Fast boot and suspend/resume

Zhang Rui
rui.zhang@intel.com

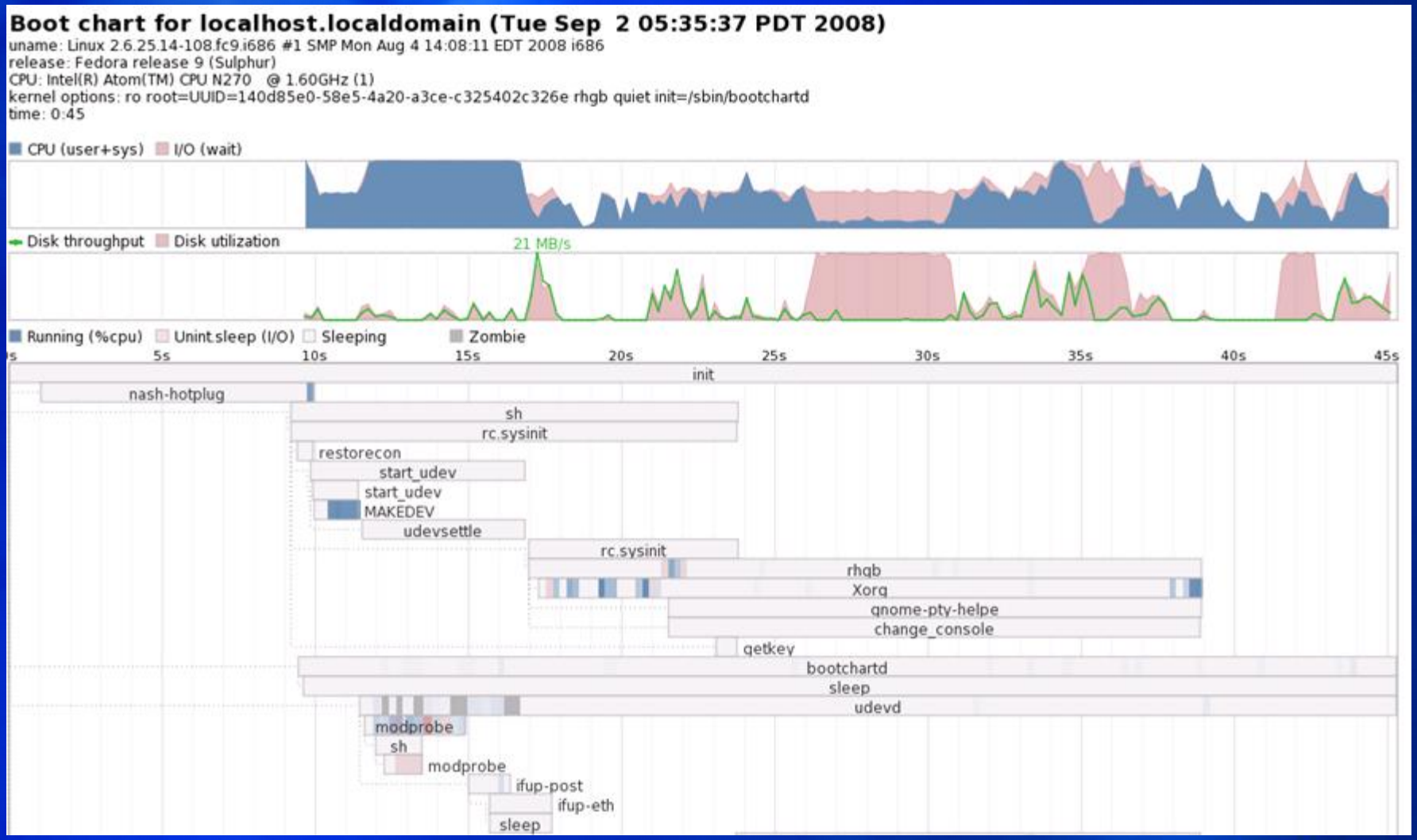
Beijing October 18th 2008

Open Source Technology Center

Agenda

- Fast boot
  - Current State of the Art
  - Splash Screens
  - Budget & Resource Scheduling
  - Kernel
  - Early Userspace
  - X.org
- Fast suspend/resume
  - Driver optimization
  - Device asynchronous resume
  - More aggressive attempts

Current state



We hate splash screens

By the time you see it.... we want to be done!

