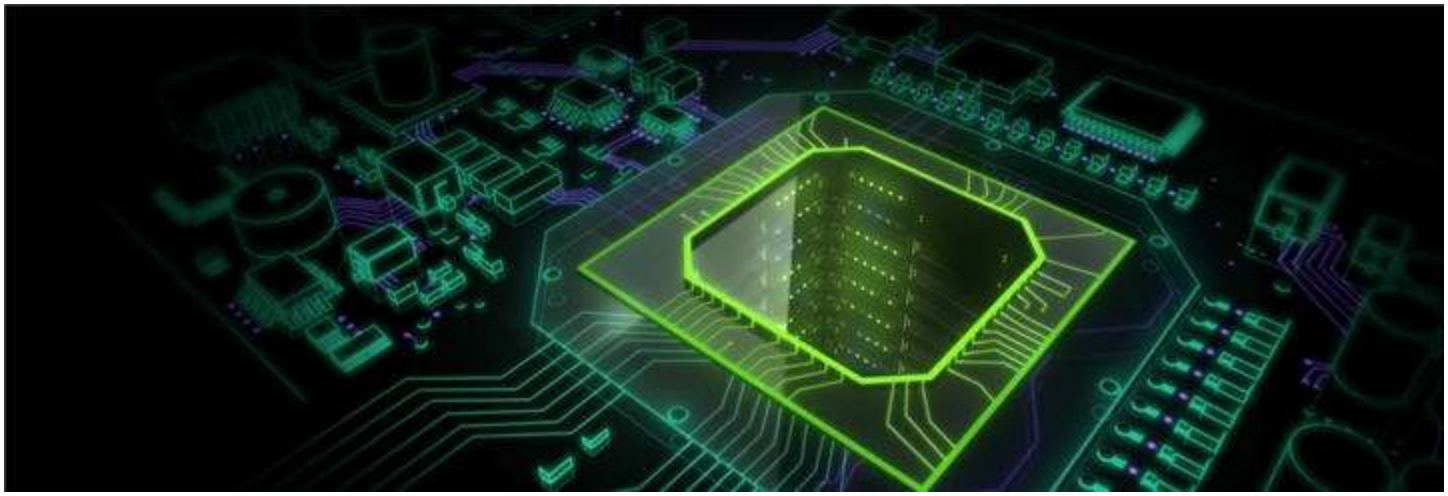
NYU
New York University

CSCI-GA.3033-012

Graphics Processing Units (GPUs): Architecture and Programming

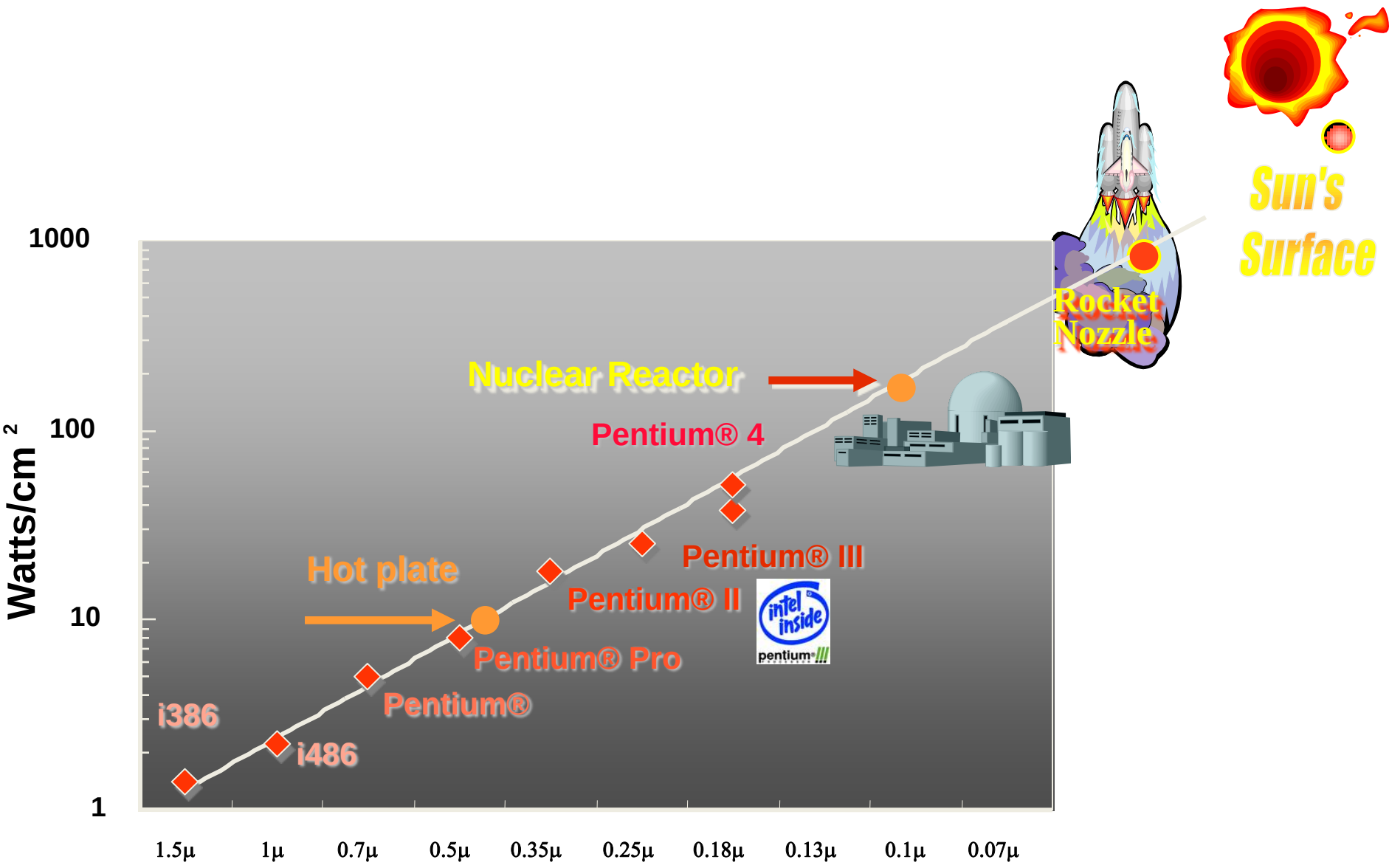
Lecture 12: Power-Wall

Mohamed Zahran (aka Z)
mzahran@cs.nyu.edu
http://www.mzahran.com



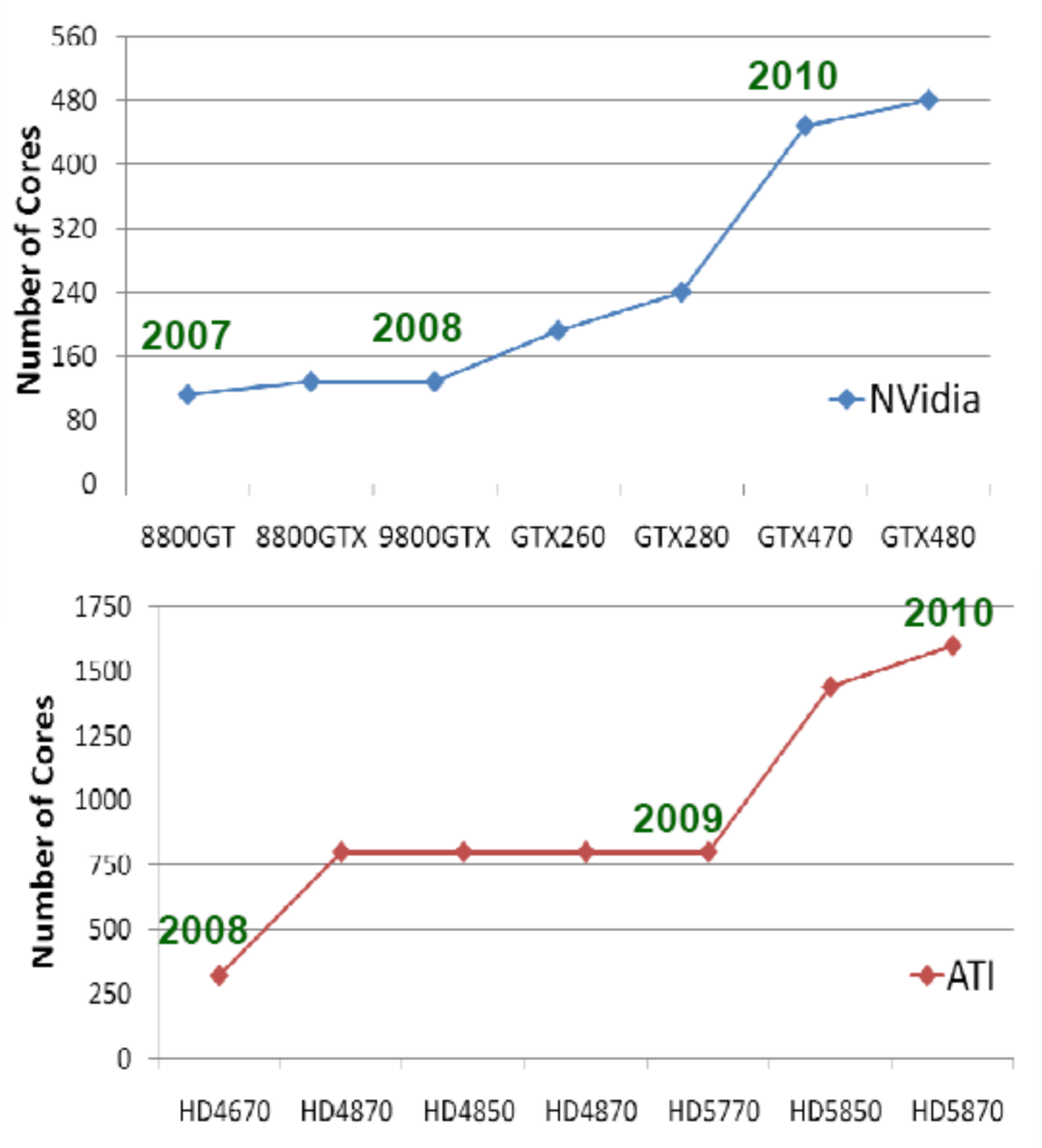
This Lecture

- Why do we have power-wall?
- Techniques to solve the problem
- Real-life example processor
- What can you do in software?
  - Is there power-aware software?

