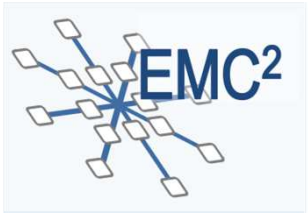


Power Optimizations and Power management of multicores real-time systems

Prof. Avi Mendelson

EE Department, Technion

and consultant for Infineon

avi.mendelson@technion.ac.il

September 30, 2015

Agenda

- Why Power matters?
- Terminology.
- SW / Hardware interfaces
- Run-to-halt vs “minimum operational point”.
- Multicore vs. multi threaded
- Schedule for performance vs. schedule for power.

I will also emphasis:
“*Myth vs. reality*” in power management

Why power matters for embedded systems?

- If battery operated; e.g., your cellular phone
  - Low power helps to extend the operational time
- If connected to utility, or to large battery; e.g., automotive
  Why do we care about low power?
  - We care about thermal
  - The "name of the game changes" → we need to look at "**achieving *maximum performance* per *power envelop***"

In the world of "general purpose core architecture" the emphasis is mainly on achieving maximum performance for power envelop

- Why Thermal is important?
  - When silicon exceed the allowed temperature it start to malfunction (soft-error) and can even melt (permanent-error)

Basic terminology

- Power → the amount of Joules a system/component consume at a certain moment.
- Energy → Power X time

- Power mainly impact thermal, Energy impact battery life.
- People tend to mix these terms and in most of the time when people refer to "power management" they mean "Energy management."
  - To keep the tradition I will keep the same mistake as everyone else ☺
- Thermal is determined by the hottest point of the system
- We also need to take care on power delivery issues → mainly out of the scope of this talk

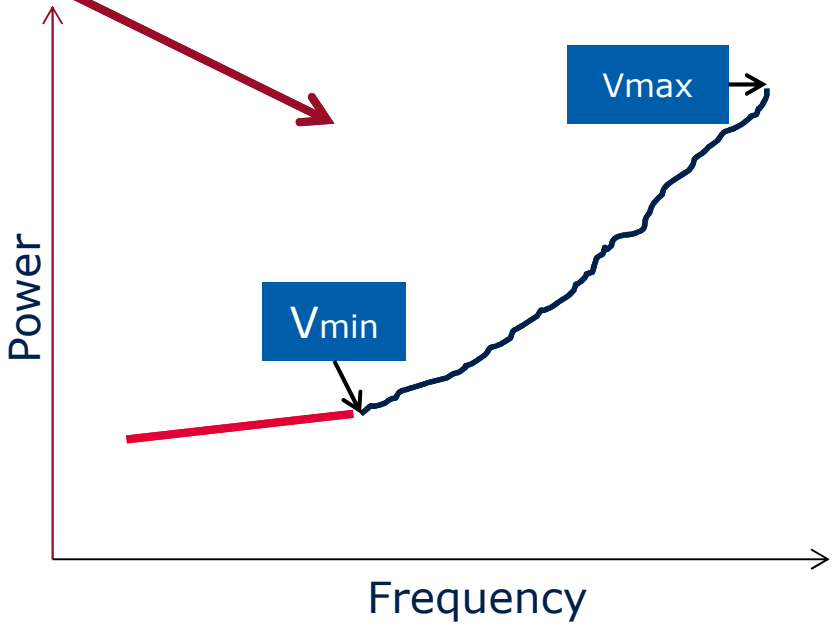
Power vs. Performance – basic equations – theory

- Energy has two components the '*active power*' (when the chip is active) and the "*leakage power*" (when the chip is idle).
- Active power is calculated by
  - Active power: Power = αCV^2f
    (α : activity, C: capacitance, V: voltage, f: frequency)
    Static power is out of the scope of this model.
  - Since voltage and frequency linearly depend on each other (between V_{max} and V_{min}), approximate power change vs. freq. :
    $\Delta Power \sim (\Delta f)^3$
- Leakage in many cases is proportional to the temperature but simplicity many tools consider it to be either constant or proportional to the active power.
- Operational point: for any V between Vmin and Vmax, there is maximum F it can "optimally" run at. The <F,V> is called an operational point. (At lower point we still keep Vmin)

Power vs. Performance – Practice

- Active power is calculated by
  - Active power: Power = αCV^2f
    (α : activity, C: capacitance, V: voltage, f: frequency)
    Static power is out of the scope of this model.

