D1-H Linux CPUFREQ
开发指南

版本号: 1.0
发布日期: 2021.04.13

版本历史

版本号	日期	制/修订人	内容描述
1.0	2021.04.13	AWA1442	1. 添加初始版本

目　录

1 前言

1.1 文档简介

介绍 CPUFREQ 使用方法。

1.2 目标读者

CPUFREQ 驱动及应用层的使用人员。

1.3 适用范围

表 1-1：适用产品列表

产品名称	内核版本	驱动文件
D1-H	Linux-5.4	drivers/cpufreq/*

2 模块介绍

2.1 模块功能介绍

CPUFREQ 负责系统运行过程中 CPU 频率和电压的动态调整。

2.2 相关术语介绍

表 2-1: 术语介绍

术语	说明
Sunxi	指 Allwinner 的一系列 SOC 硬件平台。
DVFS	动态频率电压调整

2.3 模块配置介绍

2.3.1 Device Tree 配置说明

设备树中存在的是该类芯片所有平台的模块配置，设备树文件的路径为：kernel/linux-5.4/arch/riscv/boot/dts/sunxi/CHIP.dtsi(CHIP 为研发代号，如 sun20iw1p1 等)。

目前的 cpufreq 驱动支持两种方法进行 VF 表的兼容。

- 通过 opp-supported-hw 进行频点兼容的方法，
  v-f 表：

```
    cpu_opp_l_table: opp_l_table {
        compatible = "allwinner,sun50i-operating-points";
        nvmem-cells = <&speedbin_efuse>;
        nvmem-cell-names = "speed";
        opp-shared;

        opp@480000000-0 {
            opp-hz = /bits/ 64 <480000000>;
            opp-microvolt = <820000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
```

```
            opp-supported-hw = <0x3>;
        };
        opp@480000000-1 {
            opp-hz = /bits/ 64 <480000000>;
            opp-microvolt = <880000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x4>;
        };
        opp@600000000-0 {
            opp-hz = /bits/ 64 <600000000>;
            opp-microvolt = <820000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x3>;
        };
        opp@600000000-1 {
            opp-hz = /bits/ 64 <600000000>;
            opp-microvolt = <880000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x4>;
        };
        opp@792000000-0 {
            opp-hz = /bits/ 64 <792000000>;
            opp-microvolt = <860000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x3>;
        };
        opp@792000000-1 {
            opp-hz = /bits/ 64 <792000000>;
            opp-microvolt = <940000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x4>;
        };
        opp@1008000000-0 {
            opp-hz = /bits/ 64 <1008000000>;
            opp-microvolt = <900000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x3>;
        };
        opp@1008000000-1 {
            opp-hz = /bits/ 64 <1008000000>;
            opp-microvolt = <1020000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x4>;
        };
        opp@1200000000-0 {
            opp-hz = /bits/ 64 <1200000000>;
            opp-microvolt = <960000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x3>;
        };
        opp@1200000000-1 {
            opp-hz = /bits/ 64 <1200000000>;
            opp-microvolt = <1100000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x4>;
        };
        opp@1296000000 {
            opp-hz = /bits/ 64 <1296000000>;
            opp-microvolt = <1100000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
```

```
            opp-supported-hw = <0x2>;
        };
        opp@1344000000 {
            opp-hz = /bits/ 64 <1344000000>;
            opp-microvolt = <1120000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x4>;
        };
        opp@1512000000 {
            opp-hz = /bits/ 64 <1512000000>;
            opp-microvolt = <1100000>;
            clock-latency-ns = <244144>; /* 8 32k periods */
            opp-supported-hw = <0x1>;
        };
    };

compatible = "allwinner,sun50i-operating-points"; ：用于匹配驱动的属性。
opp-hz ：某个频点的频率。
opp-microvolt ：频率对应的电压。
opp@480000000-0、opp@480000000-1 ：后缀的0、1仅用于区分不同节点名字，以免报错。
opp-supported-hw:  选择该频点所支持的芯片版本。如“opp-supported-hw = <0x3>;”，表示该频点支持bit0、
    bit1所表示的芯片版本。详见内核文档Documentation/devicetree/bindings/opp/opp.txt关于opp-
    supported-hw的说明。
需要注意，因为不能存在多个相同频率的频点，所以要避免相同频率的频点都被选择的情况，如opp@480000000-0、
    opp@480000000-1不能被同一个芯片版本选择。
```