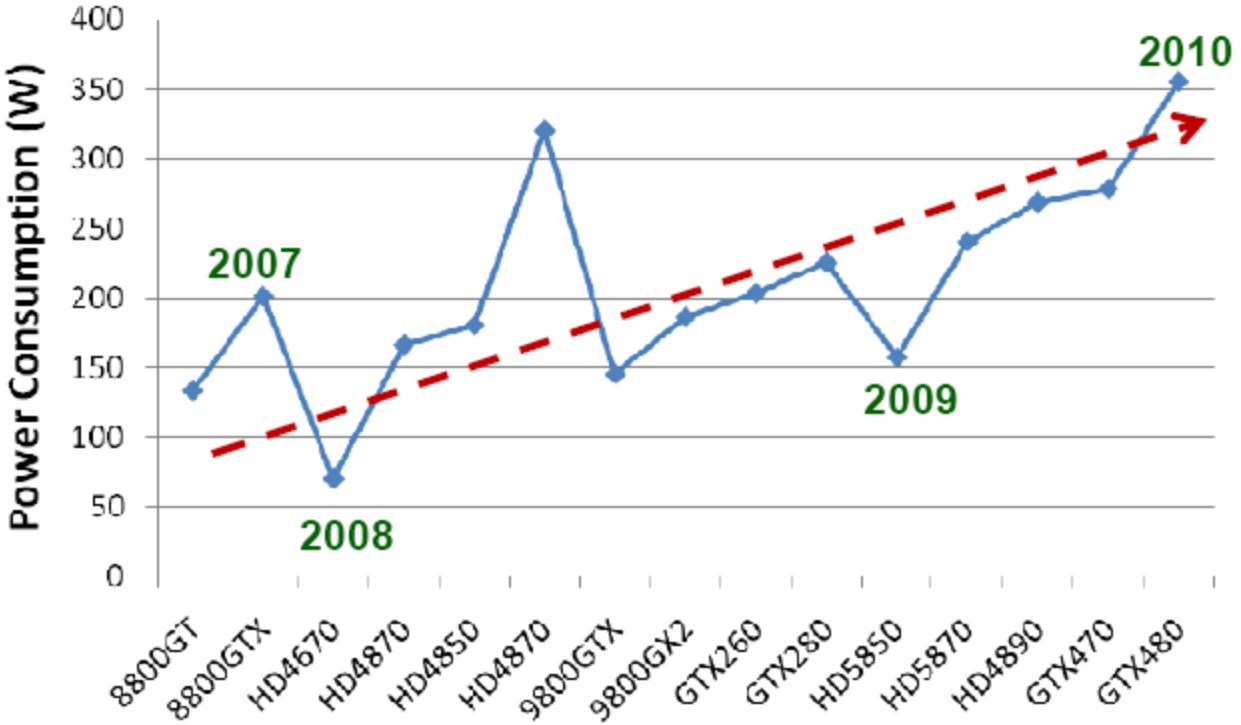


* "New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies" – Fred Pollack, Intel Corp. Micro32 conference key note - 1999.

Increasing number of cores



Source: http://www.prism.gatech.edu/~shong9/ISCA_2010.pptx



Power is also increasing!

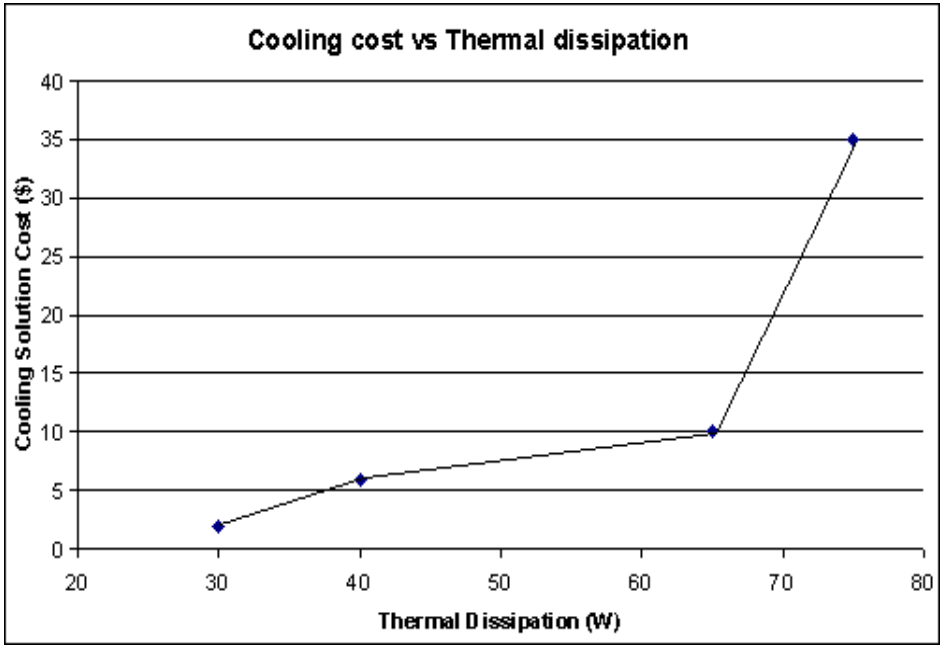
Source: http://www.prism.gatech.edu/~shong9/ISCA_2010.pptx

The Problem

- Cooling for GPUs is becoming prohibitively expensive.
  - Exasperated by the low profit margins in these market segments
- Today's cooling solutions are designed for worst-case behavior.
- Reducing the hot spots will help reduce cooling requirements.

Why Power Aware?

- Servers and Workstations
  - Packaging cost
  - High temperature = more expensive cooling system
- Embedded Devices
  - Battery Life
  - No place for fans, etc.

