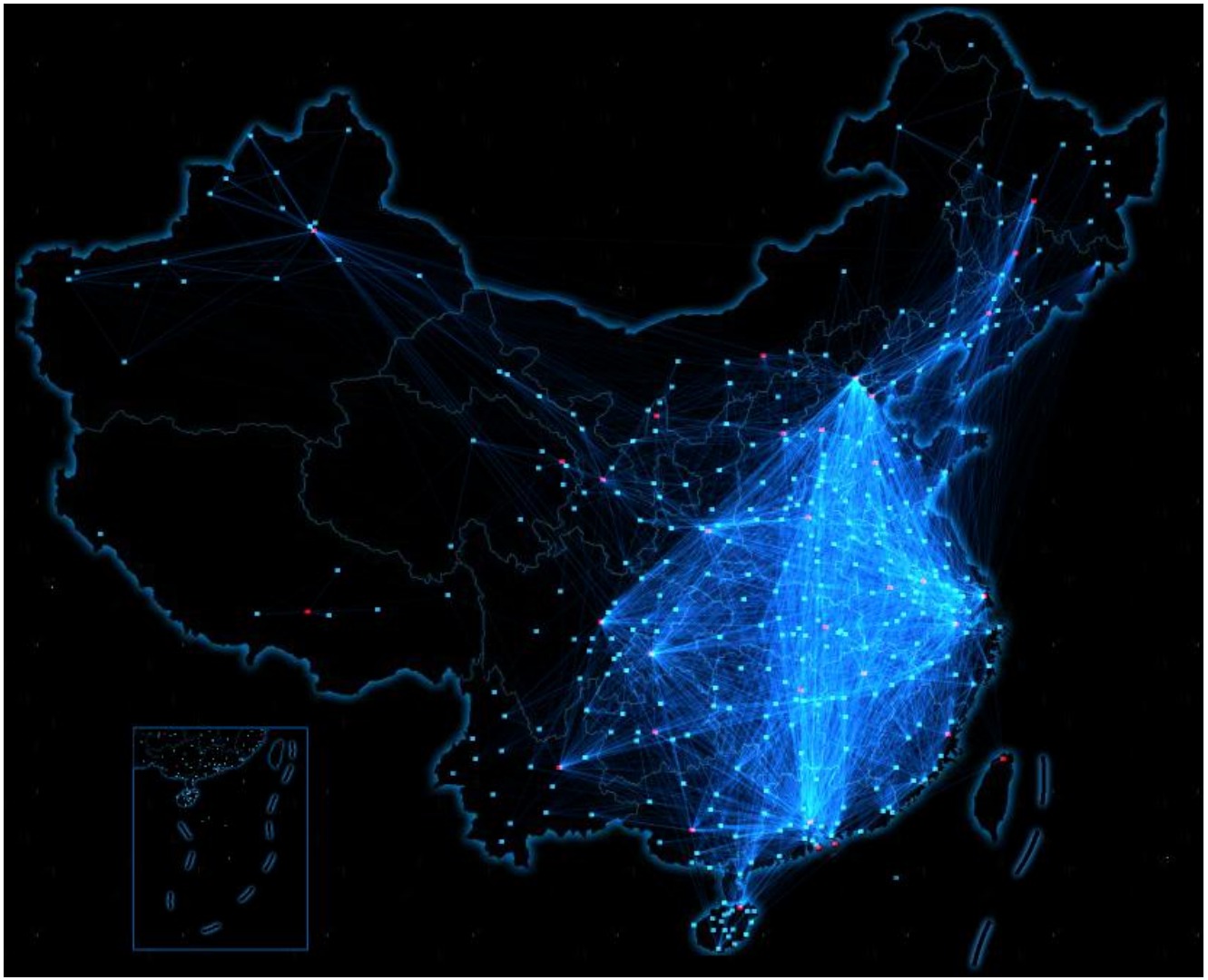
Analysis (III)
Low Power Design

Kai Huang

Chinese new year: 1.3 billion urban exodus

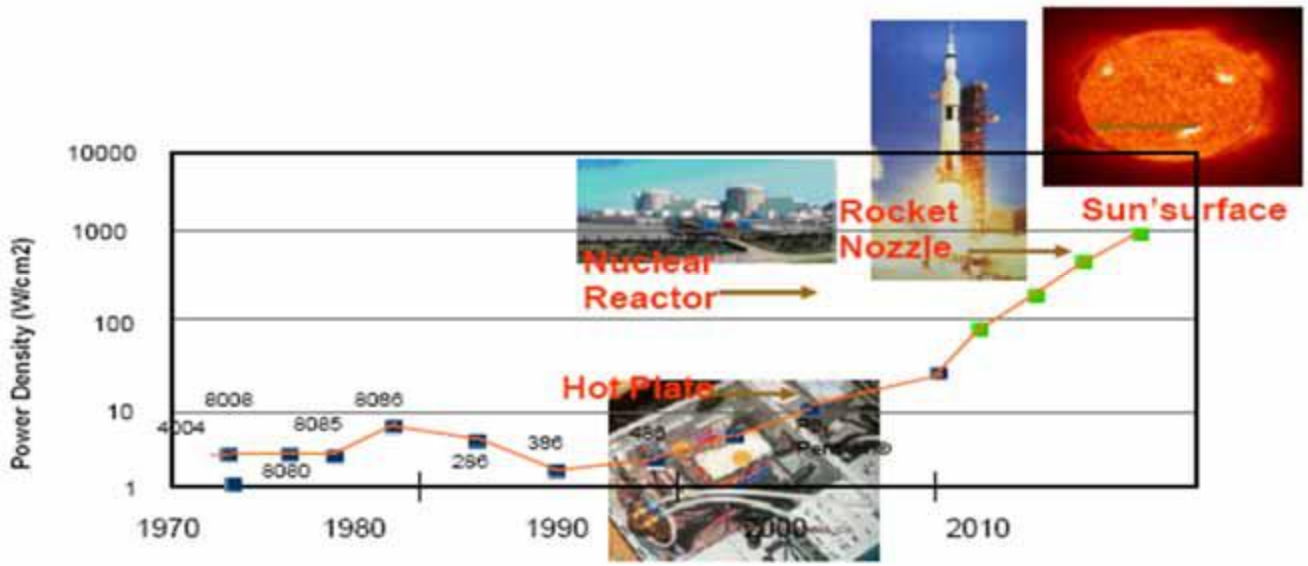


- The interactive map, which is updated hourly
- The thicker, brighter lines are the busiest routes.
- Current view 28.01.2014 9am by Baidu

Outline

- General Remarks
- Power and Energy
- Basic Techniques
  - Parallelism
  - VLIW (parallelism and reduced overhead)
  - Dynamic Voltage Scaling
  - Dynamic Power Management

Power and Energy Consumption



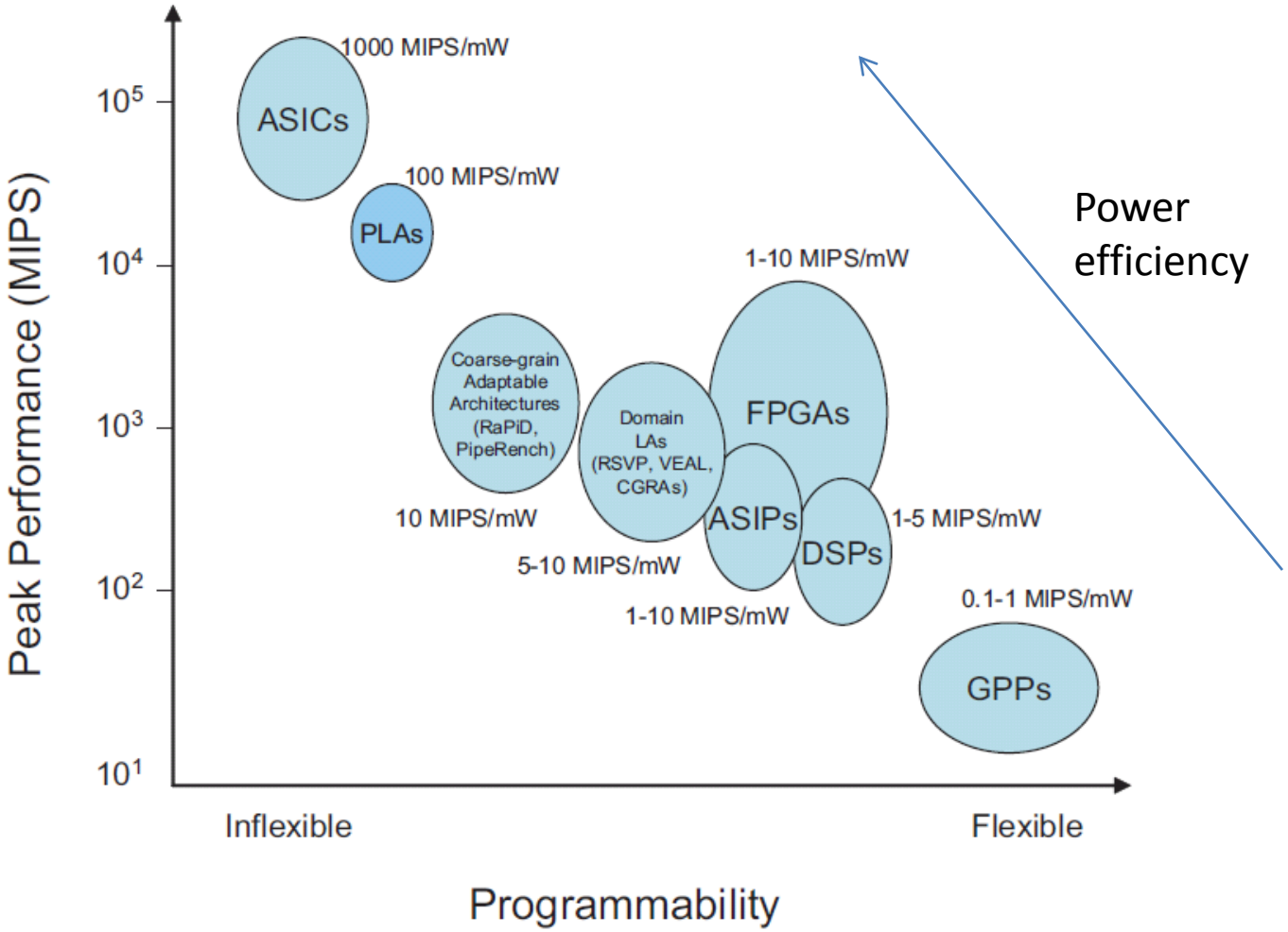
"Power is considered as the most important constraint in embedded systems."

