Android PM, Deadline(RM, EDF)

국민대학교 임베디드 연구실
경 주 현

Outline

- **Android PM**

- **Deadline Scheduler**

Android PM

Android PM

- Android inherits the Linux PM
- Wakelocks
- Scheduling Wakeups
- Early Suspend

Task may be blocked waiting on I/O

- **Problem:**
  - Task may be blocked waiting on I/O
  - Ex) Weather application
    - Waiting for a response from the remote server

Wakelocks

- **Problem:**
  - Task may be blocked waiting on I/O
  - Weather application
    - Waiting for a response from the remote server
- **Solution:**
  - Android uses wakelock
- **Wakelock**
  - While any wakelock is held the **system does not suspend**

Wakelock's problem

- **Wakelock's problem**
  - CPU can not go to suspend.
  - High power consumption

Scheduling Wakeups

- **Wakelock's problem**
  - CPU can not go to suspend.
  - High power consumption

- **Solution**
  - To allow applications to wake up at a specified time

- **Scheduling Wakeups**

Scheduling Wakeups

- To allow applications to wake up at a specified time
- Android introduces wakeup alarms
- /dev/alarm virtual device and AlarmManager
- Set next alarm -> wake up

Wake-lock for kernel

- **Problem**
  - User press the power button
  - Users want display and backlight power down

- **Solution**

Wake-lock for kernel

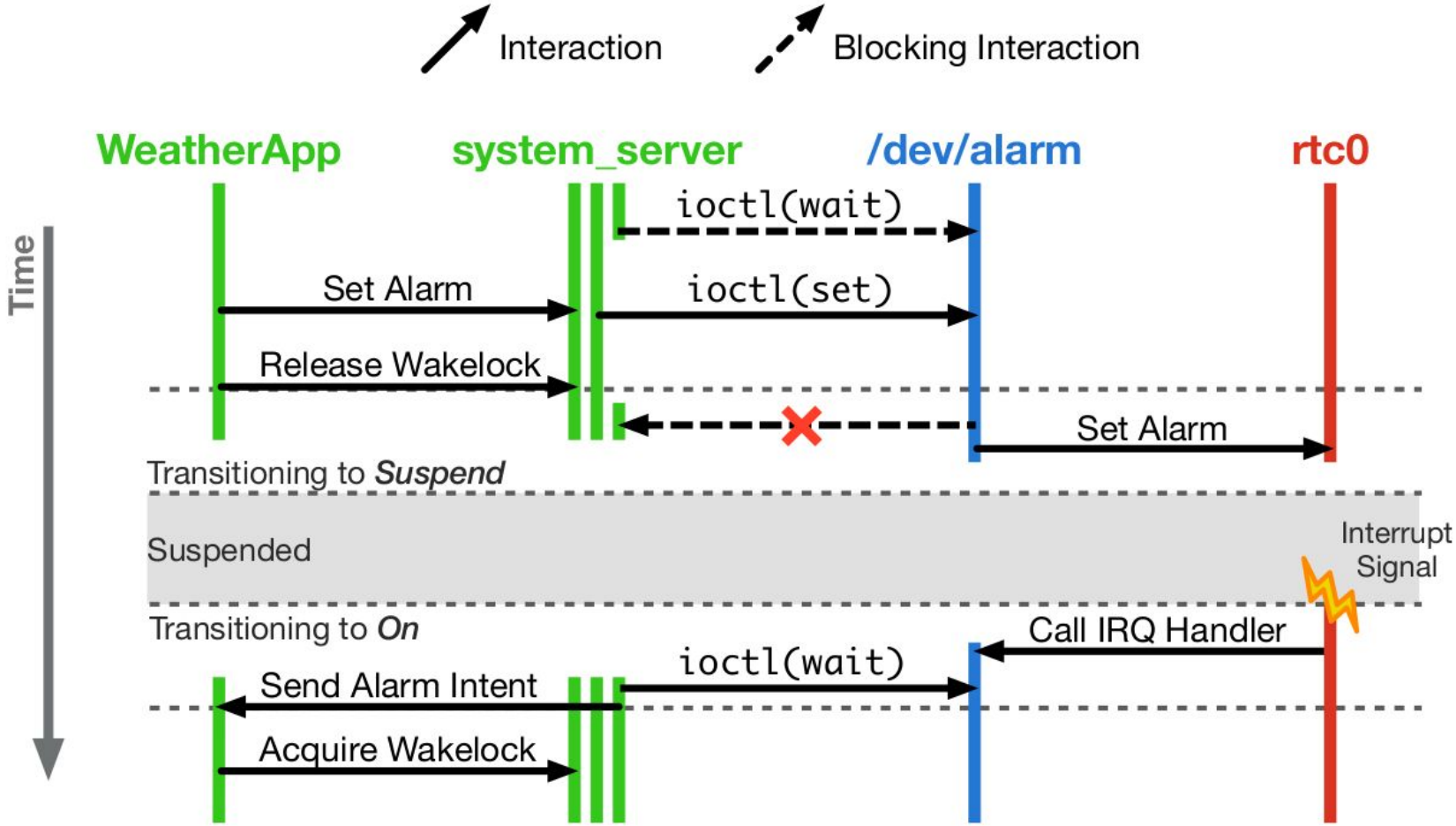
- **Problem**
  - User press the power button
  - Users want display and backlight power down

