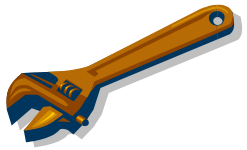
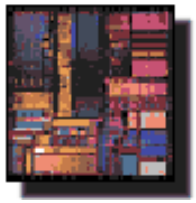






ECE 595Z
Digital Systems Design Automation

Module 9 (Lectures 29-30): Low Power Design
Lecture 30



Anand Raghunathan

MSEE 348

raghunathan@purdue.edu

Lecture 29: Re-cap

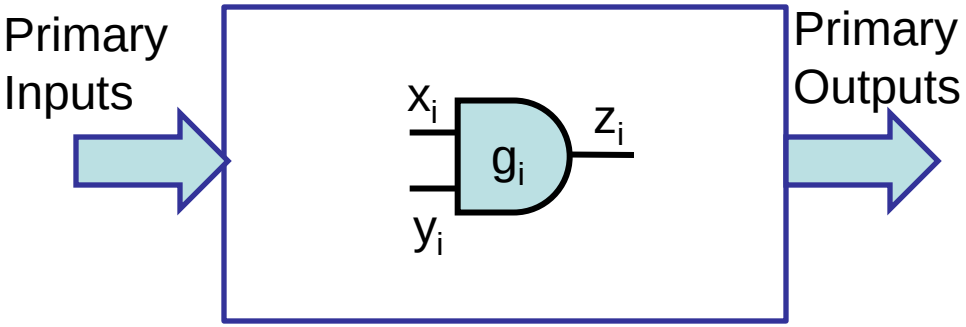
- Virtually every IC design today is impacted by power consumption
- Role of design automation
  - Power estimation
  - Power reduction
- Power estimation at the logic level
  - Power depends on values and transitions at signals in the circuit

Approaches to Power Estimation

- **Simulation-based**
  - Given (user-provided) test benches
    - Simulate the circuit (modeling gate delays)
      - Online: Evaluate power models during simulation
      - Offline: Record switching activities and signal probabilities during simulation and post-process to estimate power
- Problem: Long test benches may be necessary, can be very slow
- Solutions
  - Statistical sampling of input traces
    - Still requires simulation of shorter traces
  - Probabilistic analysis of circuits
    - Static (does not require simulation)

Approaches to Power Estimation

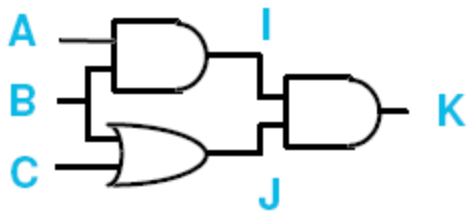
- **Probabilistic analysis**
  - Compute "average" switching activities and value probabilities at internal signals in the circuit
  - Evaluate power models
- Advantage: No simulation needed (fast), no need for test bench



$$N_{z_i} = \frac{\sum_{\text{all } v_1, v_2} N_{z_i}(v_1, v_2)}{2^{\#inputs} \times 2^{\#inputs}}$$

Value Probabilities

- Define P_x as the probability that signal x=1
- Value probability at a gate output can be computed using probabilities at its inputs



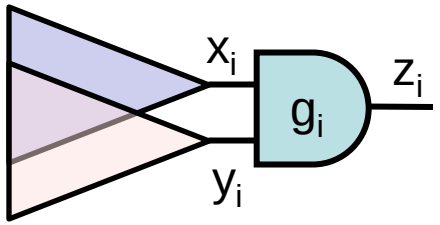
AND gate: $P_I = P_A * P_B$
OR gate: $P_J = P_B + P_C$

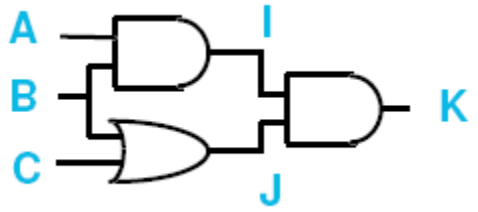
Assumption:

A, B, C are uncorrelated (we will re-visit this since it is not valid for sequential circuits)

Value Probabilities: Spatial Correlation

- In general, the inputs to a gate may be correlated
  - Failure to account for the correlation will lead to an incorrect estimate for the probability at the gate output





I and J are correlated since both depend on B

$P_K \neq P_I * P_J$

Computing Value Probabilities Considering Spatial Correlation

- Procedure for computing value probability considering correlations
  1. Write a Boolean expression for the signal in terms of primary inputs
  2. Convert the expression into a disjoint SOP expression (cubes are pair-wise disjoint, *i.e.*, intersection is NULL)
  3. Compute the output probability for the disjoint SOP expression

$F = ab + bc$

$F = ab + a'bc$

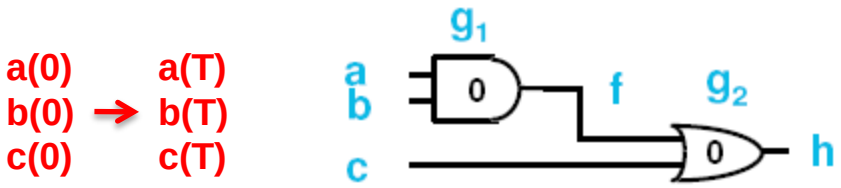
$P_F = P_a*P_b + P_b*P_c$

$P_F = P_a*P_b + (1-P_a)*P_b*P_c$

Disjoint cubes can be obtained from a _____ representation

Computing Switching Activities Considering Gate Delays and Correlations

- First, consider zero-delay model
  - All gate outputs switch instantaneously after input vector is applied



