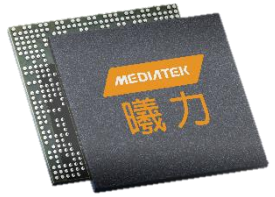
MTK平台優化方案提供

Rex Lin

MTK平台優化方案提供

- 安卓平台架構
- MTK平台優化方案
  - **EAS+** (energy aware scheduler plus, 線程排程)
  - **FPSGO** (FPS global optima, 帧率調整機制)
  - **PowerHAL** (hint)
- 進階合作方案 (Open Discussion)
  - APP hints for FPS target
  - APP hints for workload variance
  - Hint APP for bounding
  - 遊戲回放檔
  - 案例分享: UnityWorker 異常行為

介紹思路與內容連結(1/2)

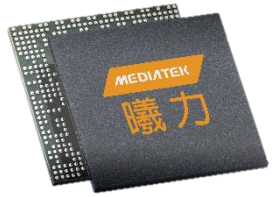


1 玩的順

2 玩的久

- 如何玩的順,玩的久?
  - 線程怎麼擺放?
  - 大小核能力怎拉?
  - CPU0是大核?
  - CPU有哪幾種架構?
  - 遊戲需求怎麼滿足?
  - 如何讓電池續航力變久?
  - …

介紹思路與內容連結(2/2)



一般設計行為

優化方案導入

應用商

線程綁定

定頻定核

特定手法

- 如何玩的順,玩的久?
  - 線程怎麼擺放?
  - 大小核能力怎拉?
  - CPU0是大核?
  - CPU有哪幾種架構?
  - 遊戲需求怎麼滿足?
  - 如何讓電池續航力變久?
  - …

EAS+

FPSGO

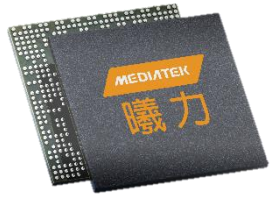
PowerHAL

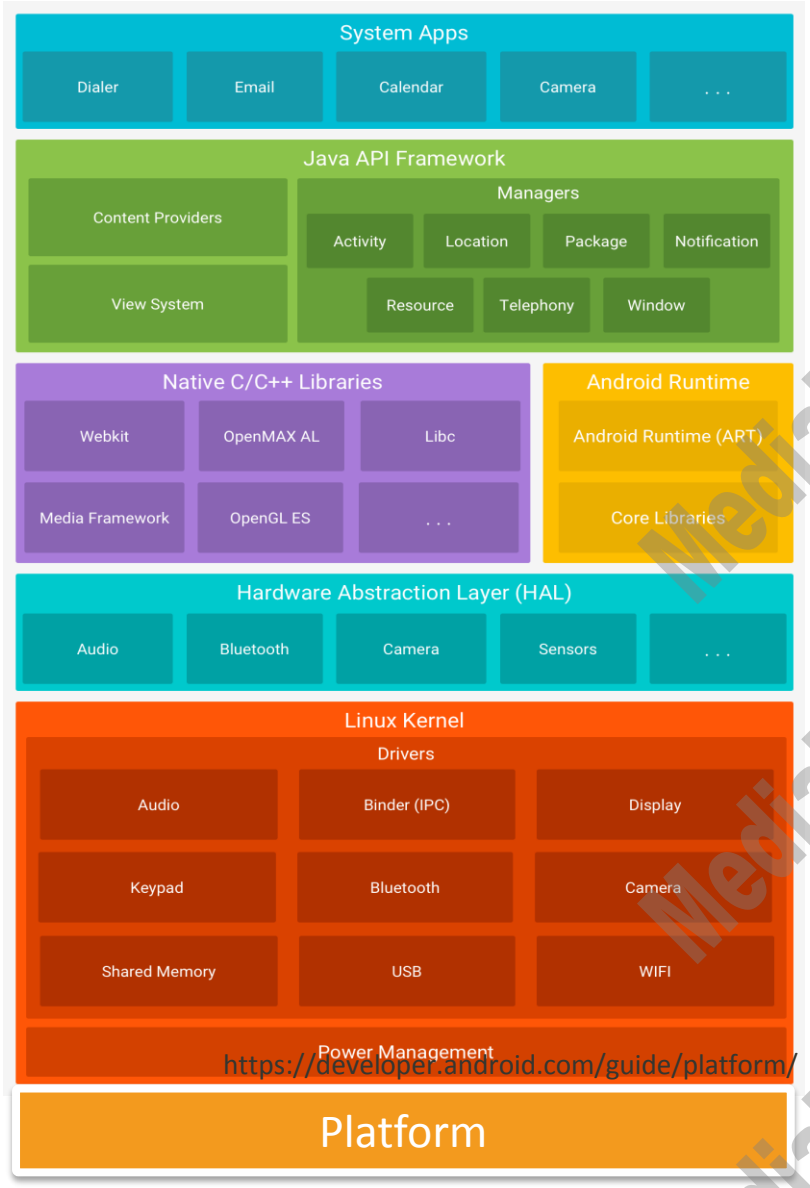
定頻定核

手機商/
平台商

應用商

安卓平台架構 (1/2)