- **Solution**
  - Early Suspend

Early Suspend

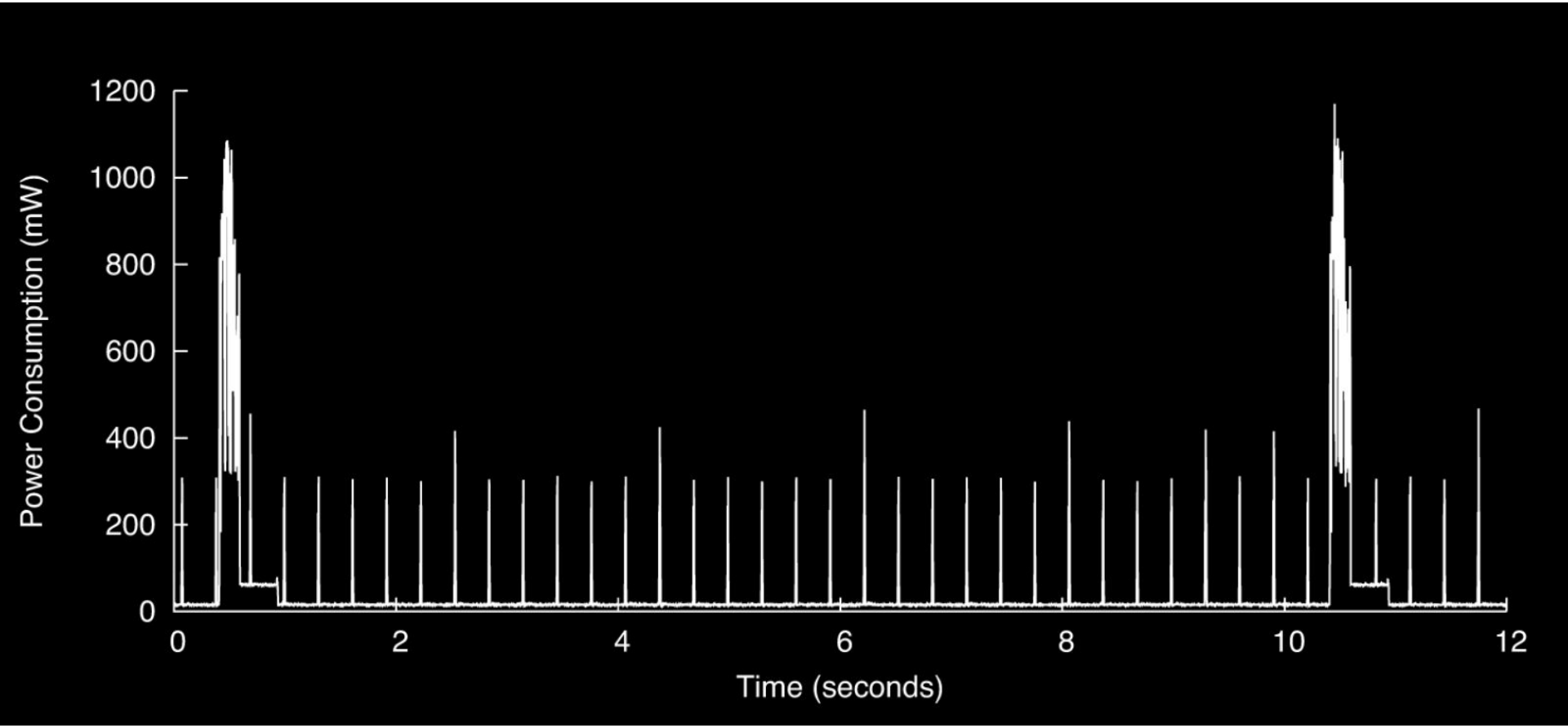
- **Android register early suspend(and late resume)**
- **User presses the power button**
- **The system invokes all early suspend callbacks**
  - E.g., display and backlight power down

Message sequence of system interactions in Android



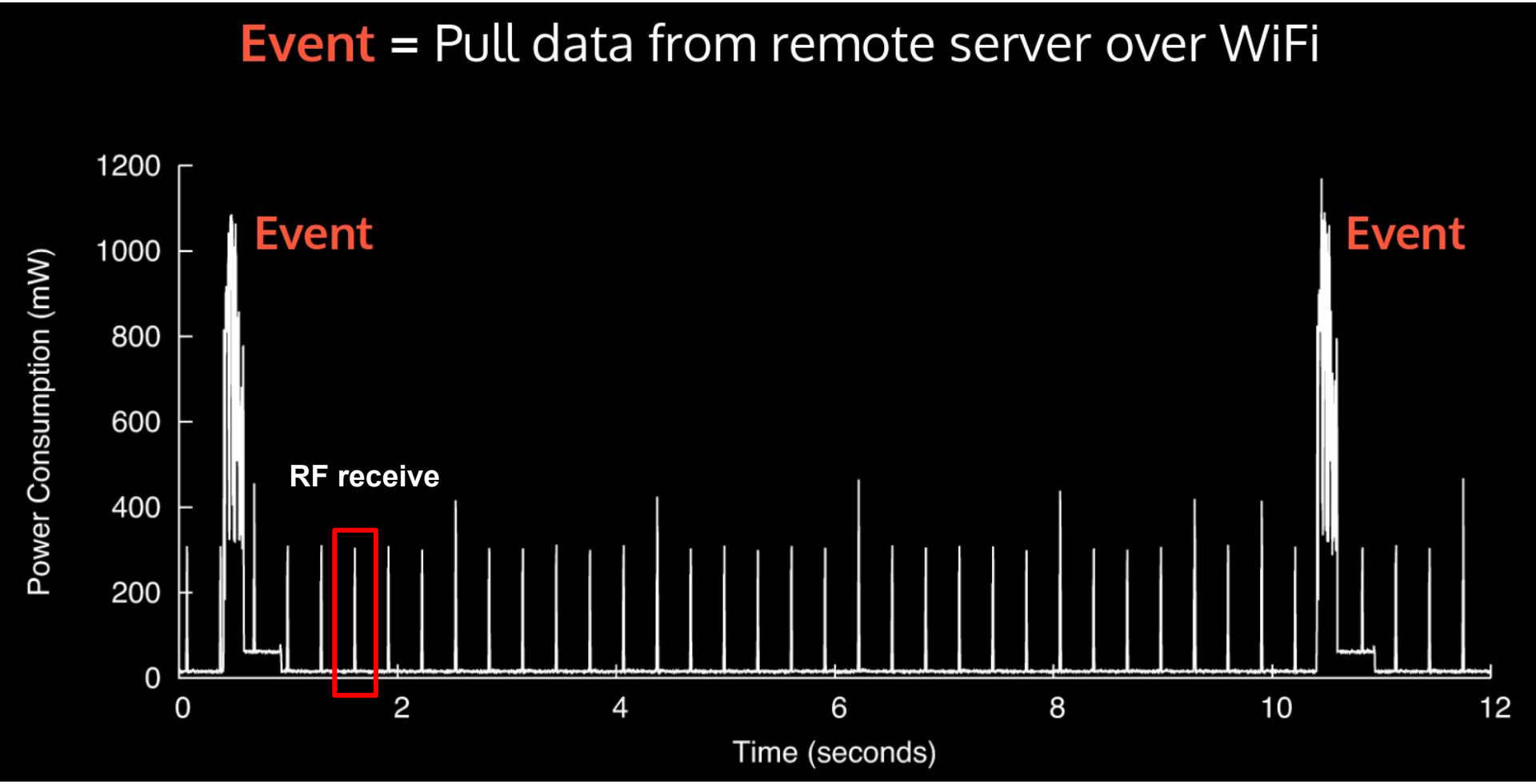
"Drowsy Power Management", Matthew Lentz, SOSP 2015