Switching functions for f and h:

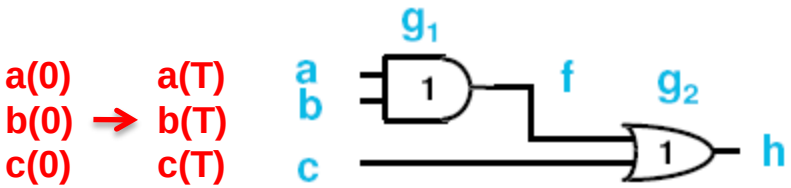
$S_f(t=T) = f(t=0) \oplus f(t=T) = (a(0)\, b(0)) \oplus (a(T)\, b(T))$

$S_h(t=T) = h(t=0) \oplus h(t=T) = (a(0)\, b(0) + c(0)) \oplus (a(T)\, b(T) + c(T))$

Under the zero-delay model, switching activity at a gate output is just the probability of the switching function evaluating to True

Computing Switching Activities Considering Gate Delays and Correlations

- Now, consider general delay model



f(0) = a(0) b(0)
f(T) = a(0) b(0)
f(T+1) = a(T) b(T)

h(0) = f(0) + c(0)
h(T) = f(0) + c(0)
h(T+1) = f(T) + c(T)
h(T+2) = f(T+1) + c(T)

Switching functions

$S_f(1) = f(T) \oplus f(T+1)$

$N_f = P(S_f(1))$

$S_h(1) = h(T) \oplus h(T+1)$
$S_h(2) = h(T+1) \oplus h(T+2)$

$N_h = P(S_h(1)) + P(S_h(2))$

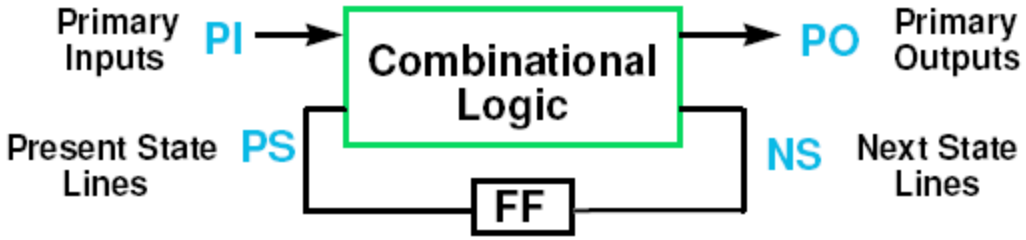
Switching activity for a signal is the sum of all its switching function probabilities

Computing Switching Activities Considering Gate Delays and Correlations

- For each gate
  - Enumerate potential times at which gate can switch.
  - For each potential switching time
    - Construct gate switching function
      - Boolean expression that represents conditions for gate switching
      - Function of current and previous input vectors
    - Compute probability of the gate switching function evaluating to True
  - Compute switching activity as sum of all switching function probabilities

Sequential Circuits

- Two additional challenges
  - Need to compute probabilities at present state lines
  - Account for temporal correlation between values appearing at PS lines in one cycle and the next
    - PS(**T**) = f (PS(**0**), PI(**0**))



Computing Present State Input Probabilities

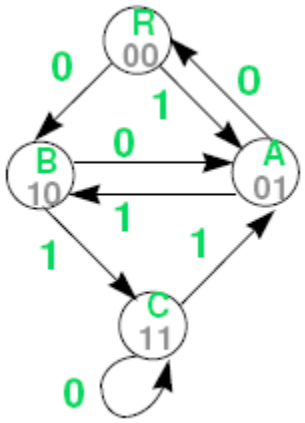
- **Exact method**
  - Extract the state transition graph (STG) from the circuit
  - Compute state probabilities by solving Chapman-Kolmogorov equations

$$prob(s_i) = \sum_m prob(s_m) \times prob(edge_{m \rightarrow i})$$

$$\sum_{i=1}^{K} prob(s_i) = 1$$

  - Use state probabilities and state encoding to compute present state input probabilities

Computing Present State Input Probabilities: Example

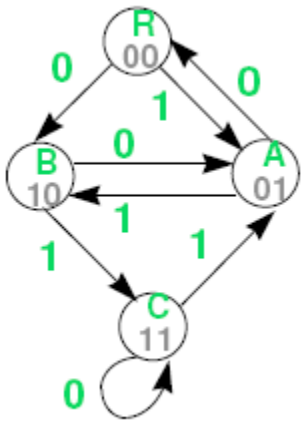


$$prob(R) = 0.5 \times prob(A)$$
$$prob(A) = 0.5 \times (prob(R) + prob(B) + prob(C))$$
$$prob(B) = 0.5 \times (prob(R) + prob(A))$$

$$prob(R) + prob(A) + prob(B) + prob(C) = 1$$

$$prob(R) = 0.167 \quad prob(A) = 0.333$$
$$prob(B) = 0.25 \quad prob(C) = 0.25$$

