



SDMMC device tree configuration



Contents

1. SDMMC device tree configuration	3
2. SDMMC internal peripheral	6
3. MMC overview	10
4. Device tree	13
5. Pinctrl device tree configuration	17
6. STM32CubeMX	21



SDMMC device tree configuration

Stable: 11.02.2019 - 11:21 / Revision: 21.01.2019 - 13:52

A quality version of this page, accepted on 11 February 2019, was based off this revision.

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	3
2 DT bindings documentation	3
3 DT configuration	3
3.1 DT configuration (STM32 level)	4
3.2 DT configuration (board level)	4
3.3 DT configuration examples	5
4 How to configure the DT using STM32CubeMX	6
5 References	6

1 Article purpose

This article explains how to configure the **SDMMC** internal peripheral when it is assigned to the Linux[®]OS. In that case, it is controlled by the MMC framework.

The configuration is performed using the **device tree** mechanism that provides a hardware description of the SDMMC peripheral, used by the STM32 SDMMC Linux driver and by the MMC framework.

2 DT bindings documentation

The SDMMC device tree bindings are composed of:

- generic MMC device tree bindings ^[1].
- SDMMC MMC/SD/SDIO interface bindings ^[2].

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 SDMMC peripheral node is located in *stm32mp157c.dtsi*^[3] file.

<pre>sdmmc1: sdmmc@58005000 { compatible = "arm,pl18x", "arm,primecell"; arm,primecell-periphid = <0x00253180>; reg = <0x58005000 0x1000>, register location <0x58006000 0x1000>; register location interrupts = <GIC_SPI 49 IRQ_TYPE_LEVEL_HIGH>; used interrupt-names = "cmd_irq"; clocks = <&rcc SDMMC1_K>; clock-names = apb_pclk resets = <&rcc SDMMC1_R>; status = "disabled"; };</pre>	Comments --> The controller --> The delay block --> The interrupt number
--	--



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

3.2 DT configuration (board level)

The SDMMC peripheral may connect to one SD card, one eMMC™ device or one SDIO card.

<pre>&sdmmc1{ pinctrl-names = "default", "opendrain", "sleep"; configuration, please refer to Pinctrl device tree configuration pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>; pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>; pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>; st,neg-edge; command on sdmmc clock falling edge st,sig-dir; direction polarity of an external transceiver st,use-ckin; from an external transceiver to sample the receive data bus-width = <4>; can be 1, 4 or 8 vmmc-supply = <&vdd_sd>; card's power vqmmc-supply = <&sd_switch>; line power status = "okay"; };</pre>	Comments --> For pinctrl --> Generate data and --> Allow to select --> Use sdmmc_ckin pin --> Number of data lines, --> Supply node for --> Supply node for IO --> Enable the node
---	---

Below optional properties have to be used when an external transceiver is connected:

- `st,sig-dir`: This property allows to select external transceiver direction signals polarity. When this property is set, the voltage transceiver IOs are driven as output when the direction signals are high. Without setting this property, the voltage transceiver IOs are driven as output when the direction signals are low.

- `st,use-ckin`: By setting this property, the `sdmmc_ckin` pin from an external transceiver is used to sample the receive data.

3.3 DT configuration examples

Below example shows how to configure the SDMMC when an eMMC™ is connected with 8 data lines ^[4].

<pre> &sdmmc2{ pinctrl-names = "default", "opendrain", "sleep"; pinctrl-0 = <&sdmmc2_b4_pins_a &sdmmc2_dir_pins_a>; pinctrl-1 = <&sdmmc2_b4_od_pins_a &sdmmc2_dir_pins_a>; pinctrl-2 = <&sdmmc2_b4_sleep_pins_a &sdmmc2_dir_sleep_pins_a>; non-removable; assume always present no-sd; command during initialization no-sdio; command during initialization st,neg-edge; bus-width = <8>; vmmc-supply = <&v3v3>; vqmmc-supply = <&vdd>; mmc-ddr-3_3v; DR 3.3V status = "okay"; }; </pre>	<p>Comments</p> <p>--> Non-removable slot,</p> <p>--> Avoid to send SD</p> <p>--> Avoid to send SDIO</p> <p>--> Host supports eMMC™ D</p>
--	--

Below example shows how to configure the SDMMC to SD card (4 data lines) with an external transceiver ^[4].

