Clock device tree configuration

Contents

Clock device tree configuration

Stable: 19.02.2019 - 14:50 / Revision: 06.02.2019 - 12:06

A quality version of this page, accepted on *19 February 2019*, was based off this revision.

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

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

The configuration is performed using the device tree mechanism that provides a hardware description of the RCC peripheral, used by the clk-stm32mp1 Linux driver and by the Common Clock framework.

2 DT bindings documentation

The RCC is a multifunction device.

Each function is represented by a separate binding document:

- generic DT bindings[1] used by the Common Clock framework.
- vendor clock DT bindings[2] used by the clk-stm32mp1 driver: this binding document explains how to write device tree files for clocks.

3 DT configuration

3.1 DT configuration (STM32 level)

The STM32MP1 Clock node is located in the *stm32mp157c.dtsi*[3]. See Device tree for more explanations.

3.1.1 clocks node

These clocks have a fixed frequency (generally they are oscillators)

```
        clocks {
                clk_hse: clk-hse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <24000000>;
                };

                clk_hsi: clk-hsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <64000000>;
                };

                clk_lse: clk-lse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32768>;
                };

                clk_lsi: clk-lsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32000>;
                };

                clk_csi: clk-csi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <4000000>;
                };

                ...
        };
```

3.1.2 STM32MP1 Clock node

We need to specify the number of cells in a clock specifier.
For the STM32MP1 this number shoud be 1 and is configured via 'clock-cells' property in rcc node.

```
rcc: rcc@50000000 {
         compatible = "st,stm32mp1-rcc", "syscon";
        #clock-cells = <1>;
        #reset-cells = <1>;
        reg = <0x50000000 0x1000>;
        ...
};
```

This device tree part is related to STM32MP1 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

If a Linux driver needs a clock, it has to be added in its DT node:
clocks = <phandle> : List of phandle and clock specifier pairs, one pair for each clock input to the device. Note: if the clock provider specifies '0' for #clock-cells, then only the phandle portion of the pair will appear.

- Example:

```
usart3: serial@4000f000 {
        compatible = "st,stm32h7-usart";
        reg = <0x4000f000 0x400>;
        interrupt-names = "event", "wakeup";
        interrupts-extended = <&intc GIC_SPI 39 IRQ_TYPE_LEVEL_HIGH>,
                              <&exti 28 1>;
        clocks = <&rcc USART3_K>;
        wakeup-source;
        power-domains = <&pd_core>;
        status = "disabled";
};
```

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:

- Documentation/devicetree/bindings/clock/clock-bindings.txt , Clock device tree bindings
- Documentation/devicetree/bindings/clock/st,stm32mp1-rcc.txt , STM32MP1 clock device tree bindings
- stm32mp157c.dtsi STM32MP157C device tree file

Operating System

Reset and Clock Control

Device Tree

Generic Interrupt Controller

Serial Peripheral Interface

Clock device tree configuration

Stable: 04.02.2020 - 15:40 / Revision: 04.02.2020 - 15:27

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

This article explains how to configure the RCC internal peripheral when it is assigned to the Linux®OS. In that case, it is controlled by the Common Clock framework.

The configuration is performed using the device tree mechanism that provides a hardware description of the RCC peripheral, used by the clk-stm32mp1 Linux driver and by the Common Clock framework.

2 DT bindings documentation

The RCC is a multifunction device.

Each function is represented by a separate binding document:

- generic DT bindings[1] used by the Common Clock framework.
- vendor clock DT bindings[2] used by the clk-stm32mp1 driver: this binding document explains how to write device tree files for clocks.

3 DT configuration

3.1 DT configuration (STM32 level)

The STM32MP1 Clock node is located in the *stm32mp157c.dtsi*[3]. See Device tree for more explanations.

3.1.1 clocks node

These clocks have a fixed frequency (generally they are oscillators)

```
        clocks {
                clk_hse: clk-hse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <24000000>;
                };

                clk_hsi: clk-hsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <64000000>;
                };

                clk_lse: clk-lse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32768>;
                };

                clk_lsi: clk-lsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32000>;
                };

                clk_csi: clk-csi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <4000000>;
                };

                ...
        };
```

3.1.2 STM32MP1 Clock node

We need to specify the number of cells in a clock specifier.
For the STM32MP1 this number shoud be 1 and is configured via 'clock-cells' property in rcc node.

```
rcc: rcc@50000000 {
         compatible = "st,stm32mp1-rcc", "syscon";
        #clock-cells = <1>;
        #reset-cells = <1>;
        reg = <0x50000000 0x1000>;
        ...
};
```

This device tree part is related to STM32MP1 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

If a Linux driver needs a clock, it has to be added in its DT node:
clocks = <phandle> : List of phandle and clock specifier pairs, one pair for each clock input to the device. Note: if the clock provider specifies '0' for #clock-cells, then only the phandle portion of the pair will appear.