Capacitance depends on activity factor
→ **program behavior**



Implications:

- Assuming performance proportional to frequency,

  below Vmin $P \alpha F$ and above Vmin $P \alpha F^3$
- The operational point of most of MP systems is around Vmin

SW / Hardware interfaces

- In order to allow SW to control Power management, the system is divided in "power plans" each of them
  - Can run at different frequency and with different Voltage. (there are virtual power plans which are differ in frequency but not in voltage. It help to change frequency in a fast manner, but save less power.
- All elements within the same power plan runs at the same frequency and voltage.
- Power plan contains cores, but can also support memories; e.g., LLC can run at one frequency while cores runs at different frequency
- For a paper on optimal way to partition your system into set of power plans, look at *"Rotem , A. Mendelson, R. Genosar and U. Weiser, "Multiple Clock and Voltage Domains for Chip Multi Processors." Micro, 2009, PP 459-468*

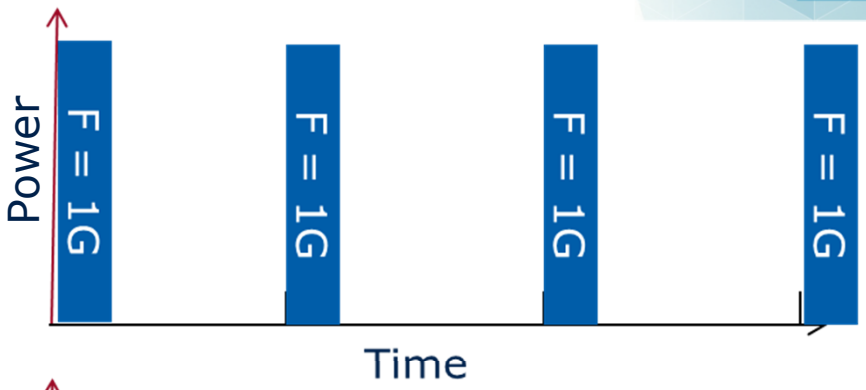
SW/HW interfaces to assist power management

- System defines a set of registers that allow the user to control Power modes. For example set a value to a MSR if you like to change the operational point of a certain power-plan
- There are performance, and recently power registered that allow the user to know how different components utilized and how much power they consume
  - Examples (from Intel)
    - Average IPC (Instructions per cycle)
    - Bus utilization
    - ALU/FPU utilization
    - Us/ALU/FPU/etc. Power consumption
    - Etc.
- There are tools to help analyzing these counters.

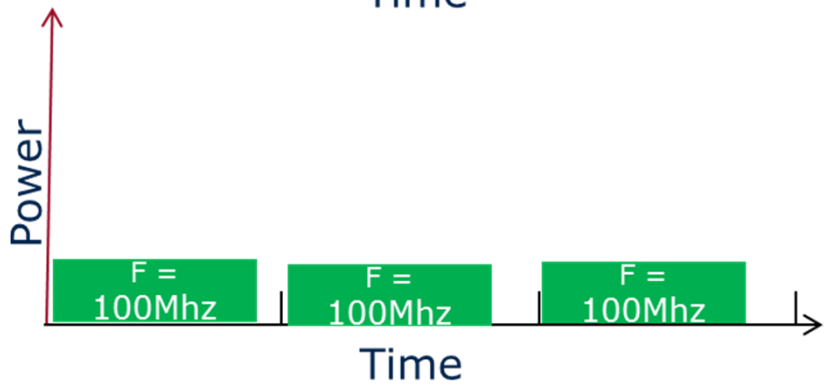
(e.g., Ahmad Yasin. Avi Mendelson, Yosi Ben-Asher: "Deep-dive Analysis of the Data Analytics Workload in CloudSuite", (IISWC, 2014)