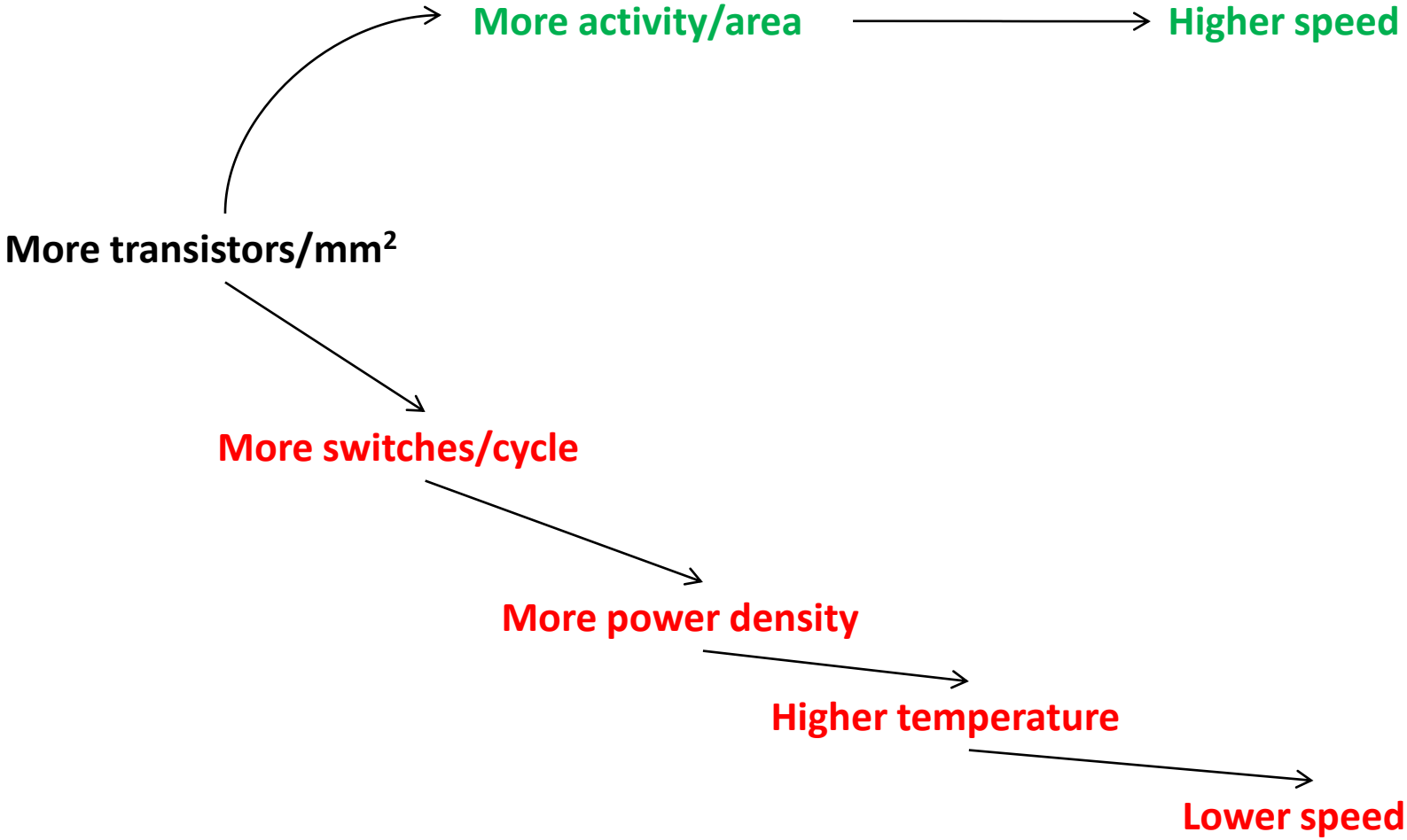


Moore's Law

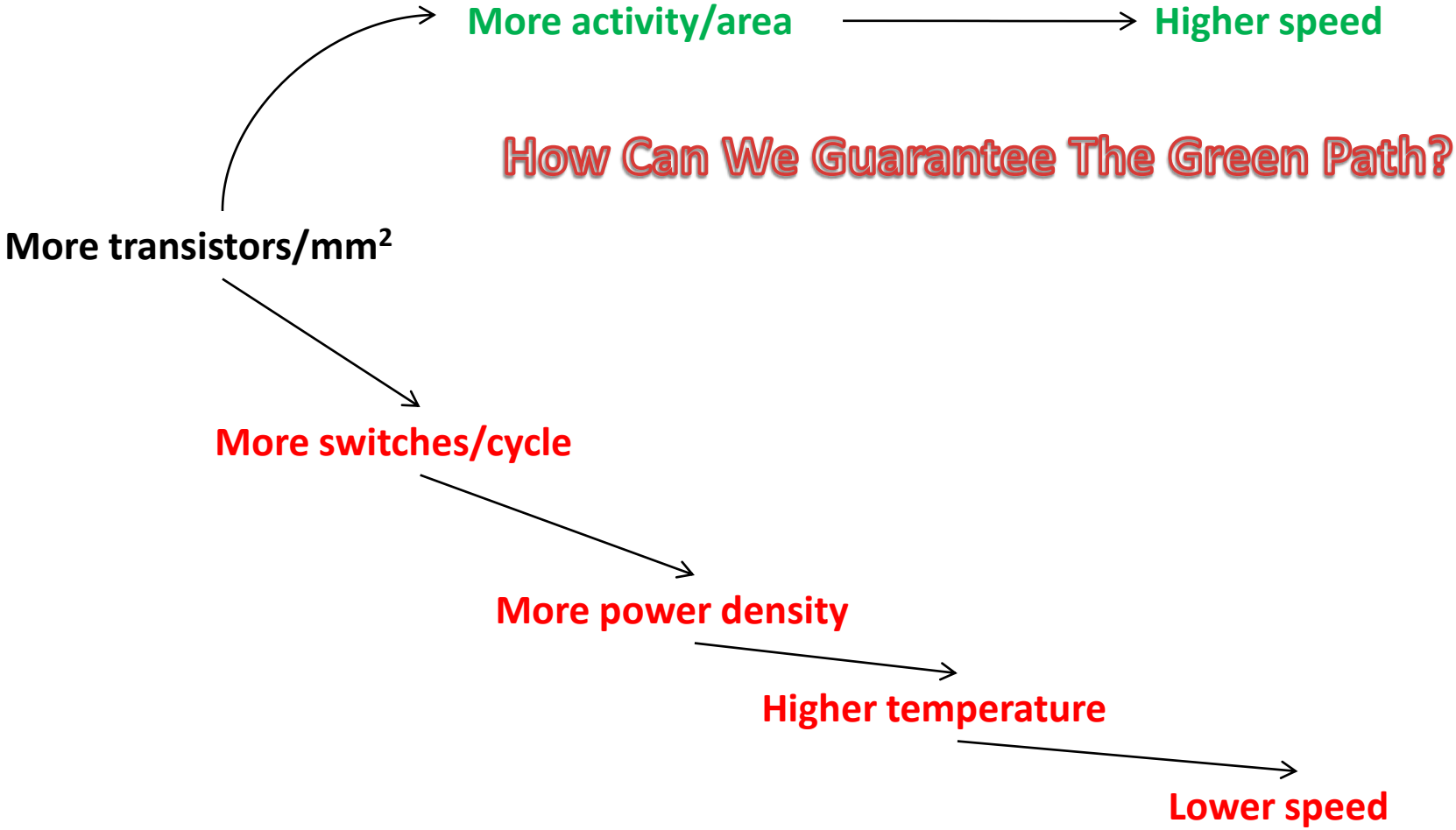
More transistors/mm^2

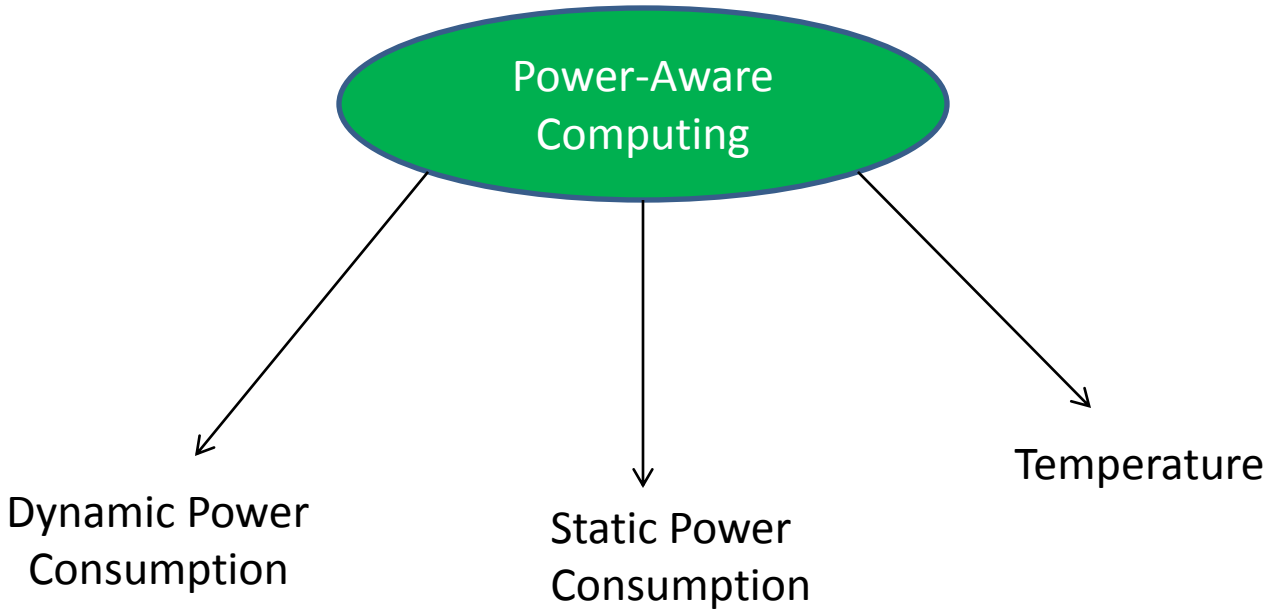
Moore's Law

Moore's Law


More activity/area
Higher speed
More transistors/mm²
More switches/cycle
More power density
Higher temperature
Lower speed

Moore's Law

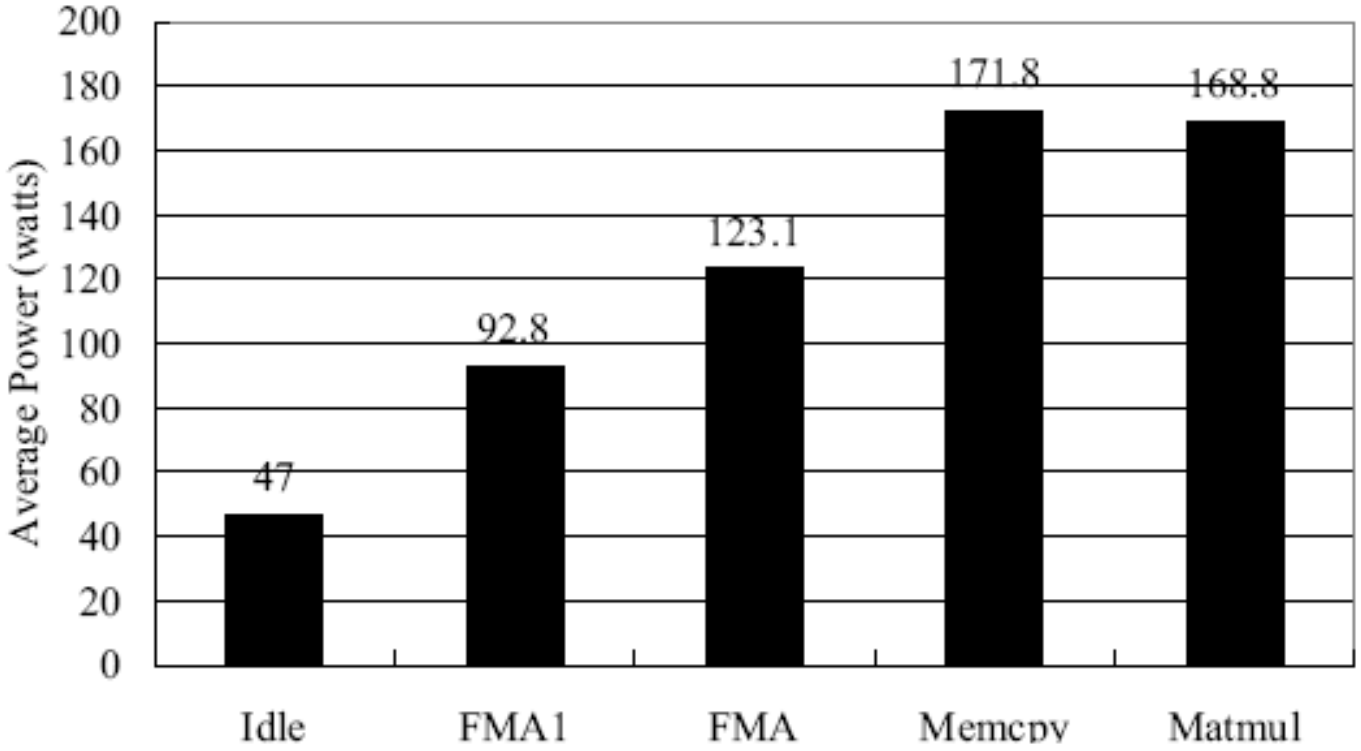

More activity/area
Higher speed
How Can We Guarantee The Green Path?
More transistors/mm²
More switches/cycle
More power density
Higher temperature
Lower speed


Power-Aware Computing
Dynamic Power Consumption
Static Power Consumption
Temperature

