

The Mobile R&D Fight: Battling Battery Drain at the HAL Layer (AOSP Expert)



Esmaeel Moradi
@esmaeel-moradi700



The Mobile R&D Fight:

Battling Battery Drain at the HAL Layer (AOSP Expert)

**How C/C++ Drivers and System Services Control Power
Consumption in Android.**



Esmaeeil Moradi
@esmaeeil-moradi700



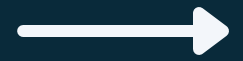
The Problem: Unwanted Device Wake-ups

The primary culprit behind rapid battery drain is unnecessary device activation (Wake-up Events), often originating from unoptimized or legacy Vendor hardware drivers and deep kernel interactions.

To precisely manage CPU/GPU frequencies and enable Deep Sleep states, we must correctly implement and tune the Power HAL and Thermal HAL using low-level C++ code.



Esmaeeil Moradi
@esmaeeil-moradi700



Power HAL: The C++ Frequency Controller

Power optimization is a C/C++ battle. We use the Power HAL to translate high-level framework demands into hardware reality:

- 1. Kernel I/O:** Define thermal and frequency parameters within the Device Tree and expose control via kernel sysfs nodes.
- 2. Power HAL Implementation:** Implement Stable AIDL methods in the C++ HAL service (e.g., `setPowerBoost()`, `setLowPowerMode()`).
- 3. The Link:** Translate Framework commands (via AIDL) into specific sysfs write operations to dynamically adjust CPU/GPU clock speeds.

Result:

Guaranteed hardware operates at peak frequency only when absolutely necessary, minimizing idle power consumption.



Esmaeeil Moradi
@esmaeeil-moradi700



Hunting Bottlenecks: Syscalls, Binders, and Latency

Pinpointing which C++ service is needlessly keeping the device awake is impossible without AOSP tooling. Our focus is on tracing every system event.

- 1. Perfetto (Modern Systrace):** Use adb shell perfetto to collect detailed traces of kernel events, Userspace threads, and Binder transactions.
- 2. Call Stack Analysis:** Trace a Binder request from the Java Framework down to the specific Syscall in the kernel to identify the exact source of a Wake-up lock.
- 3. Ftrace/Tracepoints:** Utilize Ftrace within the kernel to monitor the precise entry/exit times of hardware drivers to measure sleep/active state accuracy.



Esmaeel Moradi
@esmaeel-moradi700



OTA Stability: The Role of VNDK and A/B Updates

Our optimized HAL code must be resilient to future OTA updates and flash reliably across devices:

- 1. VNDK Compliance:** Build HAL modules using the VNDK (Vendor Native Development Kit) to ensure ABI compatibility with system libraries, even when the Framework updates.
- 2. A/B Updates:** Ensure HALs are correctly installed in logical partitions to facilitate seamless, no-downtime updates.
- 3. SELinux:** Restrict Power HAL service access using specific SELinux policies, allowing access only to power-related kernel nodes (e.g., sysfs power controls), not the entire file system.



Esmaeel Moradi
@esmaeeil-moradi700

✦ Optimizing mobile performance is a deep dive into C/C++ and the HAL.

Is your team effectively using Perfetto to root-cause power issues?

Share your expertise below! 📵



Esmaeel Moradi

@esmaeel-moradi700