Towards Tighter Integration of The System-wide and Runtime PM of Devices

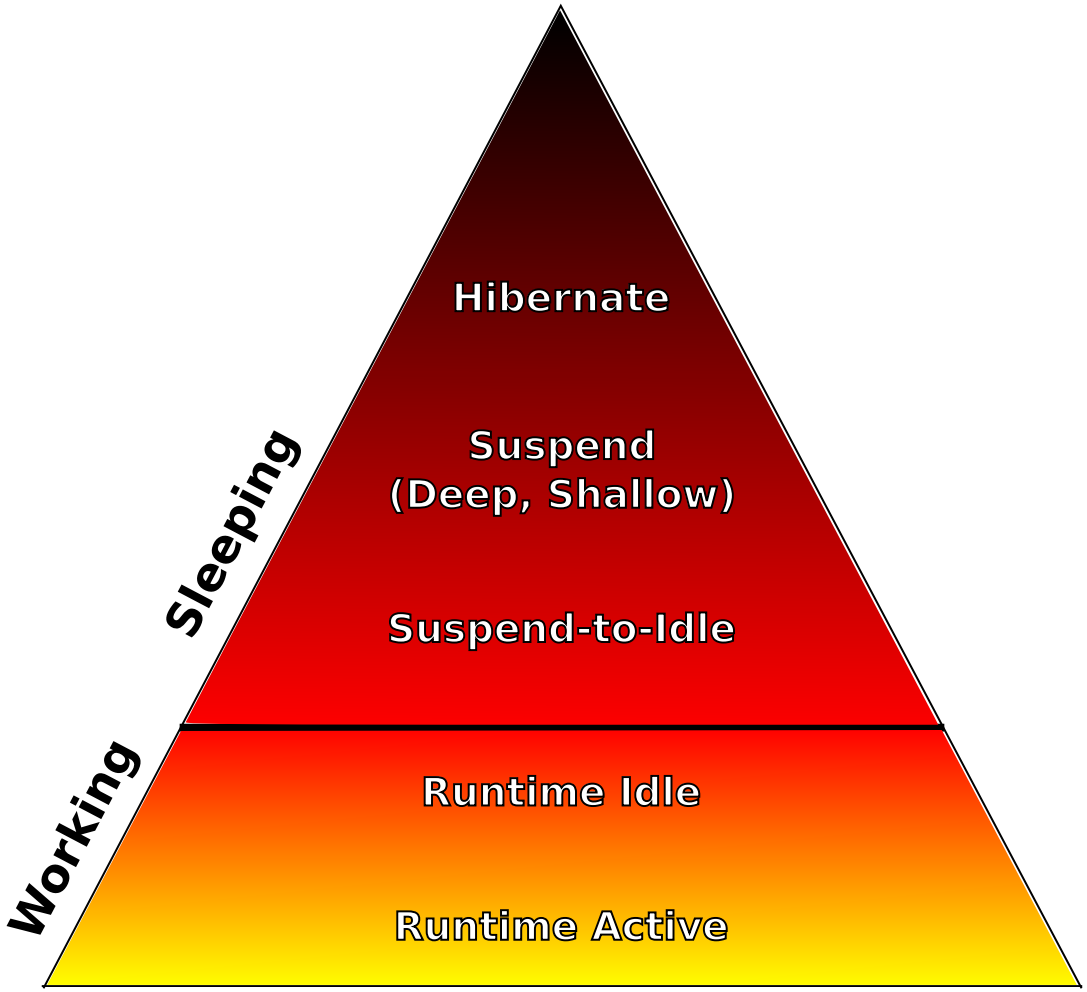
Rafael J. Wysocki

Intel Open Source Technology Center

April 16, 2018

Intel OpenSource TECHNOLOGY CENTER

Linux* Power Management Overview



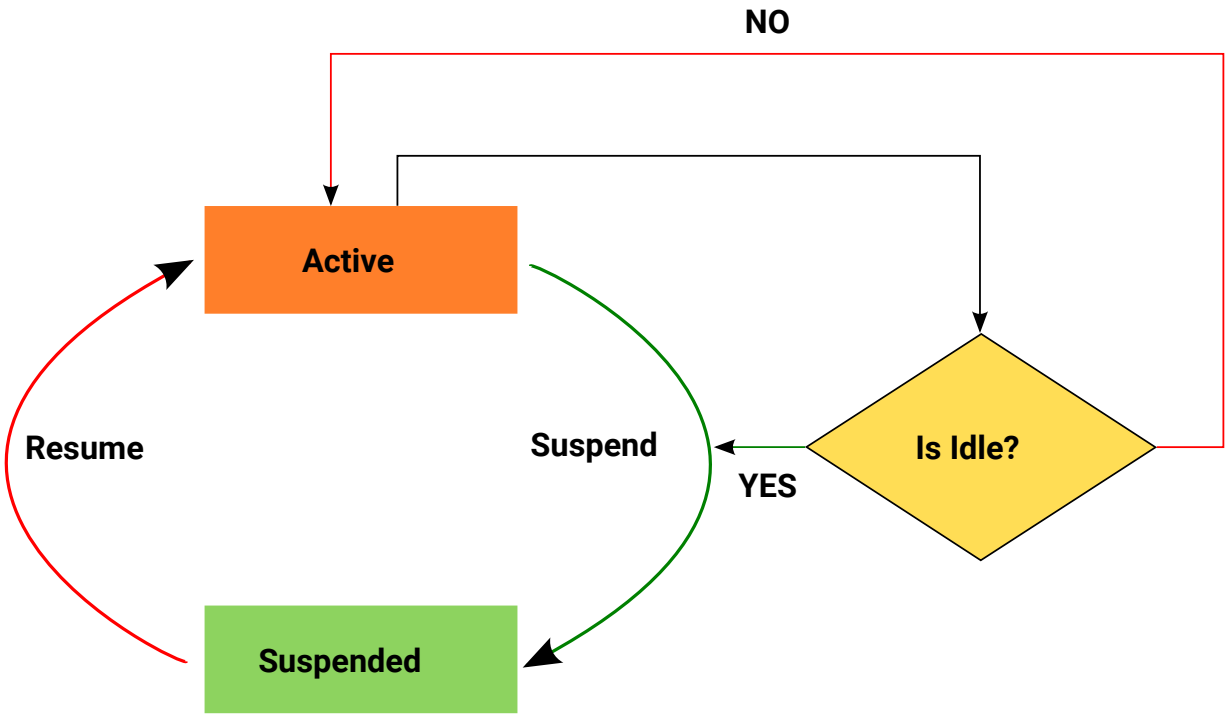
Intel OpenSource TECHNOLOGY CENTER

* Other names and brands may be claimed as the property of others

System-Wide PM (Sleep States) vs Working-State PM

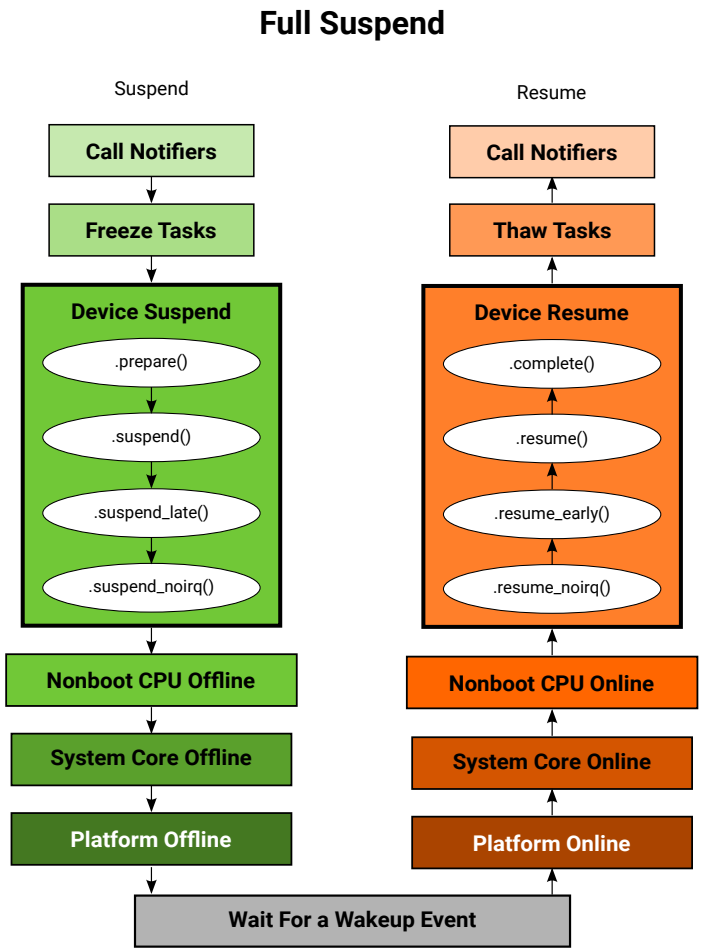
