Cluster Idle - Now and next

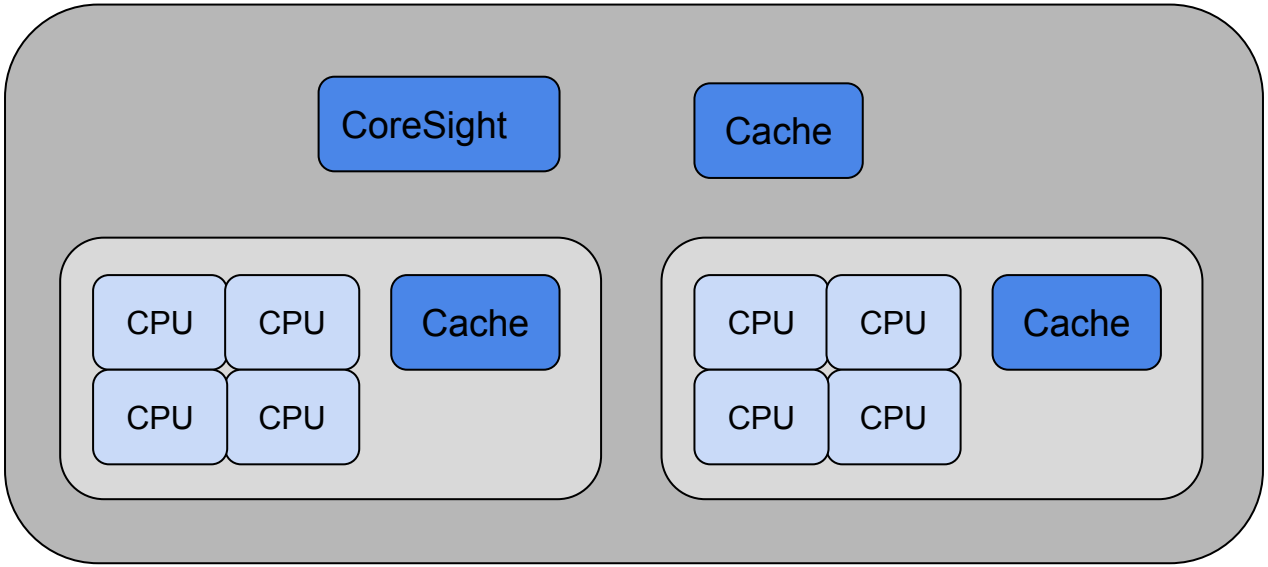
Ulf Hansson, Linaro
ulf.hansson@linaro.org

OSPM Summit 2019

Agenda

- Current status.
- Deployment for ARM/ARM64 via PSCI.
- Future improvements.

CPU Topology



CPUs and other resources sharing idle states
- doesn't always fit with the CPUIdle framework.

The problem

- Last-man activities needs to be synchronized by Linux.
  - Configure external logics for wakeups, decouple the GIC, etc.
  - Communicate sleep states of devices to FW.

- A power-rail for a group of CPUs may be shared.
  - A controller needs the power-rail on - to complete I/O.
  - Platform configurable.

Current status

Infrastructure needed - done

- Use genpd to model the topology via DT.
- Extend genpd to support CPU devices.
- Add a genpd governor targeted for a group of CPUs.
- Continue to use CPUIdle to select idle states per CPU.

Deployment for PSCI - reviewing

- Update DT bindings for PSCI to support a hierarchical CPU topology.
- Deploy genpd and runtime PM support for PSCI.
- Enable platform support for Hisilicon Hikey and Qcom 410c.

[PATCH 00/18] ARM/ARM64: Support hierarchical CPU arrangement for PSCI
https://patchwork.kernel.org/cover/10941581/

PSCI DT - flattened topology

```
CPU0: cpu@0 {
        compatible = "arm,cortex-a53", "arm,armv8";
        enable-method = "psci";
        cpu-idle-states = <&CPU_PWRDN>,
        <&CLUSTER_RET>,<&CLUSTER_PWRDN>;
};
CPU1: cpu@1 {
        compatible = "arm,cortex-a53", "arm,armv8";
        enable-method = "psci";
        cpu-idle-states = <&CPU_PWRDN>,
        <&CLUSTER_RET>,<&CLUSTER_PWRDN>;
};
psci {
        compatible = "arm,psci-0.2";
        method = "smc";
};
```

```
idle-states {
        CPU_PWRDN: cpu-power-down {
                compatible = "arm,idle-state";
                arm,psci-suspend-param = <0x0000001>;
        };
        CLUSTER_RET: cluster-retention {
                compatible = "arm,idle-state";
                arm,psci-suspend-param = <0x1000011>;
        };
        CLUSTER_PWRDN: cluster-power-down {
                compatible = "arm,idle-state";
                arm,psci-suspend-param = <0x1000031>;
        };
};
```

