BayLibre

Linux Embedded Technology Lab

Use-Case Power Management Optimization: Identifying & Tracking Key Power Indicators

ELC-E Edimburgh, 2013-10-24

Patrick Titiano

DRAFT!!!

- This is a draft version of the presentation
- Practical examples are still missing, but will be ready for the show! (plus further adjustments)
- Sincere apologies for the inconvenience!

BayLibre
Linux Embedded Technology Lab

Problem Statement

- Wireless Embedded platforms performances keep increasing
  - Multi-core processors (MPU / GPU) up to 2GHz+, H/W accelerators
  - High-Speed RAM (LPDDR3, Wide I/O) & peripheral buses (USB3)

- But power and thermal budgets remain roughly the same
  - Mobile phone: ~5W, case temperature < 45ºC, 1-day of active use

- => Power Management becomes **the** critical element.

BayLibre
Linux Embedded Technology Lab

What’s on the menu today?

- No meat, no fish, only power management stuff ☺
  - Starter
    - Critical Key Performance Indicators (KPI)
  - Main dish
    - Use-Case PM Optimization Methodology
      - Stuffed with practical examples & Thermal Management considerations
  - Dessert
    - Final Thoughts & Recommendations

BayLibre

Linux Embedded Technology Lab

Critical KPI (Key Power Indicators)

Statistics profiling platform activities, relevant to Power Management

BayLibre
Linux Embedded Technology Lab

Running Clocks

```
# cat /sys/kernel/debug/clock/summary
  ocp_abe_iclk              aess_fclk             98304000   27
  per_abe_nc_fclk           dpll_abe_m2_ck        98304000   0
  div_ts_ck                 l4_wkup_clk_mux_ck    1200000    1
  l4_wkup_clk_mux_ck        sys_clkin_ck          38400000   6
  lp_clk_div_ck             dpll_abe_m2x2_ck      12288000   0
  l4_div_ck                 l3_div_ck             100000000  62
  l3_div_ck                 div_core_ck           200000000  47
  dpll_mpu_ck               sys_clkin_ck          700000000  1
```

- Track running power resources
  - Clocks, DPLL, power switches, voltage regulators, …
- Highlight unnecessary running clocks & resources
  - Root cause of power switch(es) & voltage regulator(s) maintained ON
    - HW dependencies

BayLibre
Linux Embedded Technology Lab

C-States (Idle States) Statistics

```
# cat /sys/devices/system/cpu/cpu0/cpuidle/state*/usage
208814669
1124298
2263801
22351425
# cat /sys/devices/system/cpu/cpu0/cpuidle/state*/time
133059448774
3700489912
9190480361
943146521818
```

- Highlight
  - Cumulated time spent in each low-power states
  - Cumulated number of transitions into each low-power states
- Validate how much and deep CPU is able to sleep

Operating Point (OPP) Statistics

```
# cat /sys/devices/system/cpu/cpu0/cpufreq/stats/time_in_state
350000 1086631
700000 410910
920000 13505
1200000 401071
# cat /sys/devices/system/cpu/cpu0/cpufreq/stats/total_trans
132618
```

- Highlight
  - Cumulated time spent in each OPP (pre-defined [MHz/V] set)
  - Cumulated number of OPP transitions

- Assess processing requirements (low/medium/high MHz)
- Assess processing profile (bursty vs smooth)
- Monitor thermal management throttling

BayLibre
Linux Embedded Technology Lab

CPU & HW Accelerators Loads

CPU: # cat /proc/stat

```
cpu  7465 358 8079 3748103 6510 125 3458 0 0 0
cpu0 3713 158 4943 1868346 2681 125 3450 0 0 0
cpu1 3752 200 3135 1879757 3829 0 8 0 0 0
…
```

Other HW acc.: (GPU/DSP/ISP/…): proprietary / not standard instrumentation ☹

- Highlight
  - Processing scheduling over time
    - Processing requirements (low / medium / high / … MHz)
  - Most demanding applications – services / performance bottleneck
    - Lags, low frame rate, unresponsiveness, …
  - Root cause of Thermal Management Throttling
- Validate use-case modeling of activities