cpu 节点：

```
        CPU0: cpu@0 {
            device_type = "cpu";
            reg = <0>;
            status = "okay";
            compatible = "riscv";
            riscv,isa = "rv64imafdcvsu";
            mmu-type = "riscv,sv39";
            clocks = <&ccu CLK_RISCV>;
            clock-frequency = <24000000>;
            operating-points-v2 = <&cpu_opp_table>;
            cpu-idle-states = <&CPU_SLEEP>;
            #cooling-cells = <2>;
            CPU0_intc: interrupt-controller {
                #interrupt-cells = <1>;
                interrupt-controller;
                compatible = "riscv,cpu-intc";
            };
        };


operating-points-v2 = <&cpu_opp_l_table>;: 引用v-f表。
```

- 通过 opp-microvolt 进行频点兼容的方法，
  v-f 表：

```
        cpu_opp_l_table: opp_l_table {
                compatible = "allwinner,sun50i-operating-points";
                nvmem-cells = <&speedbin_efuse>, <&cpubin_efuse>;
                nvmem-cell-names = "speed", "bin";
                opp-shared;
```

```
                opp@408000000 {
                        opp-hz = /bits/ 64 <408000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-a0 = <900000>;
                        opp-microvolt-a1 = <900000>;
                        opp-microvolt-a2 = <900000>;
                        opp-microvolt-b0 = <900000>;
                        opp-microvolt-b1 = <900000>;
                };

                opp@600000000 {
                        opp-hz = /bits/ 64 <600000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-a0 = <900000>;
                        opp-microvolt-a1 = <900000>;
                        opp-microvolt-a2 = <900000>;
                        opp-microvolt-b0 = <900000>;
                        opp-microvolt-b1 = <900000>;
                };

                opp@816000000 {
                        opp-hz = /bits/ 64 <816000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-a0 = <940000>;
                        opp-microvolt-a1 = <900000>;
                        opp-microvolt-a2 = <900000>;
                        opp-microvolt-b0 = <900000>;
                        opp-microvolt-b1 = <900000>;
                };

                opp@1008000000 {
                        opp-hz = /bits/ 64 <1008000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-a0 = <1020000>;
                        opp-microvolt-a1 = <980000>;
                        opp-microvolt-a2 = <950000>;
                        opp-microvolt-b0 = <980000>;
                        opp-microvolt-b1 = <950000>;
                };

                opp@1200000000 {
                        opp-hz = /bits/ 64 <1200000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-a0 = <1100000>;
                        opp-microvolt-a1 = <1020000>;
                        opp-microvolt-a2 = <1000000>;
                        opp-microvolt-b0 = <1020000>;
                        opp-microvolt-b1 = <1000000>;
                };

                opp@1320000000 {
                        opp-hz = /bits/ 64 <1320000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-a0 = <1160000>;
                        opp-microvolt-a1 = <1060000>;
                        opp-microvolt-a2 = <1030000>;
                        opp-microvolt-b0 = <1060000>;
                        opp-microvolt-b1 = <1030000>;
                };
```

```
                opp@1416000000 {
                        opp-hz = /bits/ 64 <1416000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-a0 = <1180000>;
                        opp-microvolt-a1 = <1180000>;
                        opp-microvolt-a2 = <1130000>;
                        opp-microvolt-b0 = <1100000>;
                        opp-microvolt-b1 = <1070000>;
                };

                opp@1512000000 {
                        opp-hz = /bits/ 64 <1512000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-b0 = <1180000>;
                        opp-microvolt-b1 = <1130000 1130000 1140000>;
                };

                opp@1608000000 {
                        opp-hz = /bits/ 64 <1608000000>;
                        clock-latency-ns = <244144>; /* 8 32k periods */
                        opp-microvolt-b0 = <1180000>;
                        opp-microvolt-b1 = <1130000 1130000 1140000>;
                };
        };

compatible = "allwinner,sun50i-operating-points"; : 用于匹配驱动的属性。
opp-hz : 某个频点的频率。
opp-microvolt-x : 该频率下，x类型bin对应的电压。详见内核文档Documentation/devicetree/bindings/opp/
    opp.txt关于opp-microvolt-<name>的说明。
```

cpu 节点：

```
        CPU0: cpu@0 {
            device_type = "cpu";
            reg = <0>;
            status = "okay";
            compatible = "riscv";
            riscv,isa = "rv64imafdcvsu";
            mmu-type = "riscv,sv39";
            clocks = <&ccu CLK_RISCV>;
            clock-frequency = <24000000>;
            operating-points-v2 = <&cpu_opp_table>;
            cpu-idle-states = <&CPU_SLEEP>;
            #cooling-cells = <2>;
            CPU0_intc: interrupt-controller {
                #interrupt-cells = <1>;
                interrupt-controller;
                compatible = "riscv,cpu-intc";
            };
        };

operating-points-v2 = <&cpu_opp_l_table>;: 引用v-f表。
```