Intel OpenSource TECHNOLOGY CENTER

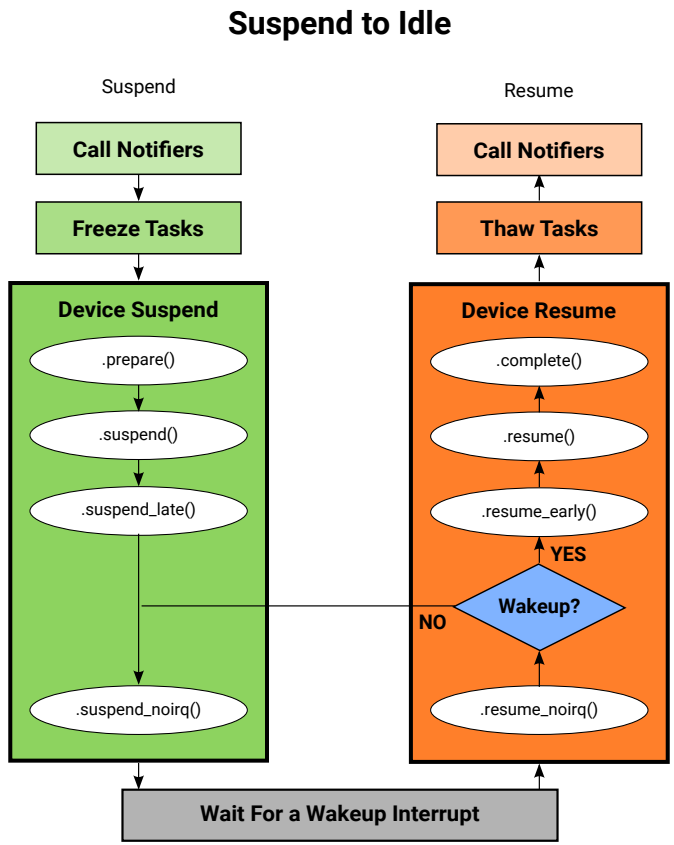
Device Runtime PM Control Flow



Intel OpenSource TECHNOLOGY CENTER

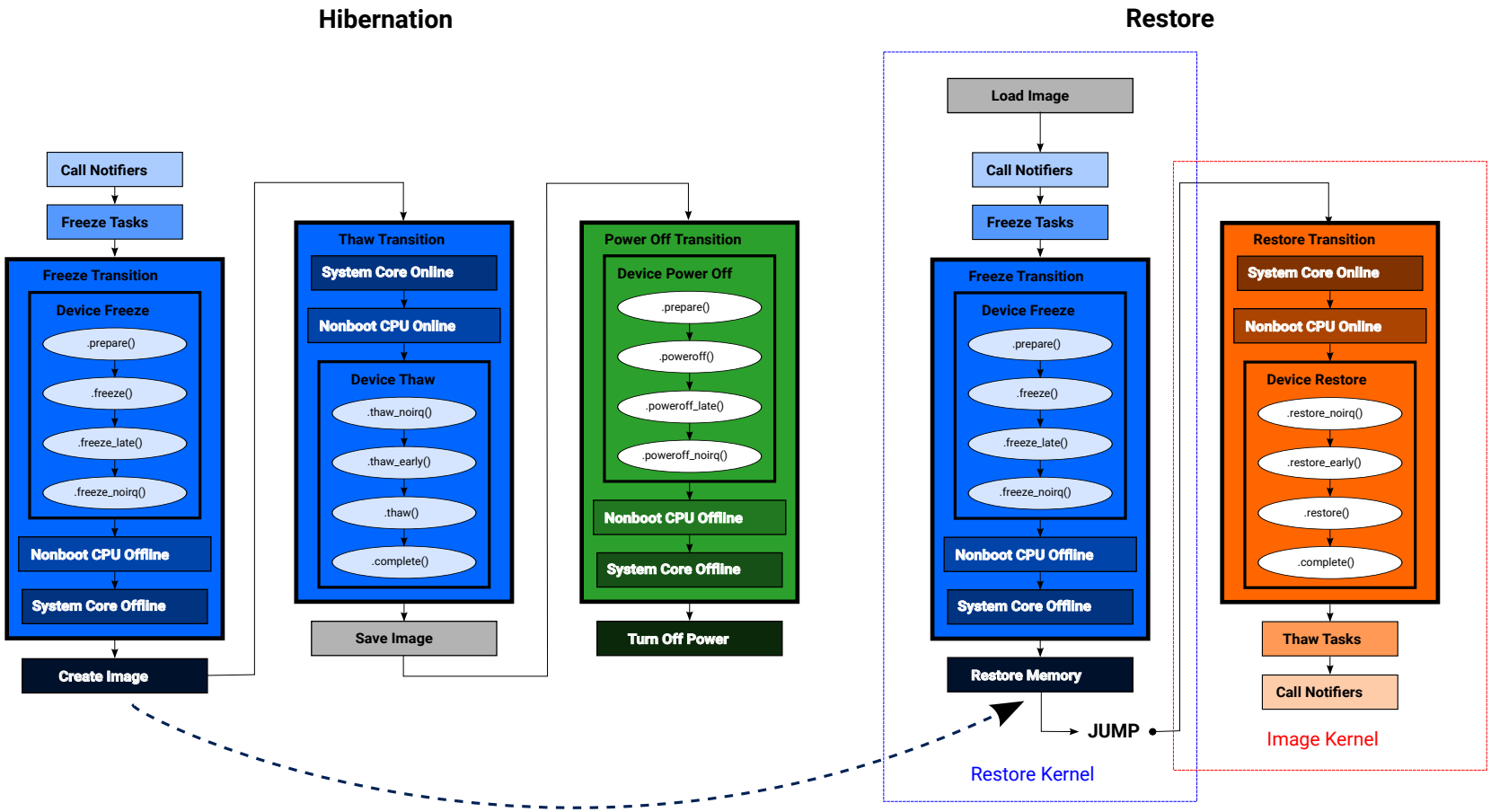
System Suspend Control Flows





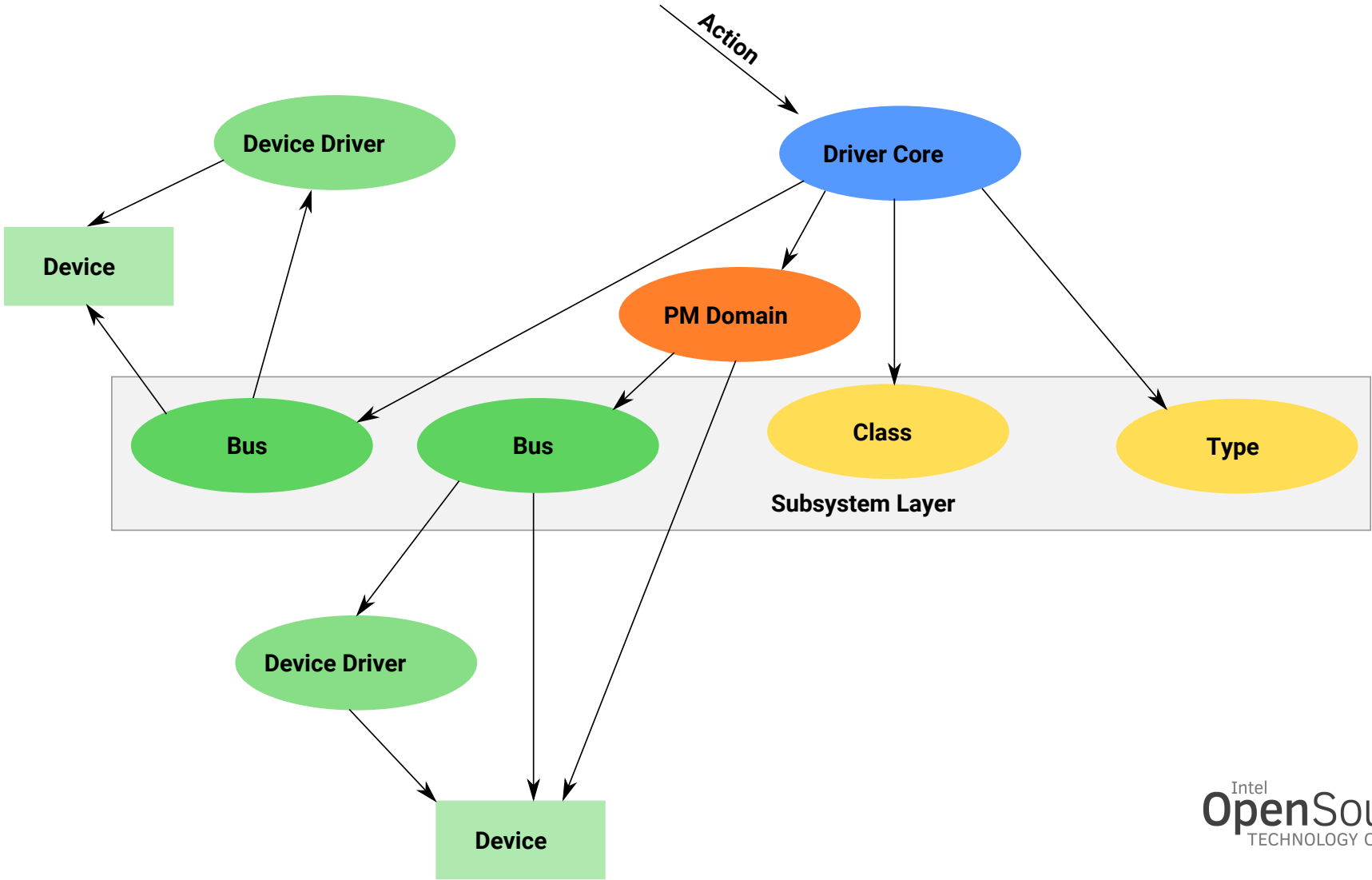
Intel OpenSource TECHNOLOGY CENTER

Hibernation and Restore Control Flows



Intel OpenSource TECHNOLOGY CENTER

Driver Core and Device PM Operations



Intel OpenSource TECHNOLOGY CENTER