Android background

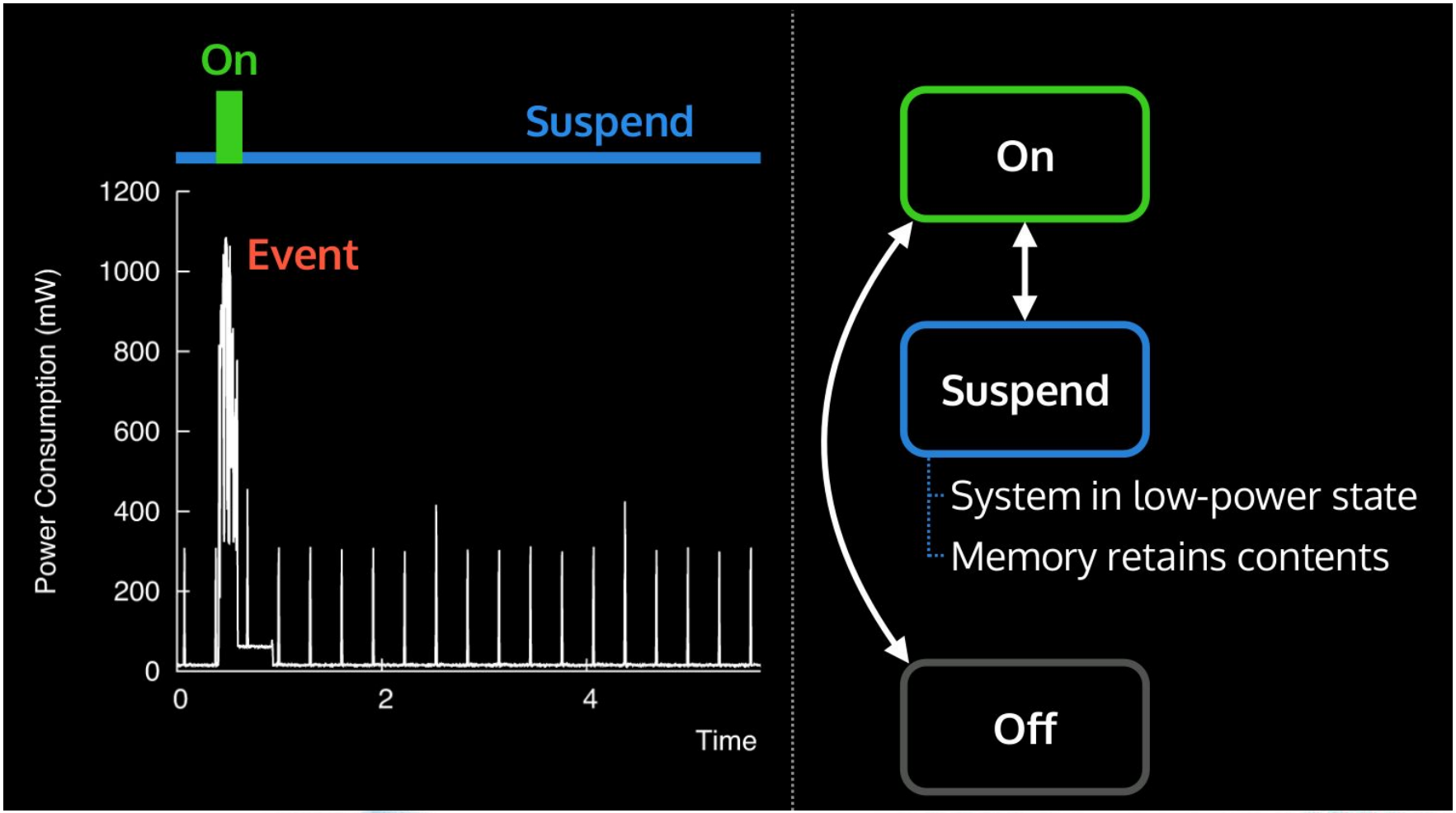


"Drowsy Power Management", Matthew Lentz, SOSP 2015

Background Energy Consumption

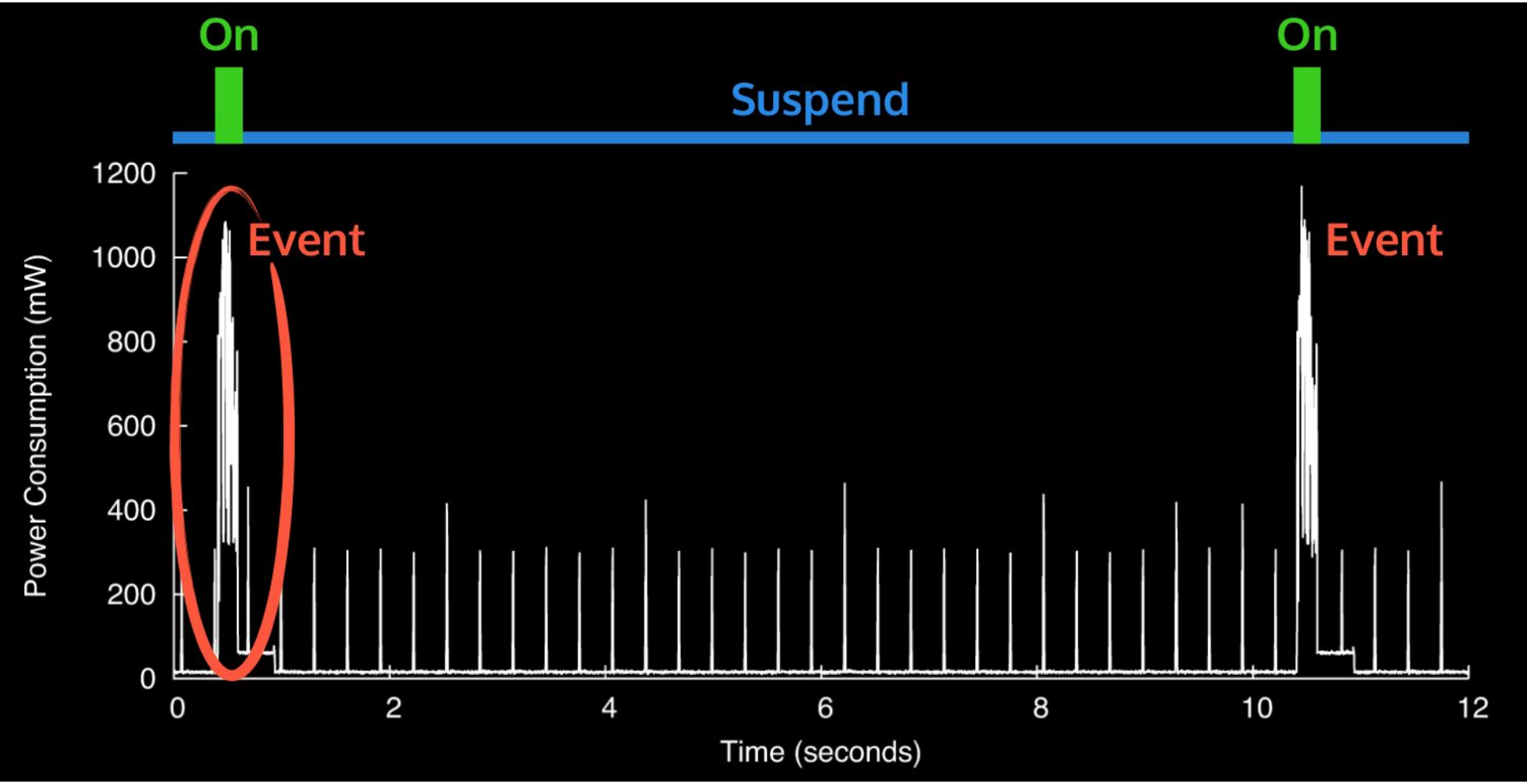


Background Energy Consumption