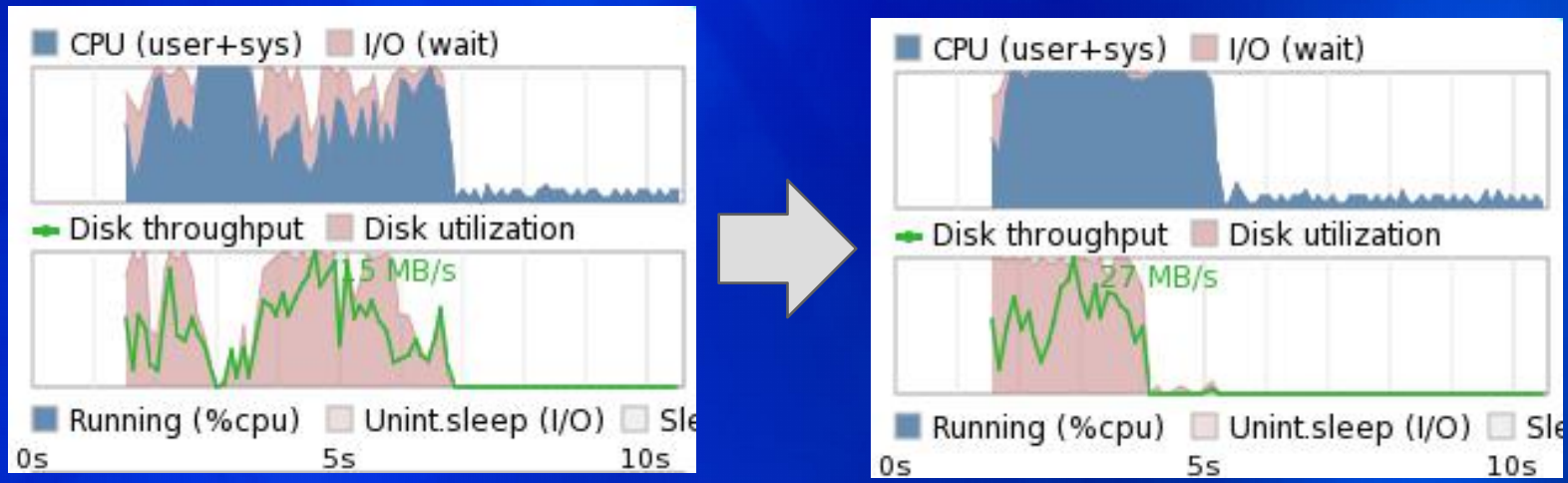
Time budgets, Resource scheduling

boot to be completely done in 5 seconds

Component	Allocated time
Kernel	1 second
Early boot	1 second
X	1 second
GUI/Desktop	2 seconds

sReadAhead

- read used portions of files in "use order" as early as possible to prime the pagecache
- "idle" IO scheduler class



Kernel
Budget: 1 second

- All system components built into the kernel image
  - Modules are slow, synchronous, and not needed for core components
- No initrd
  - All key drivers are in the kernel
  - /dev is populated with the fixed device nodes
  - initrd management just plain takes too long

Kernel
Budget: 1 second

- Asynchronous initialization of non-essential components with a new, asynchronous initcall level
- Reference: git://git.kernel.org/pub/scm/linux/kernel/git/arjan/linux-2.6-fastboot.git