[in: L. Eggermont (ed): Embedded Systems Roadmap 2002, STW]

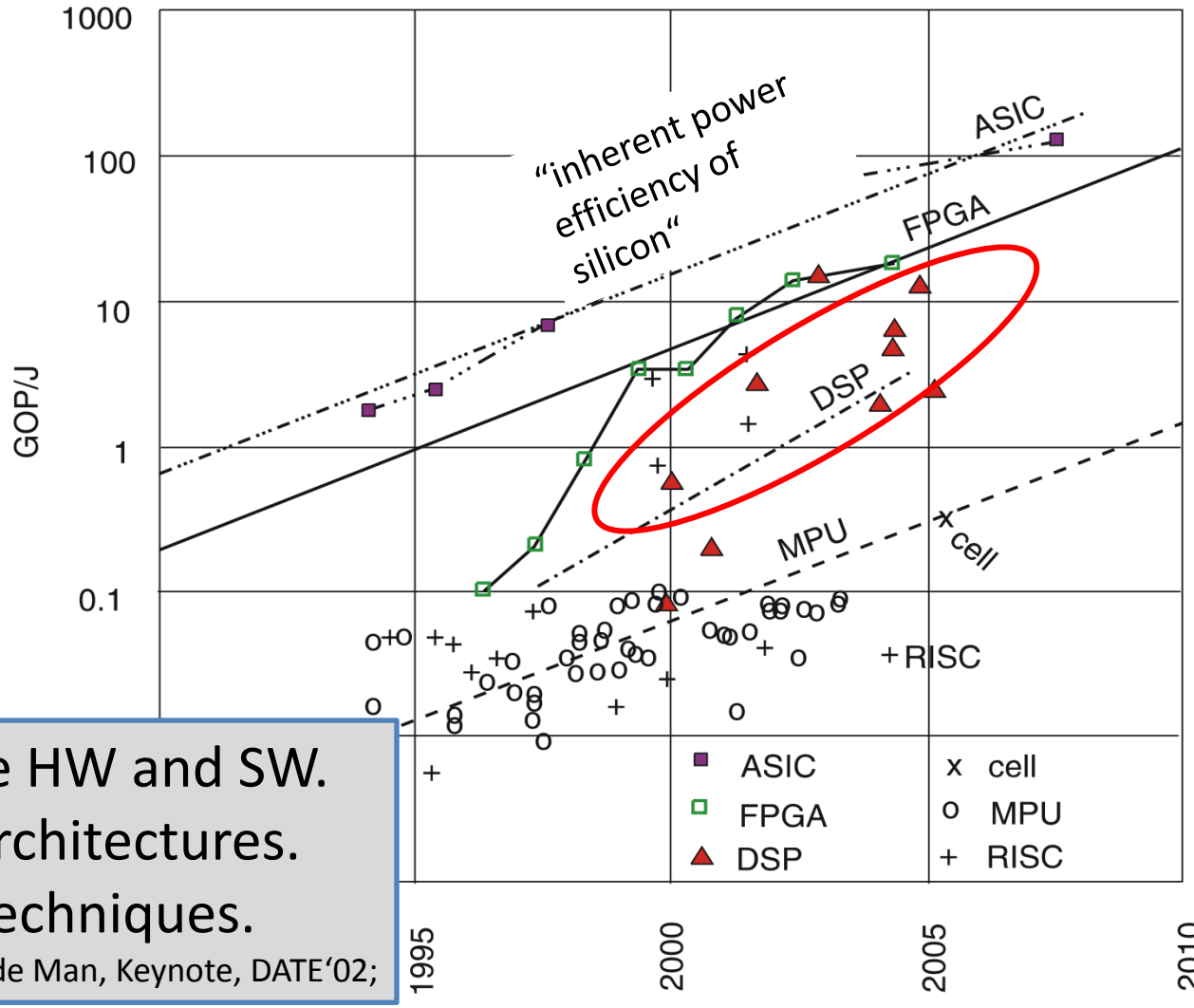
"Power demands are increasing rapidly, yet battery capacity cannot keep up."

[in Diztel et al.: Power-Aware Architecting for data-dominated applications, 2007, Springer]

Implementation Alternatives



Energy Efficiency



© Hugo De Man, IMEC, Philips, 2007

- Necessary to optimize HW and SW.
- Use heterogeneous architectures.
- Apply specialization techniques.

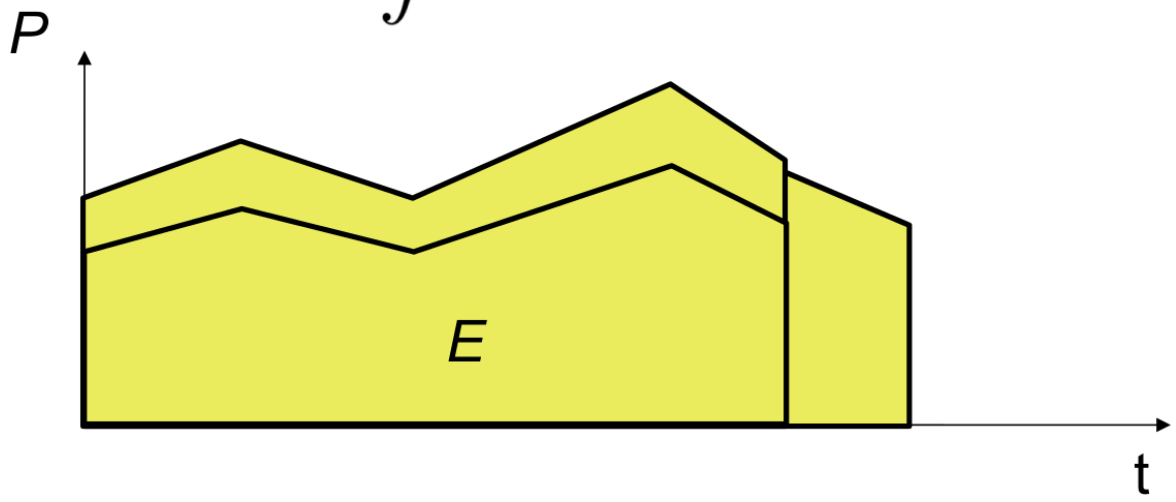
H. de Man, Keynote, DATE'02;

Outline

- General Remarks
- Power and Energy
- Basic Techniques
  - Parallelism
  - VLIW (parallelism and reduced overhead)
  - Dynamic Voltage Scaling
  - Dynamic Power Management

Power and Energy are Related

$$E = \int P(t)dt$$

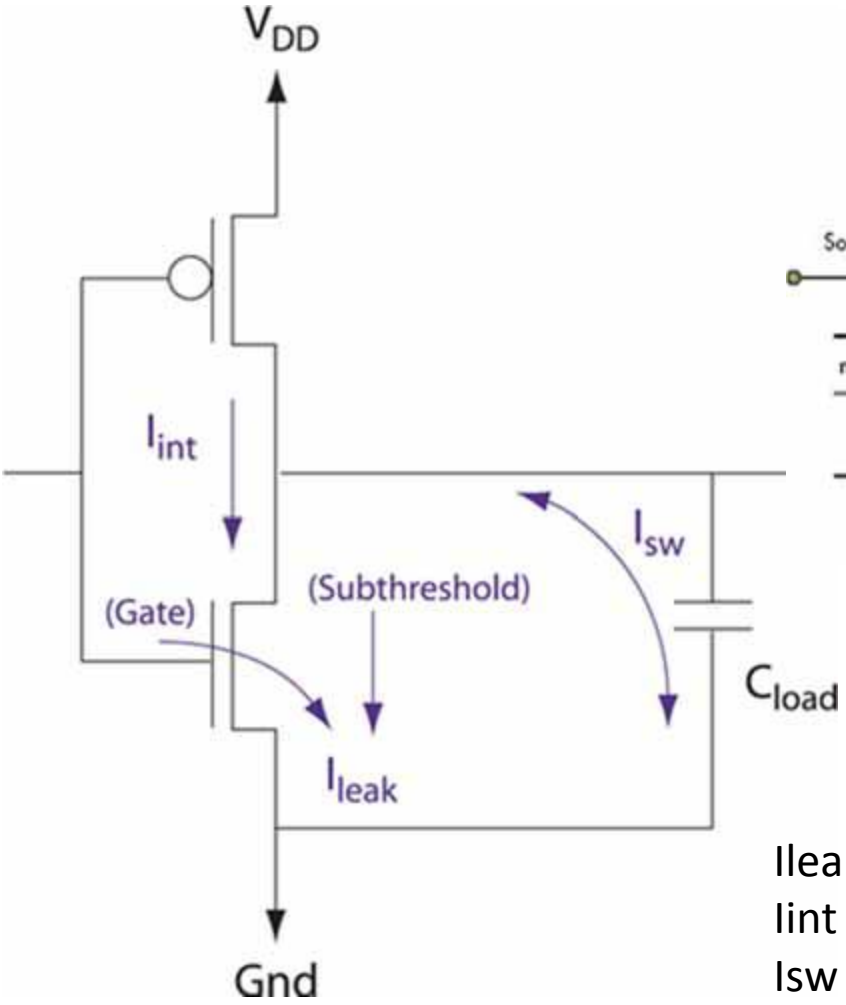


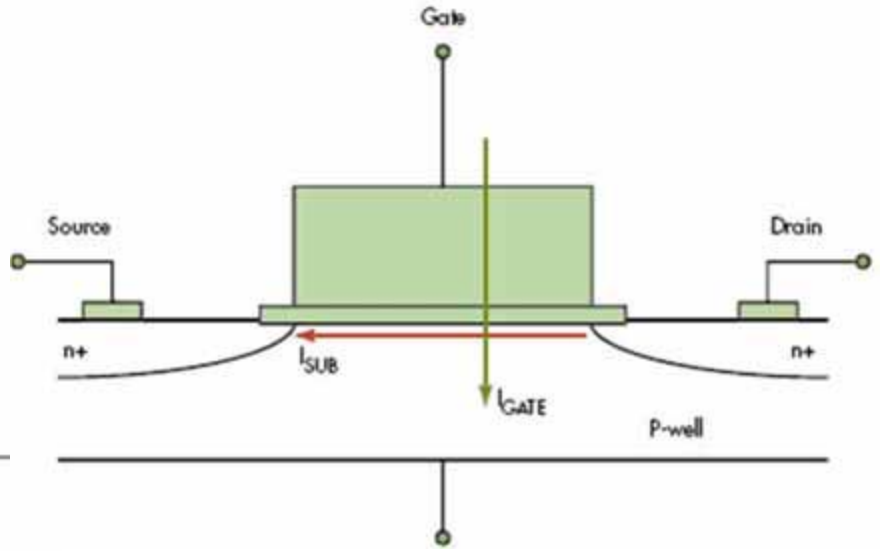
- In many cases, faster execution also means less energy, but the opposite may be true if power has to be increased to allow faster execution.

