Rockchip
瑞芯微电子

Rockchip
Thermal 开发指南

发布版本:1.00

日期:2016.07

前言

概述

本文档主要介绍 RK 平台 Thermal 配置的调试方法。

产品版本

产品名称	内核版本
RK3399	Linux4.4

读者对象

本文档（本指南）主要适用于以下工程师：
技术支持工程师
软件开发工程师

修订记录

日期	版本	作者	修改说明
2016-07-01	V1.0	XF,HYZ	初始发布

目录

1 概述 1-1
2 重要概念 2-1
3 配置方法 3-1
3.1 Tsadc 配置 3-1
3.1.1 Menuconfig 配置 3-1
3.1.2 DTS 配置 3-1
3.2 power_allocator 策略配置 3-3
3.2.1 默认使用 power_allocator 策略 3-3
3.2.2 CPU 开启温控 3-4
3.2.5 GPU 开启温控 3-5
3.2.8 thermal_zone 配置 3-6
4 调试接口 4-1
4.1 关温控 4-1
4.2 获取当前温度 4-1

1 概述

本章主要描述 Thermal 的相关的重要概念、配置方法和调试接口。

2 重要概念

在 Linux 内核中，定义一套温控框架 linux Generic Thermal Sysfs Drivers，它可以通过不同的策略控制系统的温度，目前常用的有以下几种策略：

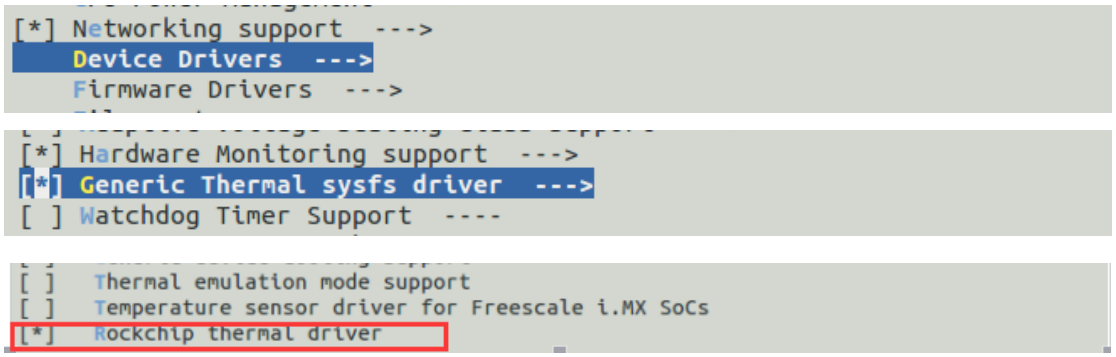
- power_allocator：引入 PID（比例-积分-微分）控制，根据当前温度，动态给各模块分配 power，并将 power 转换为频率，从而达到根据温度限制频率的效果。
- step_wise ：根据当前温度，逐级限制频率。
- userspace：不限制频率。

3 配置方法

3.1 Tsadc 配置

3.1.1 Menuconfig 配置

make ARCH=arm64 menuconfig



3.1.2 DTS 配置

如下是芯片级的 DTSI 的配置：

```
tsadc: tsadc@ff260000 {
        compatible = "rockchip,rk3399-tsadc";
        reg = <0x0 0xff260000 0x0 0x100>;
        interrupts = <GIC_SPI 97 IRQ_TYPE_LEVEL_HIGH>;
        rockchip,grf = <&grf>;
        clocks = <&cru SCLK_TSADC>, <&cru PCLK_TSADC>;
        clock-names = "tsadc", "apb_pclk";
        assigned-clocks = <&cru SCLK_TSADC>;
        assigned-clock-rates = <750000>;
        resets = <&cru SRST_TSADC>;
        reset-names = "tsadc-apb";
        pinctrl-names = "init", "default", "sleep";
        pinctrl-0 = <&otp_gpio>;
        pinctrl-1 = <&otp_out>;
        pinctrl-2 = <&otp_gpio>;
        #thermal-sensor-cells = <1>;
        rockchip,hw-tshut-temp = <95000>;
        status = "disabled";
    };
```

```
tsadc: tsadc@ff260000 {
        compatible = "rockchip,rk3399-tsadc";
```

