HYPNOS: Understanding and Treating Sleep Conflicts in Smartphones

Martín Carreiro - Pablo Rago - Juan Manuel Tastzian

Facultad de Ciencias Exactas y Naturales

21 de Mayo de 2014

Contexto

- Es la primera vez en la historia de la computación en donde la energía es un problema

Contexto

- Es la primera vez en la historia de la computación en donde la energía es un problema
- Mismo las notebooks, están pensadas para estar mayormente conectadas

Contexto

- Es la primera vez en la historia de la computación en donde la energía es un problema
- Mismo las notebooks, están pensadas para estar mayormente conectadas
- Y los teléfonos tienen que durar el día entero...

Contexto

- Es la primera vez en la historia de la computación en donde la energía es un problema
- Mismo las notebooks, están pensadas para estar mayormente conectadas
- Y los teléfonos tienen que durar el día entero...
- Se introduce el Modelo de Manejo de Energía

Modelo de Manejo de Energía

Desde el CPU

- Después de un período de inactividad, el teléfono ingresa en un estado de suspensión
- Esto incluye el SOC: CPU, ROM, micro-controladores de varios dispositivos como el GPS, gráficos, video y audio
- La RAM ingresa en un estado de actualización
- El CPU intentará acceder a dicho estado cada 20ms, a menos que alguien lo interrumpa (veremos quien puede)
- En este estado el CPU consume casi-cero energía.

Modelo de Manejo de Energía

Desde el desarrollador

- Los desarrolladores tienen que manejar el estado de cada uno de los dispositivos que utiliza (GPS, WiFi, etc)
- El manejo de energía se convierte en una preocupación primaria
- Además de todos los bugs de la app, se tienen que manejar los Energy Bugs
- Dolor de cabeza

Mecanismos de manejo de energía

- Wakelocks
- Suspend Notifiers
- Hardware wakeups

Wakelocks

Wakelocks

- Mecanismo de alto nivel para prevenir la suspensión del SOC.
- Tienen una API mediante la cual se pueden obtener y liberar.
- Opcionalmente cuando un wakelock se obtiene se puede especificar un timeout para que después de ese plazo se libere automáticamente.
- Para usarlos tienen q ser inicializados lo cual los registra en el wakelock manager del kernel

Suspend Notifiers

Suspend Notifiers

- Mecanismo de bajo nivel que permite proseguir o abortar la suspensión del SOC
- SOC notifica a los registrados que se quiere suspender
- Si nadie devuelve error, se suspende el SOC
- Si alguien devuelve error, el proceso de suspensión es abortado
- Antes que esto corrobora que no haya wakelocks tomados

Hardware Wakeups

Hardware Wakeups

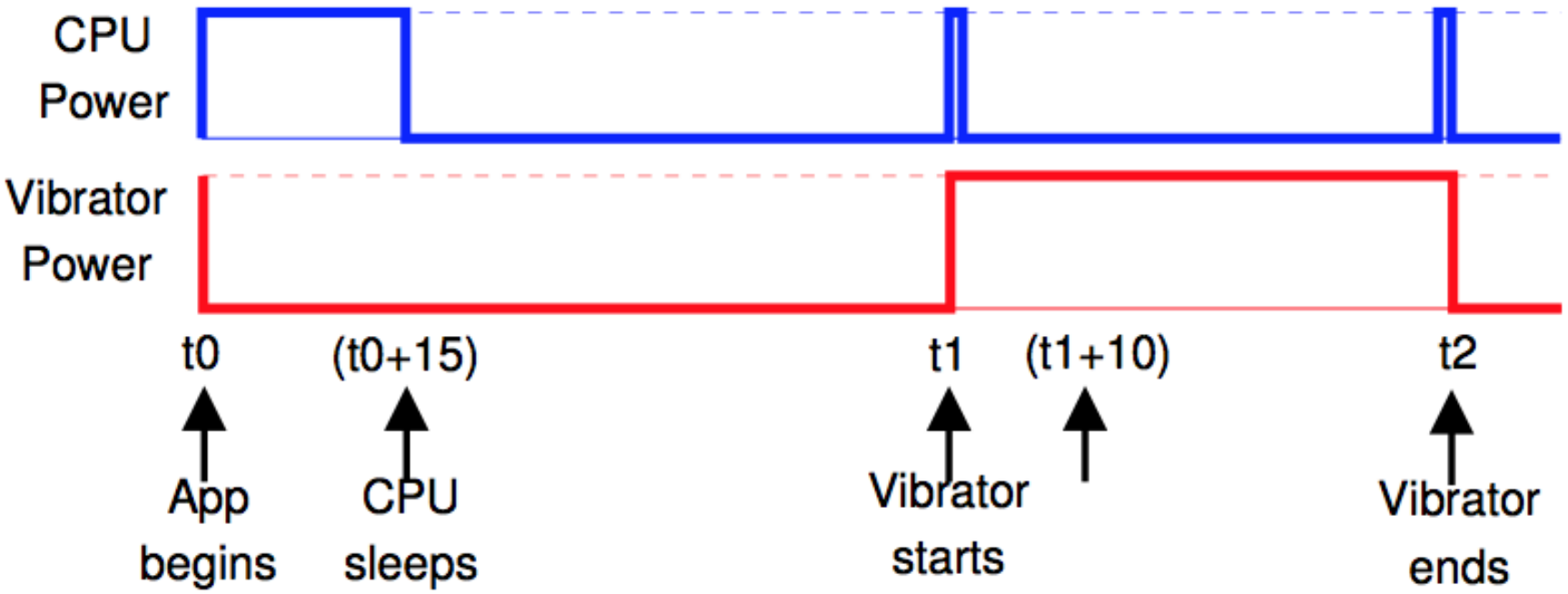
- A diferencia de los anteriores, no impiden que duerma, si no que despierta al SOC cuando lo necesita.
- Por ejemplo el Real Time Clock (RTC). Este es esencial para soportar servicios que deben ejecutarse a intervalos fijos de tiempo. Su API permite especificar un timer con un callback.
- Otros dispositivos con este poder: Wifi, wake-on-lan, llamada entrante
- Estos son los únicos que pueden!