Memory Bandwidth

- Usually HW / Proprietary non-standard instrumentation ☹
- Track memory / bus occupancy
  - Data bus load (MB/s)
  - Memory / Bus latencies
- Highlight potential root cause of Lags, low frame rate, unresponsiveness, …
- Validate estimated bus & memory power consumptions

Interrupts

# cat /proc/interrupts

```
   CPU0        CPU1
   39:          6          0        GIC  TWL6030-PIH
  213:      22218          0       GPIO  wl1271
  393:          0          1    twl6040  twl6040_irq_ready
 IPI1:      22086      94686  Rescheduling interrupts
 IPI3:      68462      59269  Single function call interrupts
  LOC:     816383     411488  Local timer interrupts
```

- Track peripheral activities over time
- Highlight
  - Unexpected interrupt sources / rates
  - Potential root cause of reduced usage of CPU low-power states
  - Potential root cause of High latency / performance degradation
- Validate use-case MPU interrupts modeling

Timers

```
# cat /proc/timer_stats
# cat /proc/timer_list
```

- Track CPU wakeup sources and rates
- Highlight
  - Unnecessary CPU wakeup sources
  - Potential root cause of reduced usage of CPU low-power states
  - Potential root cause of High latency / performance degradation

Temperatures

# cat

- Track various temperature sensors
  - CPU, GPU, PCB, SDRAM, case, …
- Highlight power and performance degradation due to over-heating / thermal management throttling
  - Power consumption increases a lot (explodes?) with temperatures
    - Thermal runaway

BayLibre

Linux Embedded Technology Lab

Use-Case Power Management Optimization: Proposed Methodology

BayLibre
Linux Embedded Technology Lab

Modelize

- Define critical use-cases for your platform
  - MP3, AV-Payback, 3D Gaming, Capture, Idle, Voice-call, Web Browsing …
- Create a power model of your platform
  - MPU / GPU / Bus / Memory / Peripherals power consumption
    - Static (leakage), Dynamic (= f(MHz))
    - Temperature
- Create a power model of targeted use-cases
  - Split use-case into simple atomic functions (slices)
    - Required peripherals, processing loads and profiles, memory / bus bandwidth, data transfer sequence diagram, …
  - Must be measurable onboard
- Define power targets and thermal budget per use-case
  - Generated from power model

Instrument

- SW
  - Kernel
  - Power Management Frameworks
  - Scripts to reproduce use-cases
  - User-space tools to collect and process power data
    - See omapconf example

- HW
  - Lab equipment with high-resolution current probes
  - Sense resistors to measure current &voltage **simultaneously**
  - Temperature sensors (embedded, external)
  - HW trace
  - Embedded power measurement capabilities is a plus

BayLibre
Linux Embedded Technology Lab

Automate

- **Automation is KEY**
  - Apples must be compared to apples
  - Power, voltages, currents are analog variables
    - Inherent variations in measurements
  - Measurements should be repeated and averaged before analysis

- Long, annoying, **approximate & source of error** if not automated!
  - Bad practice examples (real ☹☹☹):
    - Power consumption of 10 different rails for 10 different use-cases reported by hand for measurement equipment to test report
    - boot time measured with a simple watch

Characterize Silicon raw performances

- Raw Leakage current & dynamic consumption (mA / MHz / V)
  - I/O
  - Low-power Retention states
  - CPU (Dhrystone, …), GPU (GLBench, …), other processing unit(s)
  - Bus
  - Cache, RAM
  - Peripherals
  - Temperatures
- To assess power model and power targets
  - Based on estimated Silicon power performances
  - Consider process corners / worst-case

BayLibre
Linux Embedded Technology Lab

Assess Power Model

- Compare raw Silicon power performances to estimates
- Refine power model with raw Silicon power performances measurements until converged
- Re-generate power targets accordingly

Measure use-cases

- Take multiple measurements of a same use-case
- Check that all measurements are in a same ballpark
  - Not exceeding ±5%
  - Example: 3 samples of a same use-case showing 50% to 100% variation between measures
    - Bad practice: report the average value (real)
    - Good practice: report issue with the measurement setup
- Collect and save all useful KPI statistics, for further analysis.

