Subsystem Trace Points: power

The power tracing system captures events related to power transitions within the kernel. Broadly speaking there are three major subheadings:

- Power state switch which reports events related to suspend (S-states), cpuidle (C-states) and cpufreq (P-states)
- System clock related changes
- Power domains related changes and transitions

This document describes what each of the tracepoints is and why they might be useful.

Cf. include/trace/events/power.h for the events definitions.

1. Power state switch events

1.1 Trace API

A 'cpu' event class gathers the CPU-related events: cpuidle and cpufreq.

```
cpu_idle              "state=%lu cpu_id=%lu"
cpu_frequency         "state=%lu cpu_id=%lu"
cpu_frequency_limits  "min=%lu max=%lu cpu_id=%lu"
```

A suspend event is used to indicate the system going in and out of the suspend mode:

```
machine_suspend               "state=%lu"
```

Note: the value of '-1' or '4294967295' for state means an exit from the current state, i.e. trace_cpu_idle(4, smp_processor_id()) means that the system enters the idle state 4, while trace_cpu_idle(PWR_EVENT_EXIT, smp_processor_id()) means that the system exits the previous idle state.

The event which has 'state=4294967295' in the trace is very important to the user space tools which are using it to detect the end of the current state, and so to correctly draw the states diagrams and to calculate accurate statistics etc.

2. Clocks events

The clock events are used for clock enable/disable and for clock rate change.

```
clock_enable          "%s state=%lu cpu_id=%lu"
clock_disable         "%s state=%lu cpu_id=%lu"
clock_set_rate                "%s state=%lu cpu_id=%lu"
```

The first parameter gives the clock name (e.g. "gpio1_iclk"). The second parameter is '1' for enable, '0' for disable, the target clock rate for set_rate.

3. Power domains events

The power domain events are used for power domains transitions

```
power_domain_target   "%s state=%lu cpu_id=%lu"
```

The first parameter gives the power domain name (e.g. "mpu_pwrdm"). The second parameter is the power domain target state.

4. PM QoS events

The PM QoS events are used for QoS add/update/remove request and for target/flags update.

```
pm_qos_update_target              "action=%s prev_value=%d curr_value=%d"
pm_qos_update_flags               "action=%s prev_value=0x%x curr_value=0x%x"
```

The first parameter gives the QoS action name (e.g. "ADD_REQ"). The second parameter is the previous QoS value. The third parameter is the current QoS value to update.

There are also events used for device PM QoS add/update/remove request.

```
dev_pm_qos_add_request            "device=%s type=%s new_value=%d"
dev_pm_qos_update_request         "device=%s type=%s new_value=%d"
dev_pm_qos_remove_request         "device=%s type=%s new_value=%d"
```

The first parameter gives the device name which tries to add/update/remove QoS requests. The second parameter gives the request type (e.g. "DEV_PM_QOS_RESUME_LATENCY"). The third parameter is value to be added/updated/removed.

And, there are events used for CPU latency QoS add/update/remove request.

```
pm_qos_add_request         "value=%d"
pm_qos_update_request      "value=%d"
pm_qos_remove_request      "value=%d"
```

The parameter is the value to be added/updated/removed.