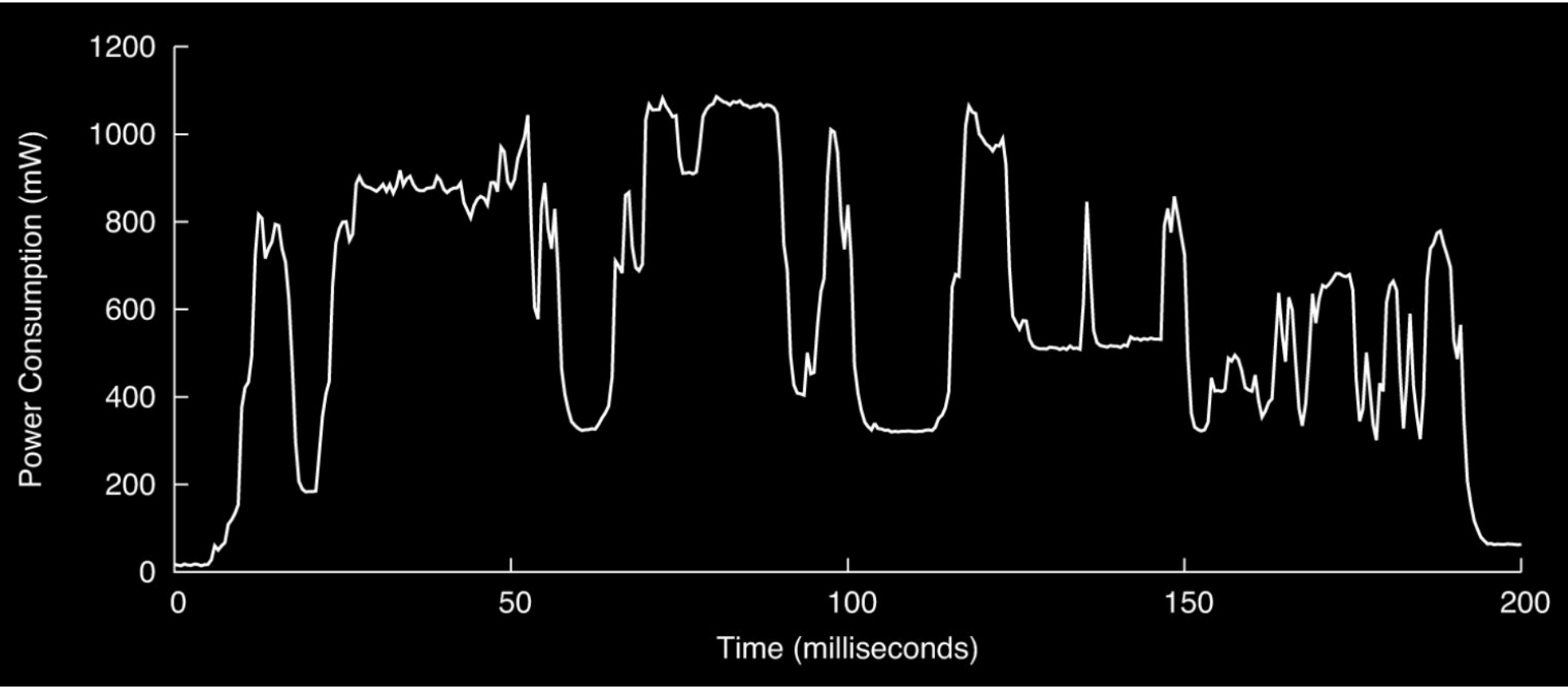
"Drowsy Power Management", Matthew Lentz, SOSP 2015

Background Energy Consumption



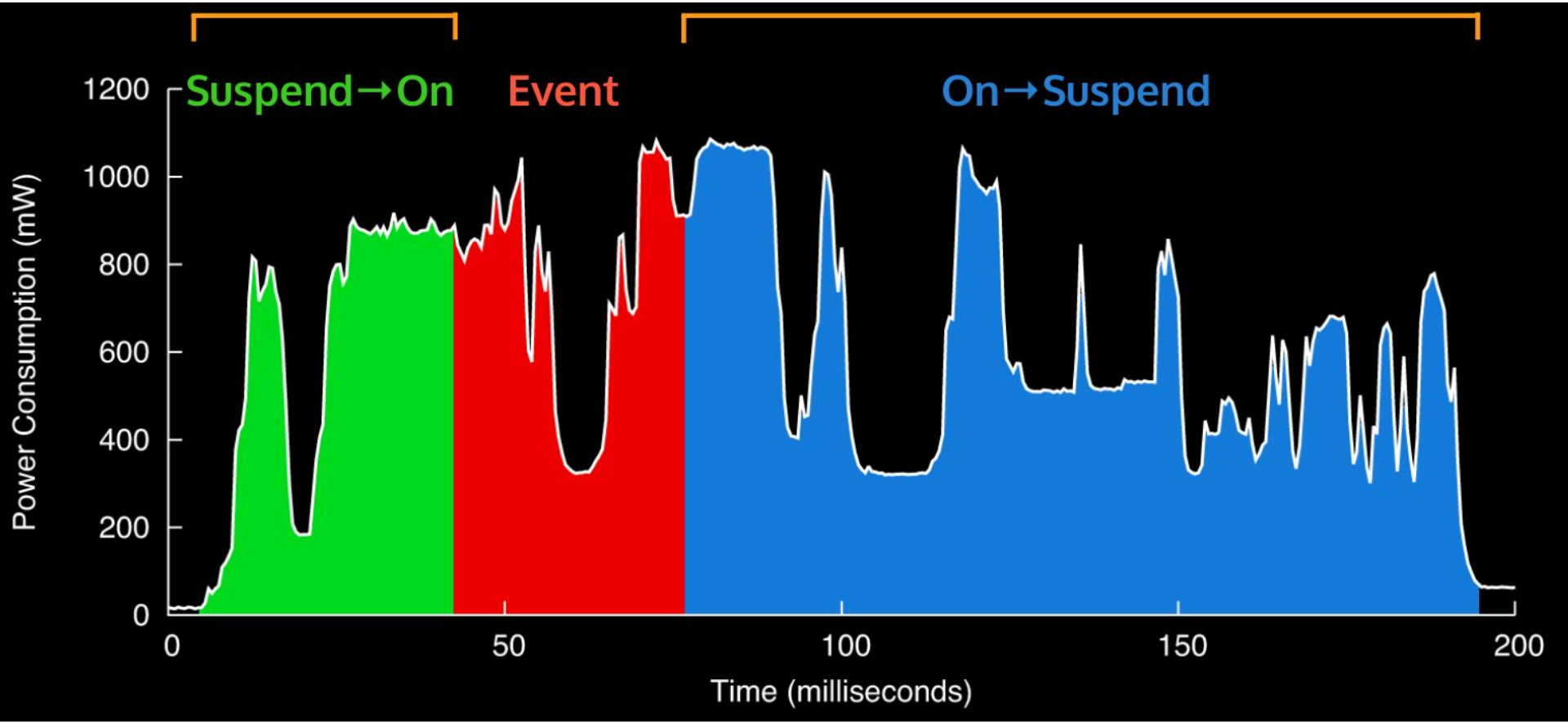
"Drowsy Power Management", Matthew Lentz, SOSP 2015