Computing Present State Input Probabilities: Example



prob(R) = 0.167 prob(A) = 0.333
prob(B) = 0.25 prob(C) = 0.25

R ≡ 00 A ≡ 01 B ≡ 10 C ≡ 11

prob(00) = 0.167 prob(01) = 0.333
prob(10) = 0.25 prob(11) = 0.25

$prob(ps_1 = 0) = prob(00) + prob(01) = 0.5$
$prob(ps_1 = 1) = prob(10) + prob(11) = 0.5$
$prob(ps_2 = 0) = prob(00) + prob(10) = 0.417$
$prob(ps_2 = 1) = prob(01) + prob(11) = 0.583$

Note that PS lines are correlated. For example:

$prob(ps_1 = 0) \times prob(ps_2 = 0) = 0.208 \neq prob(00)$

Computing Present State Input Probabilities

- STG is very large for most practical circuits
  - **Approximate method**: Compute probabilities at PS inputs independent of each other

$$
\begin{aligned}
ns_1 &= f_1(i_1, \ldots, i_M, ps_1, \ldots, ps_N) \\
ns_2 &= f_2(i_1, \ldots, i_M, ps_1, \ldots, ps_N) \\
&\ldots \\
ns_N &= f_N(i_1, \ldots, i_M, ps_1, \ldots, ps_N)
\end{aligned}
$$

$$
\begin{aligned}
prob(ns_1) &= prob(f_1(i_1, \ldots, i_M, ps_1, \ldots, ps_N)) \\
prob(ns_2) &= prob(f_2(i_1, \ldots, i_M, ps_1, \ldots, ps_N)) \\
&\ldots \\
prob(ns_N) &= prob(f_N(i_1, \ldots, i_M, ps_1, \ldots, ps_N))
\end{aligned}
$$

Computing Present State Input Probabilities

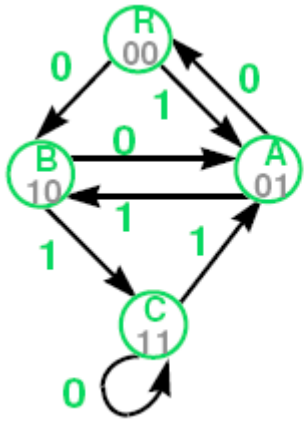
- Set of non-linear equations
  - Use iterative techniques (Newton-Raphson)

$$prob(ns_1) = prob(f_1(i_1, \ldots, i_M, ps_1, \ldots, ps_N))$$
$$prob(ns_2) = prob(f_2(i_1, \ldots, i_M, ps_1, \ldots, ps_N))$$
$$\ldots$$
$$prob(ns_N) = prob(f_N(i_1, \ldots, i_M, ps_1, \ldots, ps_N))$$

In steady-state:

$$prob(ps_i = 1) = prob(ns_i = 1) = p_i \quad 1 \leq i \leq N$$

Computing Present State Input Probabilities



Assume probability of input i is 0.5

$$ns_1 = \bar{i}\,\overline{ps_1}\,\overline{ps_2} + i\,\overline{ps_1}\,ps_2 + \bar{i}\,ps_1\,ps_2 + i\,ps_1\,\overline{ps_2}$$