Low Power vs. Low Energy

- Minimizing the power consumption is important for
  - the design of the power supply
  - the design of voltage regulators
  - the dimensioning of interconnect
  - cooling (short term cooling)
    - high cost (estimated to be rising at $1 to $3 per Watt for heat dissipation [Skadron et al. ISCA 2003])
    - limited space
- Minimizing the energy consumption is important due to
  - restricted availability of energy (mobile systems)
  - limited battery capacities (only slowly improving)
  - very high costs of energy (solar panels, in space)
  - long lifetimes, low temperatures

Power Consumption of a CMOS Gate



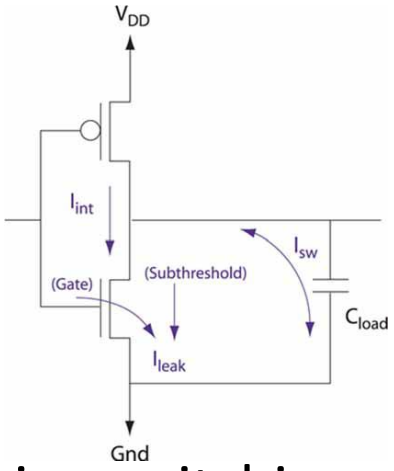


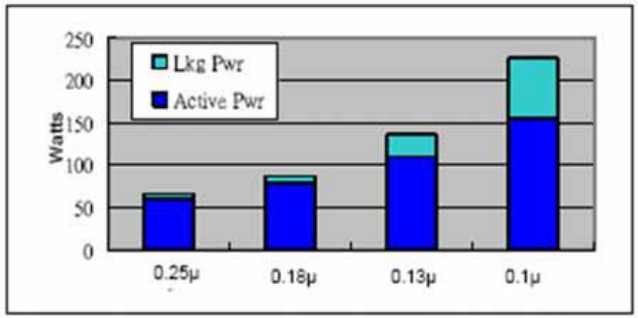
subthreshold and gate-oxide leakage

Ileak : leakage current
Iint : short circuit current
Isw : switching current

Power Consumption of CMOS Processors

- Main sources:
  - Dynamic power consumption
    - charging and discharging capacitors
  - Short circuit power consumption
    - short circuit path between supply rails during switching
  - Leakage
    - leaking diodes and translators
    - becomes one of the major factors due to shrinking feature sizes in semiconductor technology





(Micro32 Keynotes by Fred Pollack)

Dynamic Voltage Scaling (DVS)

Power consumption of CMOS circuits (ignoring leakage):

$$P \sim \alpha C_L V_{dd}^2 f$$

V_{dd} : supply voltage

α : switching activity

C_L : load capacity

f : clock frequency

Delay for CMOS circuits:

$$\tau = kC_L \frac{V_{dd}}{(V_{dd} - V_T)^2}$$

V_{dd} : supply voltage

V_T : threshold voltage

$$V_T \ll V_{dd}$$

- Decreasing V_{dd} reduces P quadratically (f constant).
- The gate delay increases only reciprocally.
- Maximal frequency f_{max} decreases linearly.

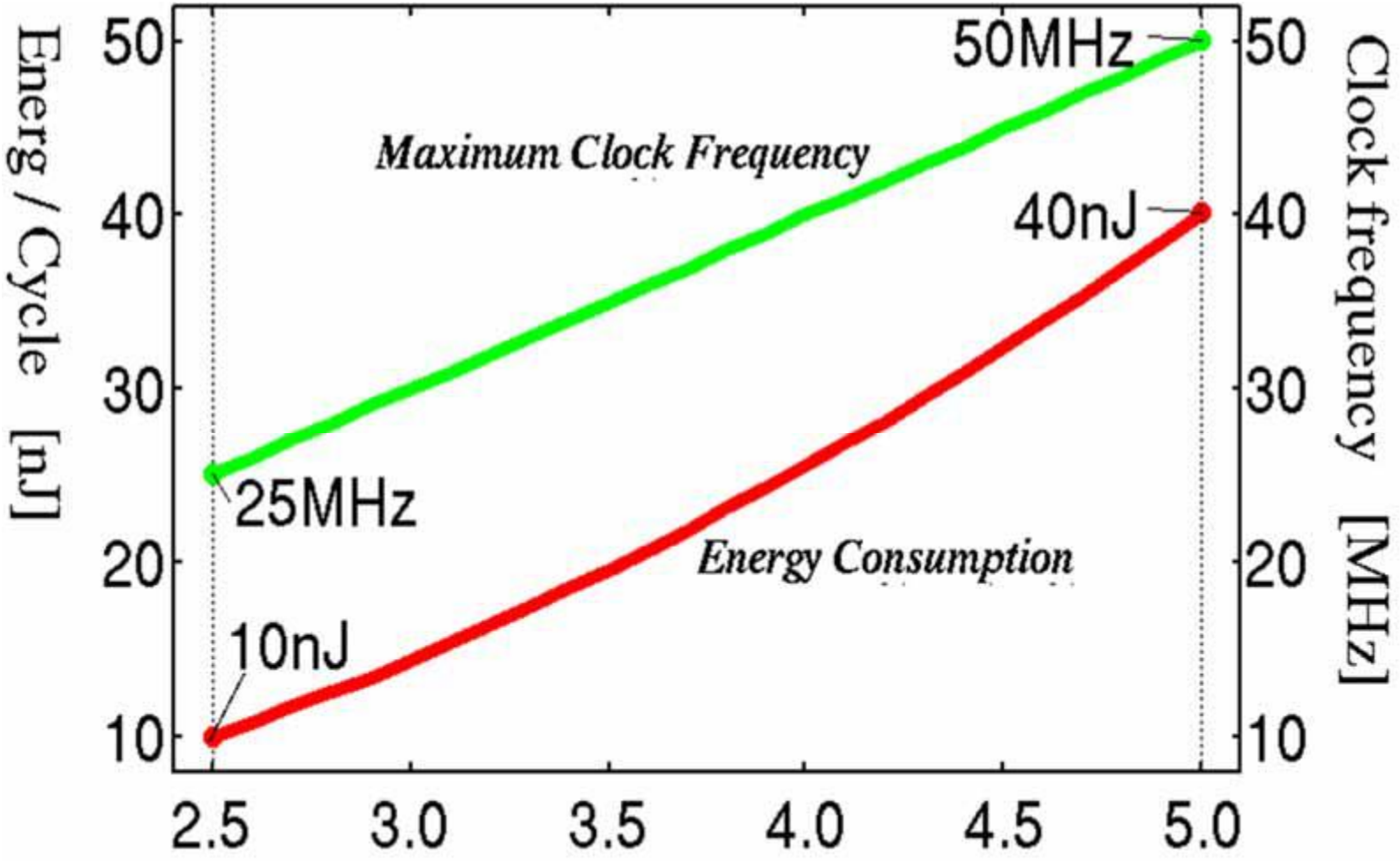
Potential for Energy Optimization: DVS

$$P \sim \alpha C_L V_{dd}^2 f$$

$$E \sim \alpha C_L V_{dd}^2 f t = \alpha C_L V_{dd}^2 \ (\#\text{cycles})$$

- Saving energy for a given task:
  - Reduce the supply voltage V_{dd}
  - Reduce switching activity α
  - Reduce the load capacitance C_L
  - Reduce the number of cycles #cycles

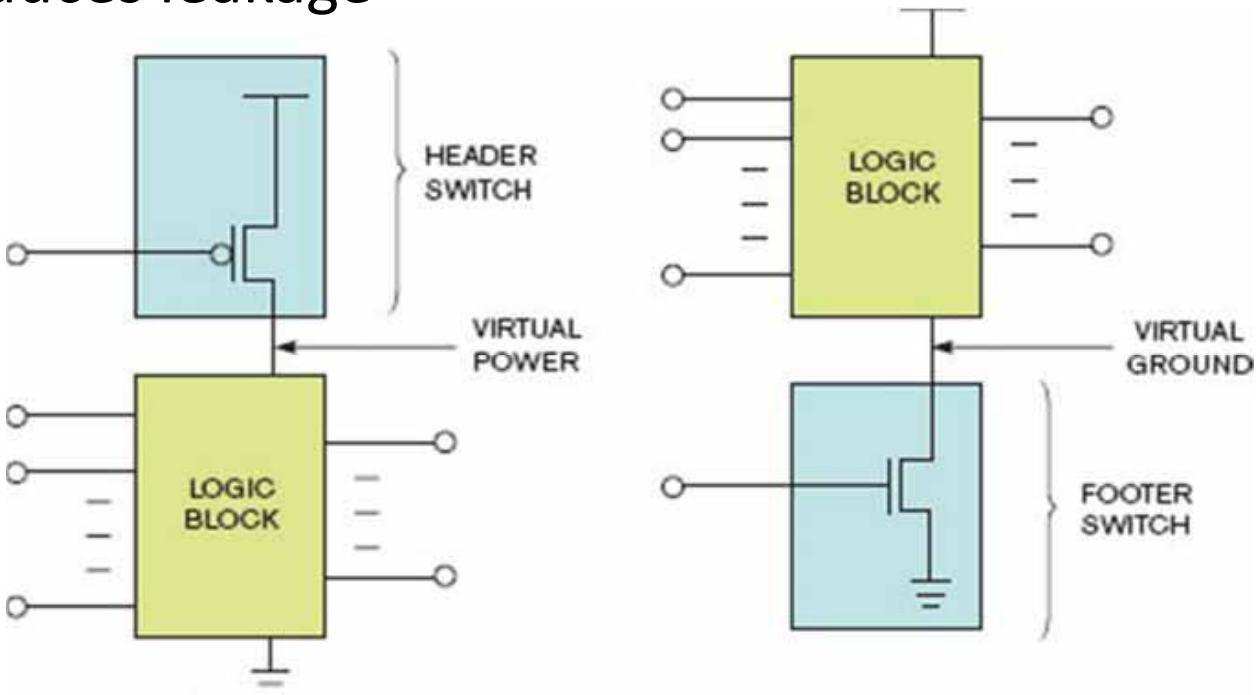
Example: Voltage Scaling



[Courtesy, Yasuura, 2000]