Event



"Drowsy Power Management", Matthew Lentz, SOSP 2015

Wake-up and Suspend

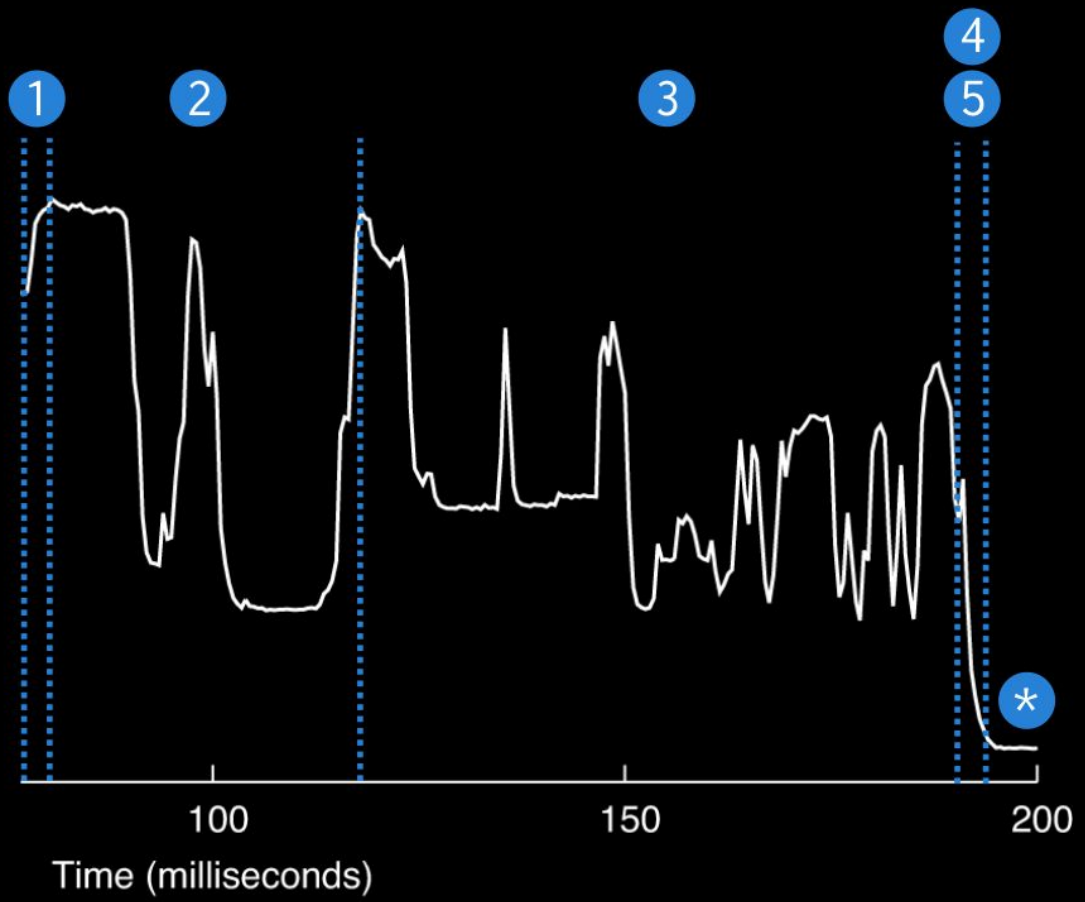


"Drowsy Power Management", Matthew Lentz, SOSP 2015

Background Energy Consumption

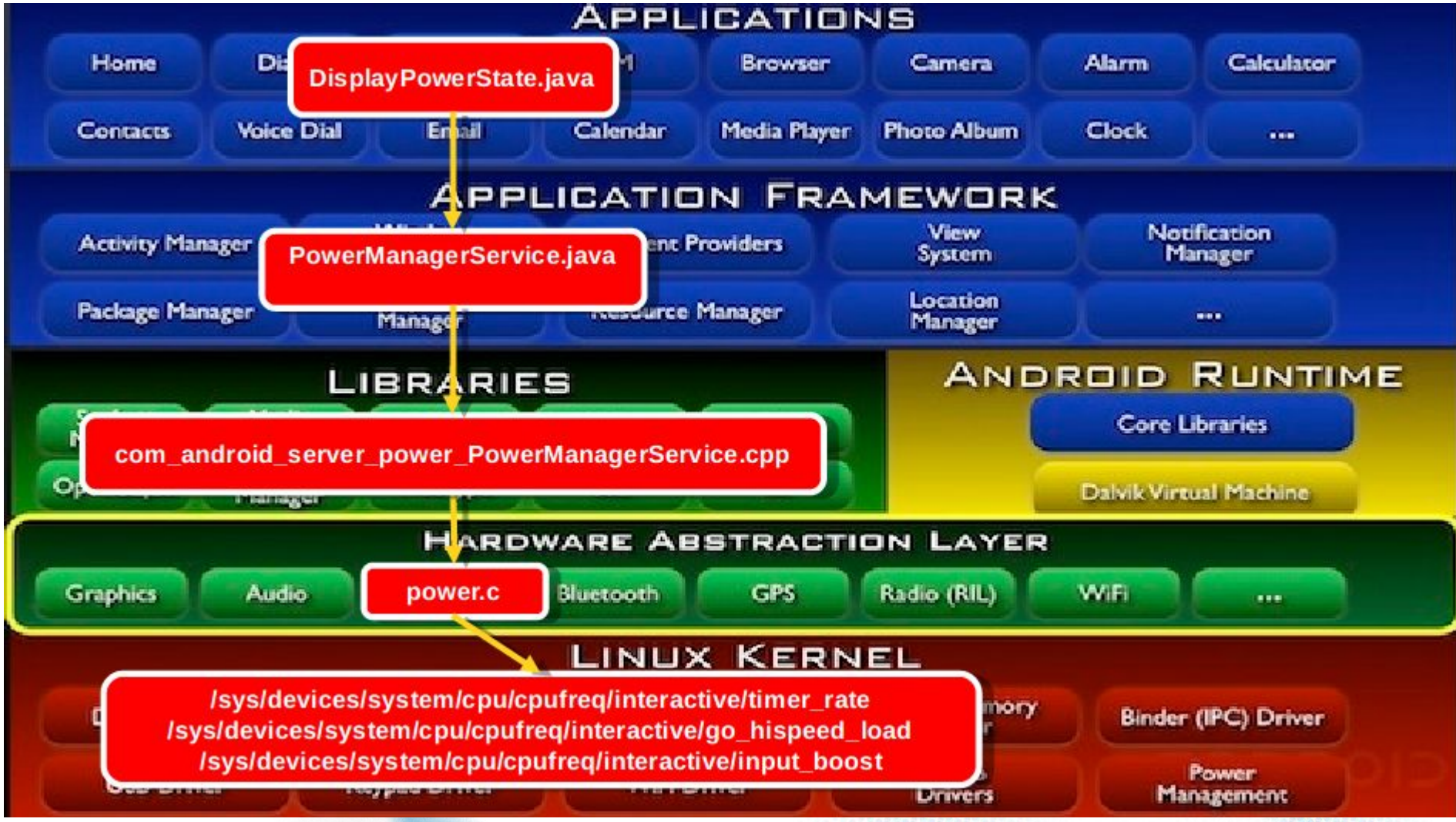
1. Flush filesystem buffers
2. Freeze all tasks
3. Suspend all devices
4. Disable non-boot CPUs
5. Set RAM to self-refresh

