The thermal framework

A short status

Eduardo Valentin

August 20, 2015

Thermal problem overview

Problem Statement

Execute the system with maximum performance while sustaining device temperature constraints.

- Device temperature constraints may refer to device operating conditions.
- Temperature constraints may also be user safety protection.

Device operating conditions

- CPUs.
- Co-processors.
- Memories.
- Batteries.
- Peripherals (Display, Wifi, Media devices, Audio).

User safety protection

- Typically, maintain device surface temperature (or skin temperature) within comfortable level.
- Avoiding life hazards, e.g. battery explosions.

General solution

Typical solution

Monitor device(s) temperature(s) and trigger mitigation action(s) based on temperature level.

- Hardware based solutions. Transparent to the operating system.
- Hardware event based. Typically seen inside the operating system.
- However, there are solution in user space.

Thermal framework

The Linux thermal framework provides abstractions for solving this problem. Also enables solutions in kernel space and in user space.

Thermal framework concepts

- Thermal zones.
- Thermal sensors.
- Trip points.
- Thermal Cooling devices.
- Thermal Governors.

Thermal zones

Thermal zone

Thermal zones represent what is the current status of a thermal constrained zone in the hardware. The zone usually is a device or component. The status of a thermal zone is mainly with respect to temperature.

```
struct thermal_zone_device {
        int id;
        char type[THERMAL_NAME_LENGTH];
        struct device device;
...
};
```

Thermal sensors

Thermal sensors

Thermal sensors provide temperature sensing capabilities on thermal zones. Typical devices are I2C ADC converters and bandgaps.

```
struct thermal_zone_device {
        int id;
        char type[THERMAL_NAME_LENGTH];
        struct device device;
        ...
        int temperature;
        int last_temperature;
        int emul_temperature;
        ...
};
```

Trip points

Trip points

The trip node describes a point in the temperature domain in which the system takes an action. This item describes just the point, not the action. Trip points are represented as temperature in miliCelsius.

```
struct thermal_trip {
        struct device_node *np;
        unsigned long int temperature;
        unsigned long int hysteresis;
        enum thermal_trip_type type;
};
```

Thermal cooling device

Thermal cooling device

Cooling devices provide control on power dissipation. There are essentially two ways to provide control on power dissipation. First is by means of regulating device performance, which is known as passive cooling (DVFS).Second is by means of activating devices in order to remove the dissipated heat, which is known as active cooling (Fans).

```
struct thermal_cooling_device {
    int id;
    char type[THERMAL_NAME_LENGTH];
    struct device device;
    struct device_node *np;
    void *devdata;
    const struct thermal_cooling_device_ops *ops;
    bool updated; /* true if the cooling device does not nee
    struct mutex lock; /* protect thermal_instances list */
    struct list_head thermal_instances;
    struct list_head node;
};
```

Thermal Governors

Thermal Governor

Thermal Governors represent a policy to manage the thermal zone device temperature. The governor targets to keep temperature in an acceptable range which correlates to the power budget, while maximizing the performance. Governors can be implemented in Kernel Space or in User Space.

```
struct thermal_governor {
        char name[THERMAL_NAME_LENGTH];
        int (*bind_to_tz)(struct thermal_zone_device *tz);
        void (*unbind_from_tz)(struct thermal_zone_device *tz);
        int (*throttle)(struct thermal_zone_device *tz,
                        int trip);
        struct list_head governor_list;
};
```

SoC thermal zones support

- Original design of the thermal framework targeted ACPI systems.
- Initial non-ACPI driver: SPEAr13xx. Followed by: Samsung Exynos and TI OMAP
- Current support:
  - ACPI based systems.
  - Marvell Armada.
  - ST DB8500.
  - Marvell Dove.
  - Hisilicon.
  - Freescale i.MX SoCs.
  - Intel 340x, SOC DTS (Quark, IOSF), PKG.
  - Marvel Kirkwood.
  - QCOM SPMI.
  - Renesas Rcar.
  - Rockchip rk3288.
  - Samsung Exynos (exynos3250, exynos4210, exynos4412, exynos5250, exynos5260, exynos5420, exynos5433, exynos5440, and exynos7).
  - TI OMAP (omap4430, omap4460, omap4470, omap5430) and DRA7.
  - ST Spear1340
  - STMicroelectronics (stih415-sas, stih415-mpe, stih416,stid127)
  - Nvidia Tegra124

Cooling devices

- cpu_cooling: exposes throttling CPU frequency (DVFS).
- clock_cooling: adds limits on a clock.
- Intel powerclamp: allow cpu idle cycles injections.
- ACPI (fan, LCD, CPU)
- GPIO-fan
- PWM-fan
- Battery.