2.3.2 board.dts 配置说明

board.dts 用于保存每一个板级平台的设备信息（如 demo 板，perf1 板等），里面的配置信息会覆盖上面的 Device Tree 默认配置信息。

cpu 节点（具体参考相应的 dts 文件）：

```
&CPU0 {
    cpu-supply = <&reg_vdd_cpu>;
};
```

2.3.3 kernel menuconfig 配置说明

在命令行中进入 longan 目录，Linux-5.4 内核版本执行：./build.sh menuconfig 进入配置主界面，并按以下步骤操作。

cpufreq 驱动由于需要兼容不同版本的 VF 表，需要依赖于 nvmem 驱动获取版本号，需要先把 nvmem 驱动支持上：

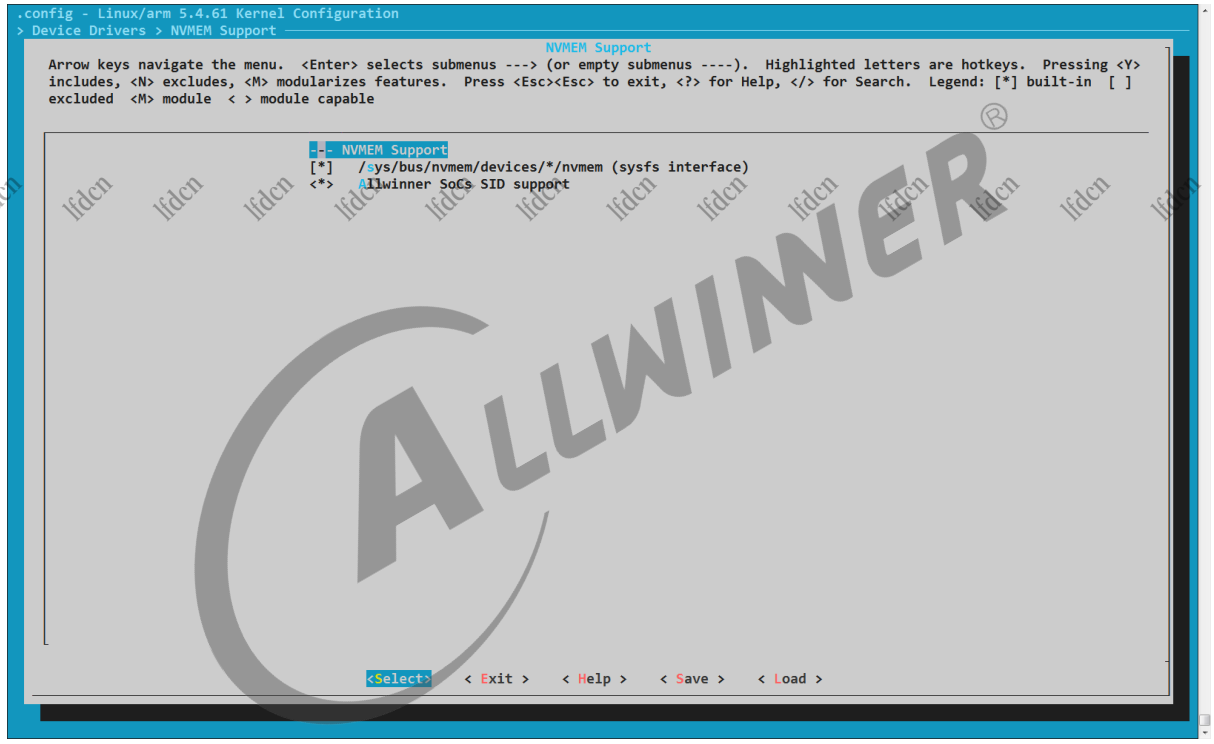

图 2-1: 配置图 1

cpufreq 驱动中，选择相关驱动及调频策略：

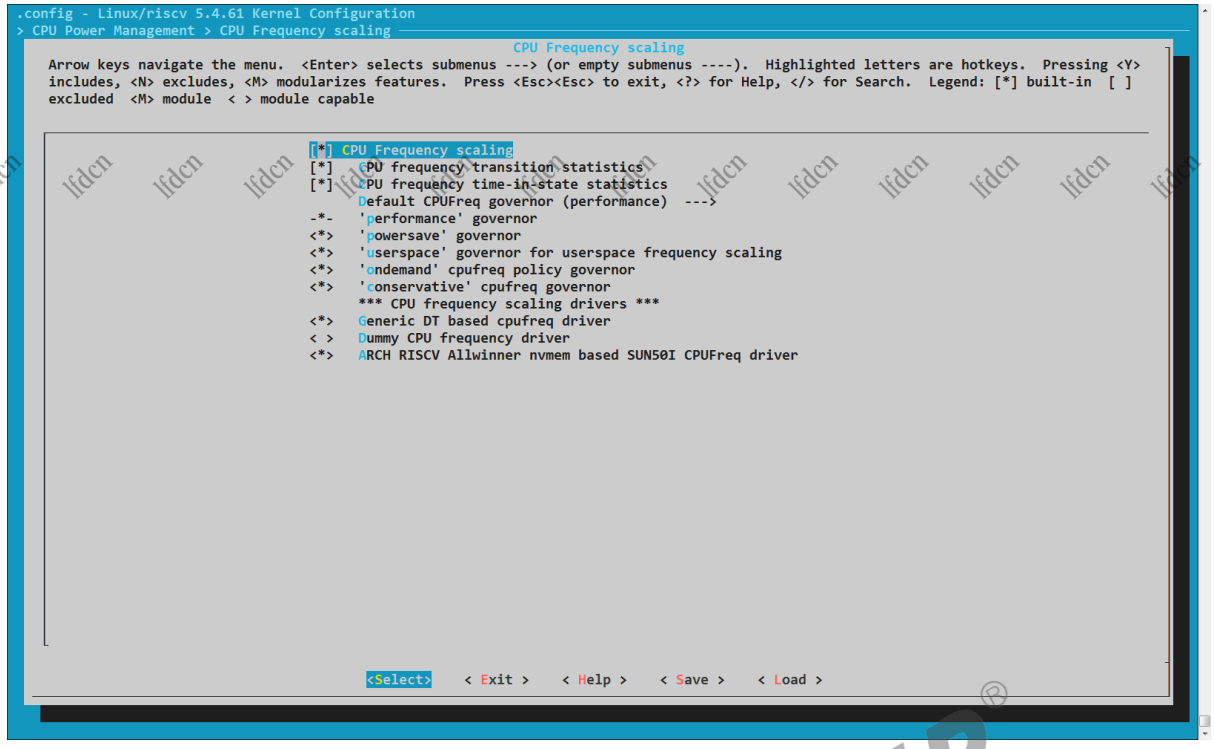

图 2-2: 配置图 2

2.4 源码结构介绍

CPUFREQ 的源代码位于内核 drivers/cpufreq/目录下：

```
drivers/cpufreq/
├── cpufreq-dt.c
├── cpufreq-dt-platdev.c
└── sun50i-cpufreq-nvmem.c
```

cpufreq-dt.c 为调频调压功能实现代码。
cpufreq-dt-platdev.c 为匹配没有 cpu 分 bin 需求的平台的代码。
sun50i-cpufreq-nvmem.c 为匹配有 cpu 分 bin 需求的平台的代码，它依赖于 nvmem 模块驱动提供芯片版本信息。

2.5 驱动框架介绍

无。

3 FAQ

3.1 调试方法

3.1.1 调试节点

节点	权限	说明
scaling_setspeed	R/W	设置频率的接口，仅当调频策略为 userspace 时可用
scaling_governor	R/W	调频策略
scaling_max_freq	R/W	软件调频最大频率
scaling_min_freq	R/W	软件调频最小频率
affected_cpus	R	受到该调频策略影响且在线的 cpu
related_cpus	R	受到该调频策略影响的所有 cpu
scaling_cur_freq	R	当前频率
scaling_driver	R	调频驱动名称
scaling_available_governors	R	可用调频策略
cpuinfo_transition_latency	R	频率转换延迟
cpuinfo_max_freq	R	硬件最大频率
cpuinfo_min_freq	R	硬件最小频率
cpuinfo_cur_freq	R	硬件实际运行频率