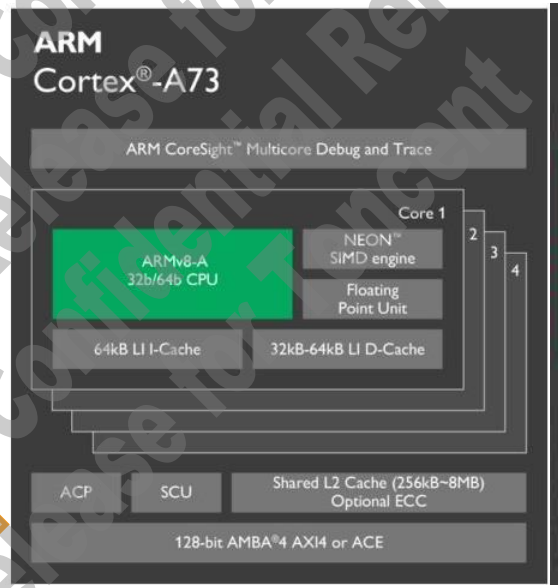


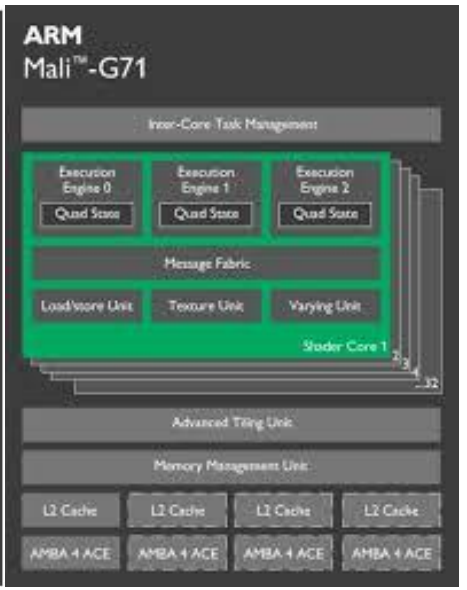


https://developer.android.com/guide/platform/

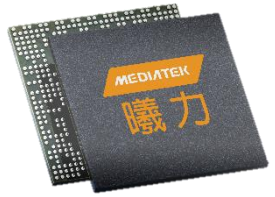