Un Tipo de Energy Bug: Sleep Conflict

Los *sleep conflicts* pasan cuando un dispositivo en un estado de alto consumo de energía es incapaz de hacer la transición de vuelta al estado de energía básico porque el CPU esta dormido y el controlador del dispositivo no puede progresar en su ejecución para hacer la transición.
Tres Factores que dan lugar a *sleep conflicts*:

- Un dispostivo puede estar en uno de varios estados de energia independientemente del estado de energia del SOC
- El driver del dispositivo controla las transiciones de estados de energia
- El SO usa una politica agresiva de suspension del sistema.

Ejemplo


CPU
Power
Vibrator
Power
t0
(t0+15)
t1
(t1+10)
t2
App
begins
CPU
sleeps
Vibrator
starts
Vibrator
ends

Evitando sleep conflicts con suspend notifiers

- Hacer que pare el dispostivo y luego permitir a la CPU suspenderse
- Abortar el proceso de suspensión
- Registrar un timer RTC que se dispare cuando el dispositivo vaya a tener que ejecutar el código que lo apaga

Clasificación de los sleep conflicts

- Esperando a que termine la utilización activa
- Esperando por una transición mediante un timeout
- Esperando por una transición por input del programa
- Esperando por una transición mediante un evento externo

HYPNOS

¿Qué es?

Es un programa que en tiempo de ejecución permite evitar bugs de sleep conflict en drivers de dispositivos. Funciona haciendo el checkeo en forma online mientras las aplicaciones se ejecutan normalmente, y ademas permite testear la aplicación en busca de sleep conflicts antes de ser deployada.