节点位于/sys/devices/system/cpu/cpufreq/policy0下

3.2 常见问题

3.2.1 调频策略使用说明

governor	说明
powersace	节能
performance	性能
userspace	由用户控制

governor	说明
ondemand	按需
conservative	保守，适用于带电池设备
schedutil	利用调度器提供信息进行调频，与 EAS 调度器一起配合使用

```
选择合适的governor，并勾选对应的调频驱动即可。

如果需要手动更改频率，选择governor为userspace。
echo userspace > /sys/devices/system/cpu/cpufreq/policy0/scaling_governor

需要注意，CPU调频功能会受到温控功能影响。所以如果有自主调频而不受温控影响的需求，要关闭温控功能。详细参考《
    thermal模块使用文档》。
```

3.2.2 怎样获取当前使用的电压频率表

方法一：使用 sunxi 自定义节点 cpufreq_table。
注意，这个节点并不是每个平台都支持，是需要 kernel 编译 drivers/soc/sunxi/vf-table.c 才有。

```
/ # mount -t debugfs none /sys/kernel/debug/
/ # cat /sys/kernel/debug/cpufreq_table
freq(kHz)       vol(mv)
-----------------------
408000          900
600000          900
816000          900
1008000         980
1200000         1020
1320000         1060
1416000         1180
```

方法二：使用内核原生的 opp 节点。

```
/ # mount -t debugfs none /sys/kernel/debug/


/* 对于Linux5.4 */
/* 获取所有频点的频率，单位是Hz */
/ # cat /sys/kernel/debug/opp/cpu0/opp*/rate_hz
1008000000
1200000000
1320000000
1464000000
408000000
600000000
816000000

/* 获取所有频点的电压，单位是mV */
/ # cat /sys/kernel/debug/opp/cpu0/opp*/supply-0/u_volt_target
1020000
1100000
1160000
1180000
```

```
900000
900000
940000
```

3.2.3 怎样修改电压频率表

方法一：直接修改 dts 文件中的 v-f 表，再重新编译固件。具体参考 Device Tree 配置说明。

方法二：在 uboot 修改 v-f 表，这样不需要重新编译固件。但是安全固件是不支持修改保存，所以机器重启后修改失效。

以某个平台为例，说明如何通过 uboot 修改 v-f 表。其他平台的节点路径和命名可能不同，具体参考 Device Tree 配置说明。

进入 uboot 命令行：

```
/* 获取v-f表 */
=> fdt list /opp_l_table
opp_l_table {
        compatible = "allwinner,sun50i-operating-points";
        nvmem-cells = <0x000000ff 0x00000100>;
        nvmem-cell-names = "speed", "bin";
        opp-shared;
        linux,phandle = <0x000000fc>;
        phandle = <0x000000fc>;
        opp@408000000 {
        };
        opp@600000000 {
        };
        opp@816000000 {
        };
        opp@1008000000 {
        };
        opp@1200000000 {
        };
        opp@1320000000 {
        };
        opp@1416000000 {
        };
        opp@1464000000 {
        };
        opp@1512000000 {
        };
        opp@1608000000 {
        };
};

/* 删除不想要的频点，如408MHz的频点 */
=> fdt rm /opp_l_table/opp@408000000

/* 保存修改，注意安全固件是不支持修改保存 */
=> fdt save

/* 从uboot启动到内核后，确认频点修改是否正确 */
/ # cat /sys/devices/system/cpu/cpufreq/policy0/scaling_available_frequencies
```

```
600000 816000 1008000 1200000 1320000 1464000
```

3.2.4 怎样验证 cpufreq 电压频率

先获取当前使用的电压频率表，再调节至某个频点，最后实测电压。

```
/ # echo userspace > /sys/devices/system/cpu/cpufreq/policy0/scaling_governor
/ # cat /sys/devices/system/cpu/cpufreq/policy0/scaling_available_frequencies
408000 600000 816000 1008000 1200000 1320000 1416000
/ # echo 1200000 > /sys/devices/system/cpu/cpufreq/policy0/scaling_setspeed
/ # cat /sys/devices/system/cpu/cpufreq/policy0/scaling_cur_freq
1200000
```

著作权声明

商标声明

免责声明