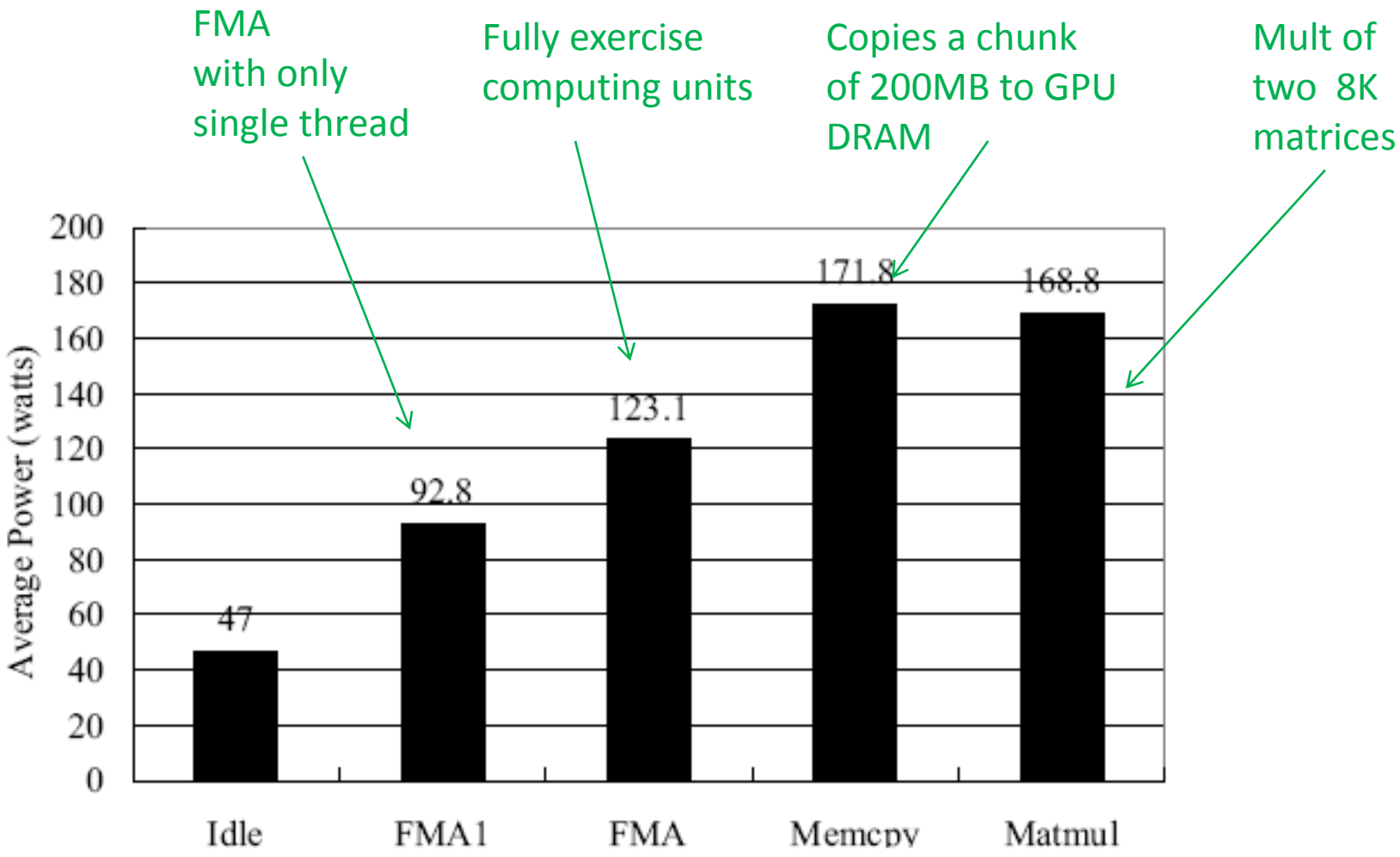
So … What is it about Moore's law?

- Power and temperature are becoming crucial
- GPU power consumption =

  Runtime power + idle power
- Power = dynamic + leakage
- Given power budget, how to get best performance?
- Given required performance, how to achieve it with lowest power?



GeForce 285 GTX



GeForce 285 GTX

Be Careful

- Static power is no longer trivial
- Higher utilization does not necessarily mean higher performance but for sure means higher power consumption/dissipation
- **Goal**: maximize performance/watt

Power-Aware Computing is:

Reducing power without loosing performance

Dynamic Power Consumption

$$P_{\text{dynamic}} = \alpha C V_{DD}^{\ 2} f A$$

Dynamic Power Consumption

$$P_{\text{dynamic}} = \alpha C V_{DD}^{\ 2} f A$$

depends on the wire lengths

supply voltage

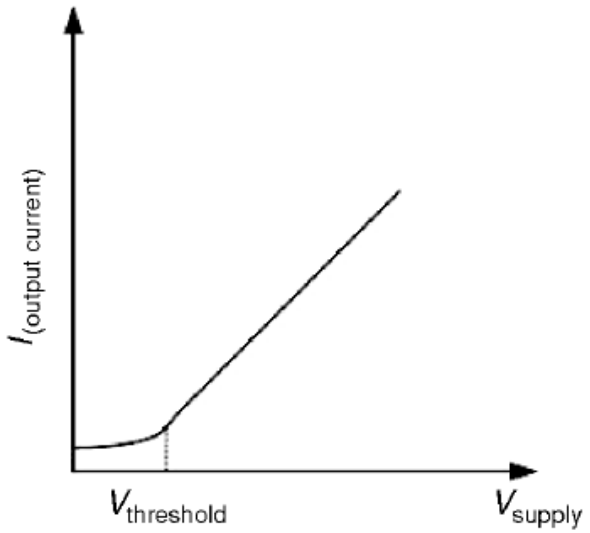
clock frequency

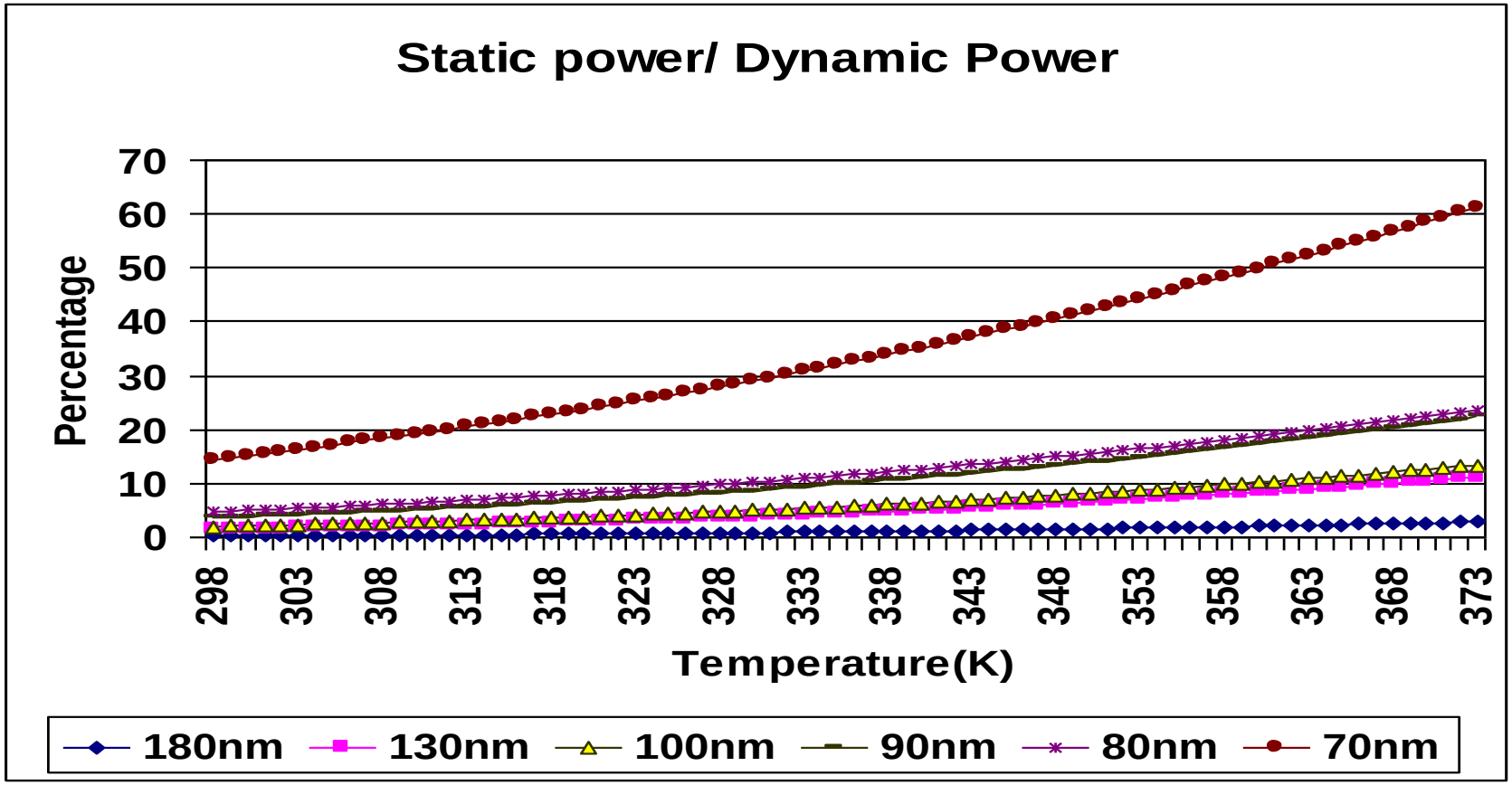
- between 0 and 1
- how often wires transition