Run-to-halt vs "minimum operational point"

- Run-to-Halt



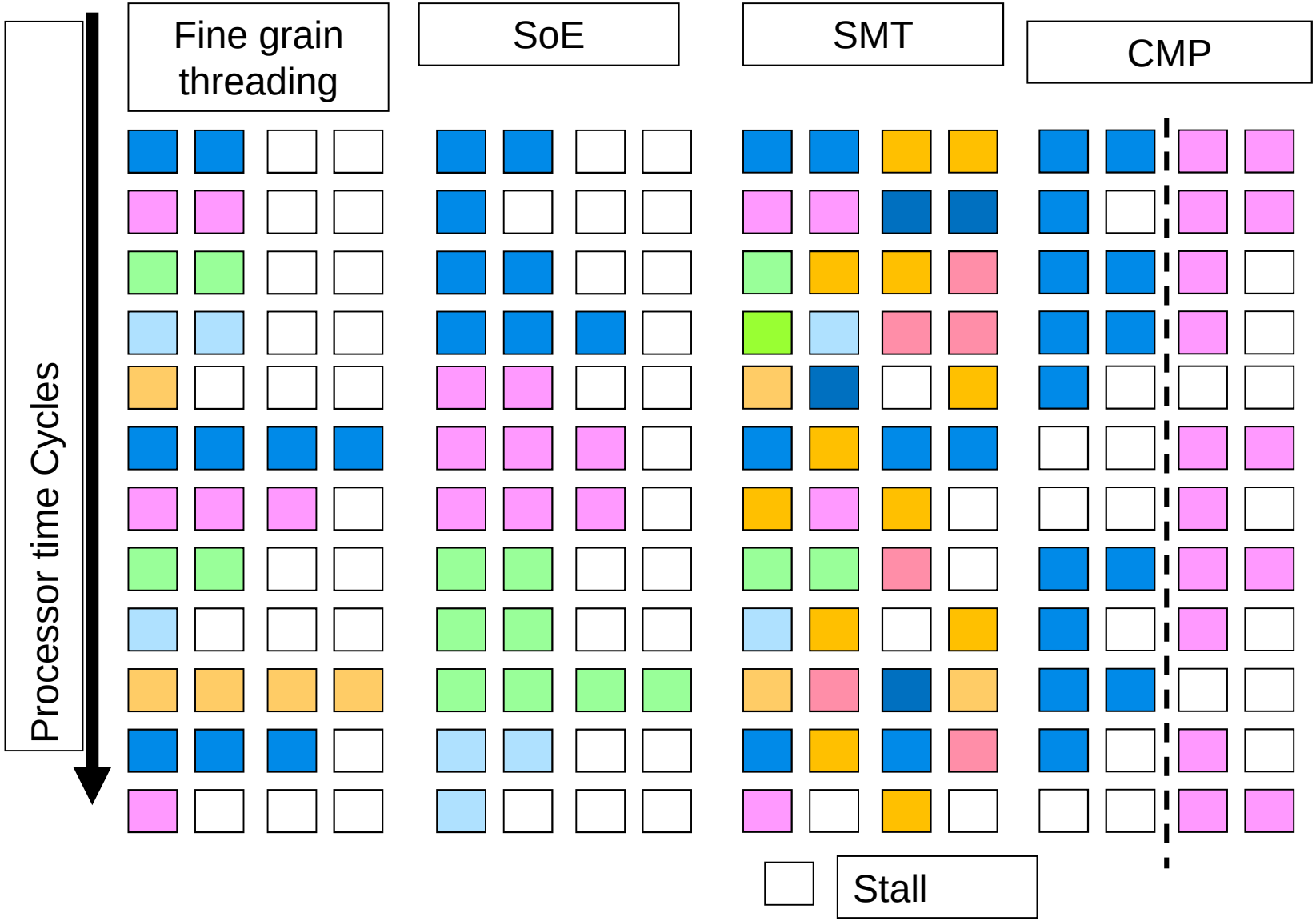
- Minimum Operational point



- What is more power efficient? It depends on
  - Static power vs. dynamic power
  - Program characteristics and utilization at different frequencies