* Wait for interrupt



"Drowsy Power Management", Matthew Lentz, SOSP 2015

Android power HAL



https://www.slideshare.net/nanik/learning-aosp-android-hardware-abstraction-layer-hal

Android power HAL

- **A way of providing user's experience information.**
- **Android Power HINT -> kernel**
- **Ex) Touch boost, vsync**

Sample implementations of power HAL
power.$(TARGET_BOARD_PLATFORM).so file can be found (hint: look at the Android.mk's for these as well as the c code):

- hardware/qcom/power
- device/asus/grouper/power
- device/samsung/manta/power
- device/samsung/tuna/power
- device/generic/goldfish/power

Real-time scheduling

Real-time scheduling

- **Determines the order of real-time task executions**

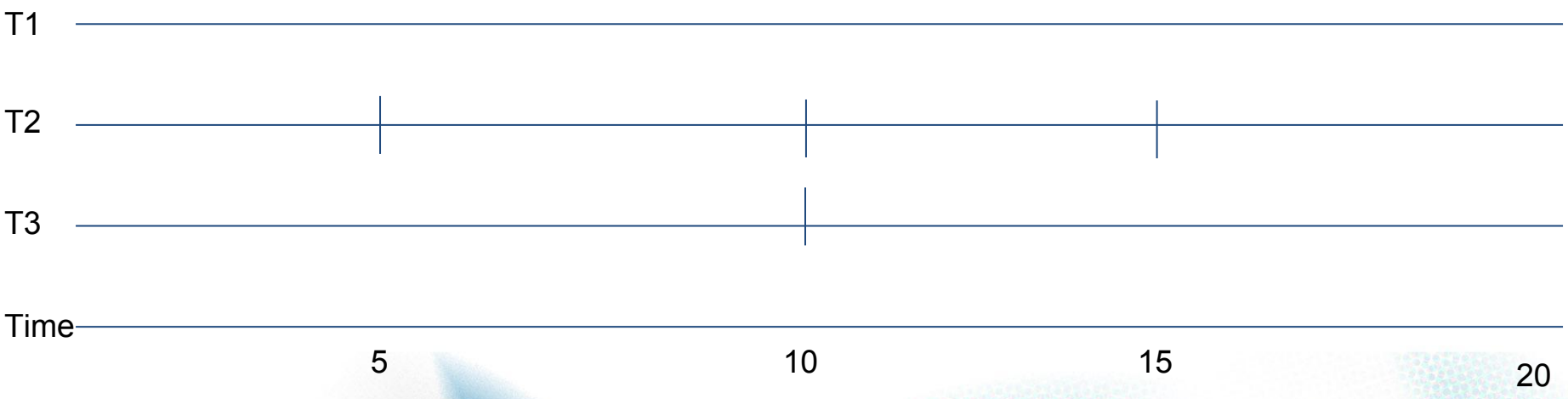
- **Static-priority scheduling**
  - **Rate Monotonic**