Tsadc 驱动加载的标识字符串

```
        reg = <0x0 0xff260000 0x0 0x100>;
```

Tsadc 操作的寄存器基地址和寄存器地址总长度

```
        interrupts = <GIC_SPI 97 IRQ_TYPE_LEVEL_HIGH>;
```

Tsadc 中断号

```
        rockchip,grf = <&grf>;
```

Tsadc 引用 grf 模块，驱动会操作配置 grf

```
        clocks = <&cru SCLK_TSADC>, <&cru PCLK_TSADC>;
        clock-names = "tsadc", "apb_pclk";
```

Tsadc 模块的两个 clock，其中"tsadc"是 Tsadc 的工作时钟，"apb_pclk"是 Tsadc 的配置时钟

```
        assigned-clocks = <&cru SCLK_TSADC>;
        assigned-clock-rates = <750000>;
```

配置 Tsadc 的工作时钟是 750000，Tsadc 的配置时间周期，是以这个时钟为基准的。

```
        resets = <&cru SRST_TSADC>;
        reset-names = "tsadc-apb";
```

Tsadc 的 reset 控制，用于 resett Tsadc 模块

```
        pinctrl-names = "init", "default", "sleep";
        pinctrl-0 = <&otp_gpio>;
        pinctrl-1 = <&otp_out>;
        pinctrl-2 = <&otp_gpio>;
```

Tsadc 工作的 GPIO 口配置，这个配置与下面的配置对应。初始化 Init 和休眠 Sleep 的时候，是 GPIO 口功能，默认 Default 的时候，是输出 Out 功能。

```
        tsadc {
            otp_gpio: otp-gpio {
                rockchip,pins = <1 6 RK_FUNC_GPIO &pcfg_pull_none>;
            };
            otp_out: otp-out {
                rockchip,pins = <1 6 RK_FUNC_1 &pcfg_pull_none>;
            };
        };
        #thermal-sensor-cells = <1>;
```

表示 Tsadc 可以向 Thermal Zone 注册，而且对应的 Thermal Zone 引用 Tsadc 的时候，带有一个参数。

```
如：thermal-sensors = <&tsadc 0>;
        rockchip,hw-tshut-temp = <95000>;
```

Tsadc 设定关机温度是 95000

```
        status = "disabled";
    };
```

注：

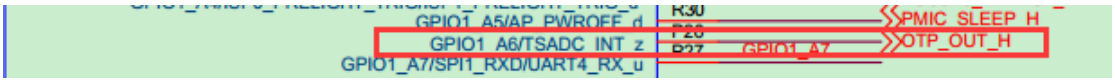
1. Tsadc 模块，默认在 DTSI 中是 Disabled 状态，要启用的时候，需要在板级的 DTS 中配置，如：

```
361 &tsadc {
362         rockchip,hw-tshut-mode = <1>; /* tshut mode 0:CRU 1:GPIO */
363         rockchip,hw-tshut-polarity = <1>; /* tshut polarity 0:LOW 1:HIGH */
364         status = "okay";
365 };
```

2. rockchip,hw-tshut-mode = <1>

配置温度超过关机温度的复位方式，配置 0 是通过复位 SoC 的 CRU 模块，配置 1 是通过配置

上文提到的 pinctrl-1 = <&otp_out>;来实现，这个引脚功能 otp_out 一般会接入 pmic 的 reset 引脚（如下图），至于这个引脚是高有效还是低有效，需要配置 rockchip,hw-tshut-polarity 来实现。