<pre> &sdmmc1{ pinctrl-names = "default", "opendrain", "sleep"; pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>; pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>; pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>; broken-cd; card detection st,neg-edge; st,sig-dir; st,use-ckin; bus-width = <4>; sd-uhs-sdr12; sd-uhs-sdr25; sd-uhs-sdr50; sd-uhs-ddr50; sd-uhs-sdr104; vmmc-supply = <&vdd_sd>; vqmmc-supply = <&sd_switch>; status = "okay"; }; </pre>	<p>Comments</p> <p>--> use polling mode for</p> <p>--> sd modes supported ^[1]</p>
---	---



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:

- 1.01.1 Documentation/devicetree/bindings/mmc/mmc.txt
- Documentation/devicetree/bindings/mmc/mmci.txt
- arch/arm/boot/dts/stm32mp157c.dtsi
- 4.04.1 arch/arm/boot/dts/stm32mp157c-ed1.dts

Operating System

MultimediaCard

Device Tree

Secure digital

Generic Interrupt Controller

Serial Peripheral Interface

SD memory card (<https://www.sdcard.org>)

SDIO is an SD-size card with extended input/output functions

input/output

Doubledata rate (memory domain)

SDMMC device tree configuration

Stable: 14.05.2020 - 07:13 / Revision: 14.05.2020 - 07:12

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	7
-------------------------	---



2 DT bindings documentation	7
3 DT configuration	7
3.1 DT configuration (STM32 level)	7
3.2 DT configuration (board level)	8
3.3 DT configuration examples	8
4 How to configure the DT using STM32CubeMX	9
5 References	9

1 Article purpose

This article explains how to configure the **SDMMC** internal peripheral when it is assigned to the Linux[®]OS. In that case, it is controlled by the **MMC** framework.

The configuration is performed using the **device tree** mechanism that provides a hardware description of the SDMMC peripheral, used by the STM32 SDMMC Linux driver and by the MMC framework.

2 DT bindings documentation

The SDMMC device tree bindings are composed of:

- generic MMC device tree bindings ^[1].
- SDMMC MMC/SD/SDIO interface bindings ^[2].

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 SDMMC peripheral node is located in *stm32mp157c.dtsi*^[3] file.

<pre>sdmmc1: sdmmc@58005000 { compatible = "arm,pl18x", "arm,primecell"; arm,primecell-periphid = <0x00253180>; reg = <0x58005000 0x1000>, register location <0x58006000 0x1000>; register location interrupts = <GIC_SPI 49 IRQ_TYPE_LEVEL_HIGH>;</pre>	<p>Comments</p> <p>--> The controller</p> <p>--> The delay block</p> <p>--> The interrupt number</p>
--	--

used

```

interrupt-names = "cmd_irq";
clocks = <&rcc SDMMC1_K>;
clock-names = apb_pclk
resets = <&rcc SDMMC1_R>;
status = "disabled";
};

```



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

3.2 DT configuration (board level)

The SDMMC peripheral may connect to one SD card, one eMMC™ device or one SDIO card.

<pre> &sdmmc1{ pinctrl-names = "default", "opendrain", "sleep"; configuration, please refer to Pinctrl device tree configuration pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>; pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>; pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>; st,neg-edge; command on sdmmc clock falling edge st,sig-dir; direction polarity of an external transceiver st,use-ckin; from an external transceiver to sample the receive data bus-width = <4>; can be 1, 4 or 8 vmmc-supply = <&vdd_sd>; card's power vqmmc-supply = <&sd_switch>; Line power status = "okay"; }; </pre>	<p>Comments</p> <p>--> For pinctrl</p> <p>--> Generate data and</p> <p>--> Allow to select</p> <p>--> Use sdmmc_ckin pin</p> <p>--> Number of data lines,</p> <p>--> Supply node for</p> <p>--> Supply node for IO</p> <p>--> Enable the node</p>
--	--

Below optional properties have to be used when an external transceiver is connected:

- `st,sig-dir`: This property allows to select external transceiver direction signals polarity. When this property is set, the voltage transceiver IOs are driven as output when the direction signals are high. Without setting this property, the voltage transceiver IOs are driven as output when the direction signals are low.
- `st,use-ckin`: By setting this property, the `sdmmc_ckin` pin from an external transceiver is used to sample the receive data.