- Example:

```
usart3: serial@4000f000 {
        compatible = "st,stm32h7-usart";
        reg = <0x4000f000 0x400>;
        interrupt-names = "event", "wakeup";
        interrupts-extended = <&intc GIC_SPI 39 IRQ_TYPE_LEVEL_HIGH>,
                              <&exti 28 1>;
        clocks = <&rcc USART3_K>;
        wakeup-source;
        power-domains = <&pd_core>;
        status = "disabled";
};
```

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:

- Documentation/devicetree/bindings/clock/clock-bindings.txt , Clock device tree bindings
- Documentation/devicetree/bindings/clock/st,stm32mp1-rcc.txt , STM32MP1 clock device tree bindings
- stm32mp157c.dtsi STM32MP157C device tree file

Operating System

Reset and Clock Control

Device Tree

Generic Interrupt Controller

Serial Peripheral Interface

Clock device tree configuration

Stable: 15.10.2019 - 11:58 / Revision: 15.10.2019 - 11:57

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

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

The configuration is performed using the device tree mechanism that provides a hardware description of the RCC peripheral, used by the clk-stm32mp1 Linux driver and by the Common Clock framework.

2 DT bindings documentation

The RCC is a multifunction device.

Each function is represented by a separate binding document:

- generic DT bindings[1] used by the Common Clock framework.
- vendor clock DT bindings[2] used by the clk-stm32mp1 driver: this binding document explains how to write device tree files for clocks.

3 DT configuration

3.1 DT configuration (STM32 level)

The STM32MP1 Clock node is located in the *stm32mp157c.dtsi*[3]. See Device tree for more explanations.

3.1.1 clocks node

These clocks have a fixed frequency (generally they are oscillators)

```
        clocks {
                clk_hse: clk-hse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <24000000>;
                };

                clk_hsi: clk-hsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <64000000>;
                };

                clk_lse: clk-lse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32768>;
                };

                clk_lsi: clk-lsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32000>;
                };

                clk_csi: clk-csi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <4000000>;
                };

                ...
        };
```

3.1.2 STM32MP1 Clock node

We need to specify the number of cells in a clock specifier.
For the STM32MP1 this number shoud be 1 and is configured via 'clock-cells' property in rcc node.

```
rcc: rcc@50000000 {
         compatible = "st,stm32mp1-rcc", "syscon";
        #clock-cells = <1>;
        #reset-cells = <1>;
        reg = <0x50000000 0x1000>;
        ...
};
```

This device tree part is related to STM32MP1 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

If a Linux driver needs a clock, it has to be added in its DT node:
clocks = <phandle> : List of phandle and clock specifier pairs, one pair for each clock input to the device. Note: if the clock provider specifies '0' for #clock-cells, then only the phandle portion of the pair will appear.

- Example:

```
usart3: serial@4000f000 {
        compatible = "st,stm32h7-usart";
        reg = <0x4000f000 0x400>;
        interrupt-names = "event", "wakeup";
        interrupts-extended = <&intc GIC_SPI 39 IRQ_TYPE_LEVEL_HIGH>,
                              <&exti 28 1>;
        clocks = <&rcc USART3_K>;
        wakeup-source;
        power-domains = <&pd_core>;
        status = "disabled";
};
```

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:

- Documentation/devicetree/bindings/clock/clock-bindings.txt , Clock device tree bindings
- Documentation/devicetree/bindings/clock/st,stm32mp1-rcc.txt , STM32MP1 clock device tree bindings
- stm32mp157c.dtsi STM32MP157C device tree file

Operating System

Reset and Clock Control

Device Tree

Generic Interrupt Controller

Serial Peripheral Interface

Clock device tree configuration

Stable: 04.02.2020 - 07:47 / Revision: 04.02.2020 - 07:34

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

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

The configuration is performed using the device tree mechanism that provides a hardware description of the RCC peripheral, used by the clk-stm32mp1 Linux driver and by the Common Clock framework.

2 DT bindings documentation

The RCC is a multifunction device.

Each function is represented by a separate binding document:

- generic DT bindings[1] used by the Common Clock framework.
- vendor clock DT bindings[2] used by the clk-stm32mp1 driver: this binding document explains how to write device tree files for clocks.

3 DT configuration

3.1 DT configuration (STM32 level)

The STM32MP1 Clock node is located in the *stm32mp157c.dtsi*[3]. See Device tree for more explanations.

