在前两篇中我们了解了DTS的背景基础知识以及发挥作用的流程，这篇文章我们以高通的MSM8953平台为例来添加一个基础的i2c设备（包含一个gpio中断）。

首先我们在该i2c设备的驱动中找到了匹配设备与驱动程序的compatible

```

static const struct of_device_id iqs263_of_match[] = {
    { .compatible = "azopteq,iqs263", },
    { },
}
```

由此compatible可以找到dts中对应的设备

kernel\arch\arm64\boot\dts\qcom\msm8953-mtp.dts

```
&i2c_8 { /* BLSP2 QUP4 */
    iqs263@44 { //Capacitive Touch Controller Driver
        compatible = "azopteq,iqs263";
        reg = <0x44>;

        pinctrl-names = "default";
        pinctrl-0 = <&iqs263_irq_config>;

        vdd-supply = <&pm8953_l5>;
        vio-supply = <&pm8953_l5>;
        interrupt-parent = <&tlmm>;
        interrupts = <48 0x2>;
        azopteq,irq-gpio =<&tlmm 48 0x2>;
    };
};
```

其中compatible属性标识的名字是与驱动程序中名字相匹配的

其中reg属性及@符号后的十六进制数字标识了该设备iqs263的i2c地址为0x44

&i2c_8前的&表明此处仅仅是对i2c_8节点的补充，我们可以找到该节点定义的地方

kernel\arch\arm64\boot\dts\qcom\msm8953.dtsi

```
&soc {
    i2c_8: i2c@7af8000 { /* BLSP2 QUP4 */
        compatible = "qcom,i2c-msm-v2";
        #address-cells = <1>;
        #size-cells = <0>;
        reg-names = "qup_phys_addr";
        reg = <0x7af8000 0x600>;
        interrupt-names = "qup_irq";
        interrupts = <0 302 0>;
        qcom,clk-freq-out = <400000>;
        qcom,clk-freq-in  = <19200000>;
        clock-names = "iface_clk", "core_clk";
        clocks = <&clock_gcc clk_gcc_blsp2_ahb_clk>,
            <&clock_gcc clk_gcc_blsp2_qup4_i2c_apps_clk>;

        pinctrl-names = "i2c_active", "i2c_sleep";
        pinctrl-0 = <&i2c_8_active>;
        pinctrl-1 = <&i2c_8_sleep>;
        qcom,noise-rjct-scl = <0>;

        qcom,noise-rjct-sda = <0>;
        qcom,master-id = <84>;
        dmas = <&dma_blsp2 10 64 0x20000020 0x20>,
            <&dma_blsp2 11 32 0x20000020 0x20>;
        dma-names = "tx", "rx";
    };
    rpm_bus: qcom,rpm-smd {
        compatible = "qcom,rpm-smd";
        rpm-channel-name = "rpm_requests";
        rpm-channel-type = <15>; /* SMD_APPS_RPM */
    };
```

这当中的大部分属性都与平台相关了，可以看到定义了该i2c接口的时钟源、中断格式等。这部分一般由平台提供，作为驱动工程师了解即可。

i2c_8节点中的pinctrl-0指向了定义其io口的节点i2c_8_active和i2c_8_sleep，代码如下可见该i2c的IO口为Gpio98和Gpio99

kernel\arch\arm64\boot\dts\qcom\msm8953-pinctrl.dtsi

```
&soc {
    tlmm: pinctrl@1000000 {
        i2c_8 {
            i2c_8_active: i2c_8_active {
                /* active state */
                mux {
                    pins = "gpio98", "gpio99";
                    function = "blsp_i2c8";
                };

                config {
                    pins = "gpio98", "gpio99";
                    drive-strength = <2>;
                    bias-disable;
                };
            };

            i2c_8_sleep: i2c_8_sleep {
                /* suspended state */
                mux {
                    pins = "gpio98", "gpio99";
                    function = "gpio";
                };

                config {
                    pins = "gpio98", "gpio99";
                    drive-strength = <2>;
                    bias-disable;
                };
            };
        };

        iqs263_irq_config: iqs263_irq_config {
            mux {
                pins = "gpio48";
                function = "gpio";
            };

            config {
                pins = "gpio48";
                drive-strength = <2>;
                bias-pull-up;
            };
        };
```

iqs263节点中的pinctrl-0 属性指向了表明其io口属性的节点为iqs263_irq_config，相关代码也在2.3.1指示的msm8953-pinctrl.dtsi文件中。

其中定义了iqs263的中断IO脚为Gpio48

iqs263节点中的vdd-supply，vio-supply属性指示了表明iqs263芯片供电的引脚的节点pm8953_l5，代码如下

kernel\arch\arm64\boot\dts\qcom\msm8953-regulator.dtsi

```
&rpm_bus {
    rpm-regulator-ldoa5 {
        status = "okay";
        pm8953_l5: regulator-l5 {
            regulator-min-microvolt = <1800000>;
            regulator-max-microvolt = <1800000>;
            qcom,init-voltage = <1800000>;
            status = "okay";
        };
    };
```

iqs263的interrupts = <48 0x2>属性表明中断号为48,2代表下降沿触发