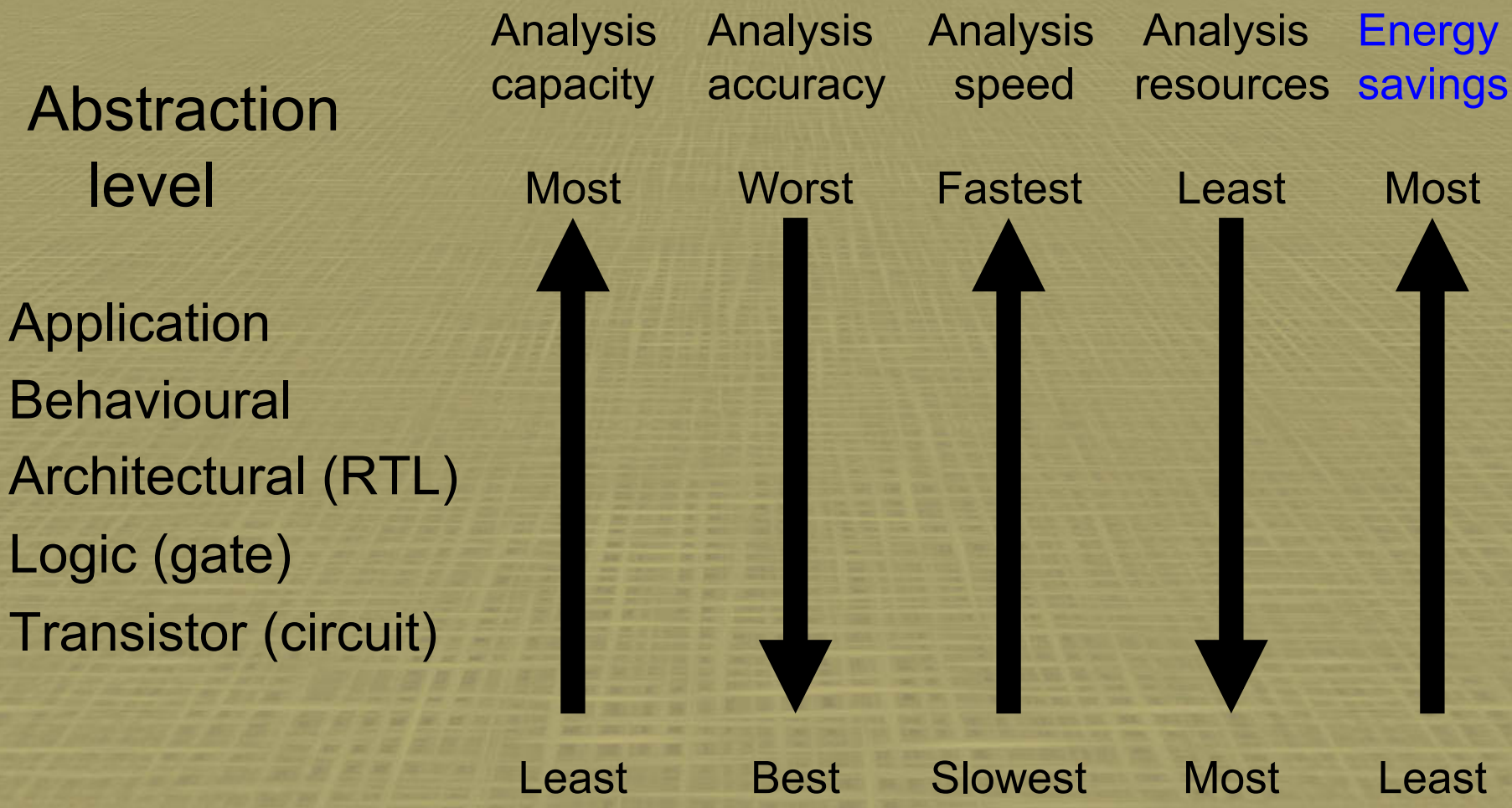
Energy-Aware Computing

Lecture 3:Modelling for power/energy

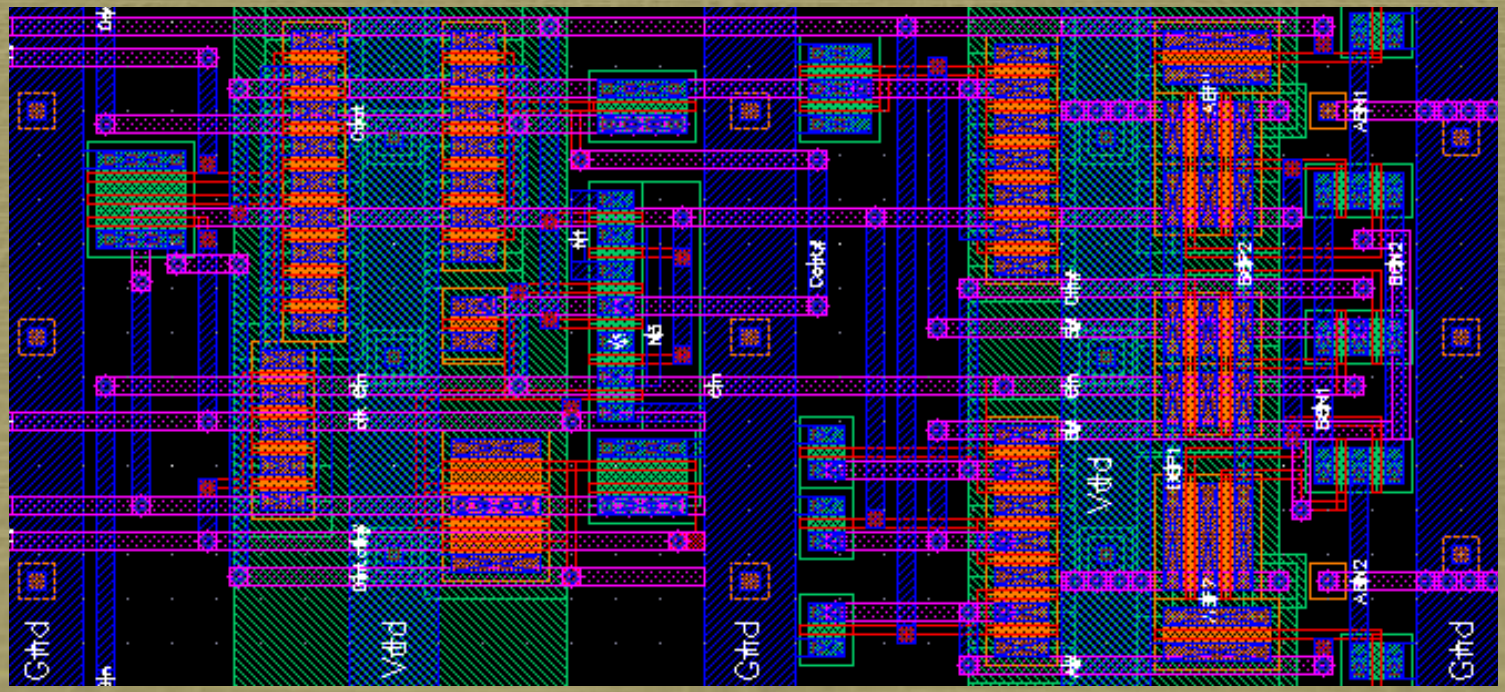
Analysis abstraction levels



Source: MJI&VP ISCA'00 tutorial

Transistor-level simulation

- Extract circuit *netlist* from actual *layout*
  - Various degrees of detail available
  - Extraction is very time consuming



