Active state management of PM domains

Viresh Kumar (PMWG)

Topics

- Problem description
- Proposed solution
- New genpd helpers
- New callbacks in “struct genpd”

Problem description

- An external entity (Cortex-M core running firmware) controls voltage rails on behalf of devices.
- That is represented as a power domain within Linux kernel.
- The firmware accepts integer values corresponding to a performance level.
- Each device under the power domain votes for a performance level.
- The highest level is selected by the M3 core for the rails.

- We can’t represent this as a regulator as we aren’t dealing with voltages.
- There can be more cases later, which handle more than just regulator.
- No existing solution works.

Proposed solution

- Kernel's power domain framework only does idle state management currently.
- We shouldn't create something new, which is almost copy of genpd.

- Extend genpd to do active state management as well.
- And provide new API/Interface for that.
- Begin with non DT solution.
- The proposed solution is currently available on LKML and isn't merged yet:

  [PATCH V9 0/7] PM / Domains: Performance state support
  https://marc.info/?l=linux-kernel&m=150166878010642&w=2

New genpd helpers

- Two new helper functions implemented in genpd core.
- Used by code that need to change perf state of a device, like OPP core.

  - int dev_pm_genpd_has_performance_state(struct device *dev)
    - Drivers can call this (At least once) to verify if performance state is supported by genpd.
    - It is optional, if the driver can guarantee that genpd do support performance states.

  - int dev_pm_genpd_update_performance_state(struct device *dev, unsigned long rate)
    - Called by drivers to change performance state of a device.
    - Genpd core finds a performance level corresponding to “rate” for device “dev” and updates the performance state of the genpd.
    - “rate” can be set to zero to remove device’s constraints.
    - It is synchronous, i.e. the state is changed when this call returns.
    - This is also propagated by the genpd core to the masters of genpd of “dev”.

New callbacks in “struct genpd”

- Two new callbacks are added in “struct genpd”.
- Provided by platform specific genpd drivers that support performance states.
- Callbacks not required if performance states aren’t supported.
- Called in following order from within dev_pm_genpd_update_performance_state().

    - int (*dev_get_performance_state)(struct device *dev, unsigned long rate);

        - Called to get performance state (integer value) corresponding to target frequency.
        - This state is used by the genpd core as device's requested performance state.
        - The same state value is used for all masters in master hierarchy.
        - Mandatory for genpd that have slave devices.
        - Not required for genpd that only have subdomains and no slave devices.

New callbacks in “struct genpd” (Cont..)

- int (*genpd_set_performance_state)(struct generic_pm_domain *genpd, unsigned int state).
  - Called to set performance state of a genpd.
  - “state” is aggregate (max) of performance states of slave devices and subdomains of genpd.
  - “state” can be zero, if we want to remove all performance constraints.
  - The same callback, if present, is also called for masters of “genpd” to propagate the request.
  - It is optional.
  - At least one genpd in the master hierarchy of device shall have this set.

Linaro connect
San Francisco 2017

Thank You

#SFO17

BUD17 keynotes and videos on: connect.linaro.org

For further information: www.linaro.org