# Real-Time Systems & Fault Tolerance

Flávia Maristela

Instituto Federal da Bahia
Especialização em Computação Distribuída e Ubíqua (ECDU)

Salvador, Outubro de 2013

# Schedule

## Computers in the past

## Computers in the past

- In the past 50 years designers believed that computers had to be "fast, small and cheap"
- Microprocessor design used to focus on:
  1. improving throughput
  2. decreasing chips area
- Recently, "lower power" was added ⇒ significantly complicated the whole picture
  1. portable personal computing
  2. communication devices
  3. reliability issues
  4. ubiquitous and pervasive applications

## Mobile Devices and Energy

- The advent of mobile computing motivated the focus to be changed to providing more power conscious solutions.
- Today, many real-time systems are battery-operated embedded devices $\Rightarrow$ restricted energy budgets.

## Power Management: Why and What?

## Power Management: Why and What?

- Battery operated: most mobile systems as Laptops, PDAs and Cell phones
- Heating: for complex Servers as the ones based on multiprocessors
- Power Aware: maintain QoS, reduce energy

## Power Management: Why and What?

The peak computing rate needed is much higher than the average throughput that must be sustained

High performance is needed only for a small fraction of time, while for the rest of time, a low-power processor would suffice.

## Power Management: How?

- Power off un-used parts: LCD, disk for Laptop
- Gracefully reduce the performance
    - CPU: dynamic power $P_d = C_{ef} V_{dd}^2 f$ [Chandrakasan-1992, Burd-1995]
        - $C_{ef}$: switch capacitance
        - $V_{dd}$: supply voltage
        - $f$: processor frequency $\Rightarrow$ linear to $V_{dd}$

## Energy Model

- Common Processors: static power + dynamic power

## Energy Model

- Common Processors: static power + dynamic power

## Energy Model

- Power consumed by microprocessors can be stated as follows:

$$P \propto C_{ef} V^2 f, \tag{1}$$

where $\mathbf{C}_{ef}$ is the switch capacitance;
$\mathbf{V}$ is the supply voltage; and
$f$ is the maximum frequency.

## Energy Model

- Most used techniques for providing power reduction:
  - Dynamic Frequency Scaling (DFS)
  - Dynamic Voltage Scaling (DVS)
  - Dynamic Power Management (DPM)

- Most used techniques for providing power reduction:
  - Dynamic Frequency Scaling (DFS)
  - Dynamic Voltage Scaling (DVS)
  - Dynamic Power Management (DPM)
  - Dynamic Power and Frequency Scaling (DVFS)

## Frequency Scaling

saves energy by reducing frequency without changing supply voltage

- At maximum frequency $f_{max}$, energy consumption is:

$$E = C_{ef} V_{max}^2 f_{max} \tag{2}$$

- We assume $f_{max} = 1$
- At frequency $f < f_{max}$, the power dissipation is

$$P = C_{ef} V_{max}^2 f \tag{3}$$

## Voltage Scaling

reduces the supply voltage for lower frequencies to save energy

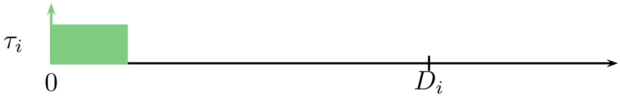
- Standard technique for managing the power consumption.
- Several energy saving approaches are based on Dynamic Voltage Scaling - DVS
- DVS can reduce power consumption at least quadratically at the expense of linearly increasing delay (reducing speed).

Considers that frequency and voltage almost always varies proportionally

- Assume that the supply voltage $V_{max} = 1$.
- Thus, for frequency $f$, the supply voltage is given by $V = V_{max}.f$
- At voltage $V < V_{max}$, the power dissipation is

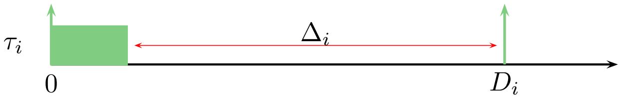
$$P = C_{ef} V^2 f \tag{4}$$

- For real-time systems, scaling down processing frequency (speed) may cause a deadline miss

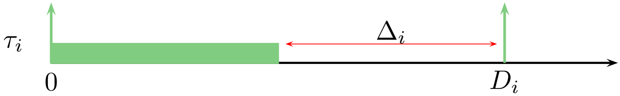



- Task executed with lower supply voltage

- The idea is to use slack $\Delta$ to save energy




- Task executed with lower voltage supply



- Basically, there are two kinds of slack: *static* and *dynamic*
  - Static: the difference between deadline and execution cost of tasks
  - Dynamic: results at runtime when a task consumes less than its worst-case execution time
  - Recent studies[1] show that DVFS has a direct impact on transient fault rates.

[1] The effects of Energy Management on Reliability in Real-Time Embedded Systems

- The idea to deal with transient faults and power consumption is very simple:
  - Exploit the available slack to schedule a recovery task
  - Use the “remaining”slack for power management
- This approach has been called *Reliability Aware Power Management* and has been extensively used explored for uniprocessor systems.

- Energy consumption and fault occurrence are co-related:
  - As the energy consumption increases so does the temperature and hence the frequency (likelihood) of fault occurrence.
  - An increment in fault occurrence causes re-execution of the task (or the execution of an alternative version) leading to higher energy consumption.
  - Also, in order to achieve energy minimization, less redundancy (time or space) should be deployed (as it add to the overheads).
  - However, most of the time, the higher the redundancy the better is the fault tolerance of the system.
- Thus, there is a trade-off which should be accurately tuned to provide better performance both in terms of energy and fault.

## Frequency Scaling and Fault Occurrence

- When supply voltage is fixed, the fault rate in circuits decrease linearly when frequency reduces [2]
- For frequency scaling with fixed supply voltage the average fault rate can be modeled as:

$$\lambda(f,V) = \lambda(f) = \lambda_0 f^b \tag{5}$$

where $\lambda_0$ is the average fault rate and $b(>0)$ is a constant. For $b=1$ the fault rate is linearly increasing with the frequency

[2] Modeling the effects of technology trends on the soft error rate of combinational logic

## Voltage Scaling and Fault Occurrence

- For technologies which have different supply voltages, the fault rate increases exponentially when supply voltage decreases[3]
- For frequency $f$ and voltage $V = fV_{max} = f$, the average fault rate is:

$$\lambda(f, V) = \lambda(f) = \lambda_0 10^{\frac{d(1-f)}{1-f_{min}}} \tag{6}$$

[3] Trends in electronic reliability: effects of terrestrial cosmic rays