手機商/平台商

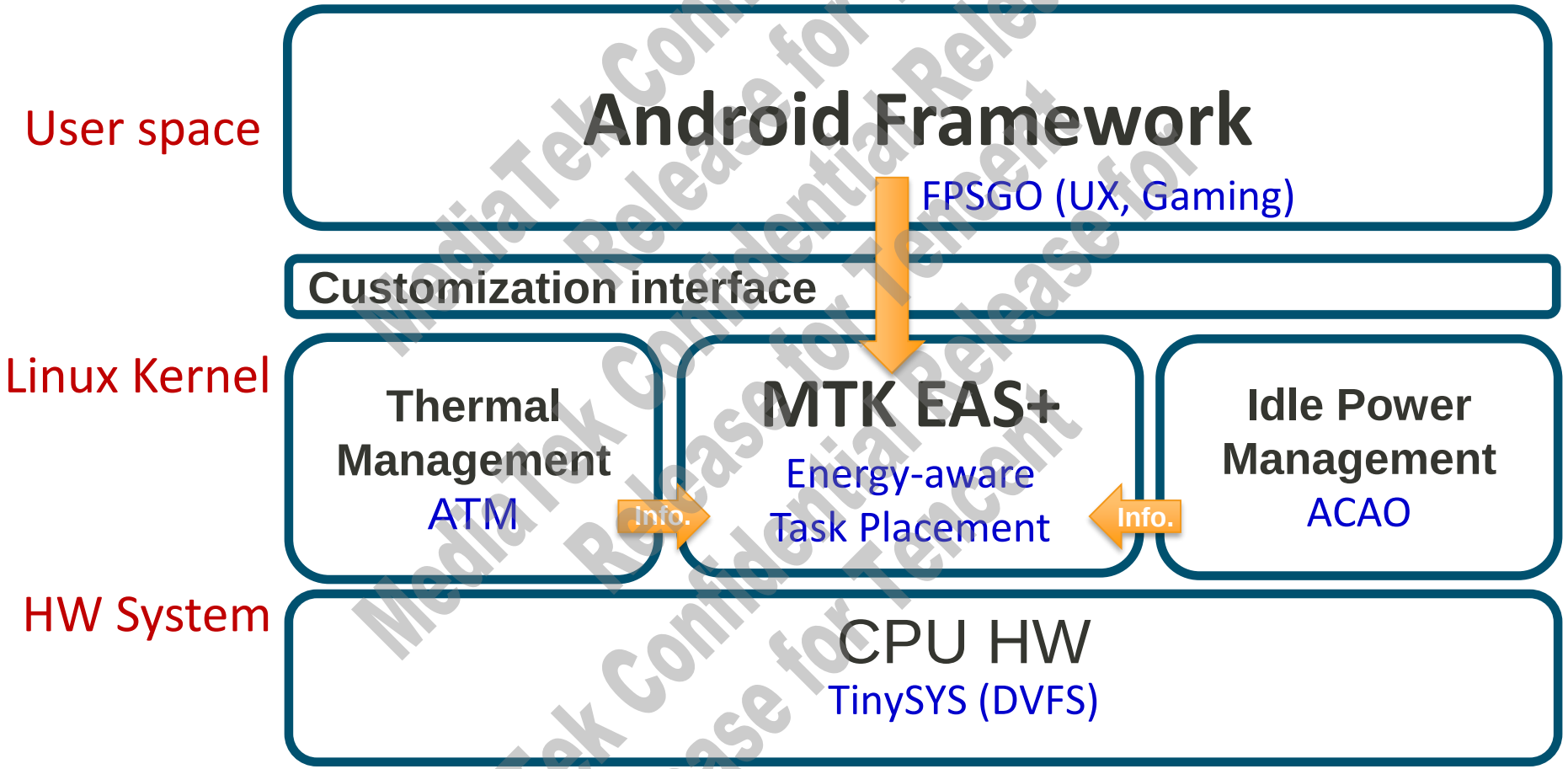
*維持整個平台設定取向 (效能/功耗/散熱)
*判讀應用需求並設定特定取向 (性能 或是 功耗 優先)
*硬件規格定義以及平台細節優化





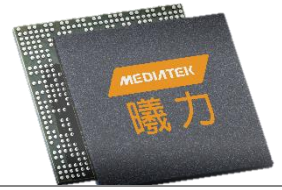
安卓平台架構 (2/2)