3.1.1 clocks node

These clocks have a fixed frequency (generally they are oscillators)

```
        clocks {
                clk_hse: clk-hse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <24000000>;
                };

                clk_hsi: clk-hsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <64000000>;
                };

                clk_lse: clk-lse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32768>;
                };

                clk_lsi: clk-lsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32000>;
                };

                clk_csi: clk-csi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <4000000>;
                };

                ...
        };
```

3.1.2 STM32MP1 Clock node

We need to specify the number of cells in a clock specifier.
For the STM32MP1 this number shoud be 1 and is configured via 'clock-cells' property in rcc node.

```
rcc: rcc@50000000 {
         compatible = "st,stm32mp1-rcc", "syscon";
        #clock-cells = <1>;
        #reset-cells = <1>;
        reg = <0x50000000 0x1000>;
        ...
};
```

This device tree part is related to STM32MP1 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

If a Linux driver needs a clock, it has to be added in its DT node:
clocks = <phandle> : List of phandle and clock specifier pairs, one pair for each clock input to the device. Note: if the clock provider specifies '0' for #clock-cells, then only the phandle portion of the pair will appear.

- Example:

```
usart3: serial@4000f000 {
        compatible = "st,stm32h7-usart";
        reg = <0x4000f000 0x400>;
        interrupt-names = "event", "wakeup";
        interrupts-extended = <&intc GIC_SPI 39 IRQ_TYPE_LEVEL_HIGH>,
                              <&exti 28 1>;
        clocks = <&rcc USART3_K>;
        wakeup-source;
        power-domains = <&pd_core>;
        status = "disabled";
};
```

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:

- Documentation/devicetree/bindings/clock/clock-bindings.txt , Clock device tree bindings
- Documentation/devicetree/bindings/clock/st,stm32mp1-rcc.txt , STM32MP1 clock device tree bindings
- stm32mp157c.dtsi STM32MP157C device tree file

Operating System

Reset and Clock Control

Device Tree

Generic Interrupt Controller

Serial Peripheral Interface

Clock device tree configuration

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

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

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

The configuration is performed using the device tree mechanism that provides a hardware description of the RCC peripheral, used by the clk-stm32mp1 Linux driver and by the Common Clock framework.

2 DT bindings documentation

The RCC is a multifunction device.

Each function is represented by a separate binding document:

- generic DT bindings[1] used by the Common Clock framework.
- vendor clock DT bindings[2] used by the clk-stm32mp1 driver: this binding document explains how to write device tree files for clocks.

3 DT configuration

3.1 DT configuration (STM32 level)

The STM32MP1 Clock node is located in the *stm32mp157c.dtsi*[3]. See Device tree for more explanations.

3.1.1 clocks node

These clocks have a fixed frequency (generally they are oscillators)

```
        clocks {
                clk_hse: clk-hse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <24000000>;
                };

                clk_hsi: clk-hsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <64000000>;
                };

                clk_lse: clk-lse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32768>;
                };

                clk_lsi: clk-lsi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <32000>;
                };

                clk_csi: clk-csi {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <4000000>;
                };

                ...
        };
```

3.1.2 STM32MP1 Clock node

We need to specify the number of cells in a clock specifier.
For the STM32MP1 this number shoud be 1 and is configured via 'clock-cells' property in rcc node.

```
rcc: rcc@50000000 {
         compatible = "st,stm32mp1-rcc", "syscon";
        #clock-cells = <1>;
        #reset-cells = <1>;
        reg = <0x50000000 0x1000>;
        ...
};
```

This device tree part is related to STM32MP1 microprocessors. It must be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

If a Linux driver needs a clock, it has to be added in its DT node:
clocks = <phandle> : List of phandle and clock specifier pairs, one pair for each clock input to the device. Note: if the clock provider specifies '0' for #clock-cells, then only the phandle portion of the pair will appear.

- Example:

```
usart3: serial@4000f000 {
        compatible = "st,stm32h7-usart";
        reg = <0x4000f000 0x400>;
        interrupt-names = "event", "wakeup";
        interrupts-extended = <&intc GIC_SPI 39 IRQ_TYPE_LEVEL_HIGH>,
                              <&exti 28 1>;
        clocks = <&rcc USART3_K>;
        wakeup-source;
        power-domains = <&pd_core>;
        status = "disabled";
};
```

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:

- Documentation/devicetree/bindings/clock/clock-bindings.txt , Clock device tree bindings
- Documentation/devicetree/bindings/clock/st,stm32mp1-rcc.txt , STM32MP1 clock device tree bindings
- stm32mp157c.dtsi STM32MP157C device tree file

Operating System

Reset and Clock Control

Device Tree

Generic Interrupt Controller

Serial Peripheral Interface