PSCI DT - hierarchical topology 1/2

```
CPU0: cpu@0 {
        compatible = "arm,cortex-a53", "arm,armv8";
        enable-method = "psci";
        power-domains = <&CPU_PD0>;
        power-domain-names = "psci";
};
CPU1: cpu@1 {
        compatible = "arm,cortex-a53", "arm,armv8";
        enable-method = "psci";
        power-domains = <&CPU_PD1>;
        power-domain-names = "psci";
};
psci {
        Next page...
};
```

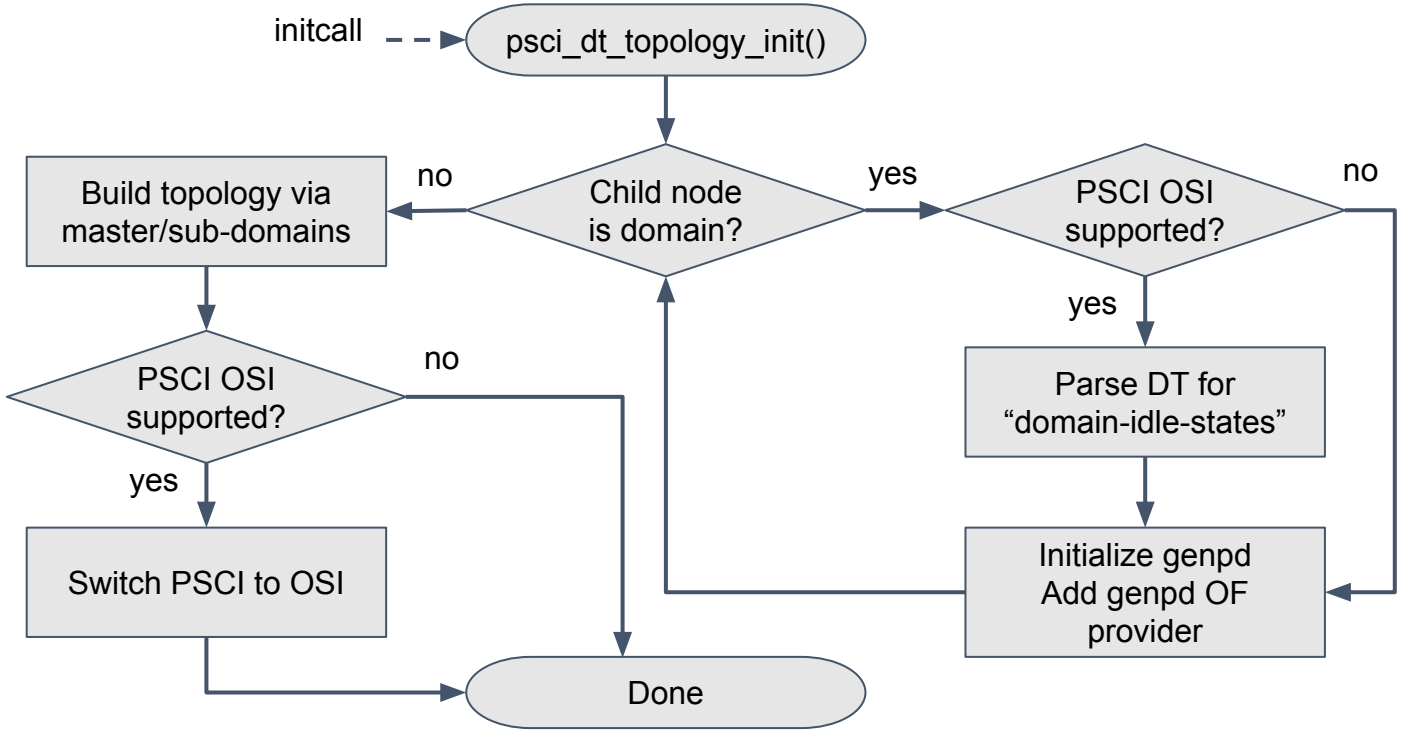
```
idle-states {
        CPU_PWRDN: cpu-power-down {
                compatible = "arm,idle-state";
                arm,psci-suspend-param = <0x0000001>;
        };
        CLUSTER_RET: cluster-retention {
                compatible = "domain-idle-state";
                arm,psci-suspend-param = <0x100001 0>;
        };
        CLUSTER_PWRDN: cluster-power-down {
                compatible = "domain-idle-state";
                arm,psci-suspend-param = <0x100003 0>;
        };
};
```