Power Supply Gating

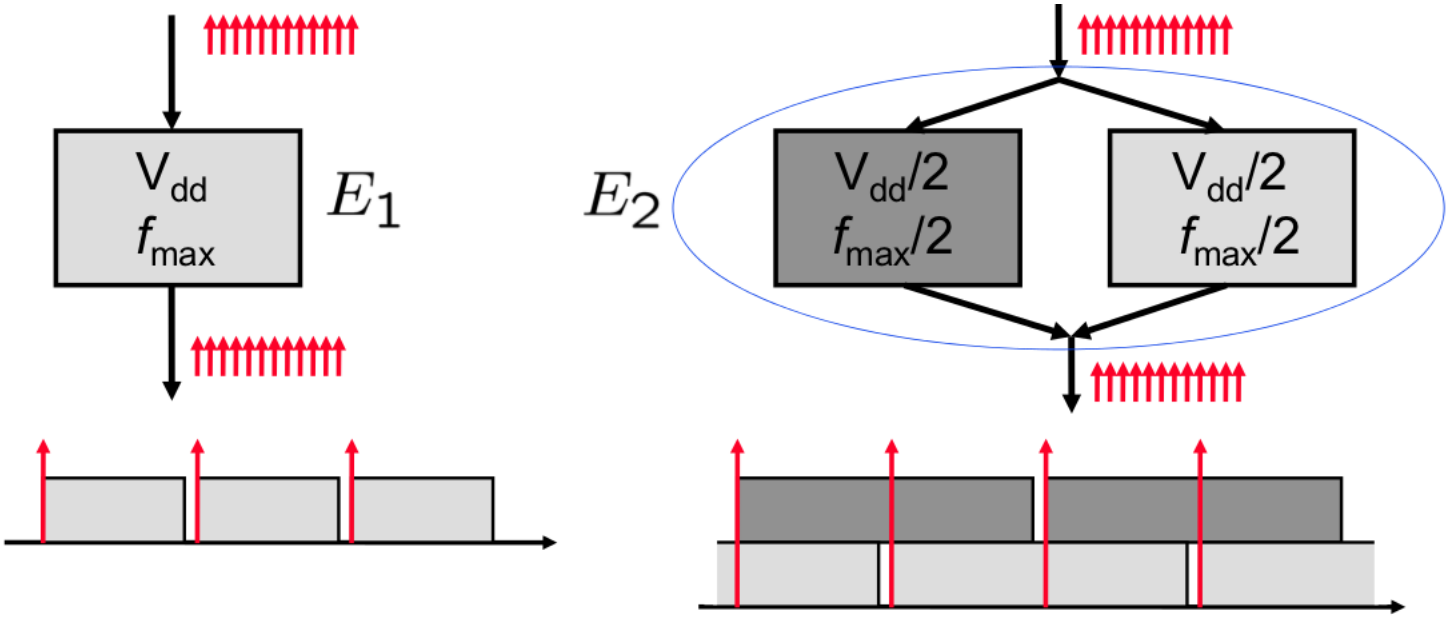
- Power gating is one of the most effective ways of minimizing static power consumption (leakage)
  - Cut-off power supply to inactive units/components
  - Reduces leakage



Outline

- General Remarks
- Power and Energy
- Basic Techniques
  - Parallelism
  - VLIW (parallelism and reduced overhead)
  - Dynamic Voltage Scaling
  - Dynamic Power Management

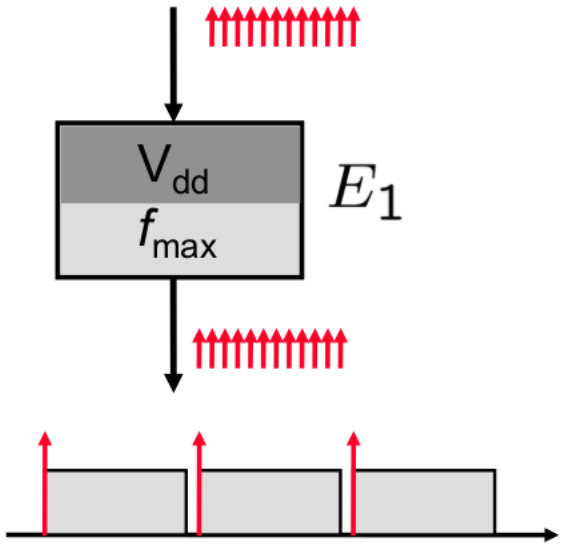
Use of Parallelism



$$E \sim V_{dd}^2 \ (\#\text{cycles})$$

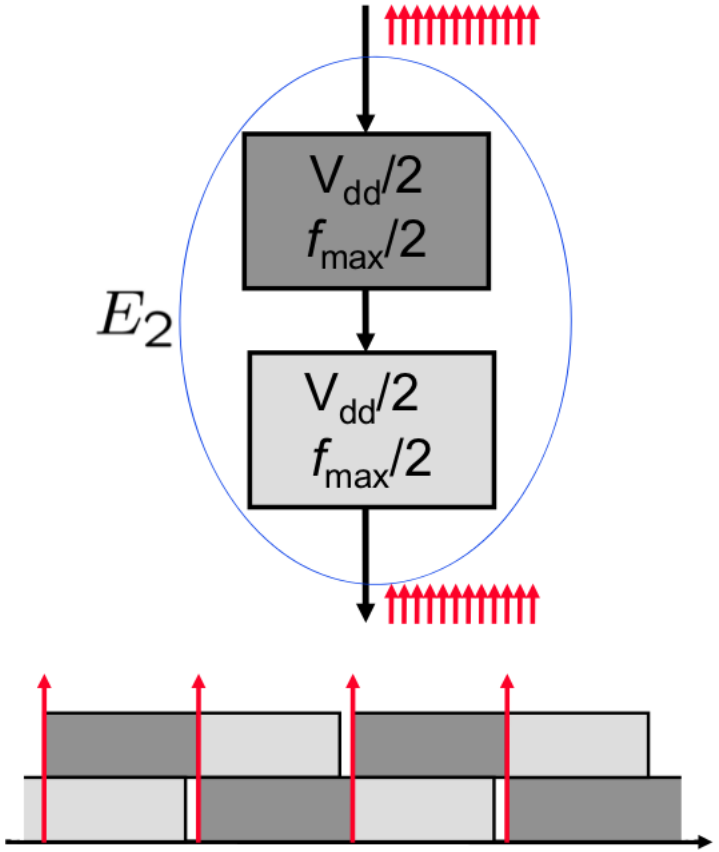
$$E_2 = \tfrac{1}{4} E_1$$

Use of Pipelining



$$E \sim V_{dd}^2 \ (\#\text{cycles})$$

$$E_2 = \frac{1}{4} E_1$$



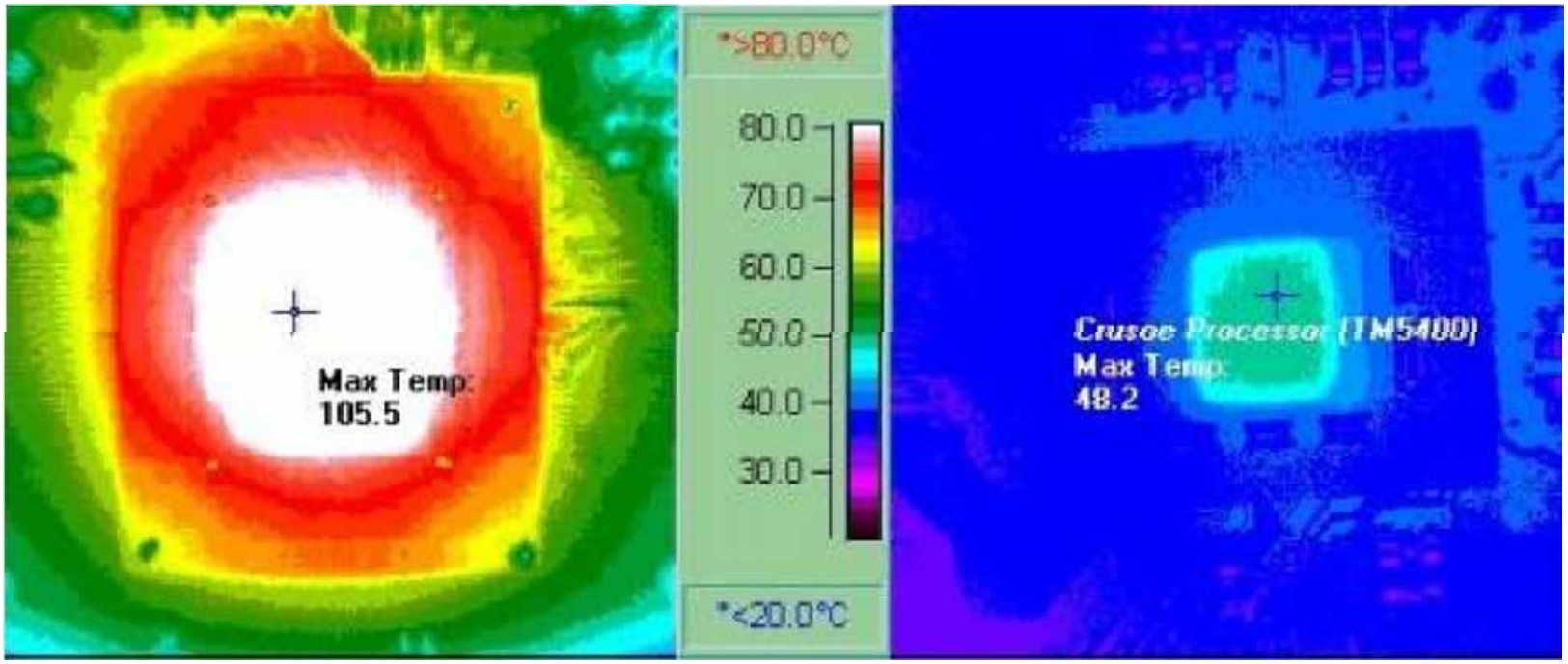
Outline

- General Remarks
- Power and Energy
- Basic Techniques
  - Parallelism
  - VLIW (parallelism and reduced overhead)
  - Dynamic Voltage Scaling
  - Dynamic Power Management

New ideas help …

Pentium

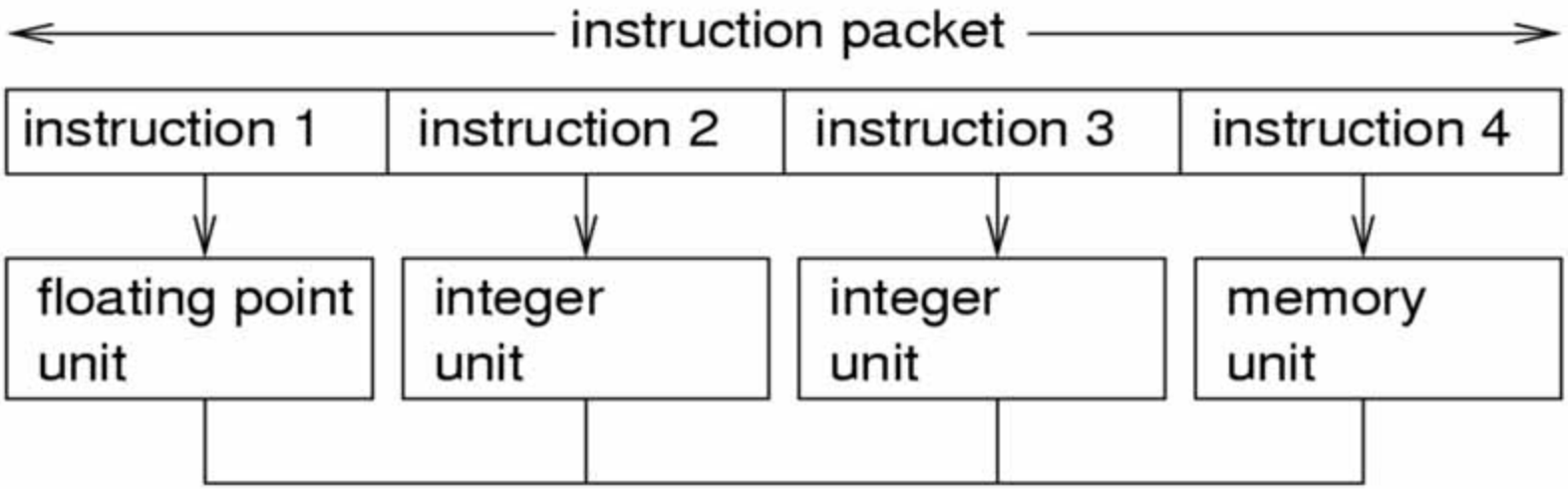
Crusoe



Running the same multimedia application.
As published by Transmeta [www.transmeta.com]

