OpenBSD's New Suspend and Resume Framework

Paul Irofti
pirofti@openbsd.org

10th European BSD Conference
October 6–9, 2011
Maarssen, The Netherlands

Outline

Early Days

KISS

- Power up
- Power off

Time Passes...

Power Management

- Computers start consuming less power
- The system gains some power control
- APM enters the scene
- Machines can suspend and resume via APM

ACPI

The Machine Gets To Be In Charge

In theory:

- Knob fiddling
- Better control
- More features

Implications

Reality Check

- Extremely complex
- Specifications that nobody respects
- Every vendor has its own quirks
- The machine has to do everything

Results

ACPI

- New power management approach
- Affects device drivers as well
- Hard to get right
- Fit into the APM logic
- Lots of system changes
- New MI suspend/resume framework

Structure

Kernel Device Tracking

- Tree hierarchy
- Everything starts at mainbus(4)
- Device drivers attach to the proper parent device

Example

Dependency view

A memory stick is attached to the system

- sd(4)
- scsibus(4)
- umass(4)
- uhub(4)
- usb(4)
- ehci(4)
- pci(4)
- mainbus(4)

The stick becomes available to the user as sd0.

Configuration

Specific functionality

- Match – proper device driver matching
- Attach – attach to a proper place in the device tree
- Activate – activate the device
- Deactivate – turn off the device
- Detach – remove it from the device tree

Suspend and Resume

Low Power States Implications

ACPI support required:

- New system states
- Driver awareness
- Device notification of state changes
- Result: The need for new activate actions in autoconf(9)

New Actions

DVACT_QUIESCE

Prepare to suspend (discussed later on).

DVACT_SUSPEND

Set the device drivers in a suspend state.

DVACT_RESUME

Resume the device drivers back to running state.

Expanding autoconf(9)

Code Changes

`config_suspend()`

- Similar with attach/detach activate/deactivate
- Signals the drivers

`config_activate_children()`

- Handle the new cases
- `config_suspend()` the device's children

What is Quiesce?

Definition

The action of pausing or modifying a given process so that data consistency can be achieved.

Why is it important?

Because...

Some devices need pre-suspend notifications to:

- Finish-up disk I/O
- Dump audio buffers
- VT switch out of X
- Wait on other actions to finish
- Do misc. operations requiring a 'normal' running state

Starting Point

APM Machines

- APM userland daemon
- Userland notifies the kernel
- Kernel APM MD state machines
- Lots of MD code, specially for devices

Integrating Other PM Implementations

Rules

- Keep the same APM mechanism.
- Mold other implementations into it.
- Make it opaque to the userland.
- Let the drivers do the work for them.
- Implementation specific bits in MD
  Mostly whacky assembly routines

ACPI Implementation

Reiterating...

- ACPI will be fit in the same model
- Create a fake apm-like kernel ACPI state-machine
- Keep the same code-paths all the way down
- No difference from a userland perspective
- Only the kernel can tell APM and ACPI apart

Improvements

More MI, Less MD

- The BIOS does most of the work
- Remove MD device related code
- Let the device drivers do it in their activate functions
- Bare MD APM state machine

On Suspend

Code Flow

- `wsdisplay_suspend()`
- `bufq_quiesce()`
- `config_suspend(DVACT_QUIESCE)`
- `splhigh()`
- `disableintr()`
- `config_suspend(DVACT_SUSPEND)`
- `sys_platform->suspend()`

On Resume

Code Flow

- `sys_platform->resume()`
- `config_suspend(DVACT_RESUME)`
- `enableintr()`
- `splx()`
- `bufq_restart()`
- `wsdisplay_resume()`

Implementations

- Microsoft Windows
- Intel ACPICA
- OpenBSD

How It Works

System Perspective

- ACPI is a proxy between the BIOS and the OS
- Access AML methods according to the ACPI spec.
- Lots of spec violations
- Lots of quirks and workarounds
- The drivers have to handle device state

APM-like

Flow

- The userland needs no change
- `acpiioctl()` – notfication ioctl
- Same commands as APM
- ACPI tasks (e.g. `acpi_sleep_task()`)

On Suspend

Flow

- `acpi_sleep_task(S3)` – checks state changes
- `acpi_sleep_mode(S3)` – handles state changes
- `acpi_prepare_sleep_state(S3)` – AML nightmare
- `acpi_sleep_machdep(S3)` – MD code
- `acpi_enter_sleep_state(S3)` – PM regs fiddling

Not APM-Like

AML Methods

- `TTS` – transition to state, before device notification
- `PTS` – prepare to sleep, after device notification
- `SST` – system status indicator
- `GTS` – firmware execution before S3
- `PM` – power management registers
- `GTE` – wake registers

On Resume

Completely different from APM

- Real-mode: ACPI trampoline
- Real-mode: Might reenable video
- Real-mode: Enable paging
- Real-mode: Restore CPU registers
- Jump to where ACPI code stopped during suspend
- Clear PM registers
- Transition to S0 (more AML methods)
- Reset the lamp
- Enable runtime GPEs
- Resume the device drivers

Devices

Problems

- The order in which we suspend/resume them
- The device registers
- The memory maps
- How much state do we need to keep?

ACPI

No Man's Land

- The specifications are just a guide in reality
- AML is Windows-targeted
- AML is autogenerated code
- Magic methods that poke into CMOS and whatnot
- The AML parser is always finding quirks in production code

Video

Reposting

Can be done by:

- Real-mode BIOS call
- x86emu
- The driver itself
- Need for an PCI ID table
- nVidia is not supported at all
- Even then, some cards don’t work

USB

Problems

- Most machines have no problems (luck?)
- Some machines get their usb ports reset on resume
- Some don’t get them at all
- Keep usb state vs whack the whole stack

Other

Miscellaneous

- Mount points for usb drives don't get restored
- Audio sometimes gets trashed
- Aucat doesn't handle suspend/resume
- X doesn't come back on some machines
- X gets some image noise, fixed by VT switching
- Taking the cpu to 1-cpu is done at the wrong place
- Some drivers are not supported yet

Don't Panic

It Works!

- Most laptops are supported
- Most workstations as well
- The sub-system is stable
- The design is good
- Loongson is the newest user
- Lots of non suspend/resume bugs in drivers got fixed as a result

So Long, and Thanks for All the Fish

Questions?