HYPNOS

Funcionamiento

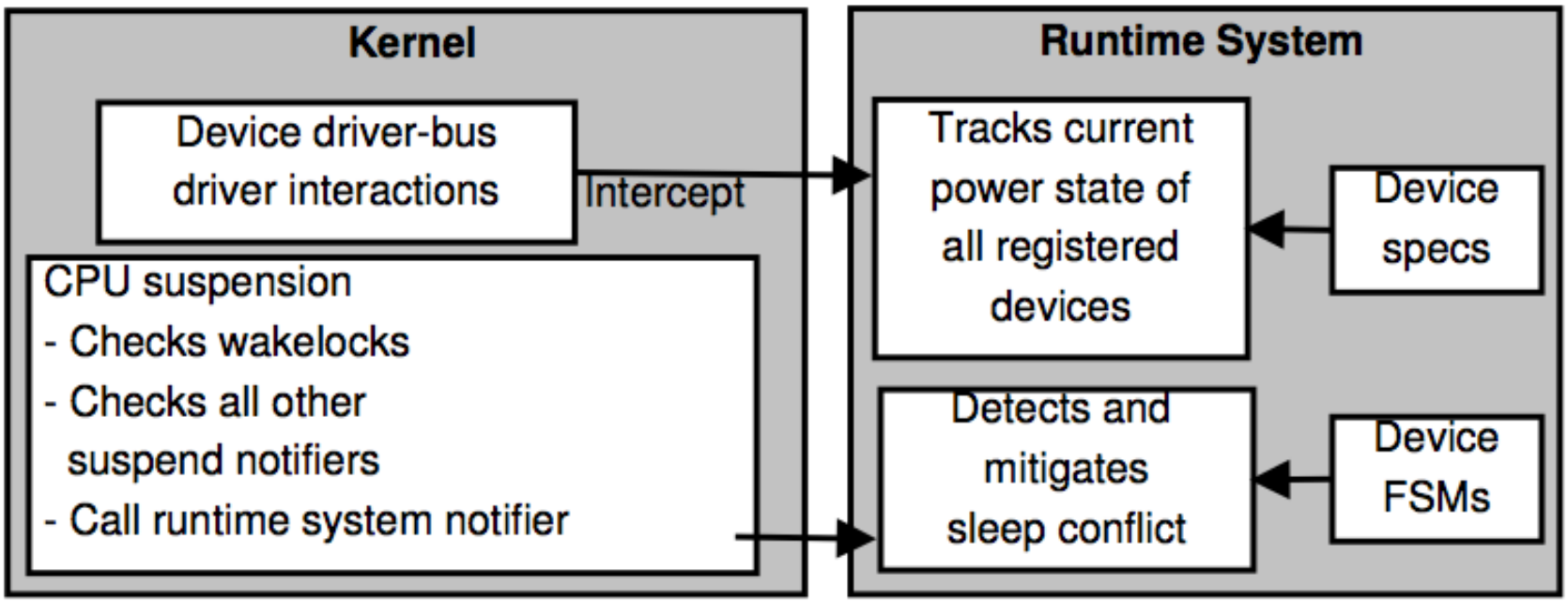
El enfoque tomado para tratar los sleep conlicts es intentar evitarlos en primer lugar. Esto se logra checkeando las condiciones que podrían llevar a un sleep conflict a cada pedido de suspención del sistema. Para esto no se necesita conocer el código fuente de los drivers de los dispositivos siempre y cuando se cuente con una forma de monitorear las transiciones de energía y se tenga acceso a los modelos de FSM de cada dispositivo. Para generar los modelos de FSM de los dispositivos de un smartphone dado se realizan procesos de ingeniería inversa offline.

HYPNOS

Arquitectura

El sistema monitorea las transiciones de energía interceptando las interacciónes con el bus de los drivers y checkeando por posibles condiciones de sleep conflict justo antes de que el sistema se suspenda para ejecutar las acciones preventivas acordemente.

HYPNOS


Kernel
Device driver-bus driver interactions
Intercept
CPU suspension
- Checks wakelocks
- Checks all other suspend notifiers
- Call runtime system notifier
Runtime System
Tracks current power state of all registered devices
Device specs
Detects and mitigates sleep conflict
Device FSMs

Chequeo de condiciones

En algún punto cuando el SOC/CPU esta a punto de entrar en estado de suspencion se checkea:

- Si todos los dispositivos estan ya en estado de suspension
- Si algunos no, entonces trata de determinar si los drivers invocaron algun mecanismo que asegure que el sistema vaya a estar despierto transcurrido el periodo de tiempo en el cual la transicion del estado de energia vaya a ser llevada a cabo.

Evitar Conflictos Online

Cuando el suspend notifier de HYPNOS es invocado al final del proceso de suspensión del sistema se checkean las condiciones necesarias para que ocurra un sleep conflict y se ejecutan los algoritmos necesarios para evitarlo.

Pseudocódigo (1)

```
for all devices not in base state do
        if next transition is end of utilization then
            return false
        end if
        ...
    end for
    return true
```

Pseudocódigo (2)

```
for all devices not in base state do
        ...
        if next transition is timeout then
            next wakeup time = rtc get alarm(..)
            if next wakeup time > timeout time then
                rtc set alarm( timeout time );
            end if
        end if
        ...
    end for
    return true
```

Pseudocódigo (3)

```
for all devices not in base state do
      ...
       if next transition is external event then
        if device has not registered hardware wakeup then
            return false
        end if
       end if
       ...
   end for
   return true
```

Pseudocódigo (4)

```
for all devices not in base state do
        ...
        if next transition is user input then
            return false
        end if
    end for
    return true
```