Early boot / Init

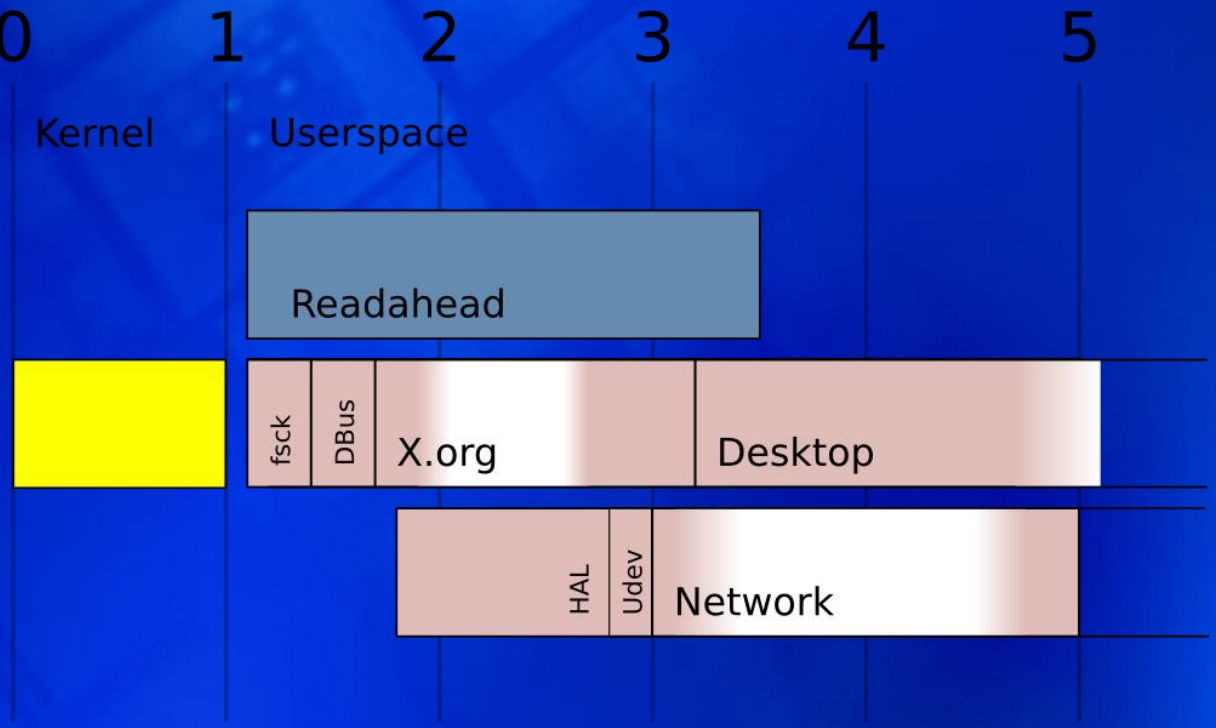
Budget: 1 second

- *sysvinit*, not *upstart*: Asynchronous, not Parallel
- Udev
  - Persistent /dev reduces overhead enormously