Transistor-level simulation

- Simulate
  - Library of detailed models of basic components characterised by the chip foundry
  - Solve current/voltage equations for all circuit nodes
  - The best accuracy possible with simulation
- Input data dependent
  - Must use typical values
- *Spice* like tools: Hspice, spectre
- Table look-up tools: NanoSim (PowerMill)
  - An order of magnitude faster
  - 15hr for a self-timed ARM9 CPU (0.8M tran) running 2 iterations of Dhrystone on a Sun Ultra 5

Gate-level simulation

- The netlist contains logic gates as its components
  - Written in structural Verilog or VHDL
  - Extraction is very simple / fast
- Logic simulation
  - Determine switching activity of each wire
- Internal gate power estimated using characterised models of all logic gates
- Capacitance of interconnect is estimated
  - Number of connections, historical data from existing designs

Gate-level simulation

- Tools: e.g. Power Compiler (Synopsys)
- Typical flow:
  - Perform logic design and optimisation
  - Create a file containing a list of nodes for simulator to count transitions
  - Perform simulation. Output file containing # transitions per node
  - Calculate pwr/energy from simulation output, capacitance estimation and logic gate internal power info
  - Re-optimise the design based on power results
  - Iterate until target reached or can’t improve further

Architectural-level simulation

- Cycle/event based simulation
  - Activity modelled at a clock-cycle granularity
- Instruction based simulation
  - Activity modelled at instruction granularity
- Simulation provides information on activity at the input of each major functional unit
- Switched capacitance of functional units is estimated/characterised

Functional unit power estimation

- Analytical energy/power models
  - Typically used for caches, register file and other regular array structures
  - E.g. the cacti tool for caches
- Transition sensitive models
  - Typically used for ALUs, system busses, “random logic”
- Calibration using low-level simulation
  - But this is highly dependent on sim input values

Architectural simulation

- The simulator is just an application running on a *host* machine
- Emulates the execution of a program on a *target* processor
- Target processor's system calls, e.g. file input/output are (eventually) handled by host machine
- Simulation time is different to wall clock time!
  - Delays of functional units are built into the simulator, which reports the total simulation time at the end

Simulators

- SimpleScalar is a widely used performance-only simulator
- Wattch - an add-on to SimpleScalar, models dynamic power
- Hot Leakage - an add-on to Wattch, includes leakage power modelling

Effective use of simulators

- Spend time to learn how to build, simulate and hack the tools!
- Simulations take very long time. Learn how to run a batch of simulations with different benchmarks or configurations and process the outputs automatically
  - Use scripts: Perl, Python, Unix shell