另外，不管配置 0 还是配置 1，这个复位方式都是硬件行为，这与下文介绍的 Thermal 配置的 soc_crit 节点不同。下文 soc_crit 节点表示温度超过 95 度，会自动重启系统，这种重启系统是软件行为，会走内核系统重启的标准流程。为防止系统重启丢失数据，rockchip,hw-tshut-temp 建议配置高于 soc_crit 节点配置的温度。

3.2 power_allocator 策略配置

3.2.1 默认使用 power_allocator 策略

make ARCH=arm64 menuconfig

```
[*] Networking support  --->
    Device Drivers  --->
    Firmware Drivers  --->
```

```
[*] Hardware Monitoring support  --->
[*] Generic Thermal sysfs driver  --->
[ ] Watchdog Timer Support  ----
```

```
--- Generic Thermal sysfs driver
[*]   Expose thermal sensors as hwmon device
[*]   APIs to parse thermal data out of device tree
[*]   Enable writable trip points
      Default Thermal governor (power_allocator)  --->
[*]   Fair-share thermal governor
[*]   Step_wise thermal governor
[ ]   Bang Bang thermal governor
[*]   User_space thermal governor
-*-   Power allocator thermal governor
[*]   generic cpu cooling support
[ ]   Generic clock cooling support
[*]   Generic device cooling support
[ ]   Thermal emulation mode support
[ ]   Temperature sensor driver for Freescale i.MX SoCs
[*]   Rockchip thermal driver
```

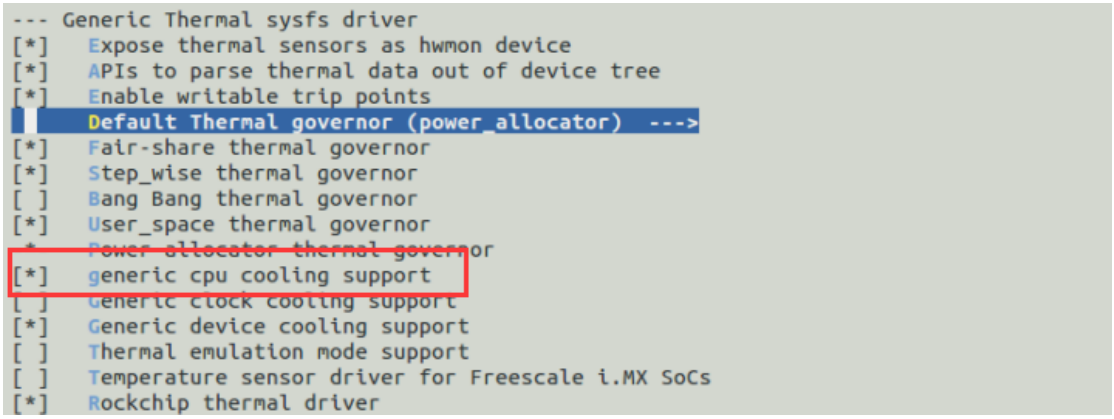
```
Default Thermal governor
Use the arrow keys to navigate this window or press the
hotkey of the item you wish to select followed by the <SPACE
BAR>. Press <?> for additional information about this

        ( ) step_wise
        ( ) fair_share
        ( ) user_space
        (X) power_allocator

        <Select>    < Help >
```

3.2.2 CPU 开启温控

3.2.3 menuconfig 配置



3.2.4 DTS 配置

```
cpu_l0: cpu@0 {
        device_type = "cpu";
        compatible = "arm,cortex-a53", "arm,armv8";
        reg = <0x0 0x0>;
        enable-method = "psci";
        #cooling-cells = <2>; /* min followed by max */
        dynamic-power-coefficient = <121>;
        clocks = <&cru ARMCLKL>;
        cpu-idle-states = <&cpu_sleep>;
        operating-points-v2 = <&cluster0_opp>;
        sched-energy-costs = <&CPU_COST_A53 &CLUSTER_COST_A53>;
    };
cpu_b0: cpu@100 {
        device_type = "cpu";
        compatible = "arm,cortex-a72", "arm,armv8";
        reg = <0x0 0x100>;
        enable-method = "psci";
        #cooling-cells = <2>; /* min followed by max */
        dynamic-power-coefficient = <1068>;
        clocks = <&cru ARMCLKB>;
        cpu-idle-states = <&cpu_sleep>;
        operating-points-v2 = <&cluster1_opp>;
        sched-energy-costs = <&CPU_COST_A72 &CLUSTER_COST_A72>;
    };
```