PSCI DT - hierarchical topology 2/2

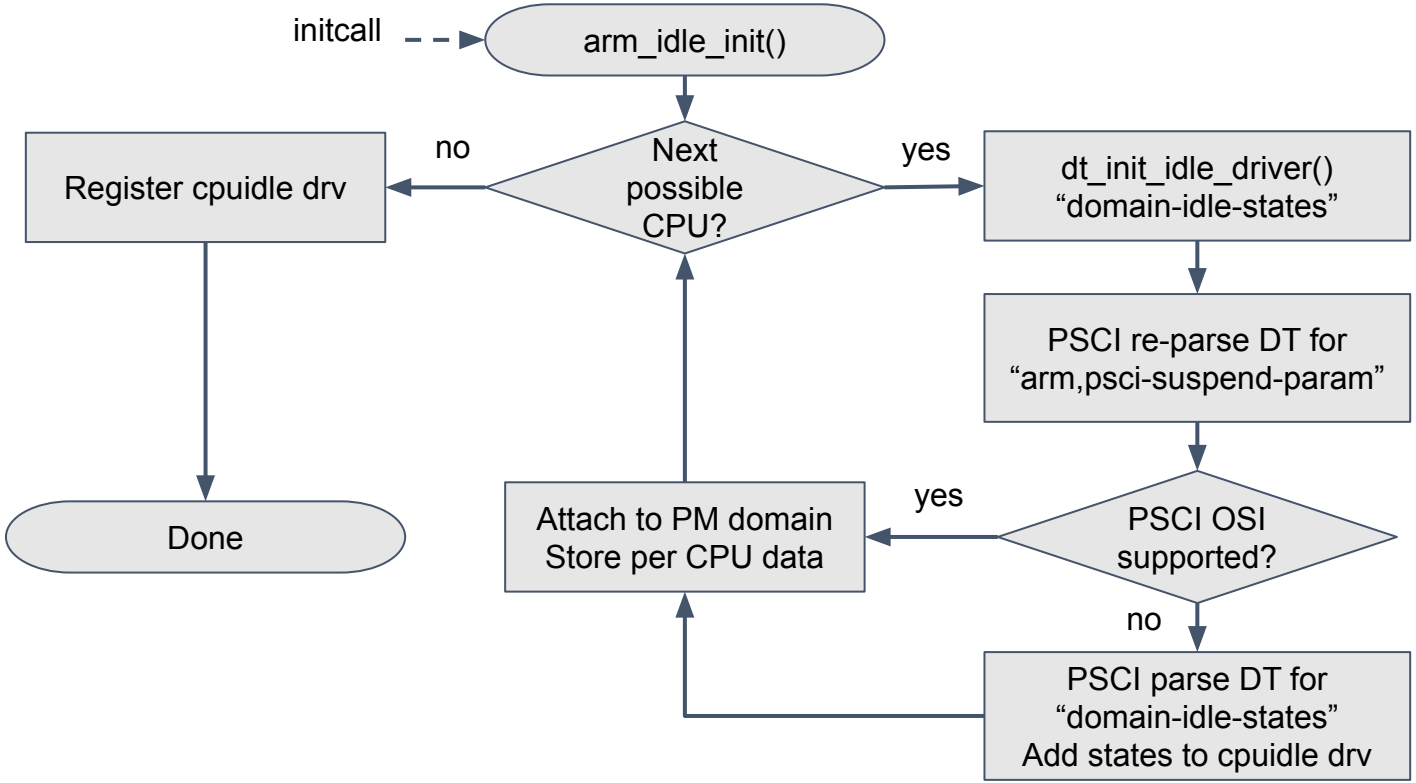
```
psci {
        compatible = "arm,psci-1.0";
        method = "smc";
        CPU_PD0: cpu-pd0 {
                #power-domain-cells = <0>;
                domain-idle-states = <&CPU_PWRDN>;
                power-domains = <&CLUSTER_PD>;
        };
        CPU_PD1: cpu-pd1 {
                #power-domain-cells = <0>;
                domain-idle-states =  <&CPU_PWRDN>;
                power-domains = <&CLUSTER_PD>;
        };
        CLUSTER_PD...
};
```

```
CLUSTER_PD: cluster-pd {
        #power-domain-cells = <0>;
        domain-idle-states = <&CLUSTER_RET>,
        <&CLUSTER_PWRDN>;
};
```

