How to get s2ram working

2006 Linus Torvalds 2006 Pavel Machek

1. Check suspend.sf.net, program s2ram there has long whitelist of "known ok" machines, along with tricks to use on each one.
2. If that does not help, try reading tricks.txt and video.txt. Perhaps problem is as simple as broken module, and simple module unload can fix it.
3. You can use Linus' TRACE_RESUME infrastructure, described below.

Using TRACE_RESUME

I've been working at making the machines I have able to STR, and almost always it's a driver that is buggy. Thank God for the suspend/resume debugging - the thing that Chuck tried to disable. That's often the _only_ way to debug these things, and it's actually pretty powerful (but time-consuming - having to insert TRACE_RESUME() markers into the device driver that doesn't resume and recompile and reboot).

Anyway, the way to debug this for people who are interested (have a machine that doesn't boot) is:

- enable PM_DEBUG, and PM_TRACE
- use a script like this:

  ```
  #!/bin/sh
  sync
  echo 1 > /sys/power/pm_trace
  echo mem > /sys/power/state
  ```

  to suspend
- if it doesn't come back up (which is usually the problem), reboot by holding the power button down, and look at the dmesg output for things like:

  ```
  Magic number: 4:156:725
  hash matches drivers/base/power/resume.c:28
  hash matches device 0000:01:00.0
  ```

  which means that the last trace event was just before trying to resume device 0000:01:00.0. Then figure out what driver is controlling that device (lspci and /sys/devices/pci* is your friend), and see if you can fix it, disable it, or trace into its resume function.

  If no device matches the hash (or any matches appear to be false positives), the culprit may be a device from a loadable kernel module that is not loaded until after the hash is checked. You can check the hash against the current devices again after more modules are loaded using sysfs:

  ```
  cat /sys/power/pm_trace_dev_match
  ```

For example, the above happens to be the VGA device on my EVO, which I used to run with "radeonfb" (it's an ATI Radeon mobility). It turns out that "radeonfb" simply cannot resume that device - it tries to set the PLL's, and it just _hangs_. Using the regular VGA console and letting X resume it instead works fine.

NOTE

pm_trace uses the system's Real Time Clock (RTC) to save the magic number. Reason for this is that the RTC is the only reliably available piece of hardware during resume operations where a value can be set that will survive a reboot.

pm_trace is not compatible with asynchronous suspend, so it turns asynchronous suspend off (which may work around timing or ordering-sensitive bugs).

Consequence is that after a resume (even if it is successful) your system clock will have a value corresponding to the magic number instead of the correct date/time! It is therefore advisable to use a program like ntp-date or rdate to reset the correct date/time from an external time source when using this trace option.

As the clock keeps ticking it is also essential that the reboot is done quickly after the resume failure. The trace option does not use the seconds or the low order bits of the minutes of the RTC, but a too long delay will corrupt the magic value.