$$ns_2 = i\,\overline{ps_1}\,\overline{ps_2} + ps_1$$

Express NS probabilities in terms of PS probabilities

$$p_1 = 0.5\,((1 - p_1)(1 - p_2) + (1 - p_1)\,p_2 + p_1\,p_2 + p_1\,(1 - p_2))$$

$$p_2 = 0.5\,(1 - p_1)(1 - p_2) + p_1$$

Solve

$$p_1 = 0.5$$

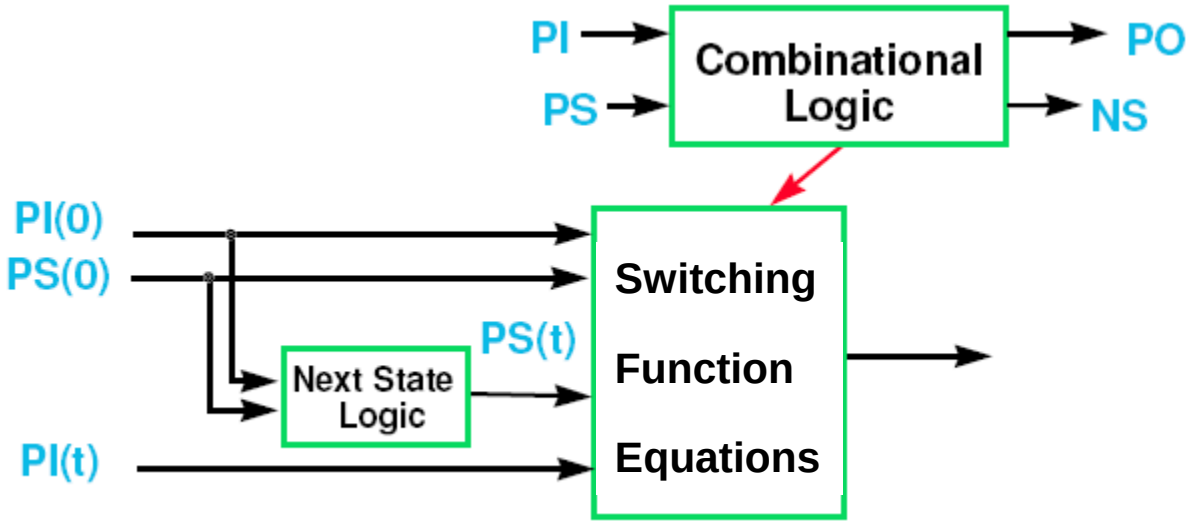
$$p_2 = 0.6$$

$p_1 = prob(ps_1 = 1)$

$p_2 = prob(ps_2 = 1)$

Modeling Temporal Correlation at Present State Inputs

- Create an additional copy of the next-state logic to feed the equations that represent the switching functions



Summary: Power Estimation

- Power estimation requires accurate computation of switching activities and value probabilities at internal signals
  - Simulation is the most commonly used approach, but often too slow
  - Probabilistic analysis leads to one-shot computation of all switching activities and value probabilities
    - Challenges: Spatial correlations due to re-convergent fanout within circuit, computing probabilities at present state inputs, temporal correlation at present state inputs
  - Alternative: Statistical sampling
- Both probabilistic and statistical techniques work well for large circuits, and are used in practice

Automatic Power Reduction

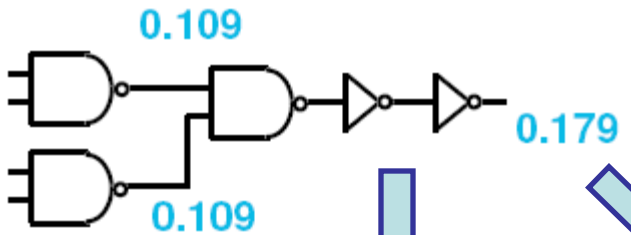
Outline

- Technology mapping for low power
- Clock gating
- Pre-computation
- Operand Isolation

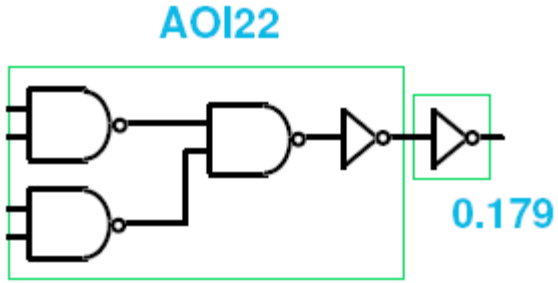
All of the above techniques reduce the switched capacitance in the circuit, and are hence complementary to frequency and supply voltage scaling

Technology Mapping for Low Power

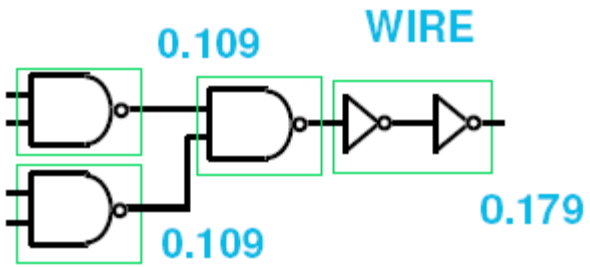
- Key idea: A desirable mapping results in high switching activity nets having low capacitance
  - Get "hidden" inside cells
  - Driven/loaded by smaller gates



Cell	Area	O/p Cap.	I/p Cap.
INV	928	0.1029	0.0514
NAND2	1392	0.1421	0.0747
AOI22	2320	0.3410	0.1033



Area =
Power =



Area =
Power =

Technology Mapping for Low Power

- Simple extension of mapping for area
  - Before mapping, compute switching activity at all signals in the subject graph
  - During mapping, use Power(match) = $\sum_{i \in pins(match)} C_i * N_i$

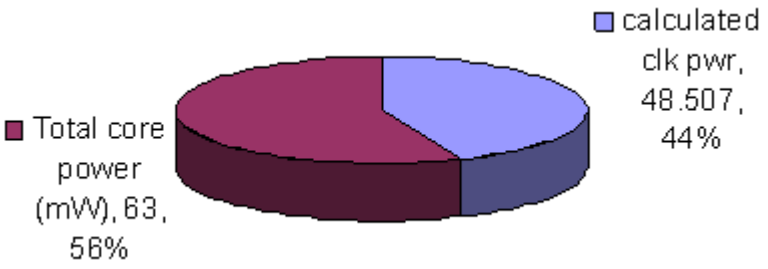
```
int min_power_map(v, P){
/* v is a vertex in the tree, P is the set of pattern graphs */
        best_cost = infinity;
        foreach(m = match(v, P)) {
            cost(m) = Power(m) + Σ_{vi ∈ inputs(m)} min_power_map(vi, P) );
            if(cost(m) < best_cost){
                match(v) = m;
                best_cost = cost(m);
            }
        }
        return best_cost;
}
```