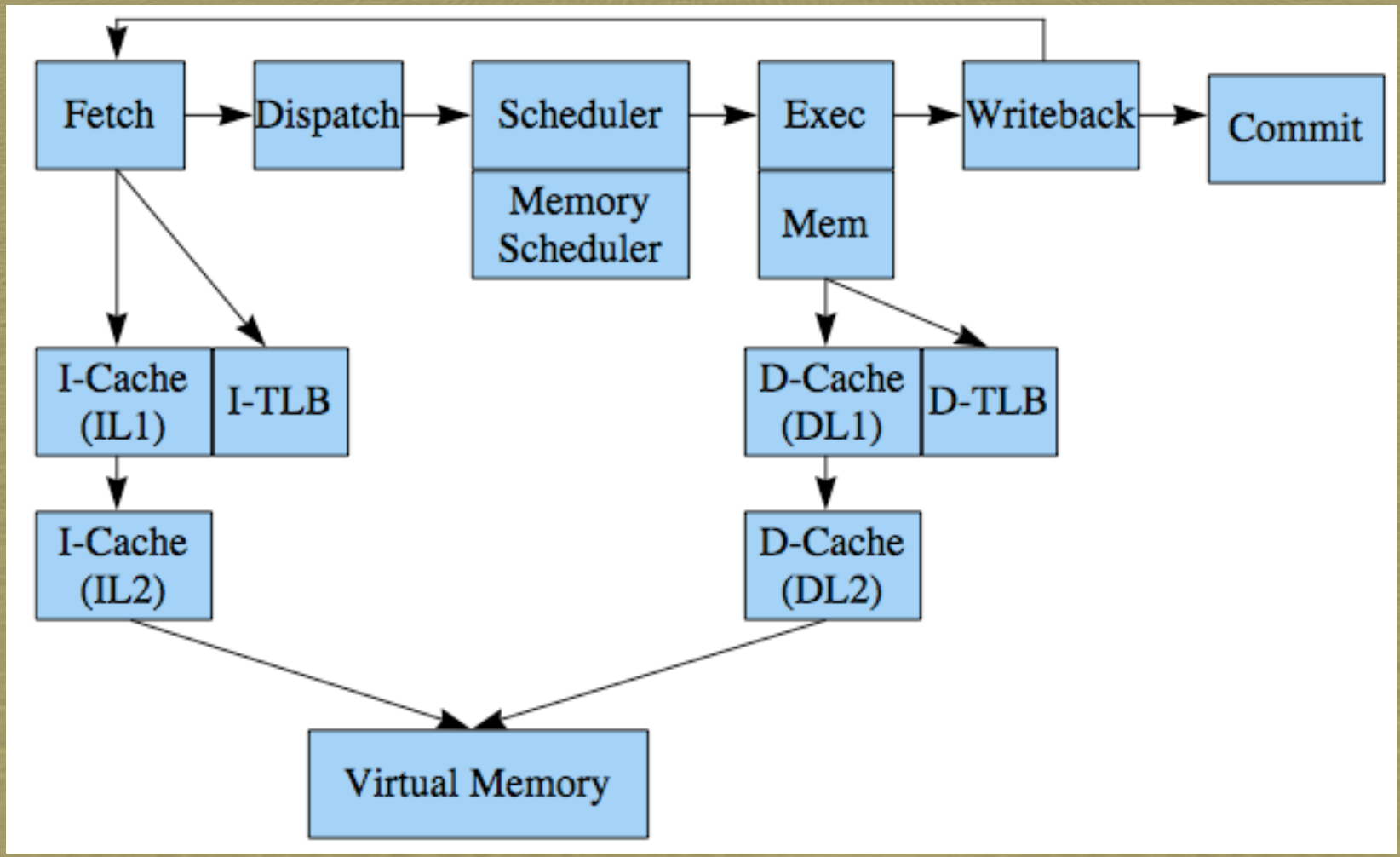
Other simulators

- System level simulators have become available recently
  - Simics, Bochs, …
  - You can boot Linux, Windows, etc on them!
- They do not currently provide power/ energy information
- Lot's more tools available. See:
  - www.cs.wisc.edu/arch/www/tools.html

