Power Management

Feng Qin
CSE Dept., The Ohio State University
Outline

• Linux Power Management
• Android Power Management
• Ongoing Research Project
Linux Power Management

• APM (Advanced Power Management)
  – Controlled by BIOS (need reboot for re-configuration)
  – Activated when system becomes idle
    • E.g., screen saver => sleep => suspend
  – OS has no knowledge about when the system change power states

• ACPI (Advanced Configuration and Power Interfaces)  ⇐ Newer
ACPI

• First released in 1996 by Compaq/HP, Intel, Microsoft, Phoenix and Toshiba

• ACPI is an interface specification comprised of both software and hardware elements

• Controlled by OS
  – During OS initialization, OSPM takes over the power management functions
ACPI Functions (I)

• System power management
  – the entire computer

• Device power management
  – ACPI tables describe motherboard devices, their power states, the power planes the devices are connected to

• Processor power management
  – When OS is idle but not sleeping, it puts processors in low-power states
ACPI Functions (II)

• Device and processor performance management
  – When system is active, OSPM balances the performance and energy conservation

• Configuration / Plug and Play
  – Information used to enumerate and configure motherboard devices

• System Events
  – A general event mechanisms (thermal events, power management events, docking, etc.)
ACPI Functions (III)

- Battery management
  - Move policy from APM BIOS to ACPI OS
  - Require a Smart Battery subsystem interface
- Thermal management
  - Allow OEMs to define thermal zones, thermal indicators, and cooling methods
- Embedded controller
- SMBus controller
Global System Power States

(Source: ACPI Specification)
Linux System Power Management States

- Three possible system states
  - Standby / Power-On Suspend
  - Suspend-to-RAM
  - Suspend-to-Disk
Standby / Power-On Suspend

• ACPI State: S1
• String: “standby”
• Offer minimal power saving, with low-latency transition back to working
• Put devices to a low-power state D1
  – If not support D1, left them on
• Transition from standby to on takes about 1-2 seconds
Suspend-to-RAM

• ACPI State: S3
• String: “mem”
• Offer significant power savings
• System and device states are saved and kept in memory
• All devices are suspended and put into D3
• Memory is placed in self-refresh mode to retain its contents
• Resume takes about 3-5 seconds
Suspend-to-Disk

- ACPI State: S4
- String: “disk”
- Offer the greatest power savings
- Similar to STA, but add an extra step to write memory contents to disk
- Can be handled by the firmware (stored in swap space) or the kernel
Power Management Interface

- Linux provides a unified sysfs interface to userspace.
- In /sys/power directory (mounting sysfs in /sys first)
- /sys/power/state controls system power state
  - Read the supported states
  - Write to the file cause state transition
Runtime Power Management

- Runtime PM (Android uses it to some extent)
- Hardware components may be powered down when they are not used and powered up when they are needed again
  - Display dimmed or turned off
  - Reduce capacity of various system components if full capacity is not needed
Android Power Management

• **Wakelocks or suspend blockers**
  – Aggressive utilization of full system suspend to save as much energy as reasonably possible
  – Nature state of the system is a sleep state
    • Energy is only used for refreshing memory and providing power to a few devices that can generate signals to wake the system up
  – Working state is only entered in response to a wakeup signal from one of the select devices
    • When the work is done, the system goes to sleep state
Wakelocks

- Two types of wakelocks:
  - WAKE_LOCK_IDLE
  - WAKE_LOCK_SUSPEND

- When a “IDLE” wakelock is held, the system will not enter idle (low power) state
- When a “SUSPEND” wakelock is held, the system will not enter suspend state
- Currently defined wakelocks are listed at /proc/wakelocks
Manipulating a Wakelock

• Kernel space
  – wake_lock_init (struct wake_lock *lock, int type, const char *name);
  – wake_lock(struct wake_lock *lock);
  – wake_unlock(struct wake_lock *lock);

• User space
  – A process write a name to /sys/power/wake_lock
  – A process write the lock name to /sys/power/wake_unlock