- **Dynamic-priority scheduling**
  - EDF

Example : RM

-

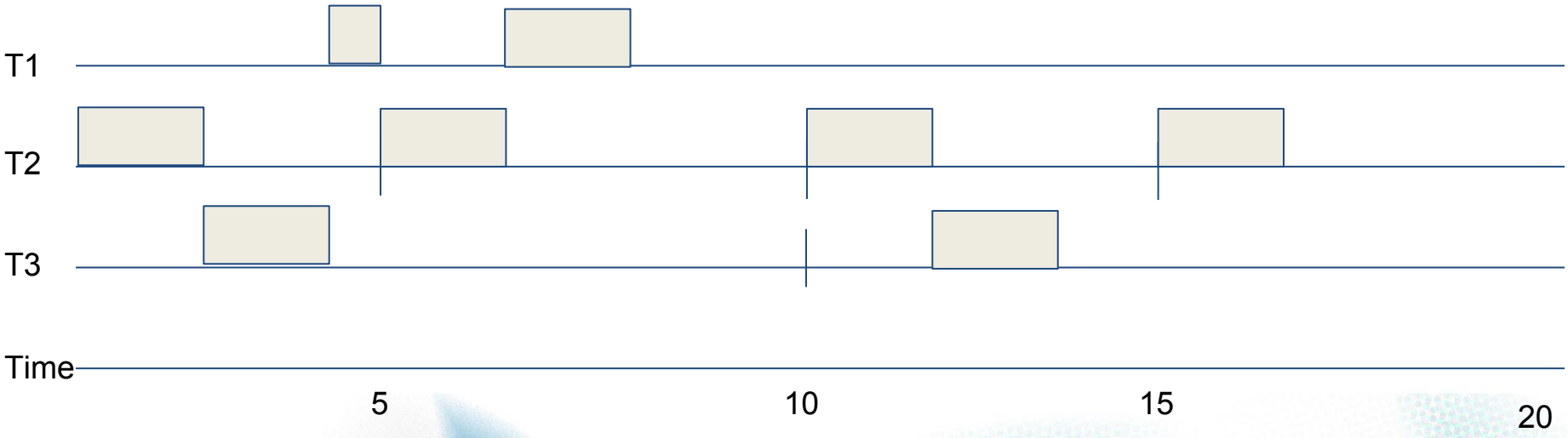
	C	P
T1	3	20
T2	2	5
T3	2	10



Example : RM

-

	C	P
T1	3	20
T2	2	5
T3	2	10



Example : EDF

-

	C	D	P
T1	3	7	20
T2	2	4	5
T3	2	8	10

T1

T2

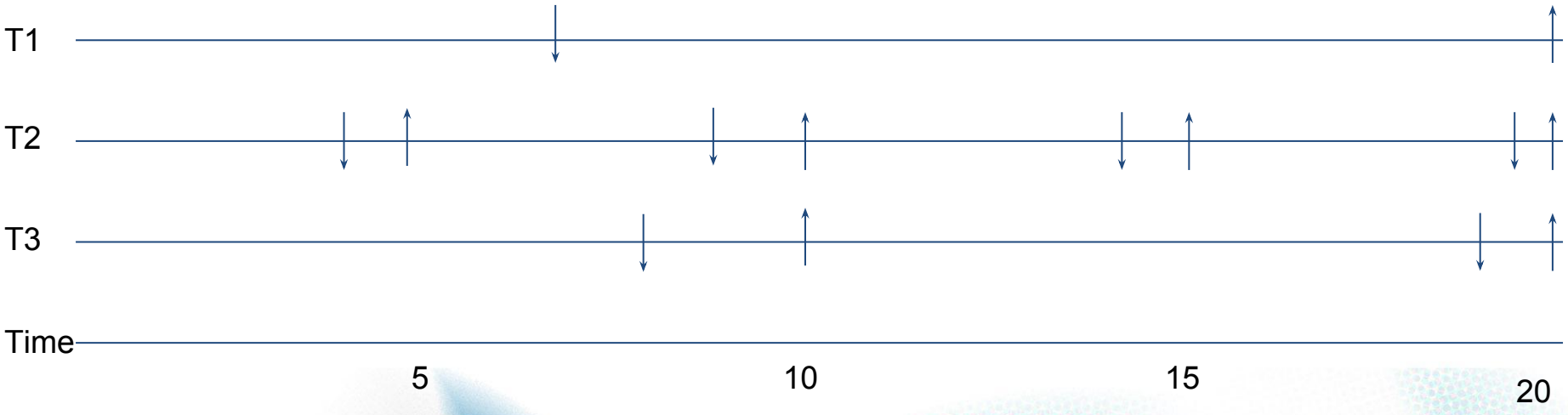
T3

Time

Example : EDF

-

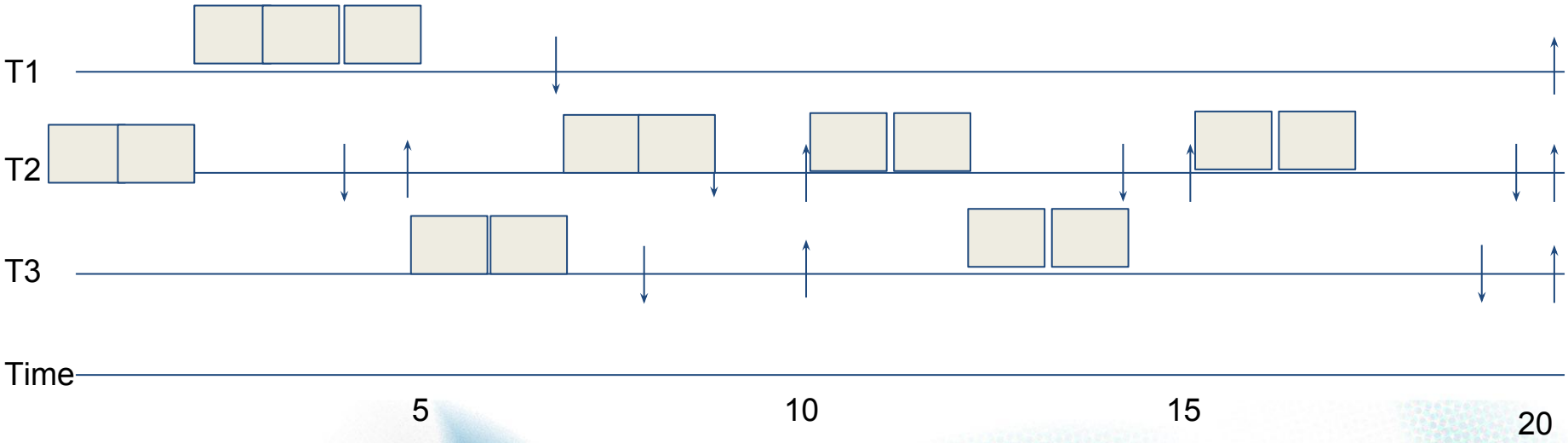
	C	D	P
T1	3	7	20
T2	2	4	5
T3	2	8	10



Example : EDF

-

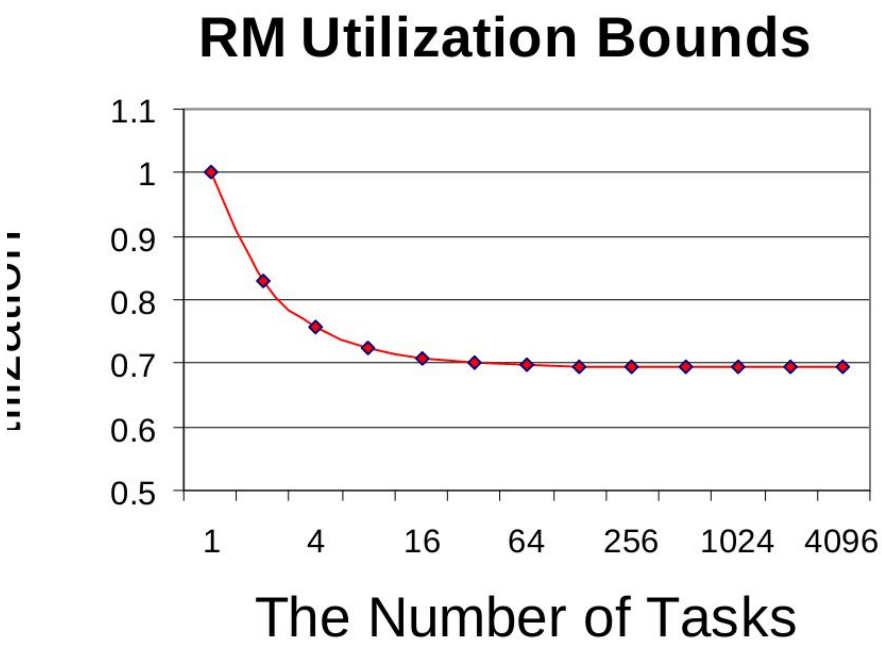
	C	D	P
T1	3	7	20
T2	2	4	5
T3	2	8	10



RM Problem – Utilization Bound

- **n = number of process**

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} \leq n(\sqrt[n]{2} - 1) \quad \rightarrow \quad \lim_{n \to \infty} n(\sqrt[n]{2} - 1) = \ln 2 \approx 0.693147\ldots$$



RM vs. EDF

- **Rate Monotonic**
  - Simpler implementation, even in systems without explicit support for timing constraints (periods, deadlines)
  - Predictability for the highest priority tasks

- **EDF**
  - Full processor utilization
  - Misbehavior during overload conditions

- **For more details: Buttazzo, “Rate monotonic vs. EDF: Judgement Day”, EMSOFT 2003.**

EDF

- **Can EDF be supported in Linux?**

- **Problem**
  - The kernel is not aware of tasks deadline

Can EDF be supported in Linux?

- **Can EDF be supported in Linux?**

- **Problem**
  - The kernel is not aware of tasks deadline

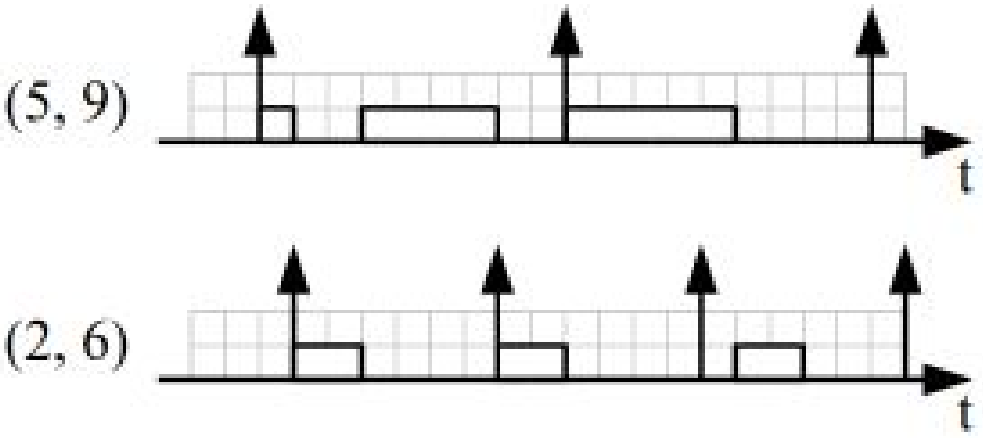
- **Simple Solution**
  - Apps send signal to OS
  - Start()/End()