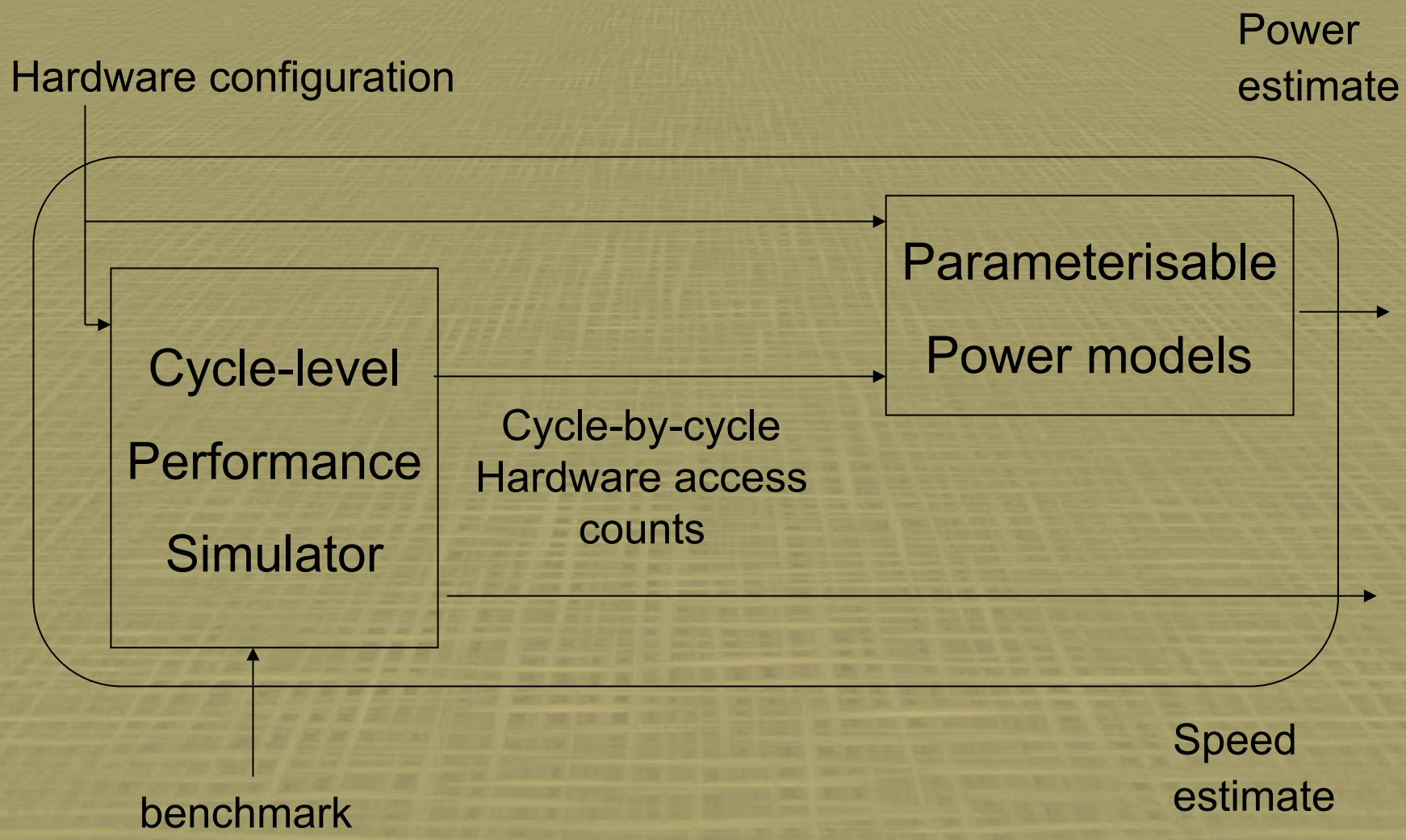
Overview of simplescalar

- Simplescalar is a suite of simulators of varying detail
- Target ISAs
  - PISA - very similar to MIPS
  - Alpha - supported by Wattch
- We use the most detailed one, sim-outorder
  - Superscalar with out of order issue
- Read the “simplescalar hacker’s guide”

Simplescalar architecture



Wattch overview



Methodology

- Know how to measure performance
  - Metrics for speed (IPC, cycle count, cache hit ratio)
  - Power, energy
    - Watch out for weird units, terminology mix-up, etc.
  - Combined metrics
    - PDP, EDP, ET^2

Methodology - benchmarks

- Which programs to run?
  - e.g. SPEC, mediabench
- How many?
  - All! As many as you have time for
- How are the programs compiled?
  - Report compiler used, optimisation flags, …
- Which inputs?
  - SPEC has reference, test inputs
- When to measure?
  - Leave some warm-up time at the beginning
  - How many instructions to simulate? ~200 Million is typical

Methodology - configurations

- Use realistic configurations for memory sizes (caches, predictors, reg. file, …), timings (#cycles for a D$1 miss), etc.
- Absolute performance metrics are less reliable than relative ones
  - Systematic errors in the model are removed
- Baseline processor - the standard against which the comparison is made
  - Model and measure even if results already published

Summary

- Levels of abstraction and relationship with accuracy, speed, flexibility
- Architectural level simulation
  - Simplescalar
  - Wattch
- Methodology