BKK19-PM02
PMWG: Cluster Idling

Ulf Hansson, PMWG, Linaro
ulf.hansson@linaro.org
Agenda

- Why do we need it.
- Update on latest achievements.
- Next steps.
CPUs and other resources sharing idle states - doesn’t fit with the CPUIdle framework.
Why?

- In some cases, last-man activities can't solely be managed by FW.
  - Configure external logics for wakeups, decouple the GIC.
  - Sleep states of devices communicated to FW.

- Power-rails for a cluster may be shared with other resources.
How?

- Extend the generic PM domain to cope with CPU devices.
- Invent a genpd governor to deal better with shared idle states.
- Keep using CPUIdle for CPU’s idle states.
- Deploy for PSCI with a DT based platform.
  - Hikey, 410c.
Status - basic functionality

- [PATCH v13 0/4] PM / Domains: Support hierarchical CPU arrangement (PSCI/ARM)
  - Reviewed!
- [PATCH v2 0/5] drivers: firmware: psci: Some cleanup and refactoring
  - Reviewed!
- PSCI/ARM64 deployment
  - Additional ~18 patches

git.linaro.org/people/ulf.hansson/linux-pm.git next - Based on RCs

git.linaro.org/people/ulf.hansson/linux-pm.git next_v13 - Based on linux.pm “next”
Status - important recent changes

- PSCI DT bindings are orthogonal to the PSCI CPU suspend mode.
  - OSI mode is optional and support is announced by the PSCI FW.

- OSI mode - the new genpd gov selects the idle state for the cluster.

- PC mode - CPUIdle selects (as before) the idle state for the cluster.
Future improvements 1/7

Deal with last-man activities

- cpu_pm_enter|exit()
  - Do we need to deliver it for all idle states?

- cluster_pm_enter_exit()
  - Do we need to deliver it for all idle states?

Scalability and latency...

- Let’s explore genpd “on/off” notifiers and let it be described via DT.
Future improvements 2/7

Fixup support for suspend-to-idle

- Runtime PM is disabled - genpd’s ->suspend_noirq() callbacks may not invoke
  ->power_off() callback with the last man, as it depends on device order in the dpm
  list.

Solutions?

- Make CPU devices as syscore device and make use of
  pm_genpd_syscore_poweroff()|poweron().
  - Problem no locking in genpd - breaks last man.
- Special runtime PM treatments for CPUs, must work when runtime PM is disabled.
  - pm_runtime_force_suspend|resume() - or other...
  - genpd_power_off() - aborts when any device is “prepared”...
Future improvements 3/7

Decrease `ktime_get()` in genpd due to overhead in the idle path.

- `ktime_get()` is called by genpd for each runtime PM suspend/resume.
- `ktime_get()` is called by genpd at each power off/on.
- `ktime_get()` is called by the genpd governor at last man point.

Optimizations?

- Avoid calling `ktime_get()` in genpd to measure latencies?
  - Convert to `ktime_get_mono_fast_ns()`.
  - Introduce a calibration phase instead of always measuring.
  - Don’t measure for CPUs and and corresponding genpd?
Future improvements 4/7

Decrease impact by `time_get_mono_fast_ns()` in runtime PM core in the idle path.

- `ktime_get_mono_fast_ns()` is called several times for each runtime PM transition (RPM_ACTIVE, RPM_SUSPENDED, RPM_SUSPENDING, RPM_RESUMING).

Optimizations?
- Call `ktime_get_mono_fast_ns()` only at RPM_ACTIVE and RPM_SUSPENDED.
  - Align with the behaviour of genpd?
Future improvements 5/7

Decrease overhead in runtime PM core for CPUs in the idle path.

- Lot of unnecessary code becomes executed when `pm_runtime_get|put_sync()` is called for CPU devices. How much is the overhead?

Optimizations?

- Invent specific runtime PM APIs which decreases the overhead.
  - In principle `pm_runtime_force_suspend|resume()`.
Future improvements 6/7

Improve the genpd governor used for CPUs.

- The next timer event is far from sufficient.

Use information about the next predicted IRQ and the next IPI.

- Based on work in progress, but we can run some tests already at this point!
Future improvements 7/7

Support greater than 2 levels of cluster hierarchy

- Gendp supports multiple idle states, but has only GPD_STATE_POWER_ON/OFF.
  - GPD_STATE_POWER_OFF == any of the idle state has been selected.

Master domains allows to be “powered off” even if a subdomain isn't in the deepest idle state.
Thank you

Join Linaro to accelerate deployment of your Arm-based solutions through collaboration

contactus@linaro.org