Linux's sched_attr

```
struct sched_attr {
    __u32 size;

    __u32 sched_policy;
    __u64 sched_flags;

    /* SCHED_NORMAL, SCHED_BATCH */
    __s32 sched_nice;

    /* SCHED_FIFO, SCHED_RR */
    __u32 sched_priority;

    /* SCHED_DEADLINE (nsec) */
    __u64 sched_runtime;
    __u64 sched_deadline;
    __u64 sched_period;
};
```

Example

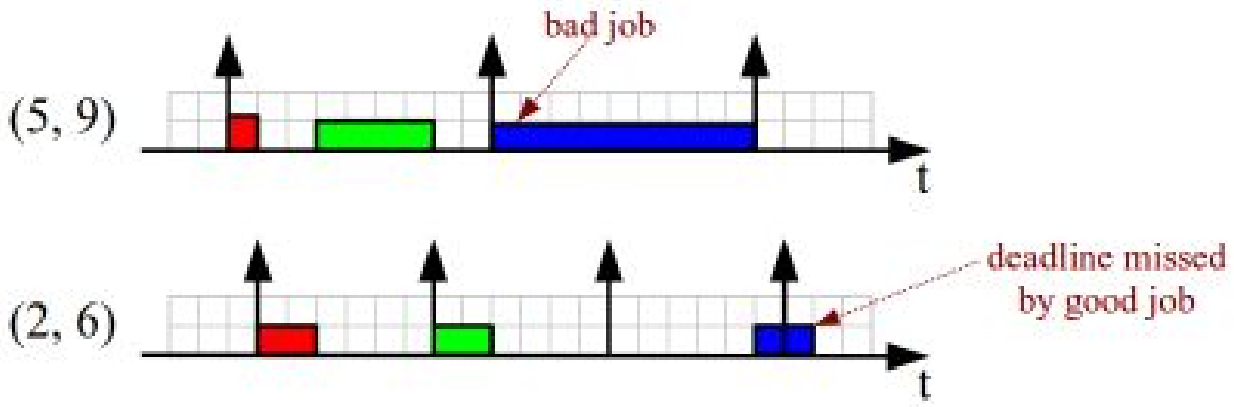
- **Two tasks with reservations**
- **5ms every 9ms and 2ms every 6ms**



~88.9%
Overall
Load

https://lwn.net/Articles/398470/

Real world Problem

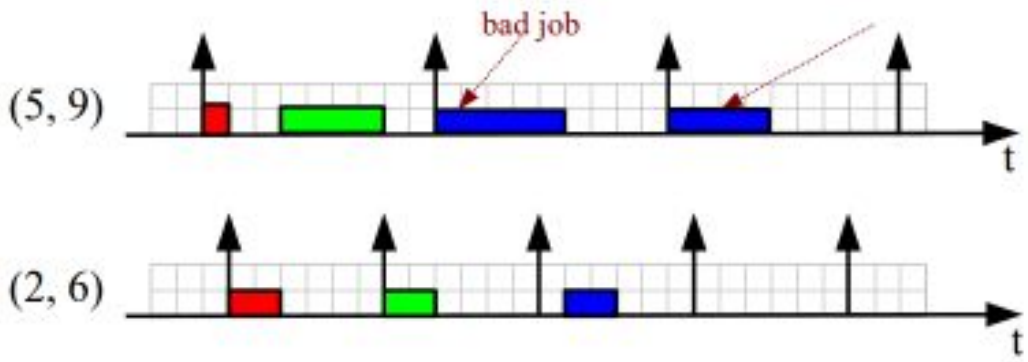


kernel is not aware of tasks deadline

- **Use dynamic scheduling deadlines**
- **Constant Bandwidth Server(CBS)**
  - A resource(CPU) reservation mechanism

Constant Bandwidth Server(CBS)

- **Current budget -> 0**
  Suspended (throttled) till the next activation period
- **Resource(CPU) reservation**



Constant Bandwidth Server(CBS)

- **Problem**
  - properly dimensioning runtime and reservation period
    - **might be too difficult**
  - What if tasks occasionally need more bandwidth?
    - e.g., occasional workload fluctuations (network traffic, rendering of particularly heavy frame, etc)

- **Solution**
  - Use resource over-allocation.
  - Greedy Reclamation of Unused Bandwidth (GRUB)

SCHED_DEADLINE: It's Alive!

- **ARM's presentation.**
  - ELC North America 17, Portland (OR) 02/21/2017
- **Near**
  - Experimenting with Android
  - Reclaiming by demotion towards lower priority class
  - Capacity awareness (for heterogeneous systems)
  - Energy awareness (Energy Aware Scheduling for DEADLINE)
- **And..**
  - Support single CPU affinity
  - Enhanced priority inheritance (M-BWI most probably)
  - Dynamic feedback mechanism (adapt reservation parameters to task' needs)

Reference

- https://pdos.csail.mit.edu/6.828/2016/schedule.html
- http://web.mit.edu/6.033
- http://www.rdrop.com/~paulmck/
- "**Is Parallel Programming Hard, And If So, What Can You Do About It?"**
- Davidlohr Bueso. 2014. Scalability techniques for practical synchronization primitives. *Commun. ACM* 58

http://queue.acm.org/detail.cfm?id=2698990

- **"CPUFreq and The Scheduler Revolution in CPU Power Management", Rafael J. Wysocki**
- **https://sites.google.com/site/embedwiki/oses/linux/pm/pm-qos**
- **https://intl.aliyun.com/forum/read-916**
- **User-level threads : co-routines**

  http://www.gamedevforever.com/291

  https://www.youtube.com/watch?v=YYtzQ355_Co

- **Scheduler Activations**
  - https://cgi.cse.unsw.edu.au/~cs3231/12s1/lectures/SchedulerActivations.pdf
- **https://en.wikipedia.org/wiki/FIFO_(computing_and_electronics)**
- **http://jake.dothome.co.kr/**
- **http://www.linuxjournal.com/magazine/completely-fair-scheduler?page=0,0**
- **https://www2.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/6_CPU_Scheduling.html**
- **"Energy Aware Scheduling", Byungchul Park, LG Electronic**
- **"Update on big.LITTLE scheduling experiments", ARM**
- **"EAS Update" 2015 september ARM**
- **"EAS Overview and Integration Guide", ARM TR**
- **"Drowsy Power Management", Matthew Lentz, SOSP 2015**
- **https://www.slideshare.net/nanik/learning-aosp-android-hardware-abstraction-layer-hal**
- https://www.youtube.com/watch?v=oTGQXqD3CNI
- https://www.youtube.com/watch?v=P80NcKUKpuo
- https://lwn.net/Articles/398470/
- "SCHED_DEADLINE: It's Alive!", ARM, 2017