VLIW Architectures

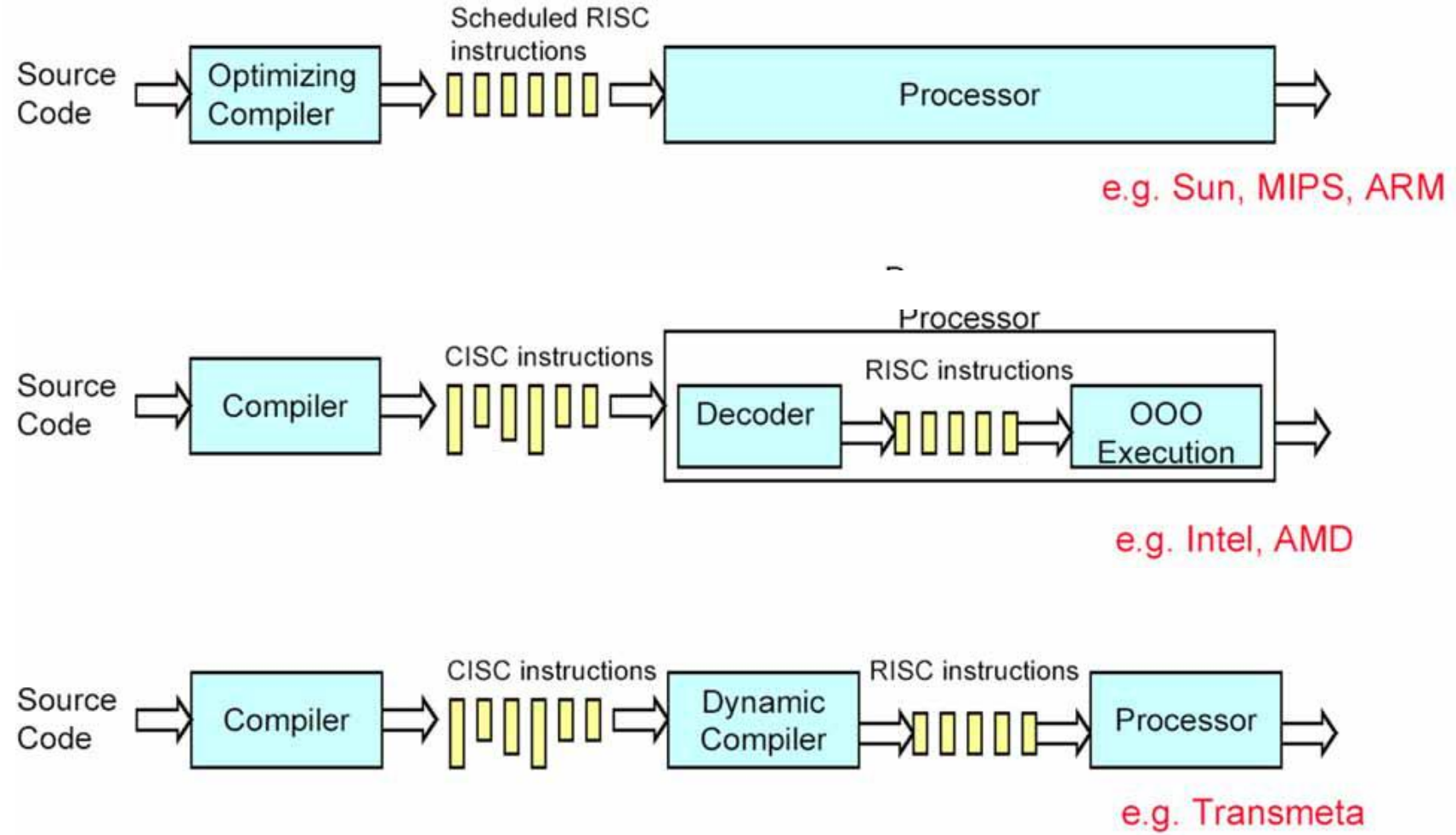
- Large degree of parallelism
  - many computational units, (deeply) pipelined
- Simple hardware architecture
  - explicit parallelism (parallel instruction set)
  - parallelization is done offline (compiler)



Transmeta is a typical VLIW Architecture

- 128-bit instructions (bundles):
  - 4 operations per instruction
  - 2 combinations of instructions allowed
- Register files
  - 64 integer, 32 floating point
- Some interesting features
  - 6 stage pipeline (2x fetch, decode, register read, execute, write)
  - X86 ISA execution using software techniques
    - Skip the binary compatibility problem !!
    - Interpretation and just-in-time binary translation
  - Speculation support

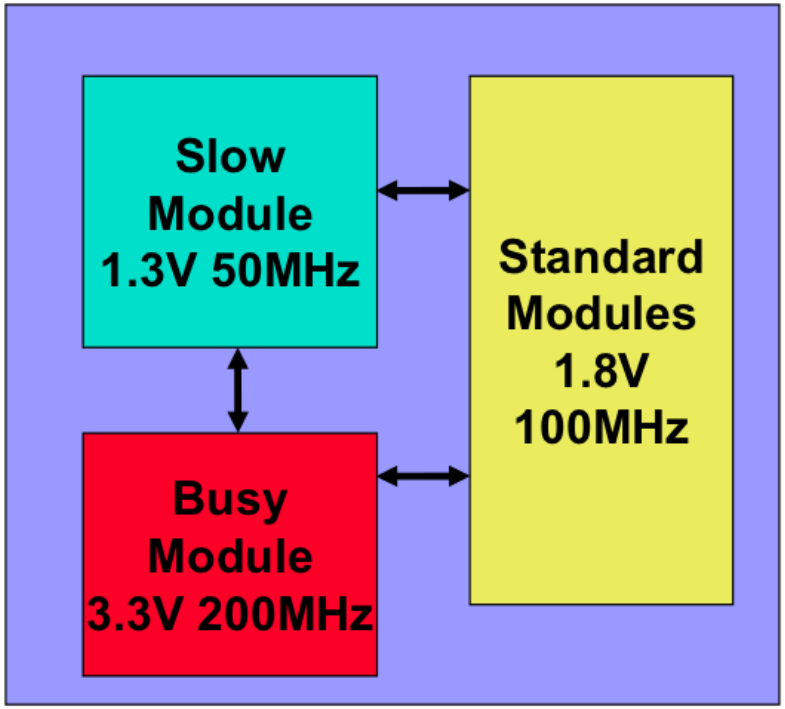
Transmeta


Source Code
Optimizing Compiler
Scheduled RISC instructions
Processor
e.g. Sun, MIPS, ARM
Source Code
Compiler
CISC instructions
Processor
RISC instructions
Decoder
OOO Execution
e.g. Intel, AMD
Source Code
Compiler
CISC instructions
Dynamic Compiler
RISC instructions
Processor
e.g. Transmeta

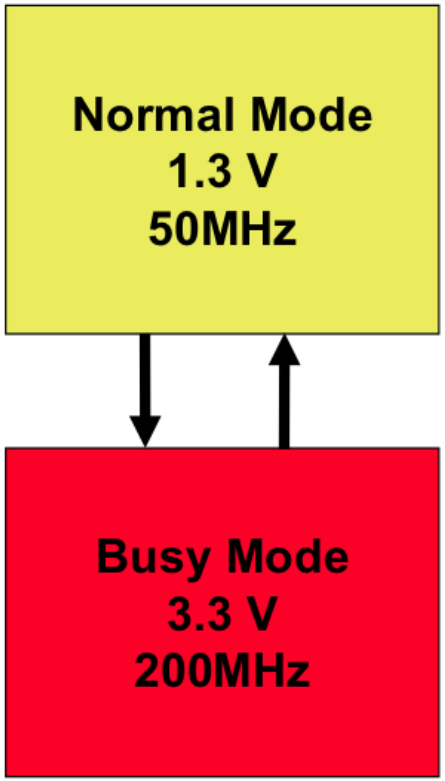
Outline

- General Remarks
- Power and Energy
- Basic Techniques
  - Parallelism
  - VLIW (parallelism and reduced overhead)
  - Dynamic Voltage Scaling
  - Dynamic Power Management

Spatial vs. Dynamic Voltage Management



Not all components require same performance.