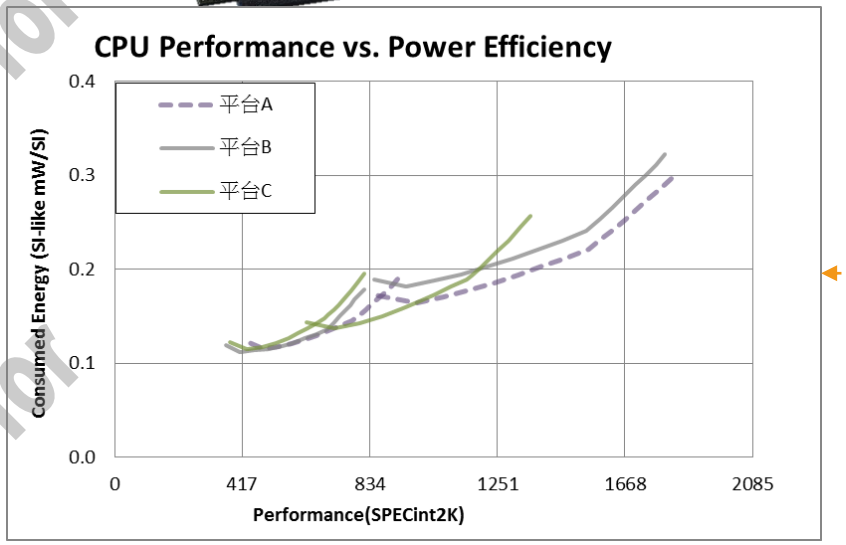


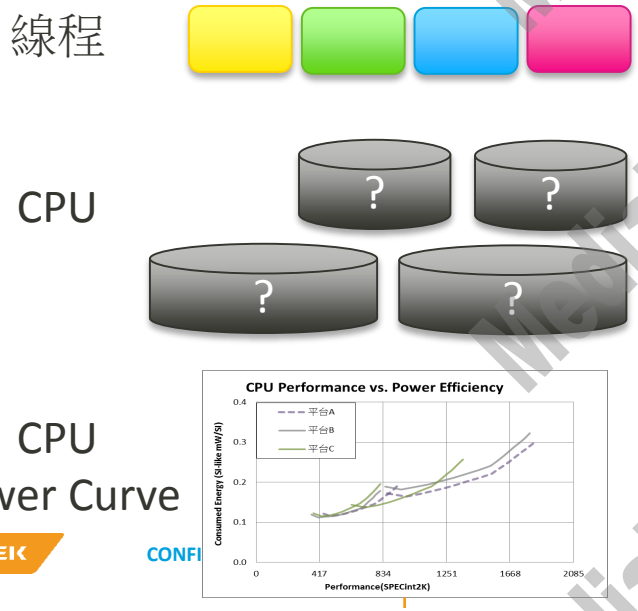


MTK EAS+

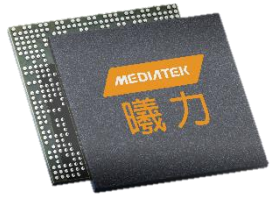
- 技術特點
  - EAS+ scheduler: energy-aware task placement
    - 將線程根據workload放在合適CPU上
    - 有標準接口可調整workload的變化曲線
  - 性能/功耗 客製化接口
    - Google Android standard (CPUSet, STune)
    - MTK擴充支援更多元化的需求





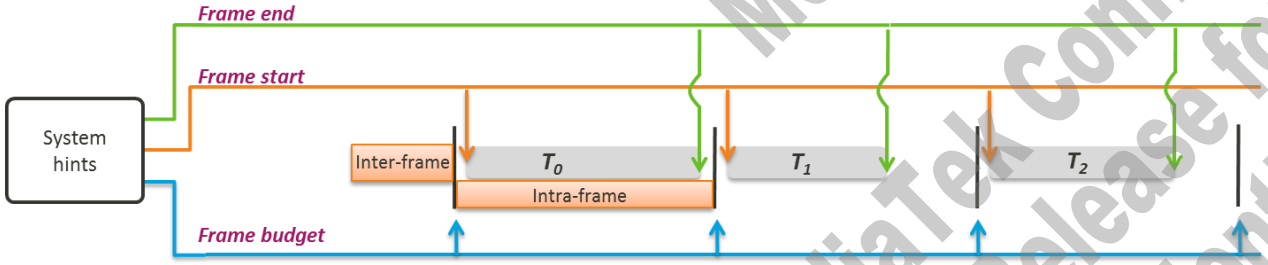


MTK FPSGO

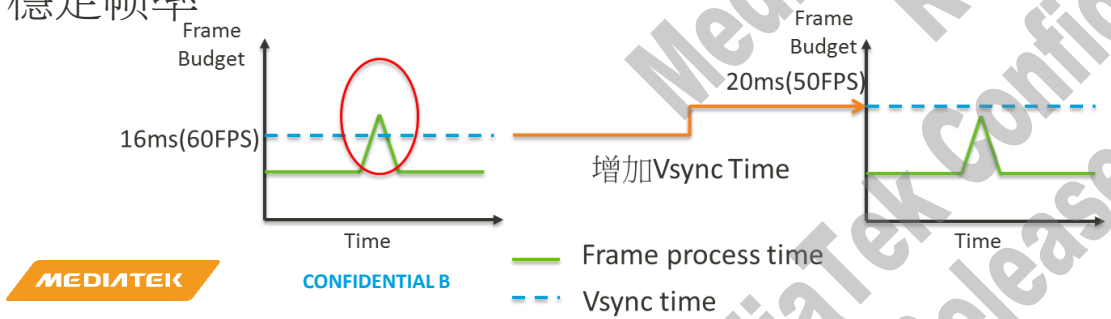


- 技術特點
  - 不需要特調
  - 預估每一框需求並提供剛好系統效能, 省下power.
  - 追帧設計減少掉幀行為
  - 系統能力夠,可以達到最佳FPS帧率
  - 系統能力有限制,透過adaptive機制調整FPS帧數得到更好的體感
  - 遊戲過程中,系統能力可以跑60FPS就跑60FPS,若過程有重載只能跑到50,則會動態調整到50FPS,回到輕載時再切回60FPS.