3.3 DT configuration examples

Below example shows how to configure the SDMMC when an eMMC™ is connected with 8 data lines ^[4].

<pre> &sdmmc2{ pinctrl-names = "default", "opendrain", "sleep"; pinctrl-0 = <&sdmmc2_b4_pins_a &sdmmc2_dir_pins_a>; pinctrl-1 = <&sdmmc2_b4_od_pins_a &sdmmc2_dir_pins_a>; pinctrl-2 = <&sdmmc2_b4_sleep_pins_a &sdmmc2_dir_sleep_pins_a>; non-removable; }; </pre>	<p>Comments</p> <p>--> Non-removable slot,</p>
---	--



```
assume always present
no-sd;                                --> Avoid to send SD
command during initialization
no-sdio;                               --> Avoid to send SDIO
command during initialization
st,neg-edge;
bus-width = <8>;
vmmc-supply = <&v3v3>;
vqmmc-supply = <&vdd>;
mmc-ddr-3_3v;                          --> Host supports eMMC™ D
DR 3.3V
status = "okay";
};
```

Below example shows how to configure the SDMMC to SD card (4 data lines) with an external transceiver ^[4].

```
&sdmmc1{                               Comments
    pinctrl-names = "default", "opendrain", "sleep";
    pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>;
    pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>;
    pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>;
    broken-cd;                          --> use polling mode for
card detection
    st,neg-edge;
    st,sig-dir;
    st,use-ckin;
    bus-width = <4>;
    sd-uhs-sdr12;                        --> sd modes supported [1]
    sd-uhs-sdr25;
    sd-uhs-sdr50;
    sd-uhs-ddr50;
    sd-uhs-sdr104;
    vmmc-supply = <&vdd_sd>;
    vqmmc-supply = <&sd_switch>;
    status = "okay";
};
```

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:



SDMMC device tree configuration

- 1.01.1 Documentation/devicetree/bindings/mmc/mmc.txt
- Documentation/devicetree/bindings/mmc/mmci.txt
- arch/arm/boot/dts/stm32mp157c.dtsi
- 4.04.1 arch/arm/boot/dts/stm32mp157c-ed1.dts

Operating System

MultimediaCard

Device Tree

Secure digital

Generic Interrupt Controller

Serial Peripheral Interface

SD memory card (<https://www.sdcard.org>)

SDIO is an SD-size card with extended input/output functions

input/output

Doubledata rate (memory domain)

SDMMC device tree configuration

Stable: 14.05.2020 - 09:26 / Revision: 14.05.2020 - 09:25

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	10
2 DT bindings documentation	11
3 DT configuration	11
3.1 DT configuration (STM32 level)	11
3.2 DT configuration (board level)	11
3.3 DT configuration examples	12
4 How to configure the DT using STM32CubeMX	13
5 References	13

1 Article purpose

This article explains how to configure the **SDMMC** internal peripheral when it is assigned to the Linux[®]OS. In that case, it is controlled by the **MMC** framework.

The configuration is performed using the **device tree** mechanism that provides a hardware description of the **SDMMC** peripheral, used by the STM32 **SDMMC** Linux driver and by the **MMC** framework.

2 DT bindings documentation

The SDMMC device tree bindings are composed of:

- generic MMC device tree bindings ^[1].
- SDMMC MMC/SD/SDIO interface bindings ^[2].

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 SDMMC peripheral node is located in *stm32mp157c.dtsi*^[3] file.