Required performance may change over time

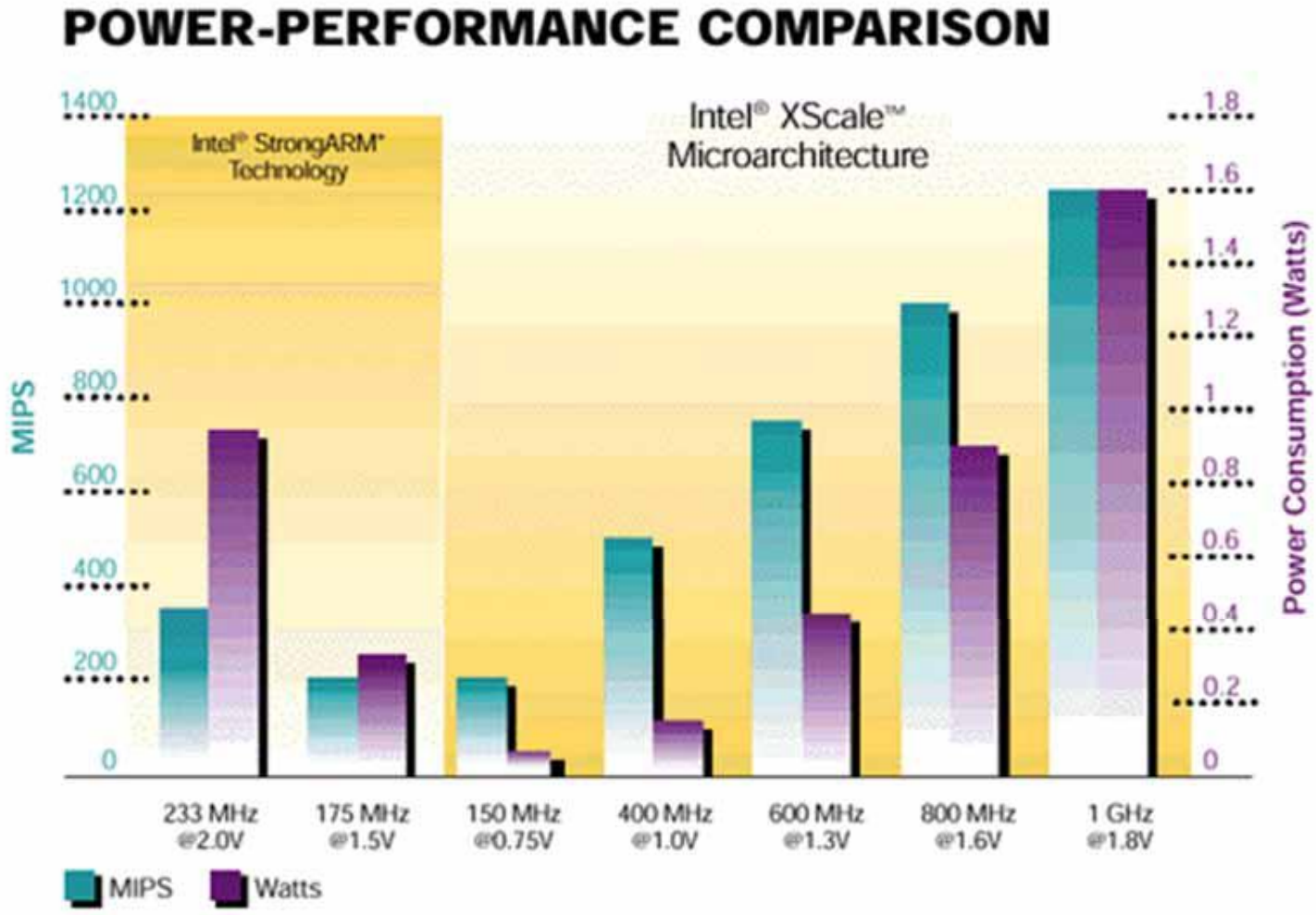
Potential for Energy Optimization: DVS

$$P \sim \alpha C_L V_{dd}^2 f$$

$$E \sim \alpha C_L V_{dd}^2 f t = \alpha C_L V_{dd}^2 \ (\#\text{cycles})$$

- Saving energy for a given task:
  - Reduce the supply voltage V_{dd}
  - Reduce switching activity α
  - Reduce the load capacitance C_L
  - Reduce the number of cycles #cycles

Example: INTEL Xscale

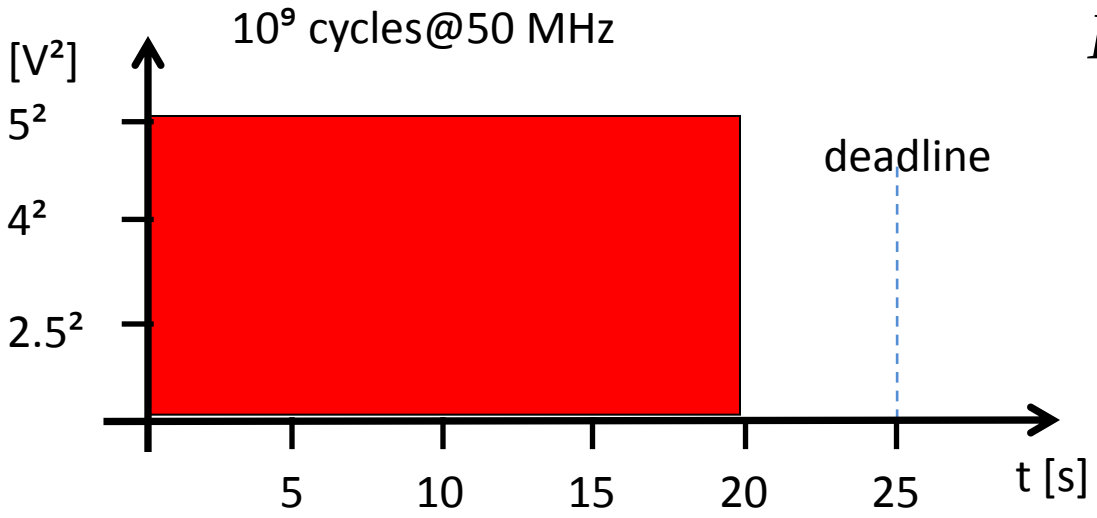


OS should schedule distribution of the energy budget.

DVS Example: a) Complete Task ASAP

- Task that need to execute 10^2 cycles within 25 seconds.

V_{dd} [V]	5.0	4.0	2.5
Energy per cycle [nJ]	40	25	10
f_{max} [MHz]	50	40	25
Cycle time [ns]	20	25	40



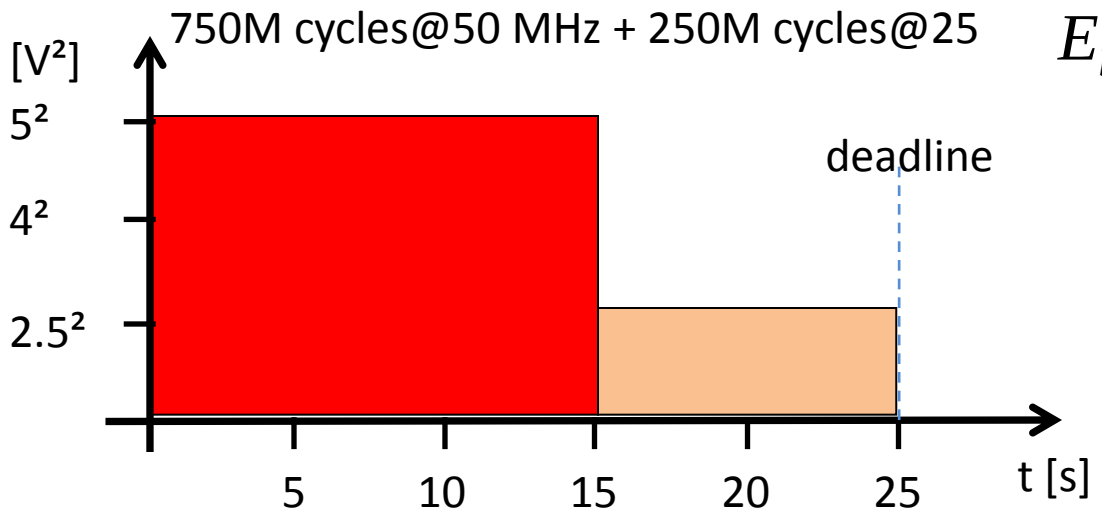
$$E_a = 10^9 \times 40 \times 10^{-9}$$

$$= 40[J]$$

DVS Example: b) Two Voltages

- Task that need to execute 10^2 cycles within 25 seconds.

V_{dd} [V]	5.0	4.0	2.5
Energy per cycle [nJ]	40	25	10
f_{max} [MHz]	50	40	25
Cycle time [ns]	20	25	40

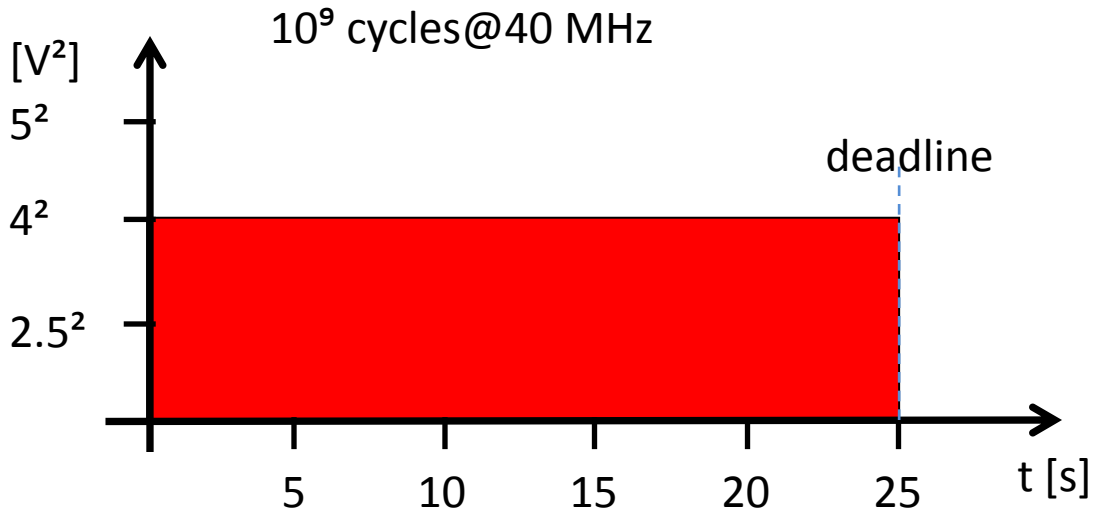


$$E_b = 750 \times 10^6 \times 40 \times 10^{-9}$$
$$+ 250 \times 10^6 \times 10 \times 10^{-9}$$
$$= 32.5[J]$$

DVS Example: c) Optimal Voltage

- Task that need to execute 10^2 cycles within 25 seconds.

V_{dd} [V]	5.0	4.0	2.5
Energy per cycle [nJ]	40	25	10
f_{max} [MHz]	50	40	25
Cycle time [ns]	20	25	40

