosen for entropy. It can be enabled to manage the clock detection.

3.3 DT configuration examples

```
&rng1 {
        status = "okay";
        clock-error-detect;
};
```

4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.
The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

Please refer to the following links for additional information:

- Device tree bindings
- STM32MP151 device tree

Linux® is a registered trademark of Linus Torvalds.

Operating System

Random Number Generator

Device Tree