Static Power Consumption

- 20% or more in sub-micron era
- Mostly leakage
  - represents the power dissipated by a transistor whose gate is intended to be off

$$P = V\left(k\mathrm{e}^{-qV_{\mathrm{th}}/(ak_{\mathrm{a}}T)}\right)$$

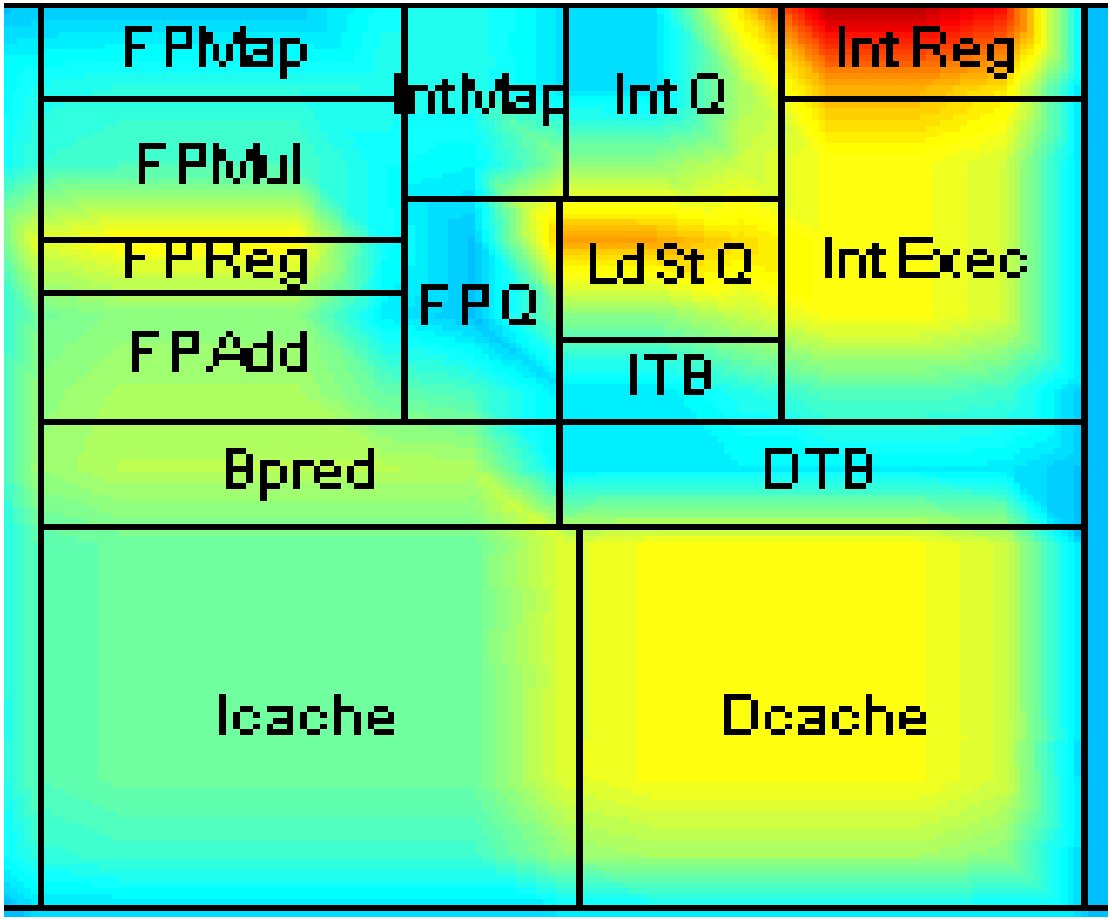



Static power/ Dynamic Power
Percentage
70
60
50
40
30
20
10
0
298
303
308
313
318
323
328
333
338
343
348
353
358
363
368
373
Temperature(K)
180nm
130nm
100nm
90nm
80nm
70nm

Temperature

- Lost power
- Leakage increases by order of magnitude at high temperature
- Higher temperature = lower mean-time-to-failure (MTTF)
- We need temperature-aware computing

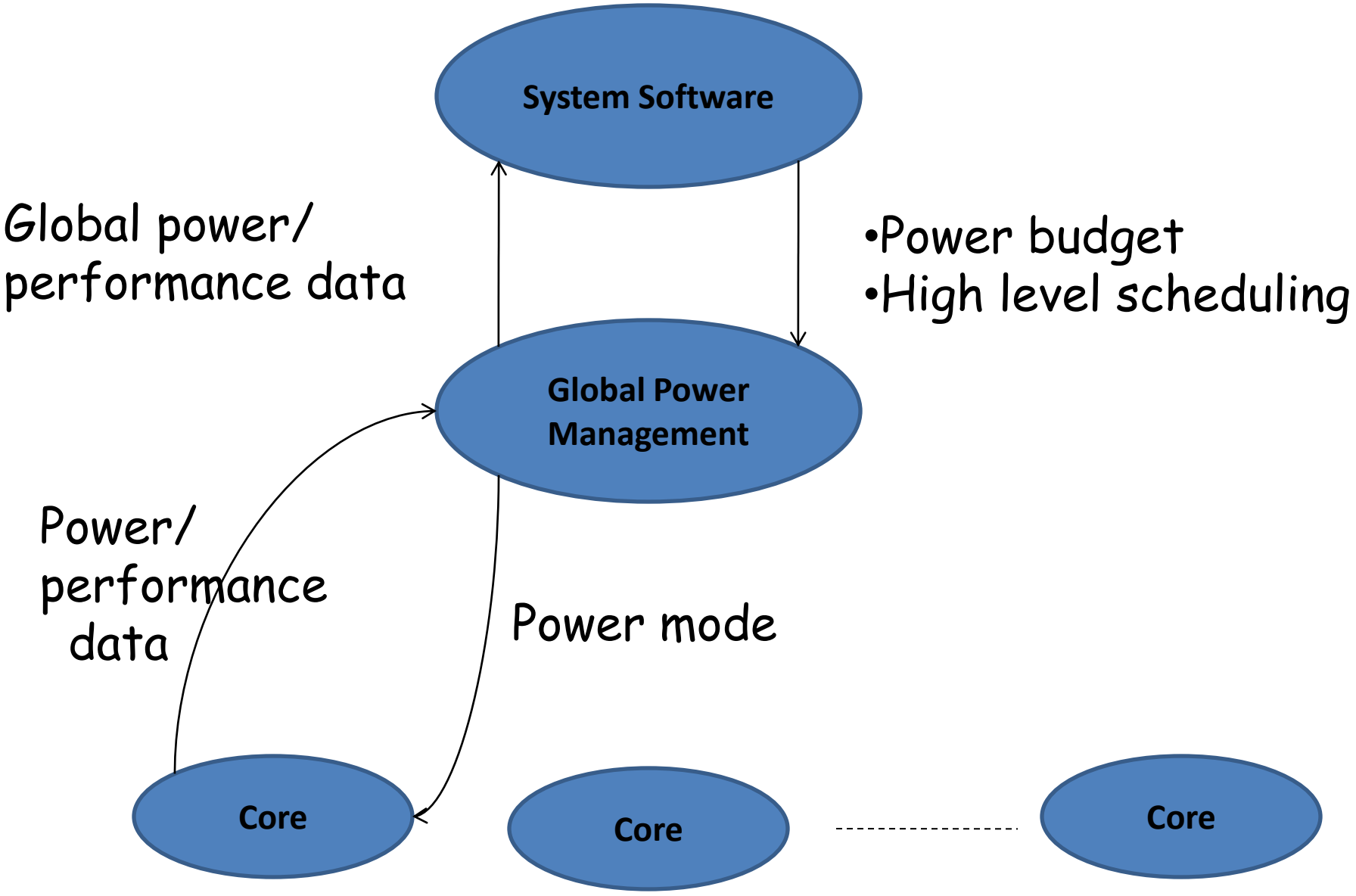
Temperature -> Hot Spot



applu benchmark on a single core (source: Kevin Skadron Tutorial in ISCA'04)

What To Do About Dynamic Power

- DFVS
  - At OS level
    - idle time represents energy waste
    - deadlines for interactive programs
  - Offline compiler analysis
    - insert mode-set instructions
    - depends on program phases
    - lowers the voltage for memory-bound sections
  - Online dynamic compiler analysis
    - phase detection
    - binary instrumentation
- Reducing switching activity


System Software
Global power/
performance data
•Power budget
•High level scheduling
Global Power
Management
Power/
performance
data
Power mode
Core
Core
Core

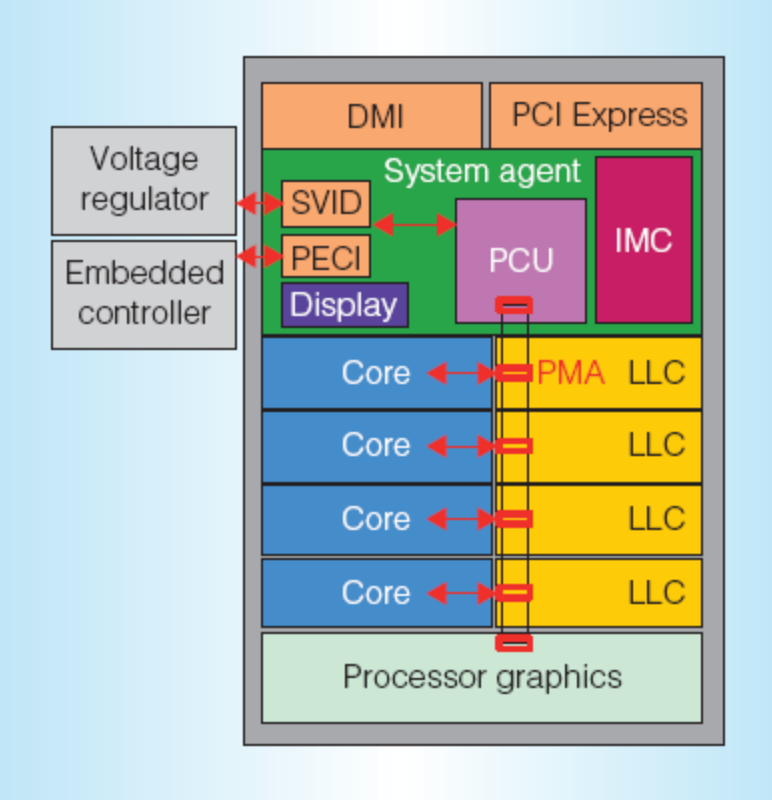
What To Do About Leakage?

- Stacking transistor
- Dynamically resized caches (mainly I-caches)
  - gated Vdd
  - Non-state-preserving
- Drowsy caches
  - Scale supply voltage to reduce leakage

What To Do About Temperature?