Early boot / Init

Budget: 1 second

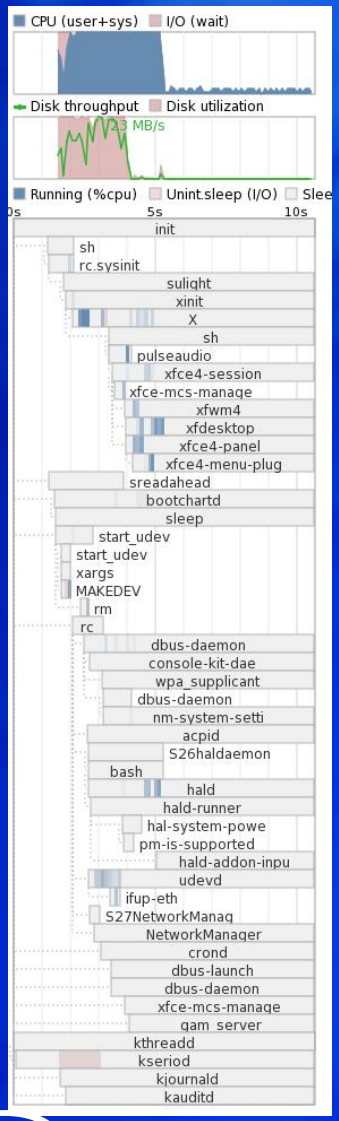
- Asynchronous for non-critical path

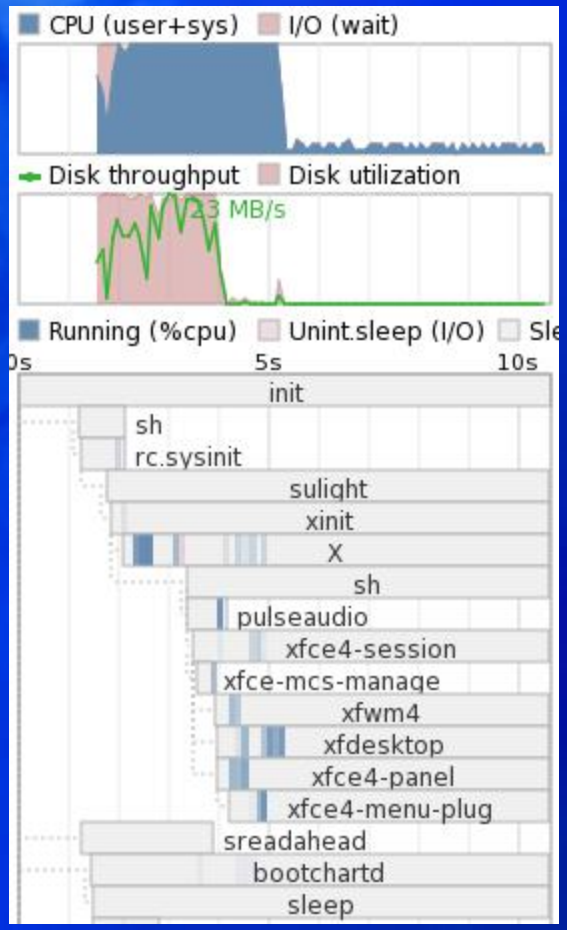


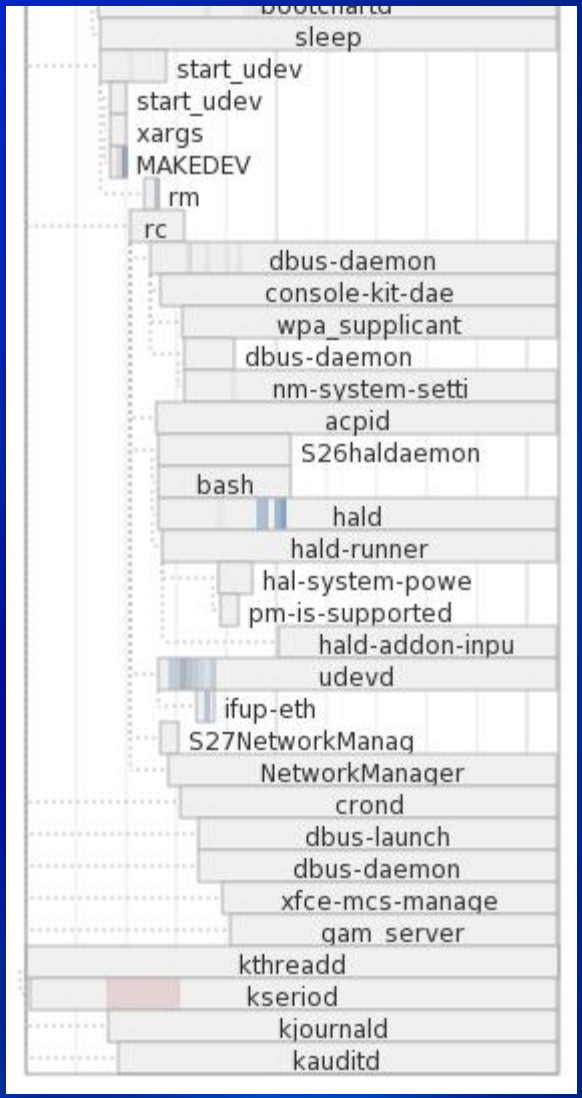
X.org

Budget: 1 second

- xorg-x11-drv-intel driver
  - Program everything during probing
  - ... then restore it to standard
  - ... then program it again for final use
  - Various "extra" delays trimmed (after fixing PCI posting bugs)
- XKB
  - Calling CPP ??? <explicit language removed>
  - Caching the result – Compute once, use forever







Fast suspend/resume

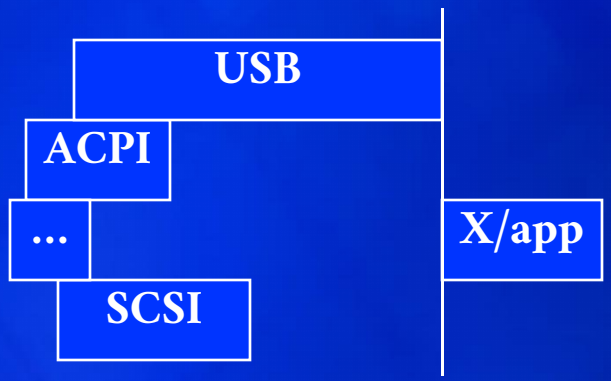
- Not as complex as fast boot
  - BIOS + kernel + X/application
  - No critical disk I/O request
- Time budget
  - Resume Kernel + X/application in 1 second

Driver optimization

- non-critical work can be deferred during device resume
  - Serio
    - offload resume to kseriod
- Fix driver poor/wrong behavior
  - AHCI
    - HBA reset speed up

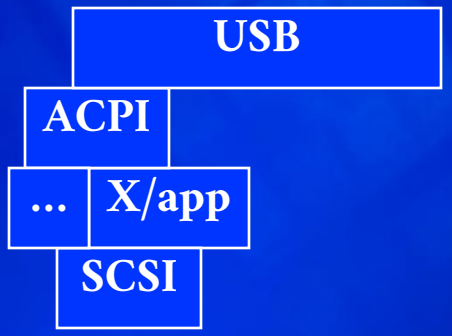
Device asynchronous resume

- Resume devices asynchronously
- Sync before resume X/applications



More aggressive attempts

- Device deferred resume instead of asynchronous resume
  - don't sync before resuming X/applications