<pre>sdmmc1: sdmmc@58005000 { compatible = "arm,pl18x", "arm,primecell"; arm,primecell-periphid = <0x00253180>; reg = <0x58005000 0x1000>, register location <0x58006000 0x1000>; register location interrupts = <GIC_SPI 49 IRQ_TYPE_LEVEL_HIGH>; used interrupt-names = "cmd_irq"; clocks = <&rcc SDMMC1_K>; clock-names = apb_pclk resets = <&rcc SDMMC1_R>; status = "disabled"; };</pre>	<p>Comments</p> <p>--> The controller</p> <p>--> The delay block</p> <p>--> The interrupt number</p>
--	--



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

3.2 DT configuration (board level)

The SDMMC peripheral may connect to one SD card, one eMMC™ device or one SDIO card.

<pre>&sdmmc1{ pinctrl-names = "default", "opendrain", "sleep"; configuration, please refer to Pinctrl device tree configuration pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>; pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>;</pre>	<p>Comments</p> <p>--> For pinctrl</p>
---	--

```

        pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>;
        st,neg-edge;
command on sdmmc clock falling edge
        st,sig-dir;
direction polarity of an external transceiver
        st,use-ckin;
from an external transceiver to sample the receive data
        bus-width = <4>;
can be 1, 4 or 8
        vmmc-supply = <&vdd_sd>;
card's power
        vqmmc-supply = <&sd_switch>;
line power
        status = "okay";
        };

```

--> Generate data and
--> Allow to select
--> Use sdmmc_ckin pin
--> Number of data lines,
--> Supply node for
--> Supply node for IO
--> Enable the node

Below optional properties have to be used when an external transceiver is connected:

- `st,sig-dir`: This property allows to select external transceiver direction signals polarity. When this property is set, the voltage transceiver IOs are driven as output when the direction signals are high. Without setting this property, the voltage transceiver IOs are driven as output when the direction signals are low.
- `st,use-ckin`: By setting this property, the `sdmmc_ckin` pin from an external transceiver is used to sample the receive data.

3.3 DT configuration examples

Below example shows how to configure the SDMMC when an eMMC™ is connected with 8 data lines ^[4].

```

&sdmmc2{
    pinctrl-names = "default", "opendrain", "sleep";
    pinctrl-0 = <&sdmmc2_b4_pins_a &sdmmc2_dir_pins_a>;
    pinctrl-1 = <&sdmmc2_b4_od_pins_a &sdmmc2_dir_pins_a>;
    pinctrl-2 = <&sdmmc2_b4_sleep_pins_a &sdmmc2_dir_sleep_pins_a>;
    non-removable;
assume always present
    no-sd;
command during initialization
    no-sdio;
command during initialization
    st,neg-edge;
    bus-width = <8>;
    vmmc-supply = <&v3v3>;
    vqmmc-supply = <&vdd>;
    mmc-ddr-3_3v;
DR 3.3V
    status = "okay";
};

```

Comments
--> Non-removable slot,
--> Avoid to send SD
--> Avoid to send SDIO
--> Host supports eMMC™ D

Below example shows how to configure the SDMMC to SD card (4 data lines) with an external transceiver ^[4].

```

&sdmmc1{
    pinctrl-names = "default", "opendrain", "sleep";
    pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>;
    pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>;
    pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>;
    broken-cd;
card detection
    st,neg-edge;
};

```

Comments
--> use polling mode for



```
st,sig-dir;
st,use-ckin;
bus-width = <4>;

sd-uhs-sdr12;
sd-uhs-sdr25;
sd-uhs-sdr50;
sd-uhs-ddr50;
sd-uhs-sdr104;
vmmc-supply = <&vdd_sd>;
vqmmc-supply = <&sd_switch>;
status = "okay";
};
```

--> sd modes supported [1]

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:

- [1.01.1 Documentation/devicetree/bindings/mmc/mmc.txt](#)
- [Documentation/devicetree/bindings/mmc/mhci.txt](#)
- [arch/arm/boot/dts/stm32mp157c.dtsi](#)
- [4.04.1 arch/arm/boot/dts/stm32mp157c-ed1.dts](#)

Operating System

MultimediaCard

Device Tree

Secure digital

Generic Interrupt Controller

Serial Peripheral Interface

SD memory card (<https://www.sdcard.org>)

SDIO is an SD-size card with extended input/output functions

input/output

Doubledata rate (memory domain)



SDMMC device tree configuration

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

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	14
2 DT bindings documentation	14
3 DT configuration	14
3.1 DT configuration (STM32 level)	15
3.2 DT configuration (board level)	15
3.3 DT configuration examples	16
4 How to configure the DT using STM32CubeMX	17
5 References	17

1 Article purpose

This article explains how to configure the **SDMMC** internal peripheral when it is assigned to the Linux[®]OS. In that case, it is controlled by the **MMC** framework.

