

FDCAN device tree configuration

Stable: 11.02.2019 - 13:21 / Revision: 21.01.2019 - 16:02

Contents

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

1 Article purpose

This article explains how to configure the [FDCAN](#) when it is assigned to the Linux[®] OS. In that case, it is controlled by the [CAN framework](#) for Bosch M_CAN controller.

The configuration is performed using the [device tree](#) mechanism that provides a hardware description of the FDCAN peripheral, used by the M_CAN Linux driver and by the NET/CAN framework.

2 DT bindings documentation

M_CAN device tree bindings^[1] describe all the required and optional properties.

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)

All M_CAN nodes are described in `stm32mp157c.dtsi`^[2] file with disabled status and required properties such as:

- Physical base address and size of the device register map

- Message RAM address and size (CAN SRAM)
- Host clock and CAN clock
- Message RAM configuration

This is a set of properties that may not vary for a given STM32 device.

```
m_can1: can@4400e000 {
    compatible = "bosch,m_can";
    reg = <0x4400e000 0x400>, <0x44011000 0x1400>; /* FDCAN1 uses only the first ha
    reg-names = "m_can", "message_ram";
    interrupts = <GIC_SPI 19 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 21 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "int0", "int1";
    clocks = <&rcc CK_HSE>, <&rcc FDCAN_K>;
    clock-names = "hclk", "cclk";
    bosch,mram-cfg = <0x0 0 0 32 0 0 2 2>;
    status = "disabled";
};

m_can2: can@4400f000 {
    compatible = "bosch,m_can";
    reg = <0x4400f000 0x400>, <0x44011000 0x2800>; /* The 10 Kbytes of the CAN_SRAM
    reg-names = "m_can", "message_ram";
    interrupts = <GIC_SPI 20 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 22 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "int0", "int1";
    clocks = <&rcc CK_HSE>, <&rcc FDCAN_K>;
    clock-names = "hclk", "cclk";
    bosch,mram-cfg = <0x1400 0 0 32 0 0 2 2>; /* Set mram-cfg offset to write
    status = "disabled";
};
```

The required and optional properties are fully described in the [bindings files](#).



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)

Part of the [device tree](#) is used to describe the FDCAN hardware used on a given board. The DT node ("**m_can**") must be filled in:

- Enable the CAN block by setting **status = "okay"**.
- Configure the pins in use via [pinctrl](#), through **pinctrl-0** (default pins), **pinctrl-1** (sleep pins) and **pinctrl-names**.

3.3 DT configuration examples

The example below shows how to configure and enable FDCAN1 instance at board level:

```
&m_can1 {
    pinctrl-names = "default", "sleep"; /* configure pinctrl modes for m_can1
    pinctrl-0 = <&m_can1_pins_a>; /* configure m_can1_pins_a as default
    pinctrl-1 = <&m_can1_sleep_pins_a>; /* configure m_can1_sleep_pins_a as sl
    status = "okay"; /* enable m_can1 */
};
```

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. ↑ [Documentation/devicetree/bindings/net/can/m_can.txt](#) M_CAN device tree bindings
2. ↑ [arch/arm/boot/dts/stm32mp157c.dtsi](#) , STM32MP157C device tree file

Operating System

Controller Area Network (robust bus mainly used for automotive applications)

Device Tree

Random Access Memory

Generic Interrupt Controller

Serial Peripheral Interface