Technology Mapping for Low Power

- Extensions
  - If C_i (power coefficients) depend on output load, use similar approach as technology mapping for delay
    - Discretize range of possible loads and compute best match for each
  - Considering leakage power:
    - Modify Power(m) to consider leakage in addition to switched capacitance
  - Multi-Vdd and multi-Vth mapping
    - Model as richer library

Clock Gating

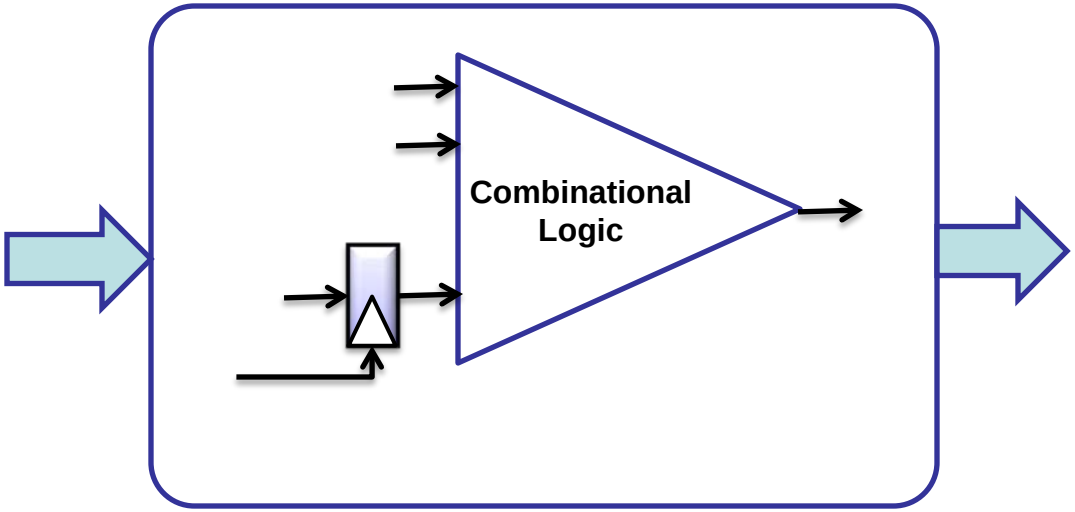
- Clock (distribution network + FF loads) can account for a significant portion of the total circuit's power
  - > 33% in a high-performance microprocessor
  - Huge capacitance (need to route all over the chip, buffers), high activity (2 transitions per cycle)



Power breakdown for LEON2 embedded processor core

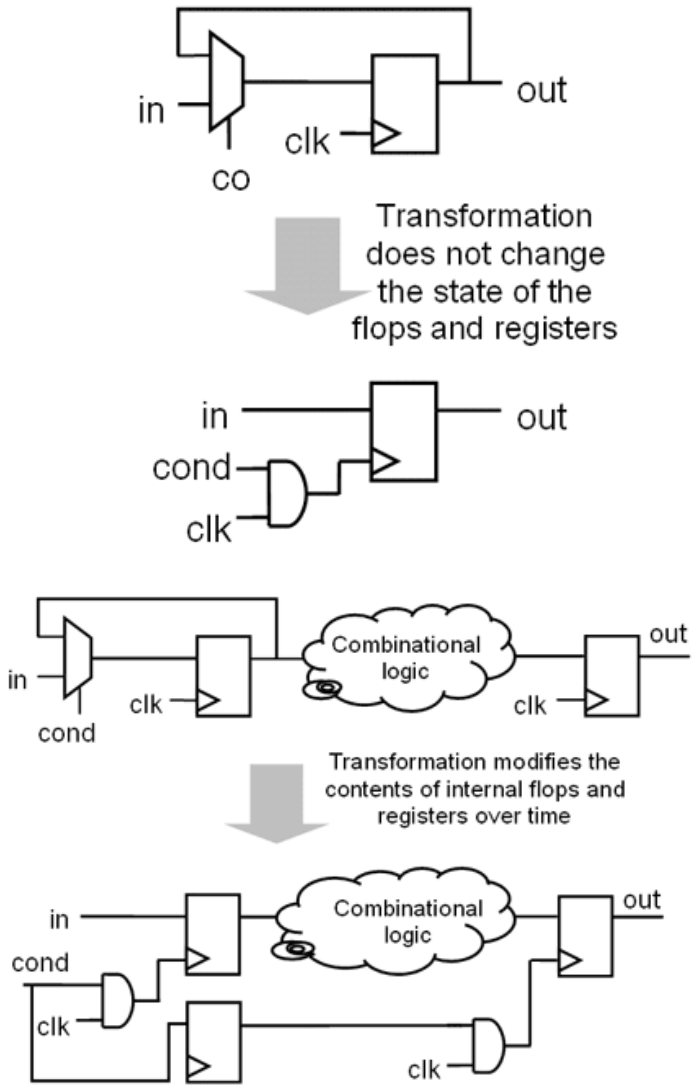
Clock Gating

- Basic idea: Suppress the clock signal from propagating through a part of the clock network
  - Question: When can you do this?



