Wakeup Sources Configuration and Management

Sudeep Holla <sudeep.holla@arm.com>
Lorenzo Pieralisi <lorenzo.pieralisi@arm.com>
Energy Aware Scheduling II Microconference - LPC 2015
Agenda

- Wakeup from System Suspend (S2R)
- Current issues
- Interrupt Controller & Power Domains
- Multiple Sleep States
- Shared Wakeup Interrupt
- Next Steps / Proposal?
Wakeup from System Suspend

```bash
echo mem > /sys/power/state
```

- **PM control interface for the devices**
  ```c
  device_init_wakeup(dev, bool)
  enable_irq_wake(irq)
  disable_irq_wake(irq)
  dev_pm_set_wake_irq(dev, irq)
  dev_pm_clear_wake_irq(dev, irq)
  ```

- **Userspace control per device**
  ```bash
  /sys/devices/.../power/wakeup
  ```

- How do identify the wakeup capable devices?
Current issues(1)

- Very platform specific code

```c
static const struct platform_suspend_ops imx6q_pm_ops = {
    .enter = imx6q_pm_enter,
    .valid = imx6q_pm_valid,
};
suspend_set_ops(&imx6q_pm_ops);
```

- Non-standard DT bindings

<table>
<thead>
<tr>
<th>Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arm/vic.txt</td>
<td>valid-wakeup-mask: A one cell big bit mask of interrupt sources that can be</td>
</tr>
<tr>
<td>extcon/extcon-palmas.txt</td>
<td>ti,wakeup: To enable the wakeup comparator in probe</td>
</tr>
<tr>
<td>input/ads7846.txt</td>
<td>linux,wakeup: use any event on touchscreen as wakeup event.</td>
</tr>
<tr>
<td>input/elan_i2c.txt</td>
<td>- wakeup-source: touchpad can be used as a wakeup source.</td>
</tr>
<tr>
<td>input/gpio-keys-polled.txt</td>
<td>- gpio-key,wakeup: Boolean, button can wake-up the system.</td>
</tr>
<tr>
<td>input/nvidia,tegra20-kbc.txt:</td>
<td>- nvidia,wakeup-source: configure keyboard as a wakeup source for suspend/resume</td>
</tr>
<tr>
<td>input/samsung-keypad.txt:</td>
<td>- linux,keypad-wakeup: use any event on keypad as wakeup event.</td>
</tr>
<tr>
<td>input/samsung-keypad.txt:</td>
<td>- linux,input-wakeup;</td>
</tr>
<tr>
<td>mfd/tc3589x.txt</td>
<td>- linux,wakeup: use any event on keypad as wakeup event.</td>
</tr>
</tbody>
</table>
Current issues (2)

- (Ab)use of IRQF_NO_SUSPEND

```c
drivers/i2c/busses/i2c-exynos5.c: IRQF_NO_SUSPEND | IRQF_ONESHOT,
drivers/i2c/busses/i2c-omap.c: IRQF_NO_SUSPEND, pdev->name, dev);
drivers/input/keyboard/tegra-kbc.c: IRQF_NO_SUSPEND | IRQF_TRIGGER_HIGH, pdev->name, kbc);
drivers/mfd/db8500-prcmu.c: prcmu_irq_thread_fn, IRQF_NO_SUSPEND, "prcmu", NULL);
drivers/rtc/rtc-ab8500.c: rtc_alarm_handler, IRQF_NO_SUSPEND | IRQF_ONESHOT,
drivers/rtc/rtc-p1031.c: .irqflags = IRQF_SHARED | IRQF_NO_SUSPEND,
```

- Shared interrupts
- Not scalable for ARM64
  - Standard interface - PSCI
  - No machine/platform specific code
Interrupt/Wakeup Controller dependency

- Interrupt controllers without wakeup source configuration (IRQCHIP_SKIP_SET_WAKE)
- Safe to mask all the non wakeup interrupts (IRQCHIP_MASK_ON_SUSPEND)
- Interrupt/Wakeup Controller must be in always-on domain
- IRQ domain hierarchy - irqchip->irq_set_wake, flags
- Need a way to represent in DT
  - boolean "always-on" property
  - "power-domains" property - phandle to always on power domain
Multiple system sleep states

- ACPI can specify the system states ($S$-States) from which the device can wake up from each of its power state ($D$-state).
- Do we need that complexity in DT?
- Runtime-PM + CPUIdle - possible to achieve traditional S1/S2 states?
- PSCI (ARM / ARM64) supports only S3 (i.e. Suspend to RAM)
Shared Wakeup Interrupt

- **IRQF_SHARED | IRQF_NO_SUSPEND**
  - IRQF_SHARED and enable_irq_wake

- **IRQF_SHARED | IRQF_COND_SUSPEND**
  - spurious IRQs Vs genuine wakeup

- Query the IRQ subsystem to get to know if it's shared?
What needs to be done?

- Cleanup misuse of IRQF flags in drivers
- Propose/Consolidate DT bindings
- Anything else?