Cấu trúc Frequency Governor

1. Khái niệm về CPU Performance Scaling:

- Các mức Frequency tuân theo chuẩn ACPI, được chia thành các Operating Performance Points hoặc P-states.
- Có Hardware interfaces kết hợp với các thuật toán để tính toán CPU capacity cần thiết, từ đó quyết định chế độ.

2. Performance Scaling trong Linux:

- Interface CPUFreq subsystem gồm 3 layer code:
    + Core.
    + Scaling governors.
    + Scaling drivers.
- CPUFreq core:
    + Có code infrastructure và user space interfaces cho tất cả platforms hỗ trợ CPU performance scaling.
    + Tạo nên basic framework.
- Scaling governors:
    + Thực thi các algorithms tính toán target CPU capacity.
    + Có nhiều governors.
    + Mỗi governors thực thi 1 thuật toán được tham số hóa.
- Scaling drivers:
    + Giao tiếp với phần cứng.
    + Cung cấp inputs cho scaling governors thông tin về các P-states khả dụng.
    + Truy cập phần cứng để chuyển các mức P-states the lệnh từ Governors.
- Các governors có thể dùng với mọi driver nếu thuật toán điều chỉnh hiệu năng dựa vào thông tin độc lập với phần cứng.
- Nếu thuật toán dựa vào thông tin phần cứng (như feedback registers).
- Do đó CPUFreq cho phép driver bỏ qua governor và tự triển khai thuật toán điều chỉnh hiệu năng.
- Các **Governors** trong Linux:
    + Performance.
    + Powersave.
    + On-demand.
    + Conservative.
    + Sched-util.

Các thuật toán Governor trên đều quan sát một chỉ số gọi là **CPU load**.

- CPU load:
    + Có thể tính thời gian CPU idle.
    + Tính bằng **kcpustat.**

```c
int cpu;
u64 total_idle = 0, total_time = 0;
for_each_online_cpu(cpu) {
    u64 wall, idle;
    idle = get_cpu_idle_time(cpu, &wall, 0);
    total_idle += idle;
    total_time += wall;
}
```

3. CPUFreq Policy Objects

- Trong một số trường hợp, P-state control cho các CPU chia sẻ cùng một giao diện phần cứng (ví dụ như cùng một tập các thanh ghi), điều khiển nhiều CPU cùng lúc. Tức là khi viết lệnh vào thanh ghi thì các CPU sẽ cùng bị ảnh hưởng.
- Một nhóm các CPU có cùng giao diện phần cứng điều khiển P-state được CPUFreq biểu diễn thành các objects **struct cpufreq_policy.**
- Objects vẫn được duy trì mà ko bị giải phóng ngay cả khi chỉ có một cpu hoạt động.
- CPUFreq core duy trì một pointer về **struct cpufreq_policy** cho tất cả CPU, bao gồm cả các CPU ko hoạt động.
- Các CPU cùng hardware P-state control interface dùng chung một object policy.

```c
struct cpufreq_policy {
    unsigned int cpu;               // CPU chính đại diện
    cpumask_t cpus;                 // Các CPU dùng chung policy này
    unsigned int min;               // Tần số tối thiểu
    unsigned int max;               // Tần số tối đa
    unsigned int cur;               // Tần số hiện tại
    struct cpufreq_governor *governor;  // Governor được chọn
    ...
};
```

4. Cơ chế khởi tạo CPU:

- Bước 1: Đăng ký scaling driver cho CPUFreq. Chỉ đăng ký được 1 scaling driver. Do đó scaling driver đó handle tất cả CPUs. Nếu driver đăng ký trước CPU thì CPUFreq ghi nhận CPU mới khi nó khởi tạo. Nếu CPU đăng ký trước, CPUFreq sẽ quét lại và ghi nhận các CPU đã có.

- Bước 2: CPUFreq xử lý từng CPU. Nếu chưa có policy nào cho CPU: Tạo mới policy -> Tạo thư mục policy -> Gán con trỏ policy cho CPU đó (và các CPU liên quan nếu chung P-state hardware).
- Bước 3: Gọi hàm init() của scaling driver

    ->init(struct cpufreq_policy *policy)

Có nhiệm vụ cấu hình giao diện phần cứng cho P-state
Bao gồm: min_freq, max_freq, bảng tần số hỗ trợ nếu ko liên tục. policy->cpus (bitmap CPU nào thuộc group chia sẻ P-state).Gắn con trỏ policy cho các CPU liên quan.
- Bước 4: Gắn governor mặc định vào policy

    + governor->init(policy) cấp phát biến nội bộ cho governor.

    + governor->start(policy):

    Đăng ký callback cập nhật utilization.

    Callback này tính toán P-state cần thiết, gọi scaling driver để đổi tần số.
- Bước 5: Khi CPU online lại

Nếu policy cũ vẫn còn:

- Gọi `governor->stop()` rồi `governor->start()` để cập nhật CPU mới online.

Nếu tất cả CPU trong policy offline trước đó:

- Gọi lại các bước `init/start` như trên.

5. Policy interface trong sysfs

Trong init kernel, CPUFreq core tạo một sysfs directory `/sys/devices/system/cpu/`

Đường dẫn bao gồm các subdirectory *policyX* được symbolic links trong thư mục /sys/devices/system/cpu/cpuY/cpufreq/

Các thư mục `policyX` trong `/sys/devices/system/cpu/cpufreq/` chứa các **thuộc tính cụ thể của policy** (dưới dạng file), dùng để:

- **Điều khiển hành vi CPUFreq**
- **Cho toàn bộ nhóm CPU** được liên kết với policy đó.