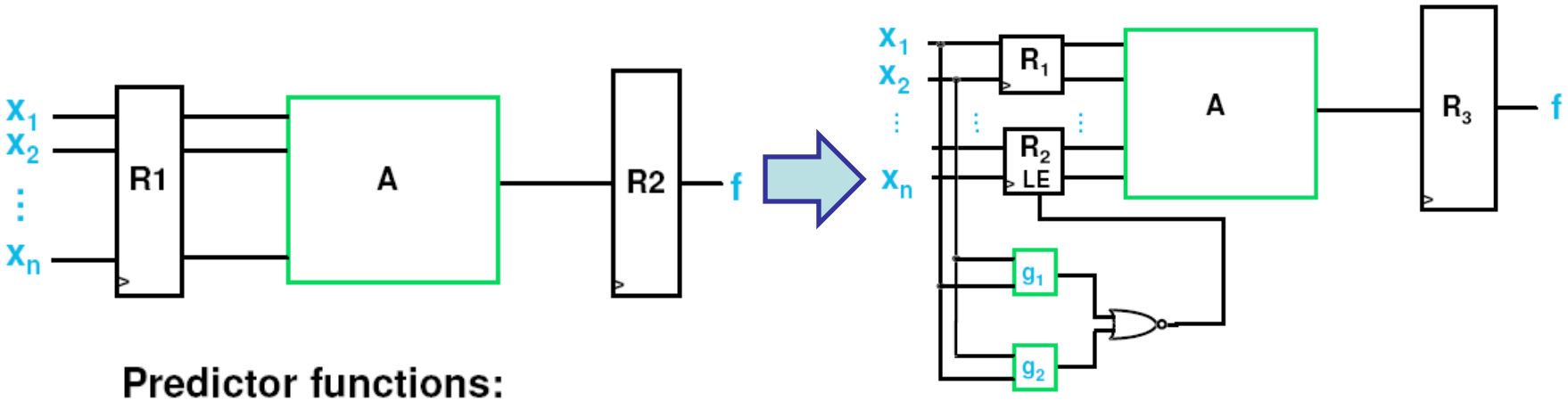
Clock Gating

- Two different approaches
  - $PS_i \oplus NS_i$ tells you when the output of the FF will change
  - Sequential ODCs for the output of a FF tell you when it's value is not useful
- Approximate (fast) method
  - Look for self-loops and derive sensitization conditions



Pre-computation

- Take clock gating to the next level
- Selectively pre-compute the output of the circuit (using simpler circuits) one cycle in advance, and disable the original circuit



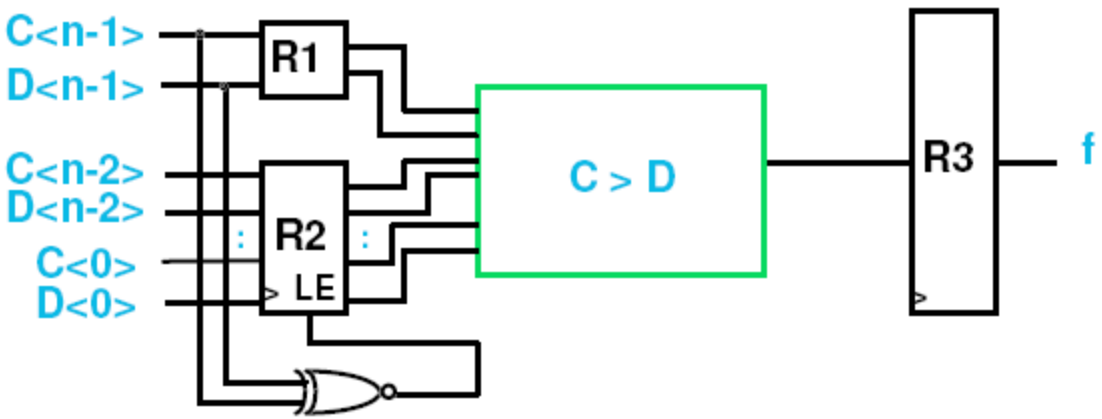
Predictor functions:

$g_1 = 1 \Rightarrow f = 1$

$g_2 = 1 \Rightarrow f = 0$

Example: Comparator

- MSBs alone can determine the output if they are unequal



$$g_1 = C\langle n-1\rangle \ \overline{D\langle n-1\rangle}$$

$$g_2 = \overline{C\langle n-1\rangle} \ D\langle n-1\rangle$$

Pre-computaton: Automatic generation of Conditions

- **Problem**: Given f(x1, …, xn) and k, the number of inputs to the pre-computation logic, determine:
  - The inputs S to the precomputation logic
  - The precomputation logic functions g1 and g2
  - Find S = { x[1], …, x[k] } that maximizes prob(g1(x[1], …, x[k]) + g2(x[1], …, x[k]) = 1)

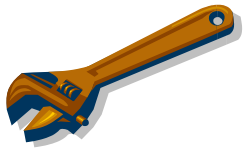
- Recall the Universal Quantification (consensus) of a function f with respect to a variable x
  - $\forall_x(f) = f_x \cap f_{x'}$
- Predictor functions can be generated as
  - $g1 = \forall_{xj[k+1]\ xj[k+2]\ldots\ xj[n]}(f)$
  - $g2 = \forall_{xj[k+1]\ xj[k+2]\ldots\ xj[n]}(f')$