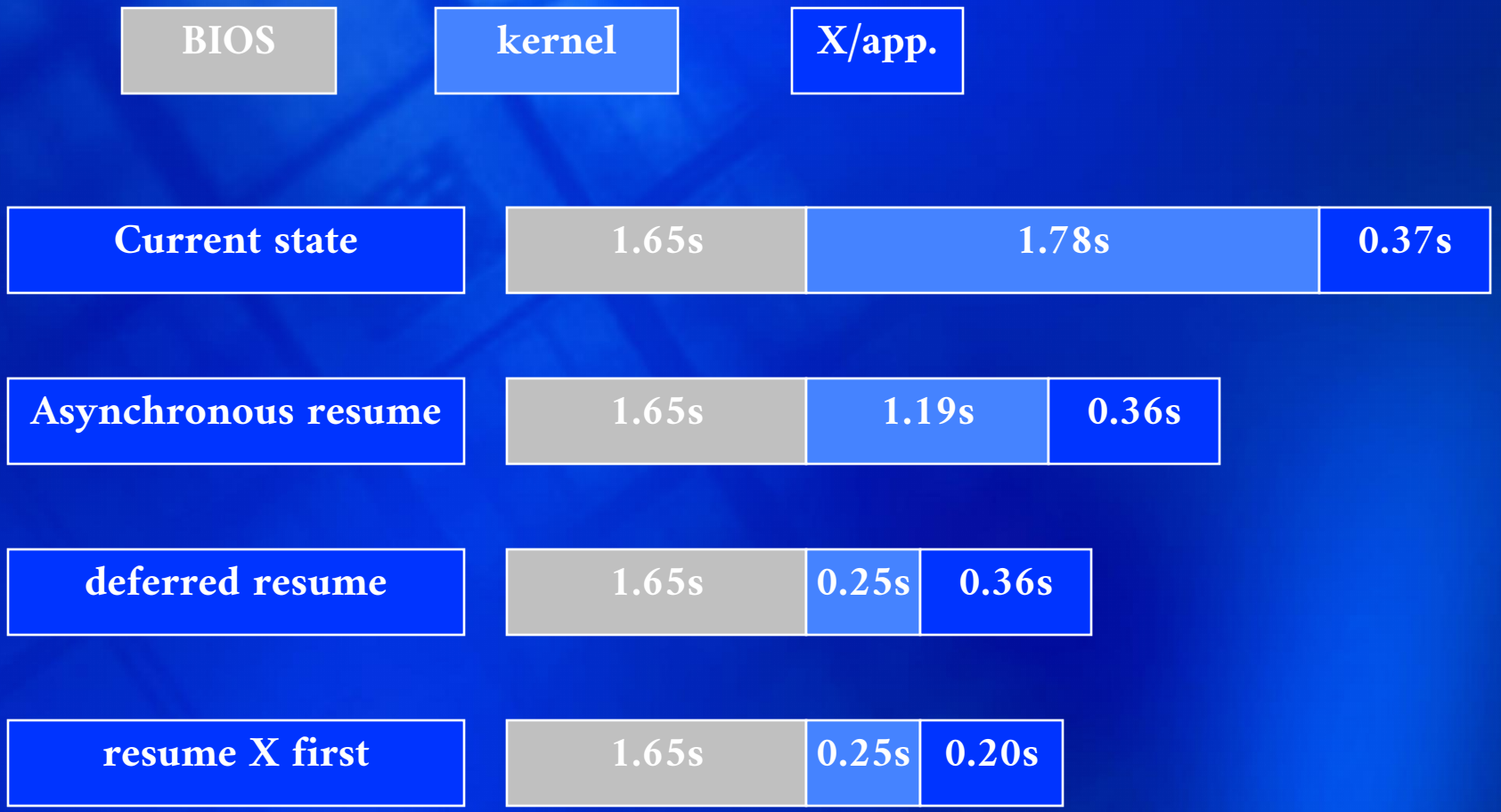


  - Screen is back before all the devices are resumed
  - works but may bring potential risks

More aggressive attempts

- Resume X with the highest priorities
  - Resume drm driver first to light the screen
  - Resume X process before the other kernel drivers
  - Works but not for upstream yet

S3 resume time Moblin



TODO

- push “deferred resume” mechanism upstream
- Fast suspend
- Fast shutdown

Acknowledgements

- Arjan van de Ven
- Auke Kok
- Li, Shaohua
- Zhao Yakui


intel®
Leap ahead™