  - Power lost at enter and exit form sleep states

In embedded systems you need to understand SW/HW interfaces in order to answer this "simple" question → you system design impact the SW power management SW algorithms.

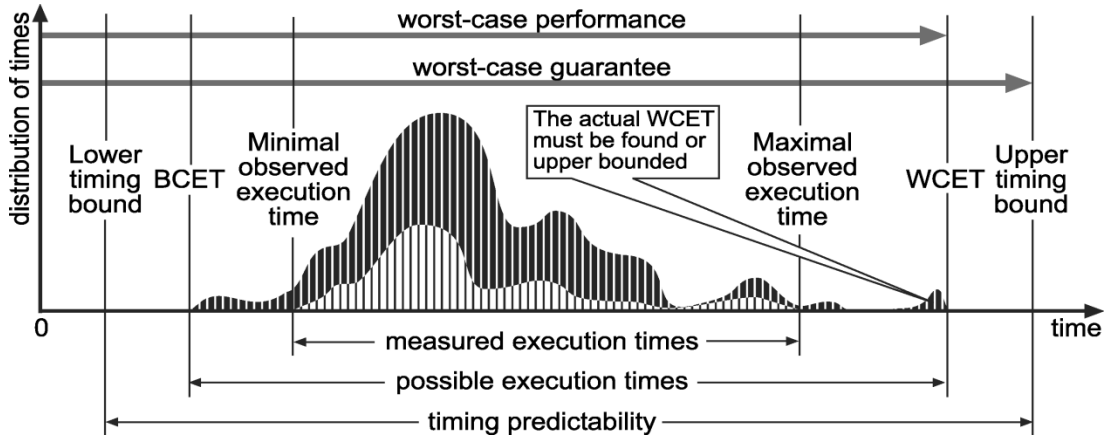
Multi-threaded/multi-tasks Switch-of-events vs. fine-grain vs. CMP

