Profiling Suspend/Resume Latency of Linux Kernel on Embedded Processors

Using pm_print_time to debug and optimize your system

Optimize suspend/resume latency to save energy

- Minimize energy consumed during suspend/resume transition.
- Eliminate unnecessary drivers to reduce time spent in suspend/resume and extend battery life.
- **Example: Portable barcode scanner device requirements:**
  - Must resume and scan barcode in less than 250ms
  - Must process greater than 500 transactions per charge

Improve user experience

- System should appear to wake “instantly” from suspend upon user interaction.
- **Example: Smart thermostat device requirement:** Screen must appear to come alive within ~100ms

Debugging suspend/resume latency issues

Pinpoint drivers that hang momentarily during suspend or resume

suspend

```
[  135.765628] calling  0-0055+ @ 1224, parent: i2c-0
[  137.262599] call 0-0055+ returned 0 after 1461860 usecs
```

…

resume

```
[  137.700684] calling  0-0055+ @ 1224, parent: i2c-0
[  139.730997] call 0-0055+ returned 0 after 1982702 usecs
```

…

For example, querying an 12C device during suspend unexpectedly stalls.

pm_print_times

- Official Linux sysfs documentation: https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-power
- Suspend and resume times are printed during resume.

```
root@am335x-evm:~# echo 1 > /sys/power/pm_print_times

root@am335x-evm:~# echo mem > /sys/power/state
[697.089698] PM: Syncing filesystems ... done.
[697.104082] Freezing user space processes ... (elapsed 0.002 seconds) done.
[697.113730] Freezing remaining freezable tasks ... (elapsed 0.001 seconds) .
[697.122848] Suspending console(s) (use no_console_suspend to debug)
[697.130636] calling  0:0:0:0+ @ 1907, parent: target0:0:0
[697.130677] call 0:0:0:0+ returned 0 after 21 usecs
[697.130735] calling  target0:0:0+ @ 1907, parent: host0
[697.130747] call target0:0:0+ returned 0 after 1 usecs
[697.131001] calling  1-1+ @ 1907, parent: usb1
[697.131046] call 1-1+ returned 0 after 31 usecs
…
```

Do you need that driver?

- Common peripherals requiring the most time to suspend/resume:
  - Ethernet (does not include DHCP time)
  - MMC
  - USB
- Notice that resume time drops off sharply after the “big 3.”

[693497.838	call	usb1+	109400	usecs
[693497.664	call	mmc0:0007+	61118	usecs
[693497.723	call	mmc1:0001+	57577	usecs
[693497.168	call	4a100000.ethernet+	16343	usecs
[693497.600	call	4a100000.ethernet+	5876	usecs
[693497.725	call	47401300.usb-phy+	980	usecs
[693497.726	call	47401b00.usb-phy+	975	usecs
[693497.173	call	44e09000.serial+	500	usecs
[693497.148	call	480c8000.mailbox+	237	usecs
[693497.723	call	4a101000.mdio:00+	232	usecs
[693497.150	call	4804c000.gpio+	169	usecs
[693497.839	call	1-1+	156	usecs
[693497.151	call	44e3e000.rtc+	103	usecs
[693497.149	call	44e09000.serial+	99	usecs
[693497.600	call	56000000.sgx+	84	usecs
[693497.145	call	4a101000.mdio+	62	usecs

252ms

AM335x – Subset of Linux driver resume times

Testing impact of removing drivers

- Mark the status property of the device tree node to “disabled.”
- Hypothetical system with NFS-mounted rootfs, no USB or MMC saves over 200ms on resume.

Linux/arch/arm/boot/dts/am335x-boneblack.dts

```
&mmc1 {     status = "disabled";};
&mmc2 {     status = "disabled";};
&usb {      status = "disabled";};
&usb_ctrl_mod {status = "disabled";};
&usb0_phy {           status = "disabled";};
&usb1_phy {           status = "disabled";};
&usb0 {     status = "disabled";};
&usb1 {     status = "disabled";};
&cppi41dma  {status = "disabled";};
```

128.848322] call	4a100000.ethernet+	14887	usecs
128.857442] call	4a100000.ethernet+	5941	usecs
128.851113] call	44e09000.serial+	279	usecs
128.859725] call	4a101000.mdio:00+	241	usecs
128.832540] call	4804c000.gpio+	171	usecs
128.833047] call	44e3e000.rtc+	103	usecs
128.857782] call	56000000.sgx+	82	usecs
128.859955] call	rtc0+	64	usecs
128.832713] call	44e09000.serial+	55	usecs
128.832310] call	49000000.edma+	52	usecs
128.857576] call	53100000.sham+	48	usecs
128.850269] call	4a101000.mdio+	46	usecs
128.859343] call	4830e000.lcdc+	40	usecs
128.858433] call	sound+	30	usecs
128.848529] call	56000000.sgx+	24	usecs
128.857621] call	53500000.aes+	20	usecs
128.848442] call	48038000.mcasp+	18	usecs

Debugging unexpected latency

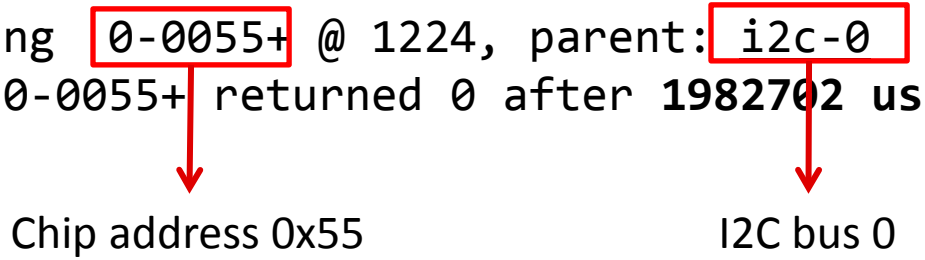
Example: I2C device is momentarily stalling suspend and resume.

suspend

```
[  135.765628] calling  0-0055+ @ 1224, parent: i2c-0
[  137.262599] call 0-0055+ returned 0 after 1461860 usecs
 ...
```

resume

```
[  137.700684] calling  0-0055+ @ 1224, parent: i2c-0
[  139.730997] call 0-0055+ returned 0 after 1982702 usecs
...
```



For more information

- Linux Kernel Archives: Documentation/ABI/testing/sysfs-power : https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-power
- Processor SDK Linux Software Developer's Guide: http://processors.wiki.ti.com/index.php/Processor_SDK_Linux_Software_Developer's_Guide
- Processor SDK Linux Kernel Performance Guide: http://processors.wiki.ti.com/index.php/Processor_SDK_Linux_Kernel_Performance_Guide
- For questions about this training, refer to the E2E Community Forums for Sitara Processors at http://e2e.ti.com/support/arm/sitara_arm/f/791/t/277411