Multi PM domains per device - yes!

Ulf Hansson
Agenda

● Why do we need it?
● Overview of implementation
● How to use it - device links!?
QCOM APQ8096 - Cam.subsystem

PM Domain
VIDEO FRONTEND H
(high resolution)

PM Domain
VIDEO FRONTEND L
(low resolution)

CAMERA SUBSYSTEM
Tegra124/210 XUSB subsystem

PM Domain
SUPERSPEED LOGIC (USB 3.0)

PM Domain
DEVICE CONTROLLER

PM Domain
HOST CONTROLLER

XUSB SUBSYSTEM
Main constraints

- Consumer driver needs control of each PM domain.
- Not another API to be called from PM callbacks!
  - Like “pm_domain_enable|disable()” - NAK!
- We have one struct dev_pm_domain * per device.

-> Device link + virtual device
The APIs to use

**Single PM domain (automagic attach by buses - power on):**

```c
int dev_pm_domain_attach(struct device *dev, bool power_on)
```

**Multi PM domain (manual by consumers - no power on):**

```c
struct device *dev_pm_domain_attach_by_id(struct device *dev, unsigned int index)
```

```c
struct device *dev_pm_domain_attach_by_name(struct device *dev, char *name)
```

```c
struct device_link *device_link_add(struct device *consumer, struct device *supplier, u32 flags)
```

```c
void device_link_del(struct device_link *link)
```

-> Use runtime PM - as before
Devicetree bindings

```plaintext
power: power-controller@12340000 {
    compatible = "foo,power-controller";
    reg = <0x12340000 0x1000>;
    #power-domain-cells = <1>;
};
leaky-device@12351000 {
    compatible = "foo,i-leak-current";
    reg = <0x12351000 0x1000>;
    power-domains = <&power 0>, <&power 1> ;
    power-domain-names = "io", "clk";
};
```
1: Extend the data structure

```c
struct my_drv_data {
    Driver specific struct
    struct device *dev;
    ..... 
    struct device *genpd_io;
    struct device *genpd_clk;
    struct device_link *link_io;
    struct device_link *link_clk;
};
```
2: Attach and probe

drv_data->genpd_io = dev_pm_domain_attach_by_name(&pdev->dev, "io");
drv_data->genpd_clk = dev_pm_domain_attach_by_name(&pdev->dev, "clk");

drv_data->link_io = device_link_add(&pdev->dev, drv_data->genpd_io,
DL_FLAG_STATELESS | DL_FLAG_PM_RUNTIME | DL_FLAG_RPM_ACTIVE);
3: Adopt to running use case

device_link_del(drv_data->link_io);

drv_data->link_clk = device_link_add(&pdev->dev, drv_data->genpd_clk,
    DL_FLAG_STATELESS | DL_FLAG_PM_RUNTIME | DL_FLAG_RPM_ACTIVE);