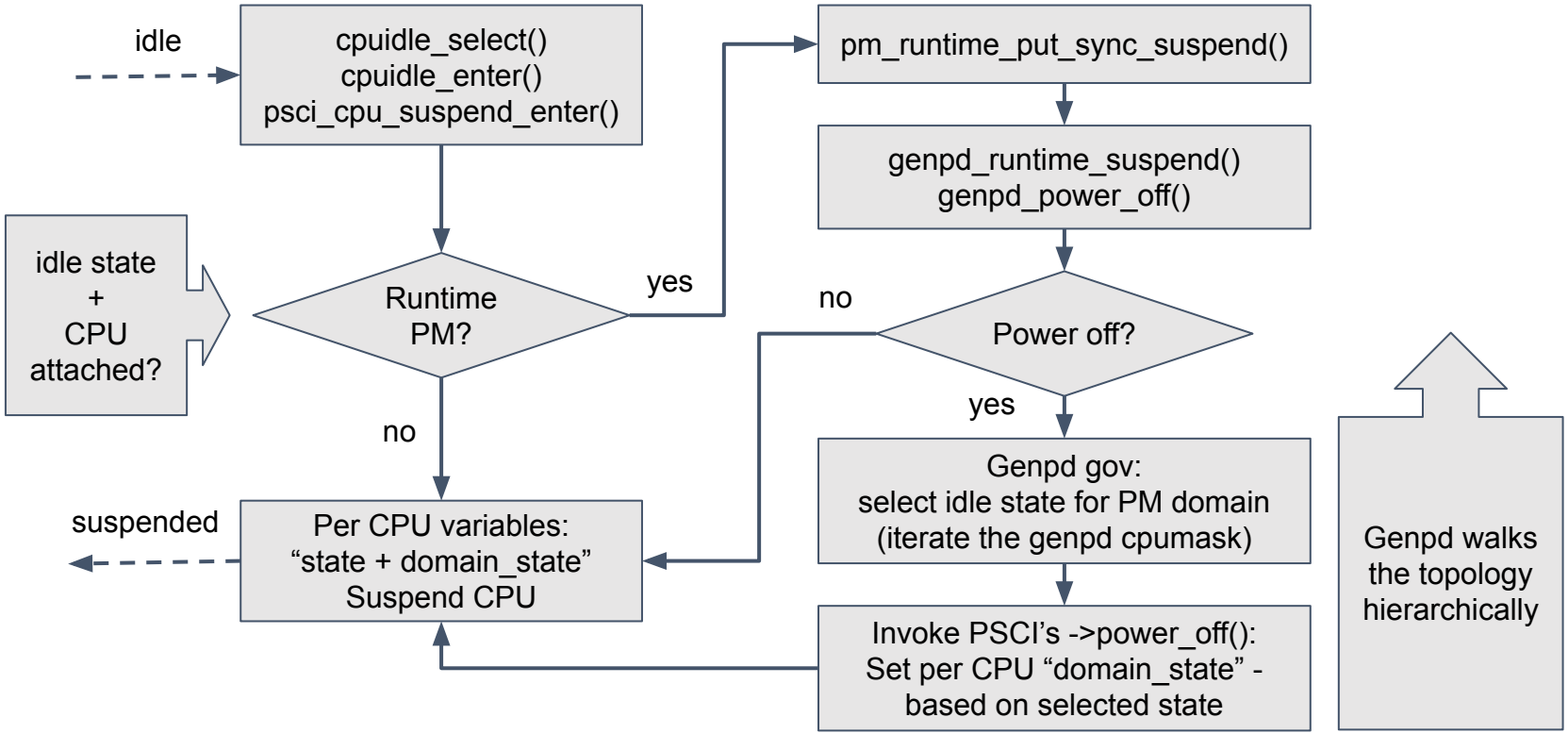
PSCI deployment - Init CPU PM topology



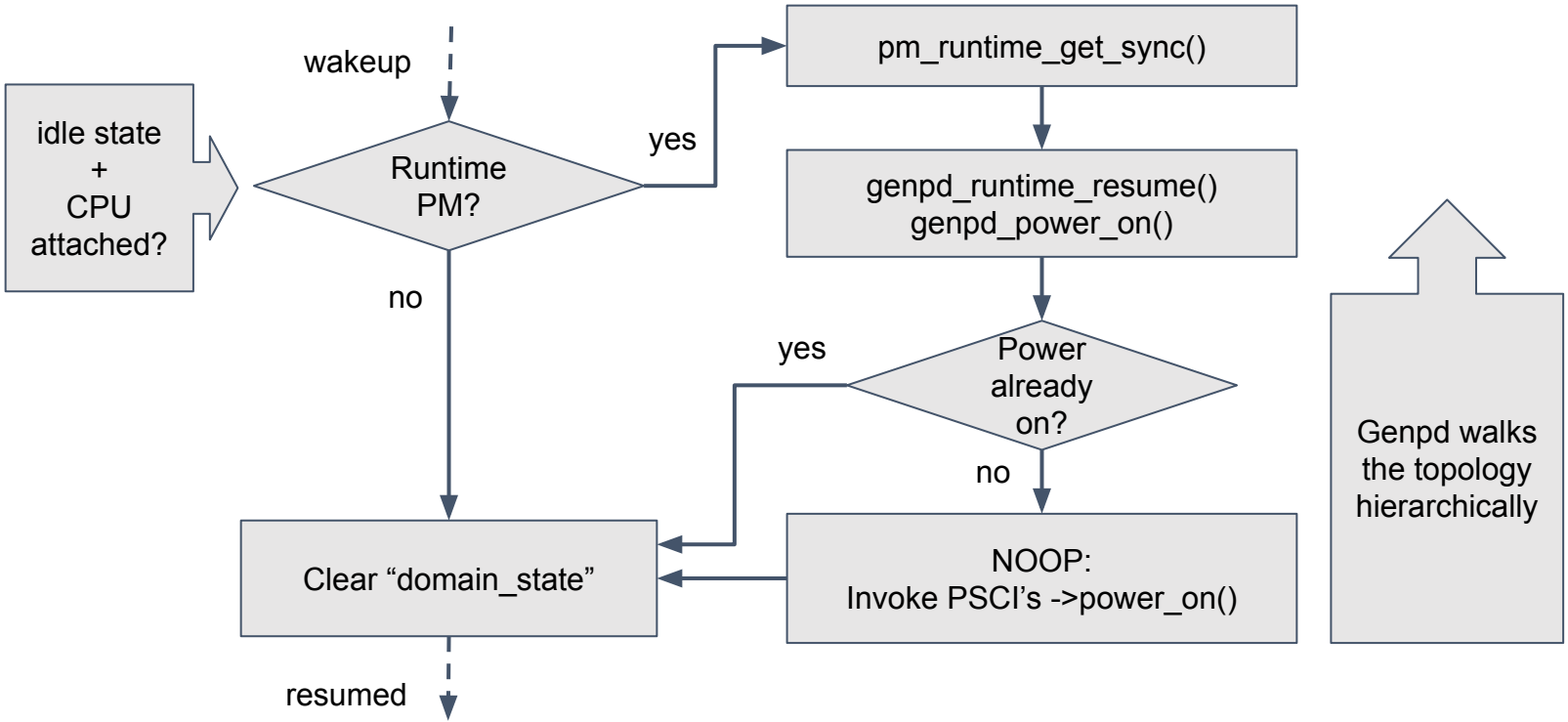
PSCI deployment - Init CPUIdle drv (attach CPU)



PSCI deployment - Idle (suspend)



PSCI deployment - Idle (resume)



Next - last man activities

- cluster_pm_enter_exit()
  - Deliver to whom and for what idle states?

- cpu_pm_enter|exit()
  - Is the per CPU notifier needed for all idle states?

Solution?

- Use genpd to inform an attached device (in a special way), described via DT.

Next - avoid ktime_get() in genpd

- ktime_get() is called to measure latencies of ->runtime_suspend|resume().
- ktime_get() is called to measure latencies of ->power_off|on().
- ktime_get() is called by the genpd governor.

Solutions?

- Convert to ktime_get_mono_fast_ns().
- Use a calibration mode, rather than always measuring latencies.
- Don’t measure for CPU PM domains.

Next - avoid ktime_get_mono_fast_ns()

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

Solution?

- Call ktime_get_mono_fast_ns() only at RPM_ACTIVE and RPM_SUSPENDED.

Next - enable deepest state for suspend-to-idle

- Runtime PM is disabled, which means 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?

- Configure CPU devices as syscore devices and make use of pm_genpd_syscore_poweroff()|poweron().
  - There is no locking in this path in genpd, but we need that anyway...

- Special runtime PM treatments of devices corresponding to CPUs.
  - genpd_power_off() - aborts when any device is "prepared".

Next - decrease overhead of runtime PM

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

Solution?

- Invent specific runtime PM APIs for CPUs, which can be used to decrease the overhead.

Next - hierarchical limitations in genpd

- 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.

- Fix bug for hierarchical locking in genpd when using GENPD_FLAG_IRQ_SAFE and when called from non-atomic context.

Next - improve the genpd governor

- The next timer event is not the only reason for a CPU to wakeup.

Solution?

- Use information about the next predicted IRQ and the next IPI.
- Further exchange information between CPUIdle governor, per CPU.

OSPM Summit 2019