Optimizations and Avoiding Code Duplication

Initial common approach

Resume devices from runtime suspend during system-wide trasitions into sleep states.

Can it be made more efficient?

1. What if a device is in runtime suspend before system suspend? Can it remain suspended and (if so) under what conditions?
2. Can devices be left in suspend when the system is resuming from system-wide suspend?
3. Can runtime PM callbacks be used for system-wide PM too and to what extent? If they can, how to make that (reasonably) easy?

Intel OpenSource TECHNOLOGY CENTER

Re: Question 1

Question

What if a device is in runtime suspend before system suspend? Can it remain suspended and (if so) under what conditions?

Yes, it can, but

- Its (current) power state has to be suitable for that.
- Its (current) wakeup settings have to match system-wide sleep requirements.

Intel OpenSource TECHNOLOGY CENTER

Re: Question 2

Question

Can devices be left in suspend when the system is resuming from system-wide suspend?

Yes, they can, but

- They must be resumed if needed for resuming children/consumers.
- Their power state and wakeup settings have to be suitable for runtime PM.
- It may not be worth it to leave the given device in suspend.

Intel OpenSource TECHNOLOGY CENTER

Re: Question 3

Question

Can runtime PM callbacks be used for system-wide PM too and to what extent? If they can, how to make that (reasonably) easy?