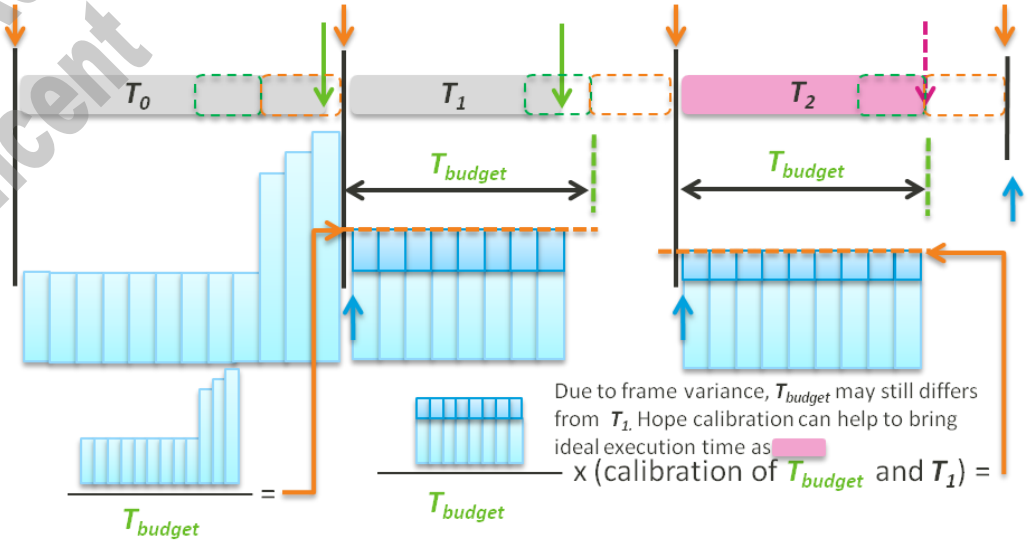
*系統hint



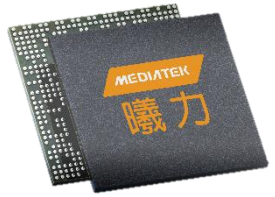
*穩定帧率



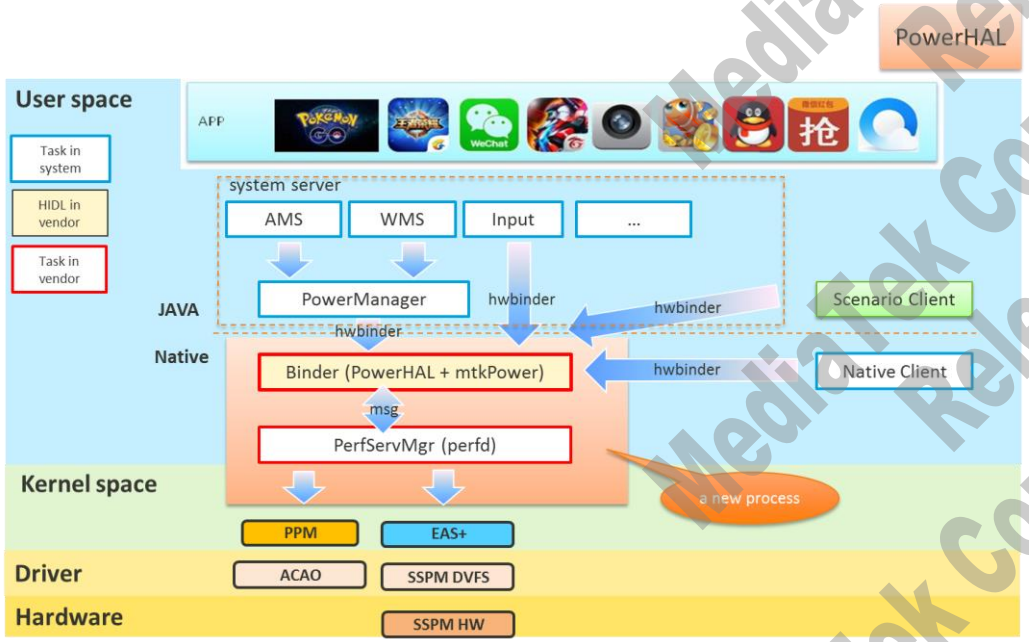
*追帧設計



MTK POWERHAL



- 技術特點
  - 提供應用控制平台設定接口
  - 提供中控邏輯,同時可滿足不同應用需求(性能優先)
  - MTK所有平台皆支援



Quick Start

Step 1: MTK Power Hal Version = ?