#cooling-cells

表示该设备可以作为一个 cooling 设备；频率随温度变化时，会限制在最大值和最小值之间。

dynamic-power-coefficient

动态功耗公式为 P= c * V^2 * f/ 1000000，dynamic-power-coefficient 是其中的参数 c，这个值跟芯片相关，由模块负责人实验数据得来，客户一般不需要修改。

3.2.5 GPU 开启温控

3.2.6 menuconfig 配置

```
--- Generic Thermal sysfs driver
[*]   Expose thermal sensors as hwmon device
[*]   APIs to parse thermal data out of device tree
[*]   Enable writable trip points
      Default Thermal governor (power_allocator)  --->
[*]   Fair-share thermal governor
[*]   Step_wise thermal governor
[ ]   Bang Bang thermal governor
[*]   User_space thermal governor
-*-   Power allocator thermal governor
[*]   generic cpu cooling support
[ ]   Generic clock cooling support
[*]   Generic device cooling support
[ ]   Thermal emulation mode support
[ ]   Temperature sensor driver for Freescale i.MX SoCs
[*]   Rockchip thermal driver
```

3.2.7 配置

```
gpu: gpu@ff9a0000 {
    compatible = "arm,malit860",
        "arm,malit86x",
        "arm,malit8xx",
        "arm,mali-midgard";

    reg = <0x0 0xff9a0000 0x0 0x10000>;

    interrupts = <GIC_SPI 19 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 20 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 21 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "GPU", "JOB", "MMU";

    clocks = <&cru ACLK_GPU>;
    clock-names = "clk_mali";
    #cooling-cells = <2>; /* min followed by max */
    operating-points-v2 = <&gpu_opp_table>;
    power-domains = <&power RK3399_PD_GPU>;
    power-off-delay-ms = <200>;
    status = "disabled";

    power_model {
        compatible = "arm,mali-simple-power-model";
        voltage = <900>;
        frequency = <500>;
        static-power = <300>;
        dynamic-power = <1780>;
        ts = <32000 4700 (-80) 2>;
        thermal-zone = "gpu-thermal";
    };
};
```

- #cooling-cells 属性表示该设备可以作为一个 cooling 设备；频率随温度变化时，会限制在最大值和最小值之间。
- voltage、frequency、static-power、dynamic-power 属性，这四个参数表示 GPU 在频率为 500MHz，电压为 900mv 下，全速运行时，静态功耗 300mW，动态功耗 1780mW，由模块负责人的实验实测所得，用于求出计算静态功耗和动态功耗所需的参数 c。

动态功耗公式：P(d)= c * v^2 * f/ 1000000

静态功耗公式：t_scale = (2* t^3) + (-80* t^2) + (4700*t) + 32000

v_scale = v^3 / 1000000

P(s)= c *t_scale*v_scale/1000000

- TS 属性，公式 t_scale = (2* t^3) + (-80* t^2) + (4700*t) + 32000 中的常数。
- Thermal-Zone 属性，指导 GPU 通过 gpu-thermal 获取温度，在 RK3399 上是 tsadc 1。