The configuration is performed using the **device tree** mechanism that provides a hardware description of the SDMMC peripheral, used by the STM32 SDMMC Linux driver and by the MMC framework.

2 DT bindings documentation

The SDMMC device tree bindings are composed of:

- generic MMC device tree bindings ^[1].
- SDMMC MMC/SD/SDIO interface bindings ^[2].

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 SDMMC peripheral node is located in *stm32mp157c.dts*^[3] file.

<pre>sdmmc1: sdmmc@58005000 { compatible = "arm,pl18x", "arm,primecell"; arm,primecell-periphid = <0x00253180>; reg = <0x58005000 0x1000>, register location <0x58006000 0x1000>; register location interrupts = <GIC_SPI 49 IRQ_TYPE_LEVEL_HIGH>; used interrupt-names = "cmd_irq"; clocks = <&rcc SDMMC1_K>; clock-names = apb_pclk resets = <&rcc SDMMC1_R>; status = "disabled"; };</pre>	Comments --> The controller --> The delay block --> The interrupt number
--	--



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

3.2 DT configuration (board level)

The SDMMC peripheral may connect to one SD card, one eMMC™ device or one SDIO card.

<pre>&sdmmc1{ pinctrl-names = "default", "opendrain", "sleep"; configuration, please refer to Pinctrl device tree configuration pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>; pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>; pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>; st,neg-edge; command on sdmmc clock falling edge st,sig-dir; direction polarity of an external transceiver st,use-ckin; from an external transceiver to sample the receive data bus-width = <4>; can be 1, 4 or 8 vmmc-supply = <&vdd_sd>; card's power vqmmc-supply = <&sd_switch>; line power status = "okay"; };</pre>	Comments --> For pinctrl --> Generate data and --> Allow to select --> Use sdmmc_ckin pin --> Number of data lines, --> Supply node for --> Supply node for IO --> Enable the node
---	---

Below optional properties have to be used when an external transceiver is connected:

- *st,sig-dir*: This property allows to select external transceiver direction signals polarity. When this property is set, the voltage transceiver IOs are driven as output when the direction signals are high. Without setting this property, the voltage transceiver IOs are driven as output when the direction signals are low.
- *st,use-ckin*: By setting this property, the *sdmmc_ckin* pin from an external transceiver is used to sample the receive data.

3.3 DT configuration examples

Below example shows how to configure the SDMMC when an eMMC™ is connected with 8 data lines ^[4].

<pre> &sdmmc2{ pinctrl-names = "default", "opendrain", "sleep"; pinctrl-0 = <&sdmmc2_b4_pins_a &sdmmc2_dir_pins_a>; pinctrl-1 = <&sdmmc2_b4_od_pins_a &sdmmc2_dir_pins_a>; pinctrl-2 = <&sdmmc2_b4_sleep_pins_a &sdmmc2_dir_sleep_pins_a>; non-removable; assume always present no-sd; command during initialization no-sdio; command during initialization st,neg-edge; bus-width = <8>; vmmc-supply = <&v3v3>; vqmmc-supply = <&vdd>; mmc-ddr-3_3v; DR 3.3V status = "okay"; }; </pre>	<p>Comments</p> <p>--> Non-removable slot,</p> <p>--> Avoid to send SD</p> <p>--> Avoid to send SDIO</p> <p>--> Host supports eMMC™ D</p>
--	--

Below example shows how to configure the SDMMC to SD card (4 data lines) with an external transceiver ^[4].