Wakelock de inactividad

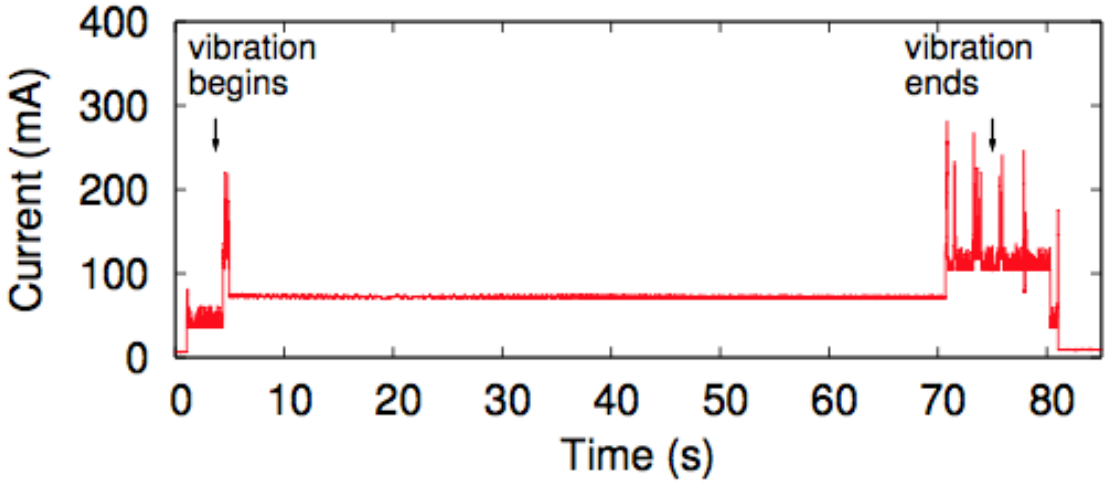
Recordemos que cuando el sistema esta prendido, el CPU usa el RTC timer para reintentar suspenderse cada 20ms. La politica de suspensión por inactividad, que suspende el sistema luego de 15 segundos de inactividad del usuario es implementada por el manager de energía del kernel que retiene un wakelock especial para este propósito por ese período.

Testeo preliminar

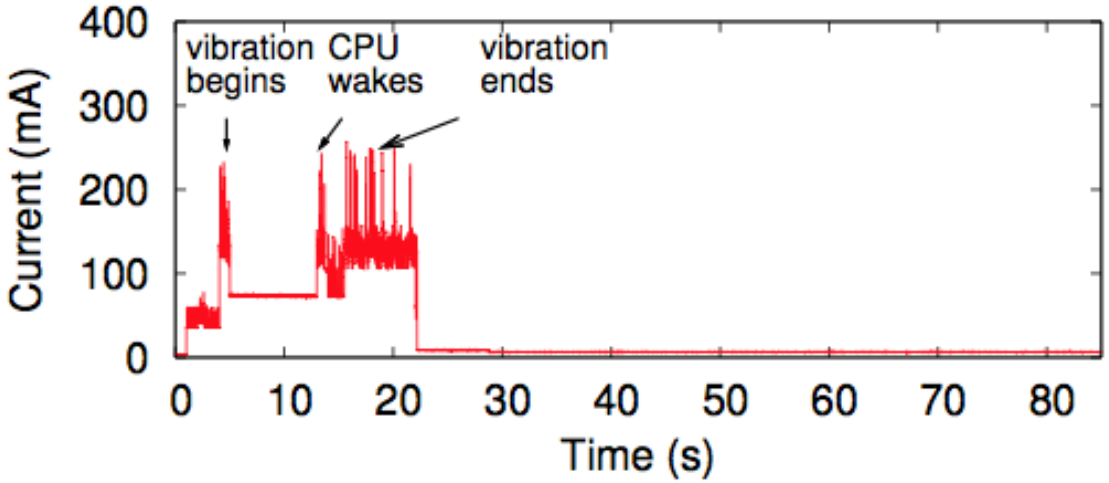
Cuando un driver realiza la transición de su dispositivo a un estado de consumo de energía activo, entonces o se ha invocado un mecanismo de prevención de sleep conflict o no. Si no, tenemos un potencial sleep conflict, pues el sistema podría suspenderse en cualquier momento. Esta condición puede ser verificada forzando al sistema a tratar de ejecutar el proceso de suspención inmediátamente después de que el dispositivo haya entrado en estado de consumo activo de energía. Se logra liberando el wakelock de inactividad

Algunos resultados

- Hypnos fue probado en un Google Nexus One y un HTC Magic con Android 2.3.
- Se probee una api para que desarrolladores de drivers puedan registrar FSMs con los modelos de energía de sus dispositivos y probar sus drivers.
- Evaluación de overhead: se testeó una transferencia de una gran cantidad de datos via WIFI, proceso con muchas interacciones con el driver, y se obtuvo, en promedio, entre un 1 % y 2 % de overhead al utilizar HYPNOS



(a) Vibrator sleep conflict.



(b) Sleep Conflict mitigation by HYPNOS.