3.2.8 thermal_zone 配置

以 RK3399 的配置为例

```
thermal-zones {
        soc_thermal: soc-thermal {
            polling-delay-passive = <20>; /* milliseconds */
            polling-delay = <1000>; /* milliseconds */
            sustainable-power = <1600>; /* milliwatts */

            thermal-sensors = <&tsadc 0>;

            trips {
                threshold: trip-point@0 {
                    temperature = <70000>; /* millicelsius */
                    hysteresis = <2000>; /* millicelsius */
                    type = "passive";
                };
                target: trip-point@1 {
                    temperature = <85000>; /* millicelsius */
                    hysteresis = <2000>; /* millicelsius */
                    type = "passive";
                };
                soc_crit: soc-crit {
                    temperature = <95000>; /* millicelsius */
                    hysteresis = <2000>; /* millicelsius */
                    type = "critical";
                };
            };
```

```
            cooling-maps {
                map0 {
                    trip = <&target>;
                    cooling-device =
                        <&cpu_l0 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>;
                    contribution = <10240>;
                };
                map1 {
                    trip = <&target>;
                    cooling-device =
                        <&cpu_b0 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>;
                    contribution = <1024>;
                };
                map2 {
                    trip = <&target>;
                    cooling-device =
                        <&gpu THERMAL_NO_LIMIT THERMAL_NO_LIMIT>;
                    contribution = <10240>;
                };
            };
        };

        gpu_thermal: gpu-thermal {
            polling-delay-passive = <100>; /* milliseconds */
            polling-delay = <1000>; /* milliseconds */

            thermal-sensors = <&tsadc 1>;
        };
    };
```

Thermal-Zones 下的第一级子节点，对应的是不同的 Tsadc，RK3399 上有两个 Tsadc，一个在 CPU 旁边，一个在 GPU 旁边，所以我们可以看到两个子节点。

polling-delay-passive = <20>;

温度超过阀值时，每隔 20ms 查询温度，并限制频率。

polling-delay = <1000>;

温度未超过阀值时，每隔 1000ms 查询温度。

sustainable-power = <1600>;

当前温度到达预设的最高值时，系统能分配给 cooling 设备的能量。

thermal-sensors = <&tsadc 0>;

当前 thermal_zone 的温度是通过 tsadc0 获取的。

- trips
  - threshold 节点表示温度超过 70 度开始限制频率，type 要设置成"passive"；
  - target 节点表示系统的最高温度会 85 度左右，type 要设置成"passive"；
  - soc_crit 节点表示温度超过 95 度，自动重启系统，type 要设置成"critical"。
- cooling-maps
  - 子节点 map0、map1、map2 表示有 3 个设备作为 cooling 设备，即会根据温度被限制频率。

- cooling-device 属性

  以 RK3399 为例，子节点的 cooling-device 属性分别引用了&cpu_b0、&cpu_l0、&gpu，表示小核 A53、大核 A72 和 GPU 会根据温度限制频率。后面两个参数表示该设备允许被限制的最低频率和最高频率，填 THERMAL_NO_LIMIT 表示按最小最大值是频率电压表中的最小值和最大值。

- trip 属性

  对与 power_allocater 策略子节点的 trip 属性都设置 target。

注：

如果板子出现过温情况，采集过温时候的温度，并可以通过 2 种方式调整参数：

1. 调整 threshold 和 target 的温度参数，如由 70，85 度，同步调整为 65，80 度，建议 threshold 和 target 的温度保持 15 度的范围。
2. 调整 soc_crit 和上文中的 rockchip,hw-tshut-temp，如由 95 度调整为 100 度。

方法 1 会损失一些性能，方法 2 的芯片温度会较高。

4 调试接口

4.1 关温控

关温控其实是把策略切换到 user_space，以 RK3399 为例，在串口中输入如下命令：

```
echo user_space > /sys/class/thermal/thermal_zone0/policy
echo user_space > /sys/class/thermal/thermal_zone1/policy
```

4.2 获取当前温度

以 RK3399 为例，获取 CPU 温度，在串口中输入如下命令：

```
cat /sys/class/thermal/thermal_zone0/temp
```

获取 GPU 温度，在串口中输入如下命令：

```
cat /sys/class/thermal/thermal_zone1/temp
```