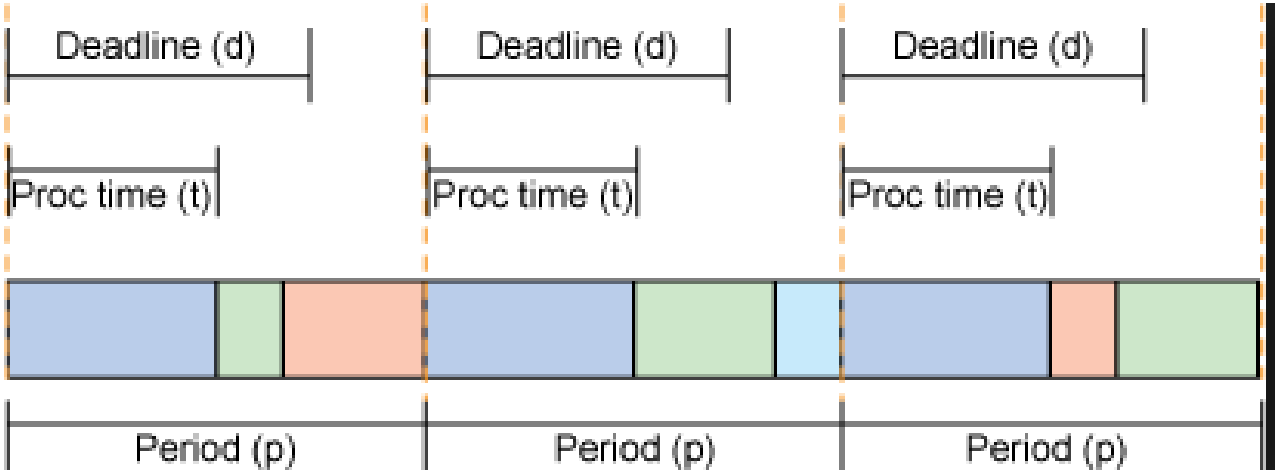


Multi-cores vs. multi-threaded

- Multi-threaded (HW) calls to share HW resources between SW processes/threads
- CMP shares minimum resources and so best fit real-time systems
  - But not power efficient
- SMT systems can be very power efficient but since share resources, may not be fitted for HRT
  - Fine grain can be used for HRT
  - SoE can be used as well, but needs more enhdncements.
- Q: what is better
- Q: can I dynamically switch between them?
- A: depends on new HW/SW interfaces.

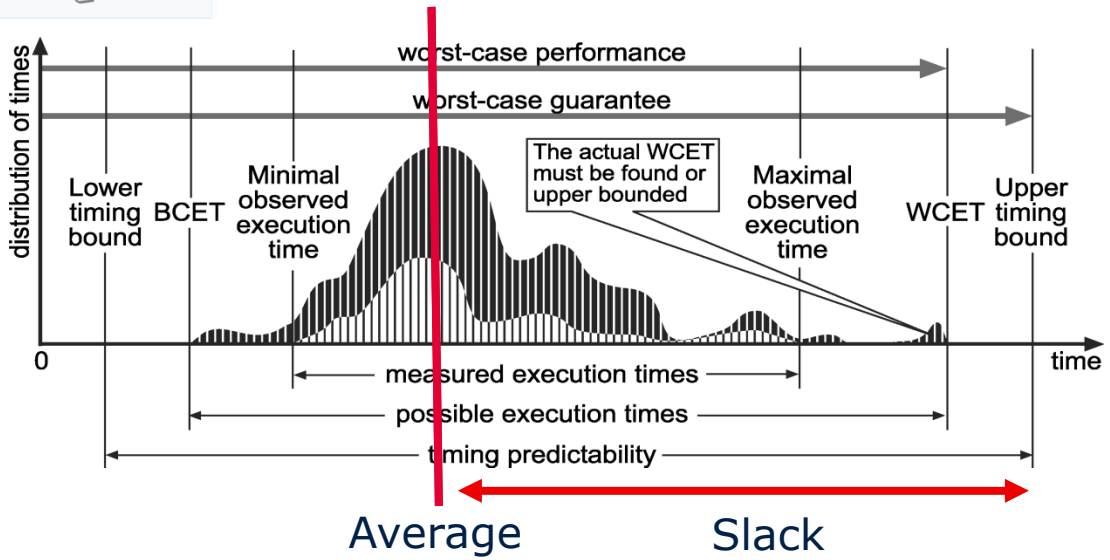
Schedule for HRT - Performance





- When schedule a set of tasks (simple case), we only care about the total sum of their WCET

Schedule for HRT - Power



- When schedule for Power, you like to schedule the one with highest slack first
  - After each task check how much time needed to complete the work of that period. If possible, move to more efficient working point.
- When schedule for multi-processors, the location of where you execute the tasks, may also impact the overall power consumption
- We are working on more advanced techniques.

Conclusions and remarks

- Power is a relatively new area force us to re-evaluate techniques we used in the past
- Usually the question is how to extract maximum performance per a given power envelop and not power saving
- New SW/HW interfaces are needed.
- The key is SW/HW co-design. This is true for general purpose and even more significant for RT and embedded systems
- The traditional way of partitioning the system is not efficient, we need to look at ways to
  - isolate the subsystems in order to guarantee the WCET
  - allowing to share resources
    → we are working on that