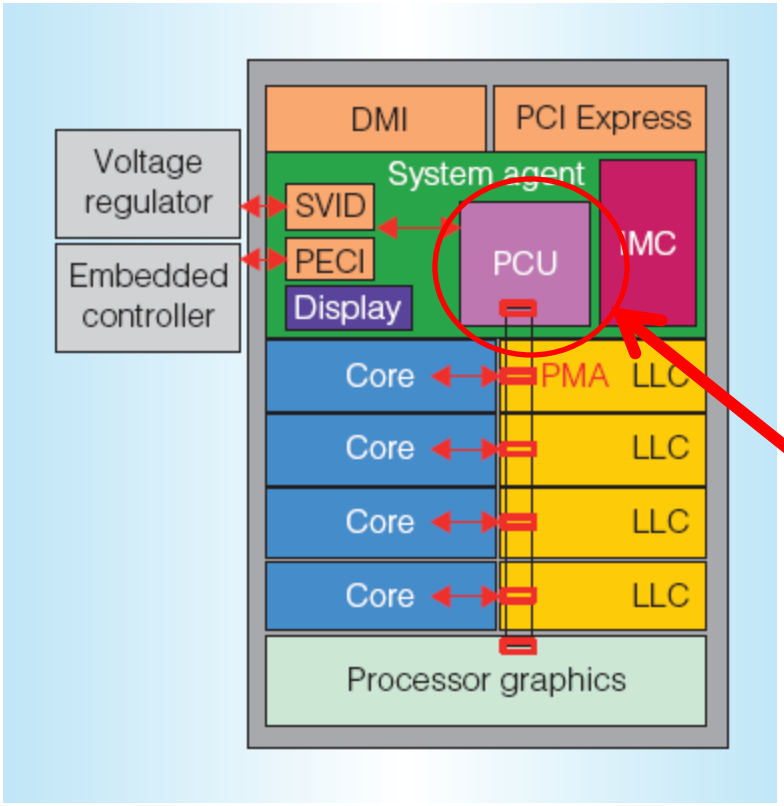
- Better sensors position
- Predicting temperature at places without sensors
- Avoid hot spots
- Must be taken care of from design-time

Real-Life Example: SandyBridge



PMA: Power Management Agent
PCU: Package Control Unit
DMI: Direct Media Interface
SVID: Serial Voltage ID
PECI: Platform Environment Control Interface
IMC: Integrated Memory Controller

Real-Life Example: SandyBridge

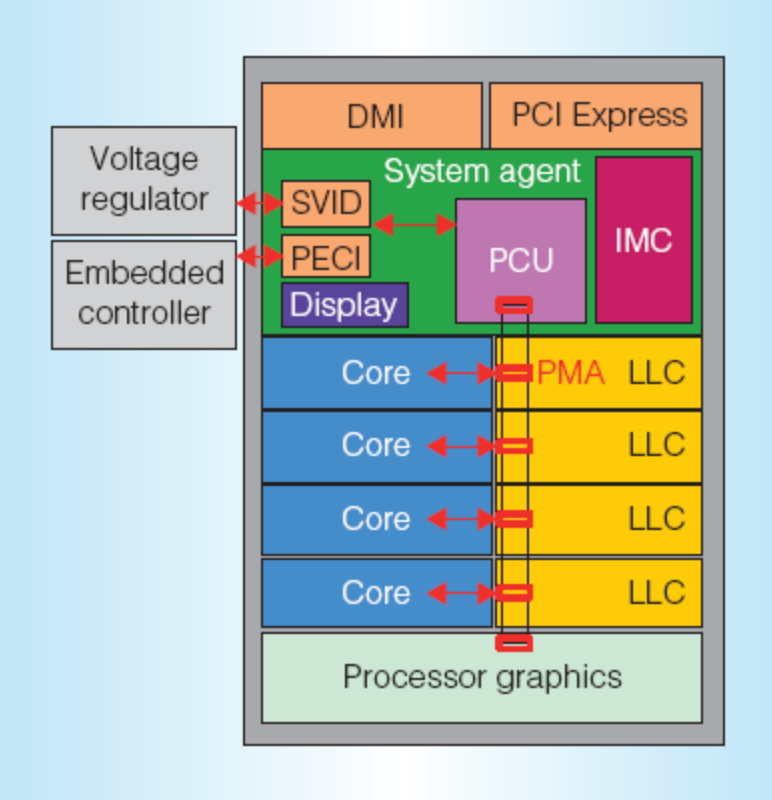


Two independent power planes:

- CPU cores, LLC, and ring
  - Each core can be turned off indept.
  - Portion of the LLC can be tuned off
- GPU

•On chip logic and embedded controller running power management firmware
•Communicates internally with cores, ring
•Monitors physical conditions Voltage, temperature, power consumption
•Controls power states CPU and GPU voltage and frequency

Real-Life Example: SandyBridge



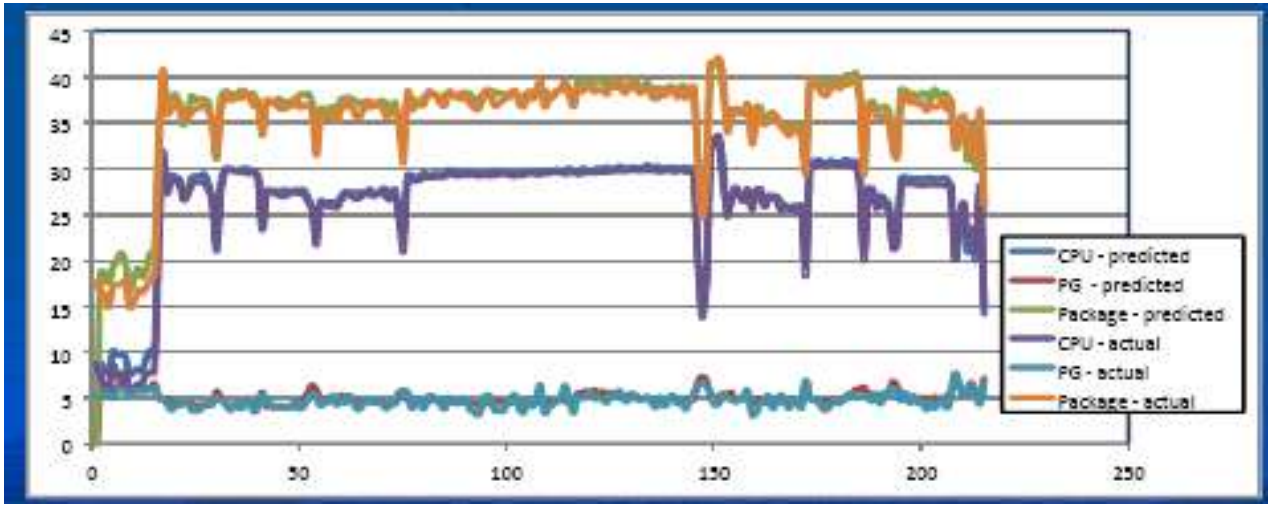
Power Performance Management Is:

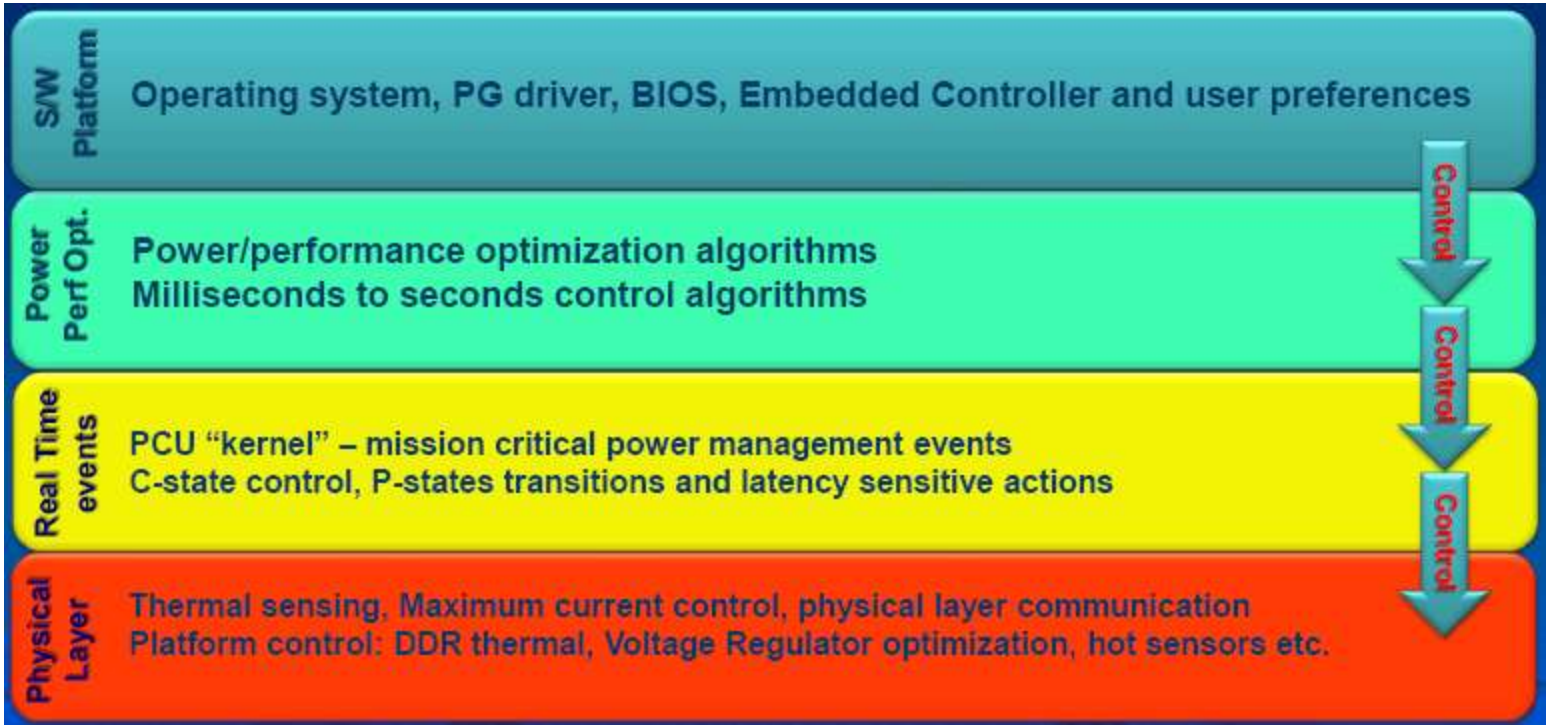
Enhance User Experience:

- Throughput performance
- Responsiveness - burst performance
- CPU / PG performance
- Battery life / Energy bills
- Ergonomics (acoustic noise, heat)

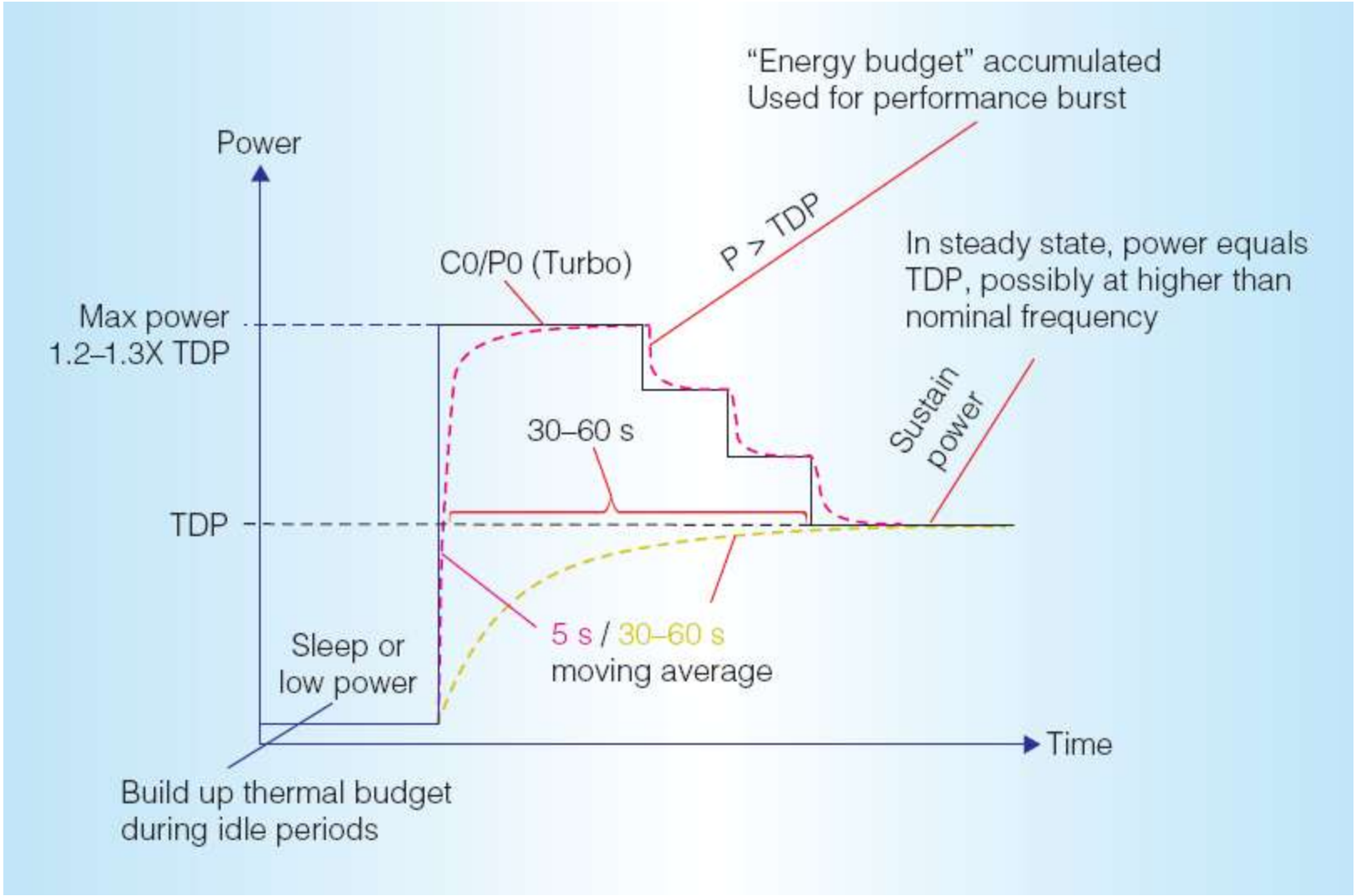
Given Physical Constraints:

- Silicon capabilities
- System Thermo-Mechanical capabilities
- Power delivery capabilities
- S/W and Operating system explicit control
- Workload and usage

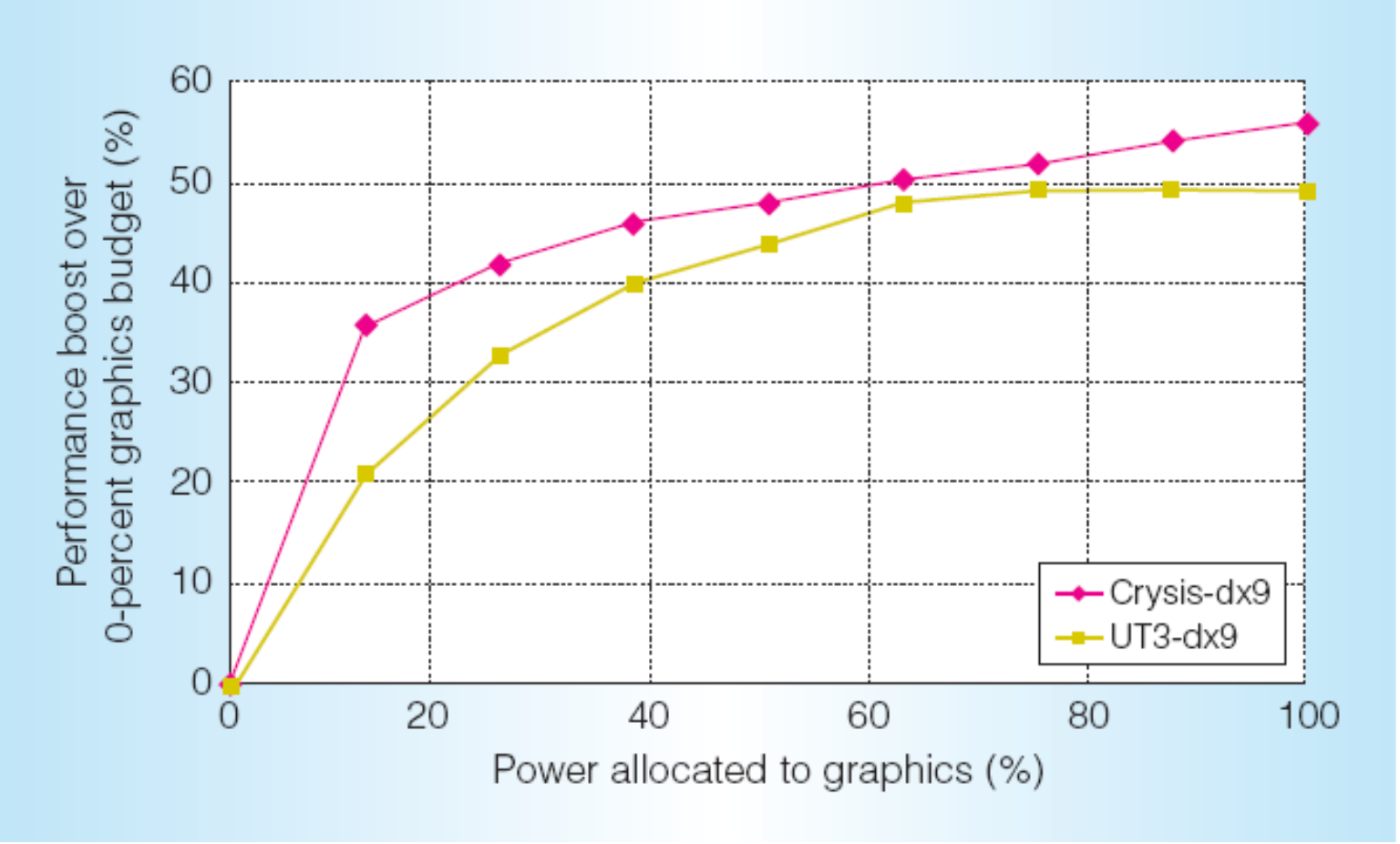

45
40
35
30
25
20
15
10
5
0
0
50
100
150
200
250
CPU - predicted
PG - predicted
Package - predicted
CPU - actual
PG - actual
Package - actual


S/W Platform
Operating system, PG driver, BIOS, Embedded Controller and user preferences
Power Perf Opt.
Power/performance optimization algorithms
Milliseconds to seconds control algorithms
Real Time events
PCU "kernel" – mission critical power management events
C-state control, P-states transitions and latency sensitive actions
Physical Layer
Thermal sensing, Maximum current control, physical layer communication
Platform control: DDR thermal, Voltage Regulator optimization, hot sensors etc.
Control
Control
Control