Yes, driver callbacks can be re-used

- Re-use of middle-layer callbacks is problematic (at best) in general.
- Runtime PM callbacks for re-use generally need to be "smart" (eg. they need to check if the device is already suspended and take care of the optimizations).

Intel OpenSource TECHNOLOGY CENTER

Direct Complete

Automatic, driven by the PM core

- `->prepare()` return value is the opt-in indicator.
- To be applied to device X, it has to be applied to all of its children/consumers and all of their children/consumers etc.
- If applied, skips all of the system-wide PM callbacks for the device, except for `->complete()`.

Weaknesses

- Significantly limited (basically, to leaf devices).
- Some middle-layer code does not take driver `->prepare()` return value into account.

Intel OpenSource TECHNOLOGY CENTER

Callback Wrappers

To be used in device drivers as system-wide PM callbacks

- `pm_runtime_force_suspend()`
- `pm_runtime_force_resume()`
- Handle both callback re-use and optimizations (limited to already suspended devices).

Weaknesses

- Children (essentially) assumed to use the same wrappers.
- Device state and wakeup settings checks may be missing.
- No opt-out for the resume-time optimization.
- Middle-layer runtime PM callbacks invoked along with driver callbacks (potentially leading to ordering issues).

Intel OpenSource TECHNOLOGY CENTER

Can The Weaknesses Be Avoided?

Guiding principle

Each layer of code must be consistent with the layers below and above it.

Idea

Allow drivers to tell the PM core and middle layers what they want to and/or can do.

Intel OpenSource TECHNOLOGY CENTER

Driver Flags for System-Wide PM

Rules

- To be set at the driver probe time.
- Cleared by the driver core on driver removal (or probe failures).

DPM_FLAG_NEVER_SKIP

Direct-complete forbidden.

DPM_FLAG_SMART_PREPARE

Take driver `->prepare()` return value into account (for direct-complete).

Intel OpenSource TECHNOLOGY CENTER

Driver Flags for System-Wide PM (Continued)

DPM_FLAG_SMART_SUSPEND

Avoid resuming the device from runtime suspend during system-wide transitions into sleep states (driver capability).

DPM_FLAG_LEAVE_SUSPENDED

Device may be left in suspend during/after system-wide resume (driver preference).

Intel OpenSource TECHNOLOGY CENTER

Development Status

- `DPM_FLAG_NEVER_SKIP` in use (eg. by `i915`).
- `DPM_FLAG_SMART_PREPARE` in use by `i2c-designware-platdrv`.
- `DPM_FLAG_SMART_SUSPEND` and `DPM_FLAG_LEAVE_SUSPENDED` taken into account by the PM core, the PCI bus type and the ACPI PM domain.
- Some drivers use these flags already (eg. `i2c-designware-platdrv`, `acpi_tad`).
- genpd support under development (I have patches).
- Integration with the callback wrappers?
- More bus types to support the existing flags?
- More flags?

Intel OpenSource TECHNOLOGY CENTER

Questions / Discussion

Intel OpenSource TECHNOLOGY CENTER

References

Linux Kernel Documentation, *Device Power Management* (https://www.kernel.org/doc/html/latest/driver-api/pm/index.html).

R. J. Wysocki, *Power Management Challenges in Linux* (https://www.linuxplumbersconf.org/2017/ocw//system/presentations/4652/original/linux_pm_challenges.pdf).

R. J. Wysocki, *PM Infrastructure in the Linux Kernel – Current Status and Future* (https://events.linuxfoundation.org/sites/events/files/slides/kernel_PM_infra_0.pdf).

R. J. Wysocki, *What Is Suspend-to-Idle and How To Make It Work* (http://events.linuxfoundation.org/sites/events/files/slides/what-is-suspend-to-idle.pdf).

R. J. Wysocki, *Why We Need More Device Power Management Callbacks* (https://events.linuxfoundation.org/images/stories/pdf/lfcs2012_wysocki.pdf).

Intel OpenSource TECHNOLOGY CENTER

Disclaimer

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at `www.intel.com`.

Intel OpenSource TECHNOLOGY CENTER