Step 2: Android.mk

Assumption: MTK Power Hal Version = 1.1

--> vendor.mediatek.hardware.power@**1.1_vendor**

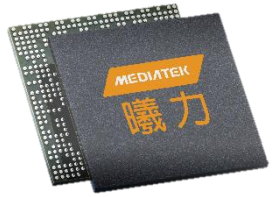
Step 3: Get Service

```
android::sp<IPower> gPowerHal;
gPowerHal = IPower::getService();
```

Step 4: Use the Hint

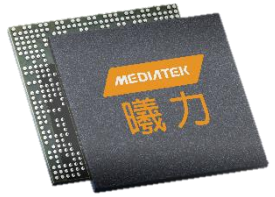
Step 5: Check sepolicy

進階合作方案

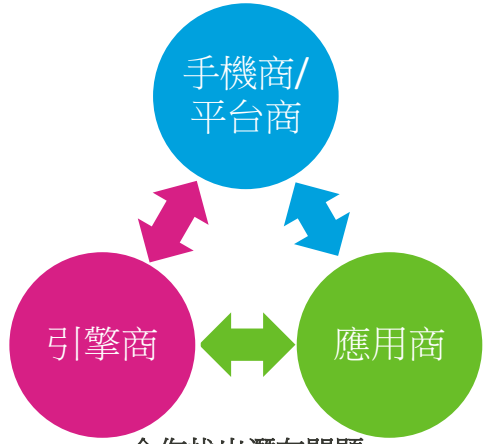


- APP hints for FPS target
  - FPS高, 相對功耗會增高
  - 若可以針對遊戲特性調整(低) FPS, 可以帶來續航力的提升
- APP hints for workload variance
  - 低負載系統跑低速, 高負載系統跑高速, 才不會浪費功耗
  - Workload variance 可能造系統跑低速,突然來高負載而造成掉張
- Hint APP for bounding
  - 當平台遇到系統極限,預期得到什麼標的? FPS? 畫質/特效?
  - APP若可以知道遇到bounding而進行調整,對於使用者體驗是否更好?
- 遊戲回放檔
  - 提高复現率可以加速問題分析

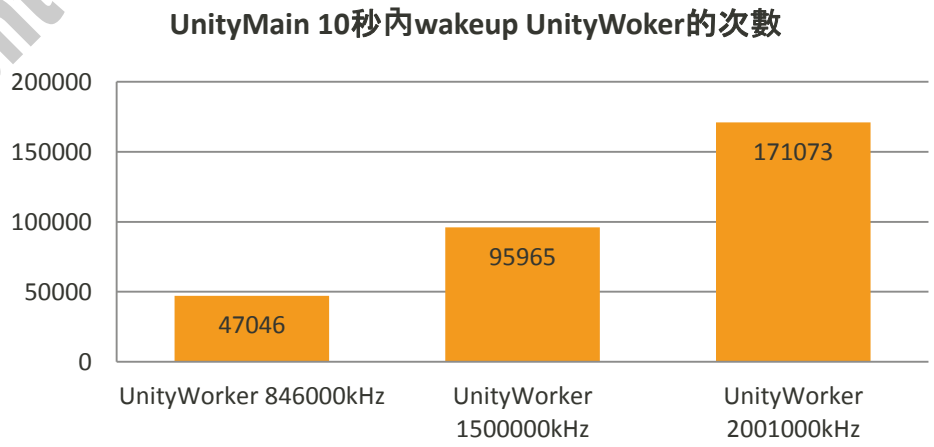
案例分享: UnityWorker 異常行為



- 問題
  - 案例遊戲 自帶的UnityWorker 會根據跑在不同CPU頻率而有不同行為
- 影響
  - 整場遊戲平均 FPS 差了約2張
- 分析
  - UnityMain 每次wakeup UnityWorker 需要承擔部份scheduling overhead
  - 若系統將UnityWorker放到高頻率CPU會導致UnityMain 頻繁wakeup 行為倍增
  - UnityMain跟UnityWorker的關係,從平台端無法看出關聯性
  - 預期UnityWorker應該是減輕UnityMain的運算量,在此場景上反而得到反效果



- 手機商/平台商釐清現象後, 需要engine協助確認行為與問題點
- 因為直接影響到遊戲體驗,因此也會需要協同應用商找engine討論



Q & A

THANK YOU!