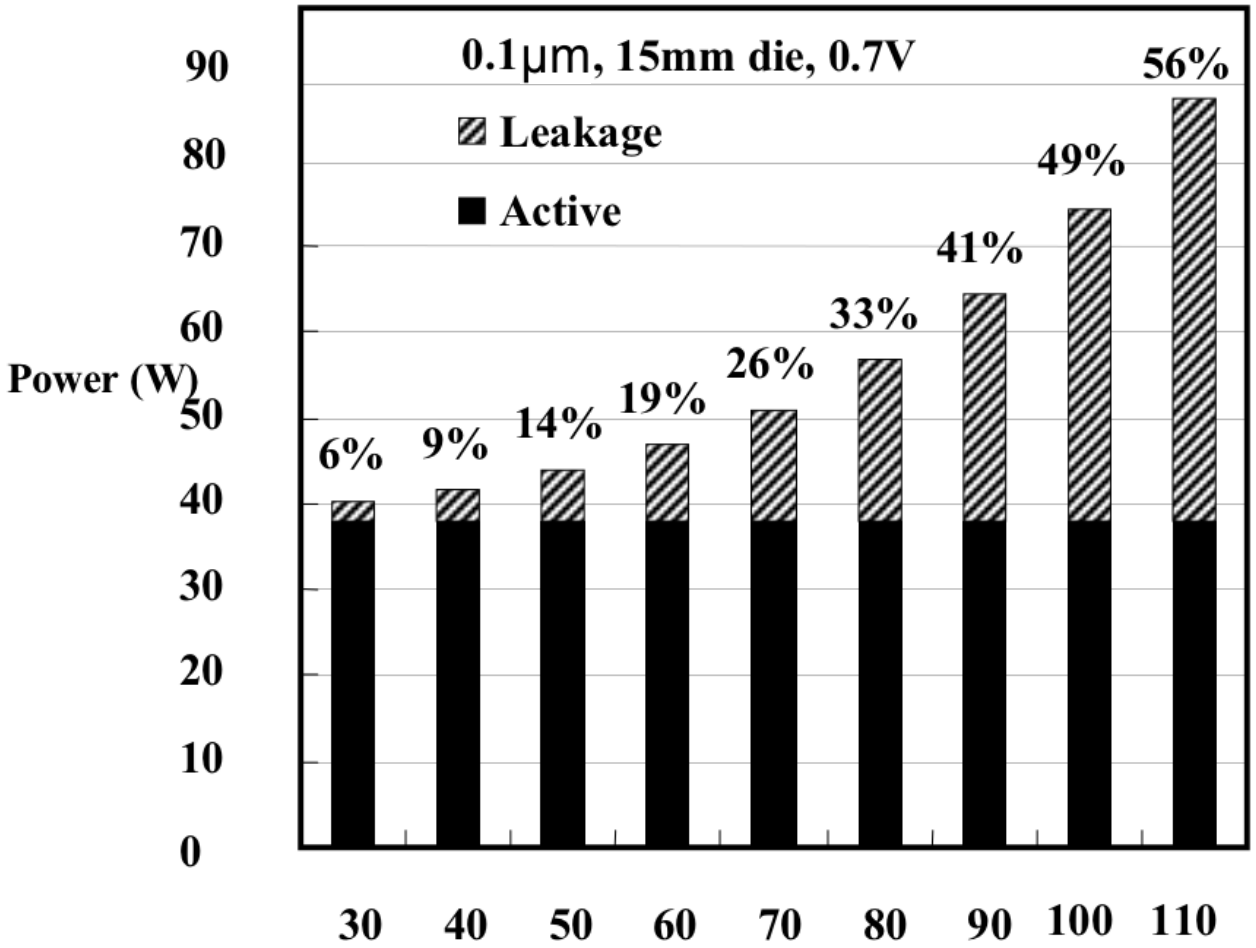


$$E_b = 10^9 \times 25 \times 10^{-9}$$
$$= 25[J]$$

Outline

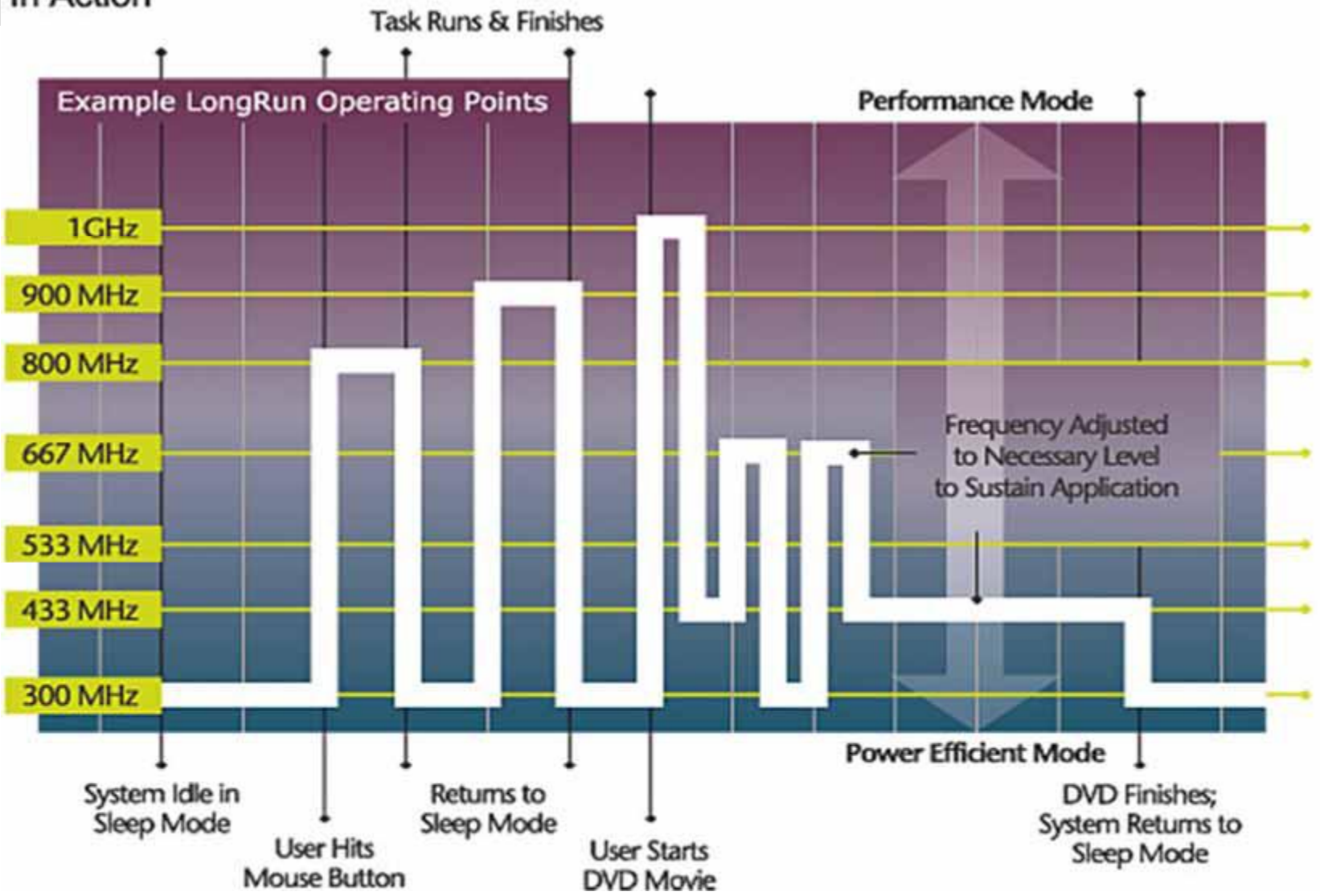
- General Remarks
- Power and Energy
- Basic Techniques
  - Parallelism
  - VLIW (parallelism and reduced overhead)
  - Dynamic Voltage Scaling
  - Dynamic Power Management

Dynamic Power V.S. Static Power



Transmeta™ LongRun™ Power Management

In Action



Dynamic Power Management (DPM)