Summary: Power Reduction

- Logic-level power reduction techniques focus on reducing the switched capacitance
  - Technology mapping for power
  - Automatic generation of clock gating, pre-computation logic, operand isolation logic
  - Commercial state-of-the-art: Technology mapping, automatic clock gating

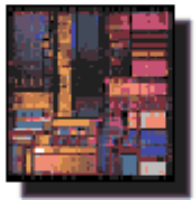






ECE 595Z
Digital Logic Synthesis :
Electronic Design Automation at the Logic Level

Selected Recent Developments / New Challenges



Anand Raghunathan

MSEE 318

raghunathan@purdue.edu

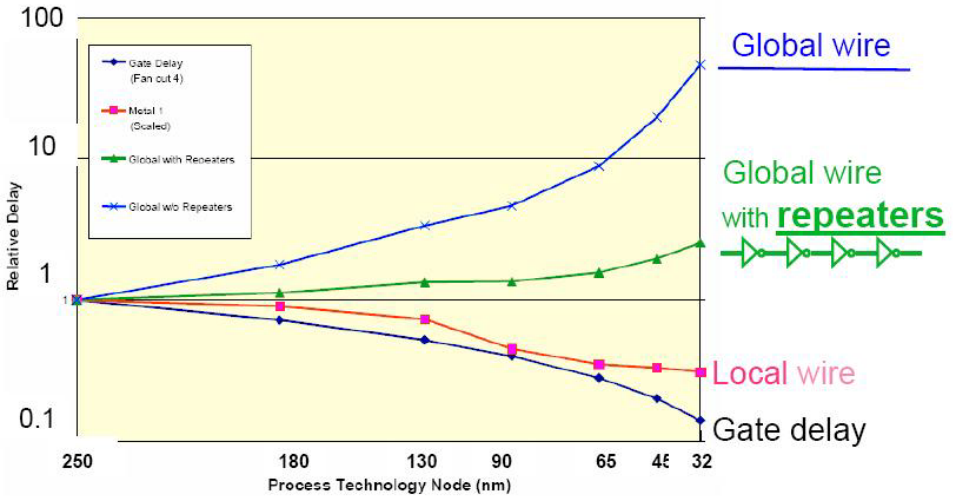
Outline

- Logic synthesis is a fairly mature area
- New challenges arise due to
  - Technology scaling trends
    - Increase in interconnect delay/power
      - Need for tighter integration with physical design, more accurate models
    - Process variations
  - Increase in circuit complexity
    - Saturation in clock frequencies of computing platforms that run tools
    - Need to parallelize algorithms

Interconnect delay/power trends

- Interconnect delay scales slower than gate delay
  - Global interconnects could become slower

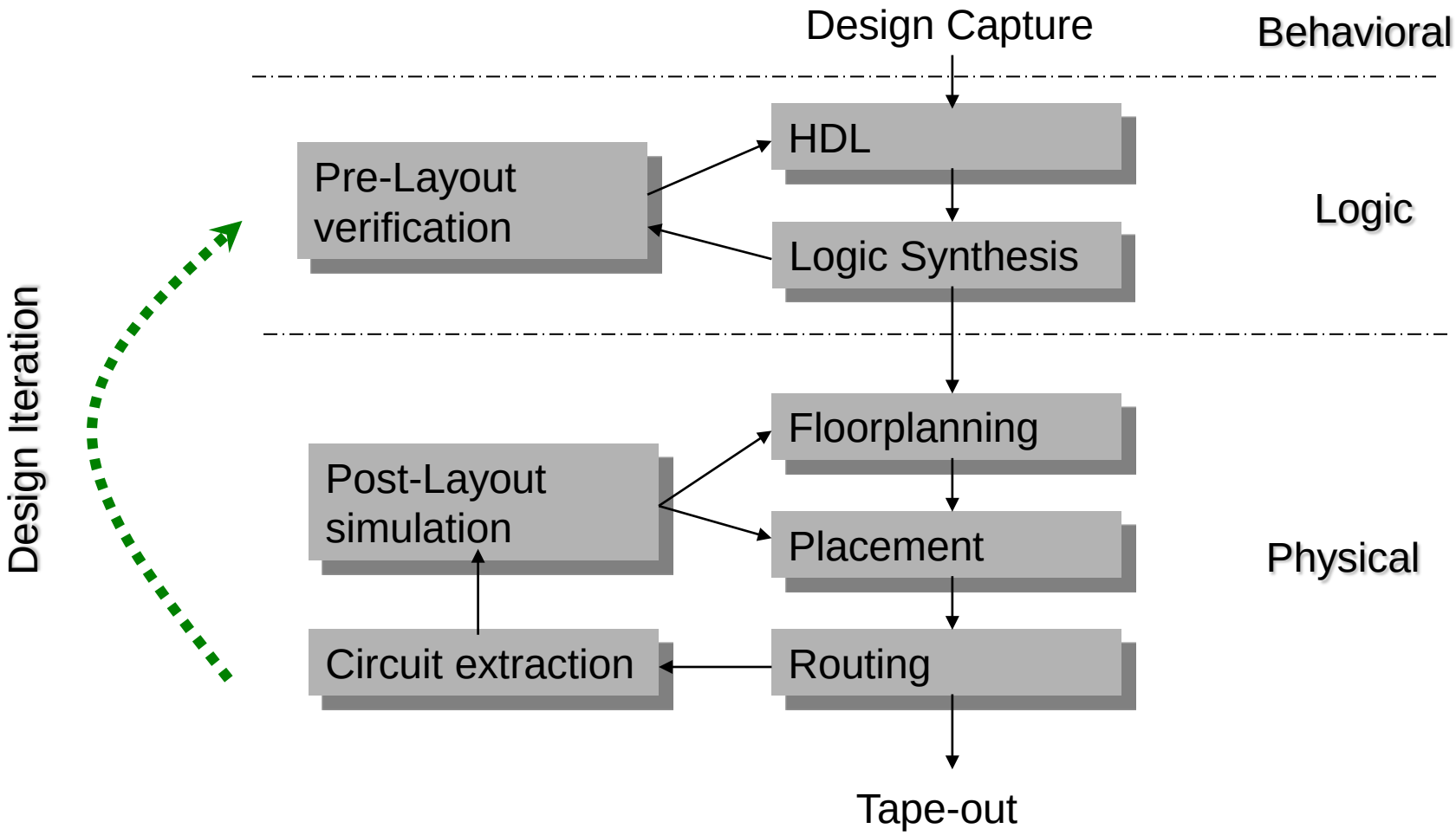
Technology (um)	*0.25*	*0.18*	*0.15*	*0.13*	*0.10*	*0.07*
Intrinsic gate delay (ns)	0.071	0.051	0.049	0.045	0.039	0.022
1mm (ns)	0.059	0.049	0.051	0.044	0.052	0.042
2cm no-opt (ns)	2.589	2.48	2.65	2.62	3.73	4.67
2cm best-opt (ns)	0.895	0.793	0.77	0.7	0.77	0.672