Governors

- step_wise: Open loop control. Temperature threshold and trend based. Walk through each cooling device cooling state, step by step.
- fair_share: Weight based. Determine the cooling device state based on assigned weight partitioning.
- bang_bang: uses a hysteresis to switch abruptly on or off a cooling device. It is intended to control fans, which can not be throttled but just switched on or off.
- power_allocator: Closed loop control. Based on power budget, temperature, and current power consumption of each involved device.
- user_space: hand off the control of a thermal zone to user space. Example: thermald and iTux.

OF thermal (1/4)

Hardware thermal constraints in DT

Describes hardware constraints using Device Tree.

```
        cpu0: cpu@0 {
                ...
                operating-points = <
                        /* kHz    uV */
                        970000  1200000
                        792000  1100000
                        396000  950000
                        198000  850000
                >;
                cooling-min-state = <0>;
                cooling-max-state = <3>;
                #cooling-cells = <2>; /* min followed by max
        };
        ...
};
```

OF thermal (2/4)


```
&i2c1 {
        ...
        /*
         * A simple fan controller which supports 10 speeds
         * (represented as 0-9).
         */
        fan0: fan@0x48 {
                ...
                cooling-min-state = <0>;
                cooling-max-state = <9>;
                #cooling-cells = <2>; /* min followed by max
        };
};

ocp {
        ...
        /*
         * A simple IC with a single bandgap temperature sen
         */
        bandgap0: bandgap@0x0000ED00 {
                ...
                #thermal-sensor-cells = <0>;
        };
```

OF thermal (3/4)


```
thermal-zones {
        cpu_thermal: cpu-thermal {
                polling-delay-passive = <250>; /* millisecon
                polling-delay = <1000>; /* milliseconds */
                thermal-sensors = <&bandgap0>;
                trips {
                        cpu_alert0: cpu-alert0 {
                                temperature = <90000>; /* m
                                hysteresis = <2000>; /* mil
                                type = "active";
                        };
                        cpu_alert1: cpu-alert1 {
                                temperature = <100000>; /*
                                hysteresis = <2000>; /* mil
                                type = "passive";
                        };
                        cpu_crit: cpu-crit {
                                temperature = <125000>; /*
                                hysteresis = <2000>; /* mil
                                type = "critical";
                        };
                };
```

OF thermal (4/4)


```
thermal-zones {
                    ...
                    cooling-maps {
                            map0 {
                                    trip = <&cpu_alert0>;
                                    cooling-device = <&fan0 THER
                            };
                            map1 {
                                    trip = <&cpu_alert1>;
                                    cooling-device = <&fan0 5 TH
                            };
                            map2 {
                                    trip = <&cpu_alert1>;
                                    cooling-device =
                                        <&cpu0 THERMAL_NO_LIMIT
                            };
                    };
            };
};
```

Thermal Framework Limitation

- Sensor API
  - Having extrapolation of hostspots is a common design need due to sensor location positioning.
  - Linear extrapolation typically uses one or more sensors.
  - Thermal zones today use a single temperature sensor source. Still doable by means of virtual sensor constructions.
- Tools
  - tmon: tools/thermal/tmon/
  - tracing: thermal:

Thermal Framework Limitation

- Sensor API
  - Having extrapolation of hostspots is a common design need due to sensor location positioning.
  - Linear extrapolation typically uses one or more sensors.
  - Thermal zones today use a single temperature sensor source. Still doable by means of virtual sensor constructions.
- Tools
  - tmon: tools/thermal/tmon/
  - tracing: thermal:
- Devices vs. Drivers
  - Today devices are added by driver code. This design is not common across kernel code.
  - Makes things a little bit challenging, e.g. of-thermal interaction with sensor driver.
  - Split between driver and devices.

Documentation

- Documentation/thermal/*
- Documentation/thermal/sysfs-api.txt
- Documentation/devicetree/bindings/thermal/*
- Documentation/devicetree/bindings/thermal/thermal.txt
- WiP: thermal-docbook.

Contacts

- linux-pm@vger.kernel.org
- ##thermal freenode
- edubezval at_ gmail.com