Dynamic Power management tries to assign optimal **power saving states**

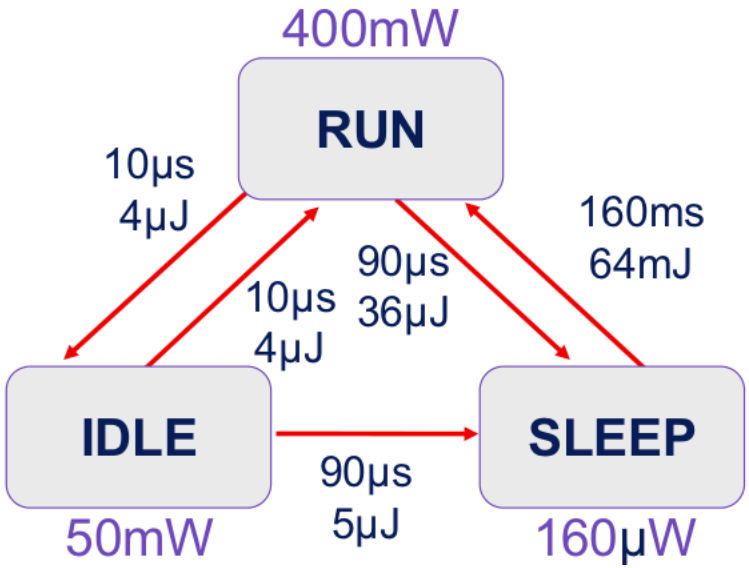
Requires Hardware Support

Example: StrongARM SA1100

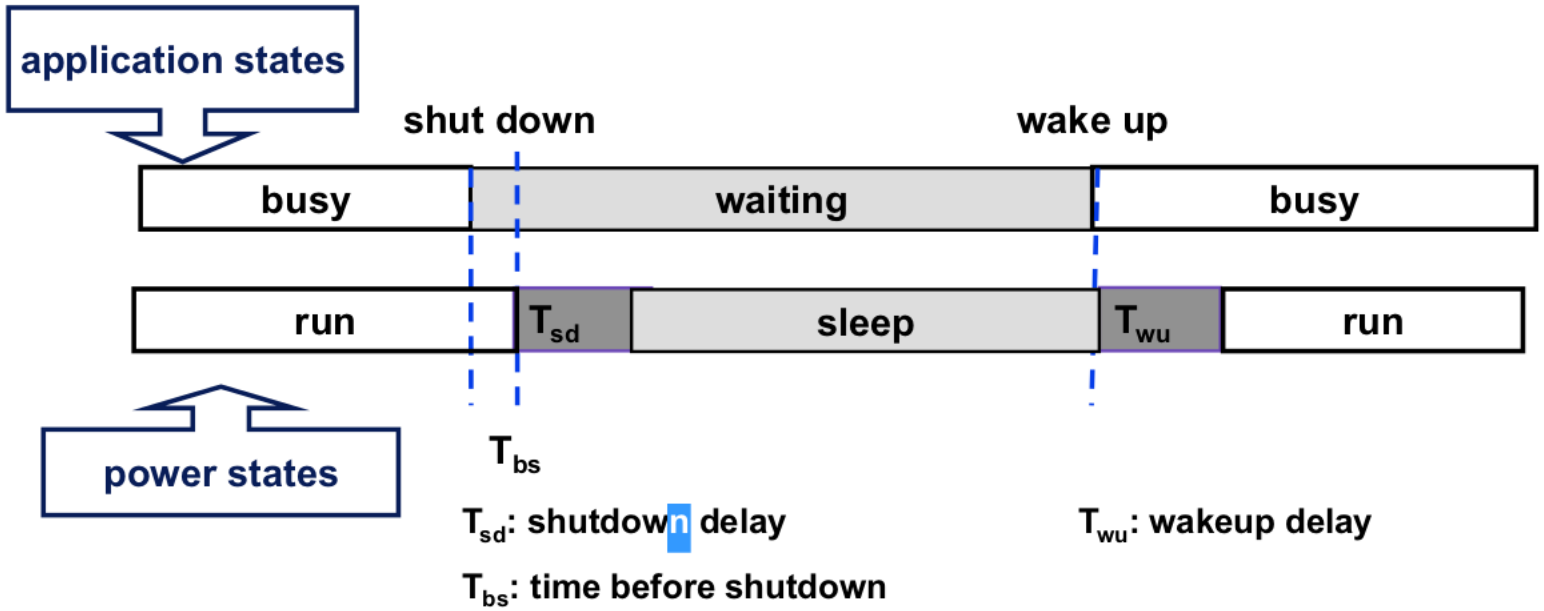
RUN: operational

IDLE: a SW routine may stop the CPU when not in use, while monitoring interrupts

SLEEP: Shutdown of on-chip activity



Reduce Power According to Workload



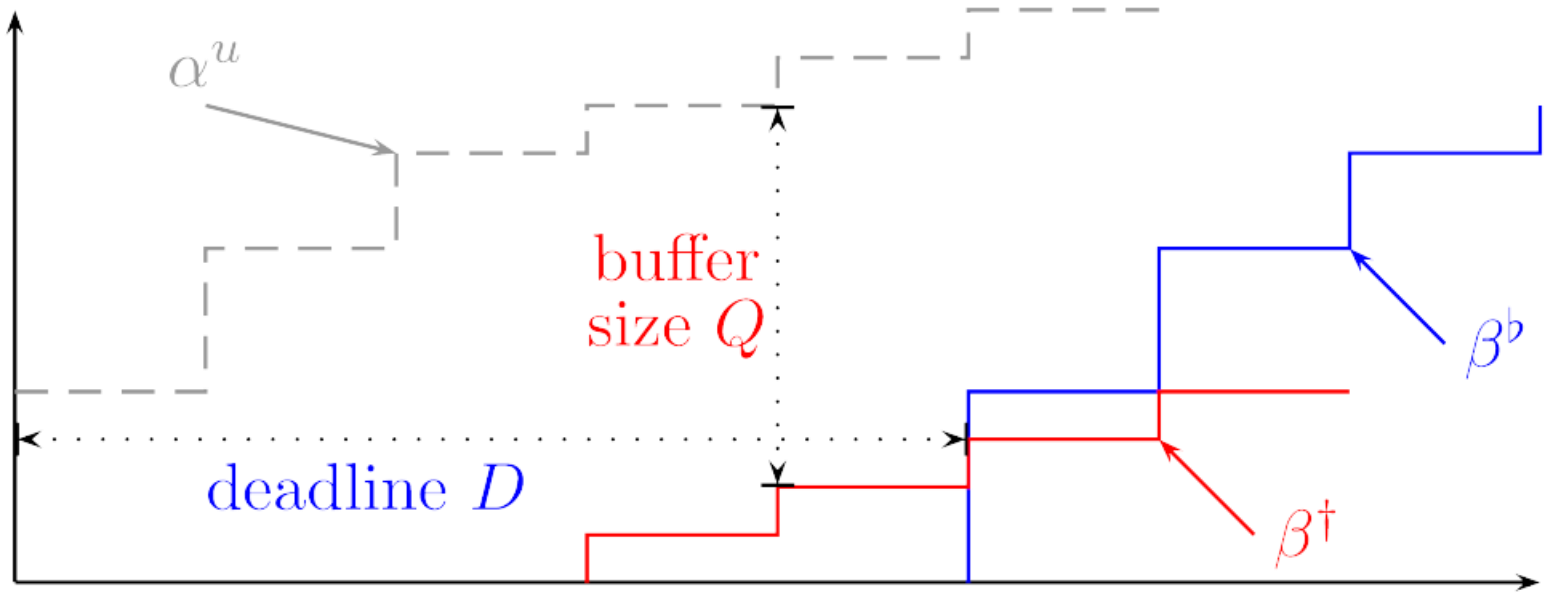
Desired: Shutdown only during long idle times
→ Tradeoff between savings and overhead

Reduce Static Power Example

- Assumption
  - Given arrival curve, buffer size and deadline requirement, power parameters
- Problem statement
  - To determine the on/off periods such that
    - energy consumption is minimized
    - no deadline violation and buffer overflow

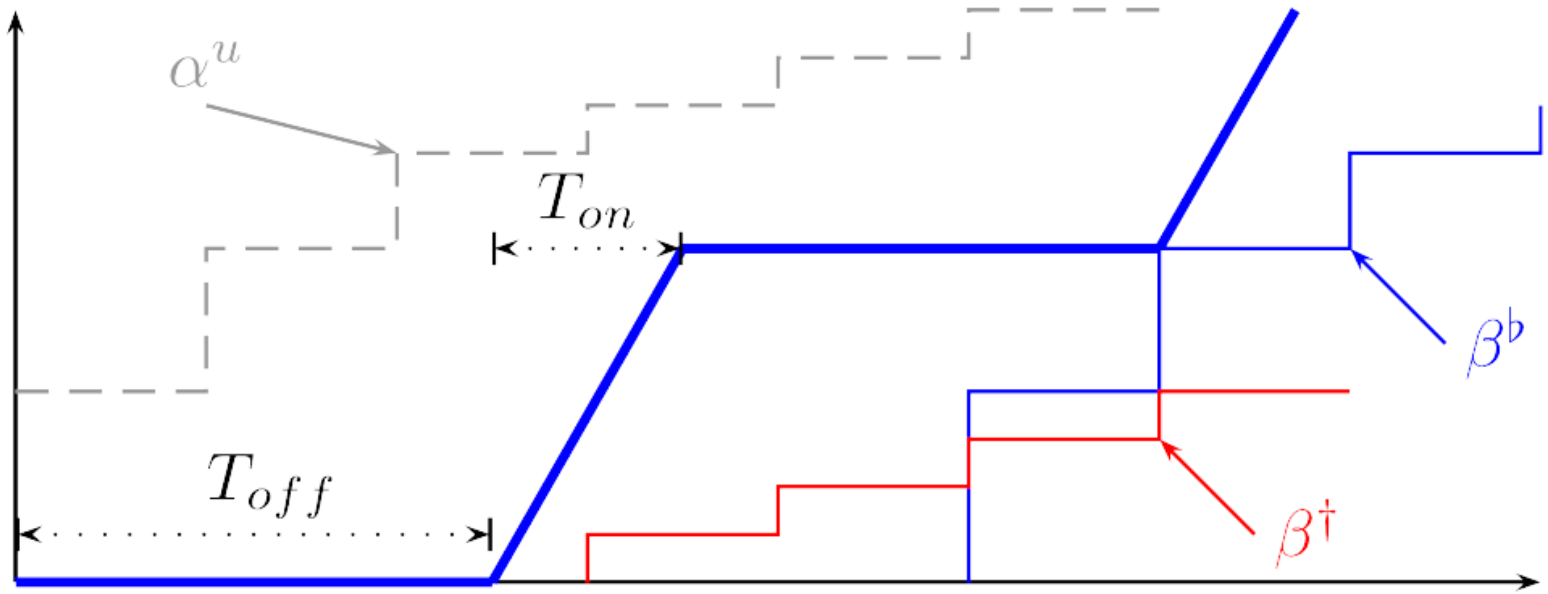
Details see the HuangDPMOffline2009 paper

Basic Idea: Use RTC to Compute Bounds



- $\beta^\flat(\Delta) = \alpha^u(\Delta - D)$ is the service demand to avoid deadline violation
- $\beta^\dagger(\Delta) = \alpha^u(\Delta) - B$ is the service demand to avoid buffer overflow

Basic Idea: Choose the Bound of Min Energy

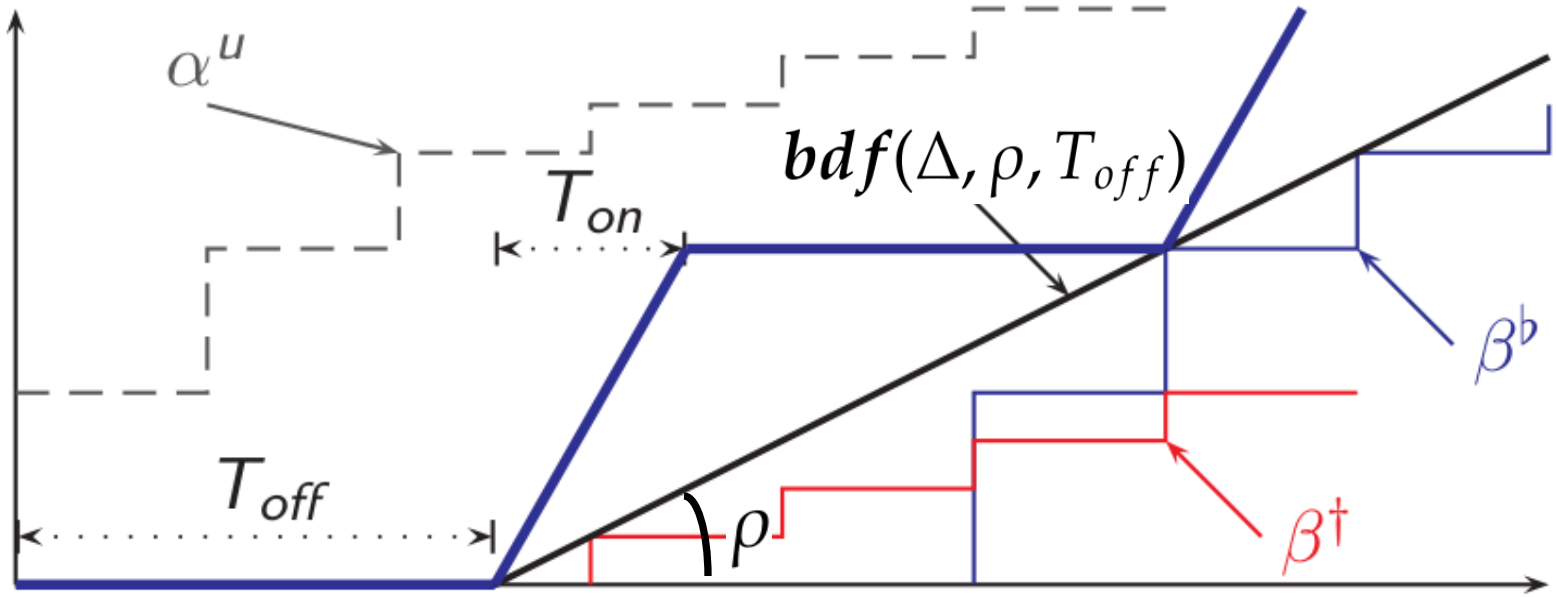


- Derive a periodic on/off curve which energy consumption is minimized

$$\beta^G(\Delta) = \max\left(\left\lfloor \frac{\Delta}{T_{on}+T_{off}} \right\rfloor \cdot T_{on},\ \Delta - \left\lceil \frac{\Delta}{T_{on}+T_{off}} \right\rceil \cdot T_{off}\right) \geq \max\{\beta^\dagger,\ \beta^\flat\}$$

$$E(L, T_{on}, T_{off}) = \frac{L}{T_{on}+T_{off}} E_{sw} + \frac{L \cdot T_{on}}{T_{on}+T_{off}} P_s + \frac{L \cdot T_{off}}{T_{on}+T_{off}} P_\sigma + \sum_{S_i \in \mathcal{S}} w_i \cdot \gamma_i(L) \cdot (P_a - P_s)$$

Bounding Delay Approximation



$$\boldsymbol{bdf}(\Delta, \rho, T_{off}) \stackrel{def}{=} \max\{0, \rho \cdot (\Delta - T_{off})\}$$

$$\tilde{T}_{on} = \frac{\rho \cdot T_{off}}{1 - \rho}$$

→ From two parameters to only T_{off}

$$\rho_{\min, T_{off}} = \inf\left\{\rho : \boldsymbol{bdf}(\Delta, \rho, T_{off}) \geq \alpha_1^u(\Delta - D_1), \forall \Delta \geq 0\right\}$$