Intel Turbo Boost in SandyBridge



CPU-GPU Interaction



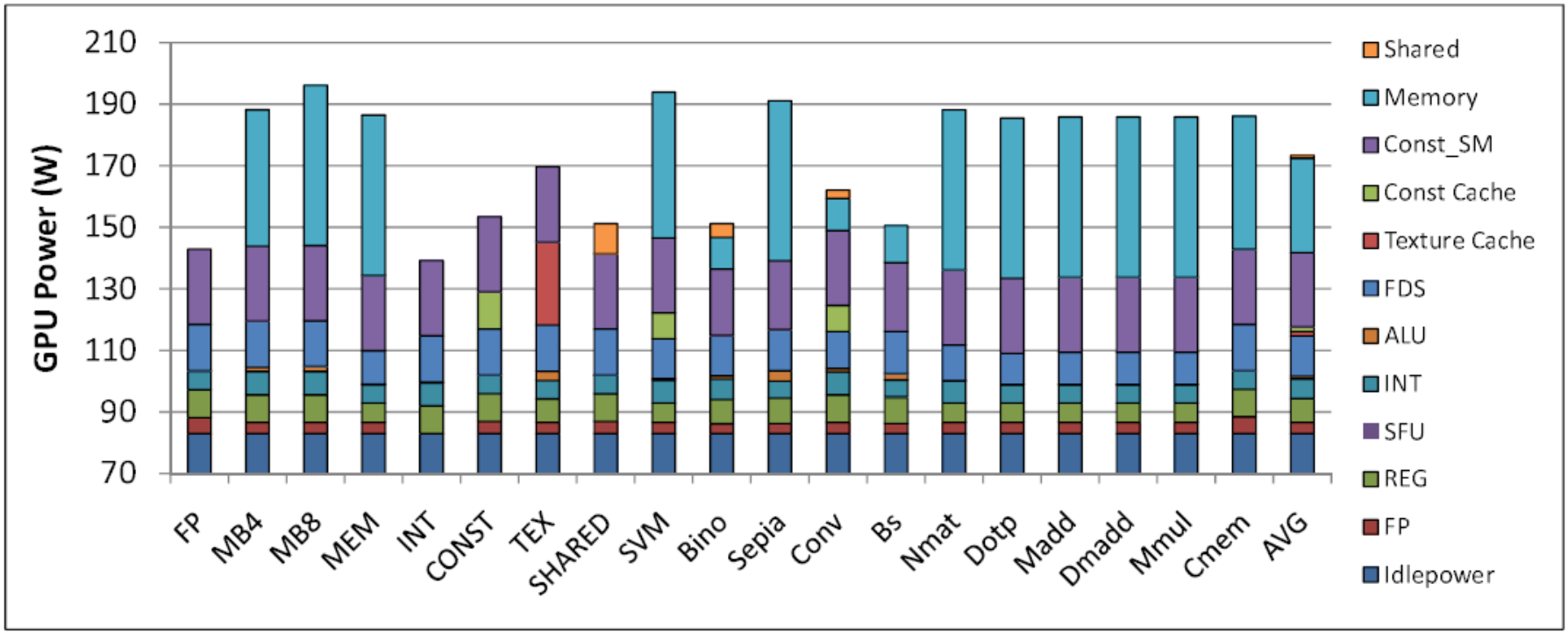
Power-Aware Software!!
Is It for Real?

What Can A Software Application Do?

- Use less expensive operations
- Less stress on power-hungry parts
- Access and make use of internal GPU performance counters
  - PAPI
  - nvidia-smi
- Pass power-related info to OS
- Interaction of three players:
  - The application software
  - The Compiler
  - The OS

Power-Aware Applications

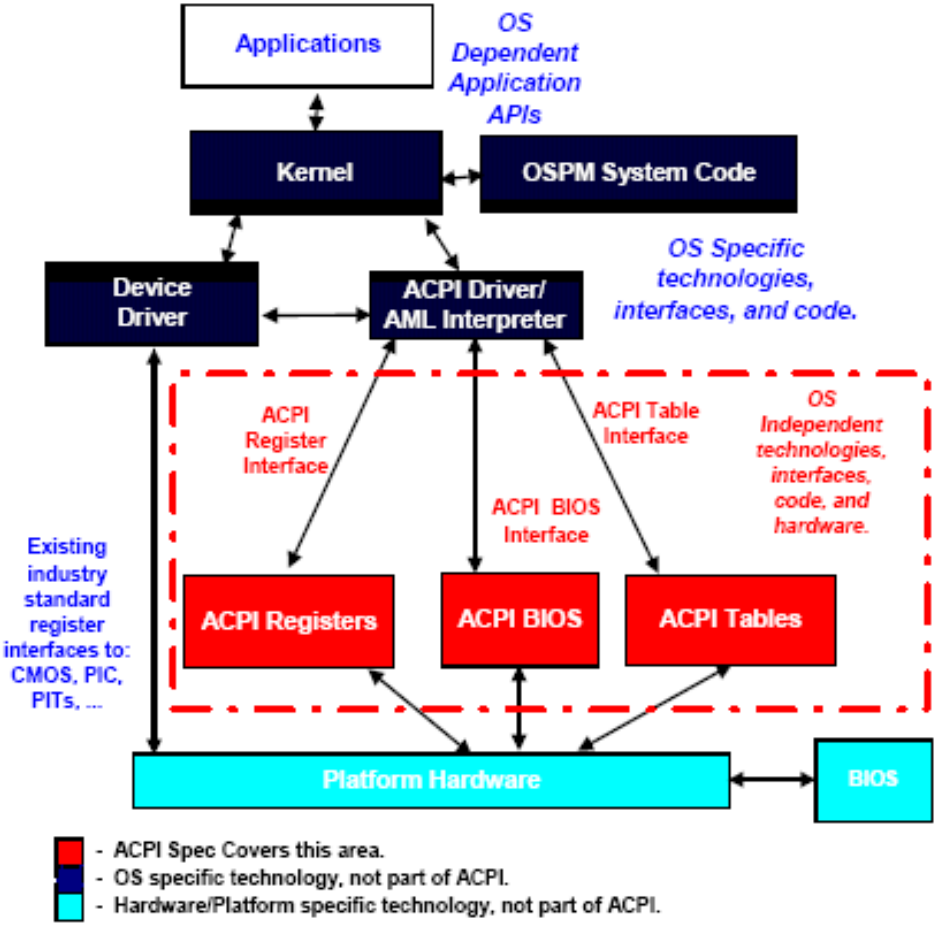
- Applications must be Designed and tested for power management
- Applications must handle sleep transitions seamlessly
- You can differentiate your application with power management features
  - Handle power management events
  - Scale behavior based on user's power preference
- OS provides APIs



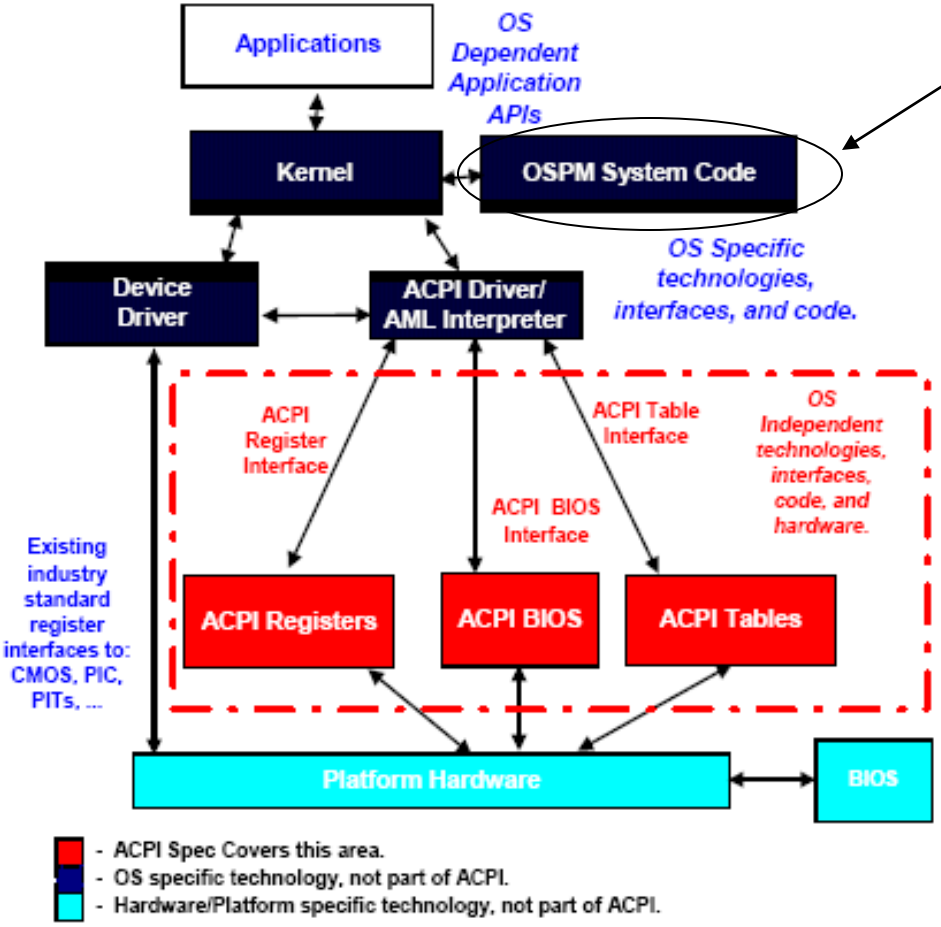
Power Breakdown: GeForce 285 GTX

Advanced Configuration and Power Interface (ACPI)

- An open industry specification co-developed by Hewlett-Packard, Intel, Microsoft, Phoenix, and Toshiba.
- Devices must support power saving modes
- ACPI must be supported by the computer motherboard, BIOS, and the operating system
- Power management platform at the hardware level
- Establishes industry-standard interfaces for OS-directed configuration and power management on laptops, desktops, and servers.



http://www.acpi.info


Applications
OS Dependent Application APIs
Kernel
OSPM System Code
OS directed Power Management
OS Specific technologies, interfaces, and code.
Device Driver
ACPI Driver/ AML Interpreter
ACPI Register Interface
ACPI Table Interface
OS Independent technologies, interfaces, code, and hardware.
ACPI BIOS Interface
Existing industry standard register interfaces to: CMOS, PIC, PITs, ...
ACPI Registers
ACPI BIOS
ACPI Tables
Platform Hardware
BIOS
- ACPI Spec Covers this area.
- OS specific technology, not part of ACPI.
- Hardware/Platform specific technology, not part of ACPI.

Conclusions

- Power is not longer to be neglected, both static and dynamic, for both CPU and GPU.
- Power-wall cannot be dealt with at one level, but requires cooperation from algorithms to circuits.