<pre> &sdmmc1{ pinctrl-names = "default", "opendrain", "sleep"; pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>; pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>; pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>; broken-cd; card detection st,neg-edge; st,sig-dir; st,use-ckin; bus-width = <4>; sd-uhs-sdr12; sd-uhs-sdr25; sd-uhs-sdr50; sd-uhs-ddr50; sd-uhs-sdr104; vmmc-supply = <&vdd_sd>; vqmmc-supply = <&sd_switch>; status = "okay"; }; </pre>	<p>Comments</p> <p>--> use polling mode for</p> <p>--> sd modes supported ^[1]</p>
--	---



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:

- 1.01.1 Documentation/devicetree/bindings/mmc/mmc.txt
- Documentation/devicetree/bindings/mmc/mhci.txt
- arch/arm/boot/dts/stm32mp157c.dtsi
- 4.04.1 arch/arm/boot/dts/stm32mp157c-ed1.dts

Operating System

MultimediaCard

Device Tree

Secure digital

Generic Interrupt Controller

Serial Peripheral Interface

SD memory card (<https://www.sdcard.org>)

SDIO is an SD-size card with extended input/output functions

input/output

Doubledata rate (memory domain)

SDMMC device tree configuration

Stable: 11.06.2020 - 09:00 / Revision: 10.06.2020 - 15:35

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	18
-------------------------	----



2 DT bindings documentation	18
3 DT configuration	18
3.1 DT configuration (STM32 level)	18
3.2 DT configuration (board level)	19
3.3 DT configuration examples	19
4 How to configure the DT using STM32CubeMX	20
5 References	20

1 Article purpose

This article explains how to configure the **SDMMC** internal peripheral when it is assigned to the Linux[®]OS. In that case, it is controlled by the **MMC** framework.

The configuration is performed using the **device tree** mechanism that provides a hardware description of the SDMMC peripheral, used by the STM32 SDMMC Linux driver and by the MMC framework.

2 DT bindings documentation

The SDMMC device tree bindings are composed of:

- generic MMC device tree bindings ^[1].
- SDMMC MMC/SD/SDIO interface bindings ^[2].

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 SDMMC peripheral node is located in *stm32mp157c.dtsi*^[3] file.

<pre>sdmmc1: sdmmc@58005000 { compatible = "arm,pl18x", "arm,primecell"; arm,primecell-periphid = <0x00253180>; reg = <0x58005000 0x1000>, register location <0x58006000 0x1000>; register location interrupts = <GIC_SPI 49 IRQ_TYPE_LEVEL_HIGH>;</pre>	<p>Comments</p> <p>--> The controller</p> <p>--> The delay block</p> <p>--> The interrupt number</p>
--	--

used

```

interrupt-names = "cmd_irq";
clocks = <&rcc SDMMC1_K>;
clock-names = apb_pclk
resets = <&rcc SDMMC1_R>;
status = "disabled";
};

```



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

3.2 DT configuration (board level)

The SDMMC peripheral may connect to one SD card, one eMMC™ device or one SDIO card.

<pre> &sdmmc1{ pinctrl-names = "default", "opendrain", "sleep"; configuration, please refer to Pinctrl device tree configuration pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>; pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>; pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>; st,neg-edge; command on sdmmc clock falling edge st,sig-dir; direction polarity of an external transceiver st,use-ckin; from an external transceiver to sample the receive data bus-width = <4>; can be 1, 4 or 8 vmmc-supply = <&vdd_sd>; card's power vqmmc-supply = <&sd_switch>; line power status = "okay"; }; </pre>	<p>Comments</p> <p>--> For pinctrl</p> <p>--> Generate data and</p> <p>--> Allow to select</p> <p>--> Use sdmmc_ckin pin</p> <p>--> Number of data lines,</p> <p>--> Supply node for</p> <p>--> Supply node for IO</p> <p>--> Enable the node</p>
--	--

Below optional properties have to be used when an external transceiver is connected:

- `st,sig-dir`: This property allows to select external transceiver direction signals polarity. When this property is set, the voltage transceiver IOs are driven as output when the direction signals are high. Without setting this property, the voltage transceiver IOs are driven as output when the direction signals are low.
- `st,use-ckin`: By setting this property, the `sdmmc_ckin` pin from an external transceiver is used to sample the receive data.