BayLibre
Linux Embedded Technology Lab

Analyze KPI for Leakage

- Static Power Consumption (a.k.a. leakage) always first

- Ensure no power is wasted
  - Supplied Voltages
  - Miss-configured I/O
    - Unused I/O not in high-impedance state, short-circuit
    - Bad pull-up /pull-down configuration:
      - Dual (at each end), combined up + down, unnecessary
  - Running clocks / DPLL instead of idle
  - Unused logic powered ON / not retained
  - Unused Voltage regulators left ON
  - Low-power states usage / Idle policies
  - SDRAM: self-refresh / power-down / other IP-specific power features

Analyze KPI for Dynamic Consumption

- Once leakage is under control, chase for extra processing / bottlenecks

- CPU / GPU / HW Accelerators
  - Supplied Voltages
  - Processes, timers, interrupts, sleep durations & levels
  - Processing loads (and profiles) vs estimations
  - CPU IPC performances (latencies, rates)
  - OPP statistics / DVFS and idle policies efficiencies
  - Cache efficiency

BayLibre
Linux Embedded Technology Lab

Analyze KPI for Dynamic Consumption

- Bus / SDRAM
  - Supplied Voltage
  - Assess loads vs estimations
  - Assess latencies
  - Assess idle duration
  - SDRAM: refresh cycle rates, …
- Filesystem

Analyze Temperature

- Keep temperature within expected limits for a given use-case
  - Fine-tune DVFS policies
  - Shutdown unnecessary logic

- Heating increases power consumption
- Heating degrades performances
  - CPU/GPU throttling

Fix!

- Code
- Power Model

- Iterate until targets and measurements converge
  - Discussions (negotiations) with architects and developers
    - Implementation? Power Estimations? Both?

  - Set an acceptable limit
    - Usually power targets cannot be reached or exceeded
    - Define when to stop optimization

BayLibre
Linux Embedded Technology Lab

Track

- Do not let power diverge again
  - Monitor power consumption over new releases until the end of the development life-cycle
  - Be strong, reject patches hurting power
    - The same way patches hurting performances and stability are
- Yes, you’re never done! ☺
  - Tracking phase should be fully automated, ultimately

BayLibre

Linux Embedded Technology Lab

Example of Power Optimizations

BayLibre
Linux Embedded Technology Lab

- To be added

BayLibre

Linux Embedded Technology Lab

Conclusion &
Final Recommendations

Anticipate

- Chip and board shall be designed for power measurement
  - Accessible probe points on voltage rails
  - Use 0-ohm resistor as placeholders to be replaced by sense resistors
  - Design power companion chip with
    - Embedded power sensors
    - HW debug logic to trace power states & transitions
      - Ultimately synchronized with SW markers
- SW shall be instrumented for tracing power management decisions

Partition HW for Power

- Do not build house with a single light switch
  - Dedicated clock switch per peripheral
  - Peripherals grouped per use-case under power switch(es)
  - Avoid sharing scalable voltage regulator(s)
  - Use retention techniques to reduce sleep/wakeup latencies
- Voltage is KEY
  - Power is proportional to the square of V
    - P = a * C * V2 * f

Fine-tune policies

- The perfect policy does not exist
  - Default policies cannot perform nicely for all use-cases

- Default Linux upstream policies made for desktops & servers, not embedded devices
  - Fine-tune parameters for critical use-cases
  - Develop your own policies

- Do not hesitate to detect use-case & switch policies on the fly

BayLibre
Linux Embedded Technology Lab

Keep Temperatures Down

- “Easier” to waste less power than find mechanical solutions to dissipate more power
  - Embedded devices are not desktop PC or servers
    - No fan, only a case … and your skin …
- Power consumption increases with temperatures
- Minimize use of performance throttling

BayLibre
Linux Embedded Technology Lab

Battery is what really matters

- Final goal is to optimize power consumption at battery level
  - Focus attention on main contributors
    - No need to save 30% of power on a rail that only accounts for 2% total
  - Think system, pay attention to side-effects
    - Doing a power optimization on one end may degrade it at another end
    - E.g.: reducing clock rates may lengthen active time and increase DPLL lock time

BayLibre
Linux Embedded Technology Lab

Q & A

Thank you!