3.3 DT configuration examples

Below example shows how to configure the SDMMC when an eMMC™ is connected with 8 data lines ^[4].

<pre> &sdmmc2{ pinctrl-names = "default", "opendrain", "sleep"; pinctrl-0 = <&sdmmc2_b4_pins_a &sdmmc2_dir_pins_a>; pinctrl-1 = <&sdmmc2_b4_od_pins_a &sdmmc2_dir_pins_a>; pinctrl-2 = <&sdmmc2_b4_sleep_pins_a &sdmmc2_dir_sleep_pins_a>; non-removable; }; </pre>	<p>Comments</p> <p>--> Non-removable slot,</p>
---	--



```
assume always present
no-sd;                                --> Avoid to send SD
command during initialization
no-sdio;                               --> Avoid to send SDIO
command during initialization
st,neg-edge;
bus-width = <8>;
vmmc-supply = <&v3v3>;
vqmmc-supply = <&vdd>;
mmc-ddr-3_3v;                          --> Host supports eMMC™ D
DR 3.3V
status = "okay";
};
```

Below example shows how to configure the SDMMC to SD card (4 data lines) with an external transceiver ^[4].

```
&sdmmc1{                               Comments
    pinctrl-names = "default", "opendrain", "sleep";
    pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>;
    pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>;
    pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>;
    broken-cd;                          --> use polling mode for
card detection
    st,neg-edge;
    st,sig-dir;
    st,use-ckin;
    bus-width = <4>;
    sd-uhs-sdr12;                        --> sd modes supported [1]
    sd-uhs-sdr25;
    sd-uhs-sdr50;
    sd-uhs-ddr50;
    sd-uhs-sdr104;
    vmmc-supply = <&vdd_sd>;
    vqmmc-supply = <&sd_switch>;
    status = "okay";
};
```

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:



SDMMC device tree configuration

- 1.01.1 Documentation/devicetree/bindings/mmc/mmc.txt
- Documentation/devicetree/bindings/mmc/mmci.txt
- arch/arm/boot/dts/stm32mp157c.dtsi
- 4.04.1 arch/arm/boot/dts/stm32mp157c-ed1.dts

Operating System

MultimediaCard

Device Tree

Secure digital

Generic Interrupt Controller

Serial Peripheral Interface

SD memory card (<https://www.sdcard.org>)

SDIO is an SD-size card with extended input/output functions

input/output

Doubledata rate (memory domain)

SDMMC device tree configuration

Stable: **Not stable** / Revision: 12.06.2020 - 13:25

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose	21
2 DT bindings documentation	22
3 DT configuration	22
3.1 DT configuration (STM32 level)	22
3.2 DT configuration (board level)	22
3.3 DT configuration examples	23
4 How to configure the DT using STM32CubeMX	24
5 References	24

1 Article purpose

This article explains how to configure the **SDMMC** internal peripheral when it is assigned to the Linux[®]OS. In that case, it is controlled by the **MMC** framework.

The configuration is performed using the **device tree** mechanism that provides a hardware description of the **SDMMC** peripheral, used by the **STM32 SDMMC Linux driver** and by the **MMC framework**.

2 DT bindings documentation

The SDMMC device tree bindings are composed of:

- generic MMC device tree bindings ^[1].
- SDMMC MMC/SD/SDIO interface bindings ^[2].

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 SDMMC peripheral node is located in *stm32mp157c.dtsi*^[3] file.

<pre>sdmmc1: sdmmc@58005000 { compatible = "arm,pl18x", "arm,primecell"; arm,primecell-periphid = <0x00253180>; reg = <0x58005000 0x1000>, register location <0x58006000 0x1000>; register location interrupts = <GIC_SPI 49 IRQ_TYPE_LEVEL_HIGH>; used interrupt-names = "cmd_irq"; clocks = <&rcc SDMMC1_K>; clock-names = apb_pclk resets = <&rcc SDMMC1_R>; status = "disabled"; };</pre>	<p>Comments</p> <p>--> The controller</p> <p>--> The delay block</p> <p>--> The interrupt number</p>
--	--



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

3.2 DT configuration (board level)

The SDMMC peripheral may connect to one SD card, one eMMC™ device or one SDIO card.

<pre>&sdmmc1{ pinctrl-names = "default", "opendrain", "sleep"; configuration, please refer to Pinctrl device tree configuration pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>; pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>;</pre>	<p>Comments</p> <p>--> For pinctrl</p>
---	--

```

        pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>;
        st,neg-edge;
command on sdmmc clock falling edge
        st,sig-dir;
direction polarity of an external transceiver
        st,use-ckin;
from an external transceiver to sample the receive data
        bus-width = <4>;
can be 1, 4 or 8
        vmmc-supply = <&vdd_sd>;
card's power
        vqmmc-supply = <&sd_switch>;
line power
        status = "okay";
        };

```

--> Generate data and
--> Allow to select
--> Use sdmmc_ckin pin
--> Number of data lines,
--> Supply node for
--> Supply node for IO
--> Enable the node

Below optional properties have to be used when an external transceiver is connected:

- `st,sig-dir`: This property allows to select external transceiver direction signals polarity. When this property is set, the voltage transceiver IOs are driven as output when the direction signals are high. Without setting this property, the voltage transceiver IOs are driven as output when the direction signals are low.
- `st,use-ckin`: By setting this property, the `sdmmc_ckin` pin from an external transceiver is used to sample the receive data.

3.3 DT configuration examples

Below example shows how to configure the SDMMC when an eMMC™ is connected with 8 data lines ^[4].

```

&sdmmc2{
    pinctrl-names = "default", "opendrain", "sleep";
    pinctrl-0 = <&sdmmc2_b4_pins_a &sdmmc2_dir_pins_a>;
    pinctrl-1 = <&sdmmc2_b4_od_pins_a &sdmmc2_dir_pins_a>;
    pinctrl-2 = <&sdmmc2_b4_sleep_pins_a &sdmmc2_dir_sleep_pins_a>;
    non-removable;
assume always present
    no-sd;
command during initialization
    no-sdio;
command during initialization
    st,neg-edge;
    bus-width = <8>;
    vmmc-supply = <&v3v3>;
    vqmmc-supply = <&vdd>;
    mmc-ddr-3_3v;
DR 3.3V
    status = "okay";
};

```

Comments
--> Non-removable slot,
--> Avoid to send SD
--> Avoid to send SDIO
--> Host supports eMMC™ D

Below example shows how to configure the SDMMC to SD card (4 data lines) with an external transceiver ^[4].

```

&sdmmc1{
    pinctrl-names = "default", "opendrain", "sleep";
    pinctrl-0 = <&sdmmc1_b4_pins_a &sdmmc1_dir_pins_a>;
    pinctrl-1 = <&sdmmc1_b4_od_pins_a &sdmmc1_dir_pins_a>;
    pinctrl-2 = <&sdmmc1_b4_sleep_pins_a &sdmmc1_dir_sleep_pins_a>;
    broken-cd;
card detection
    st,neg-edge;
};

```

Comments
--> use polling mode for



```
st,sig-dir;
st,use-ckin;
bus-width = <4>;

sd-uhs-sdr12;
sd-uhs-sdr25;
sd-uhs-sdr50;
sd-uhs-ddr50;
sd-uhs-sdr104;
vmmc-supply = <&vdd_sd>;
vqmmc-supply = <&sd_switch>;
status = "okay";
};
```

--> sd modes supported [1]

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:

- 1.01.1 [Documentation/devicetree/bindings/mmc/mmc.txt](#)
- [Documentation/devicetree/bindings/mmc/mhci.txt](#)
- [arch/arm/boot/dts/stm32mp157c.dtsi](#)
- 4.04.1 [arch/arm/boot/dts/stm32mp157c-ed1.dts](#)

Operating System

MultimediaCard

Device Tree

Secure digital

Generic Interrupt Controller

Serial Peripheral Interface

SD memory card (<https://www.sdcard.org>)

SDIO is an SD-size card with extended input/output functions

input/output

Doubledata rate (memory domain)