Delay for Metal 1 and Global Wiring versus Feature Size

Source: ITRS Roadmap

Traditional Design Flow: Separation of Logic and Physical Design

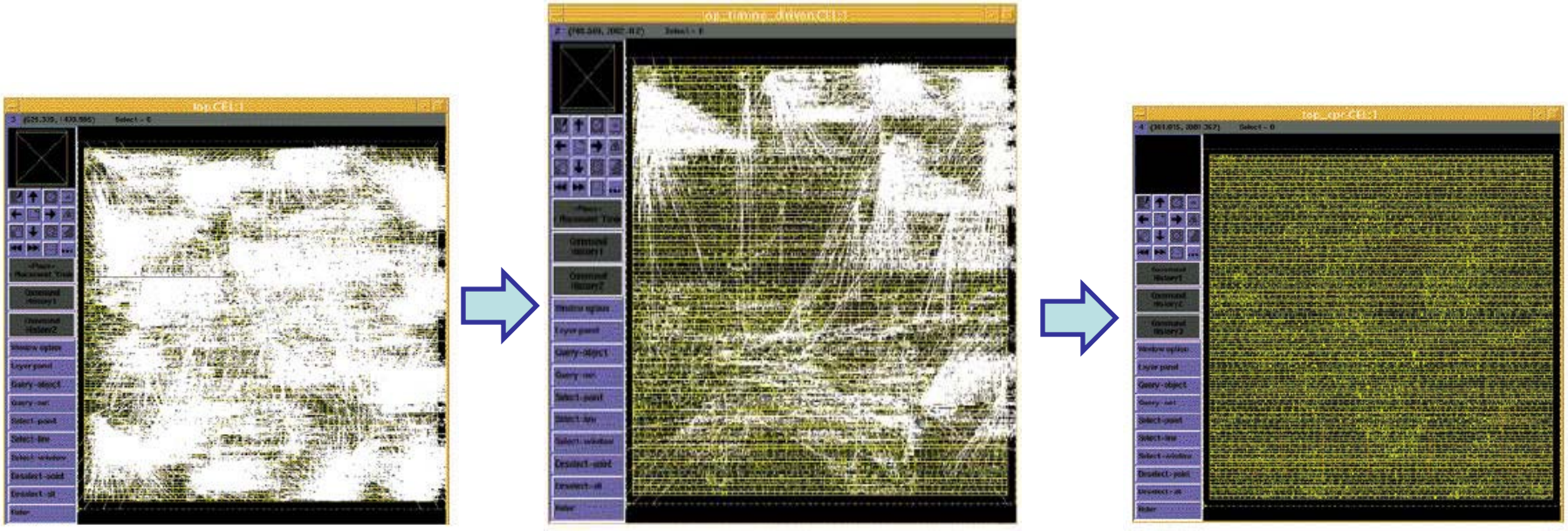


- **Earlier steps use approximate prediction models for interconnect (e.g., wire load models in logic synthesis)**

The Timing Closure Problem

- Sometimes, timing appears to be OK during earlier stages, but fails during later stages
  - Need to repeat / iterate one or more stages to fix timing problems

Example:

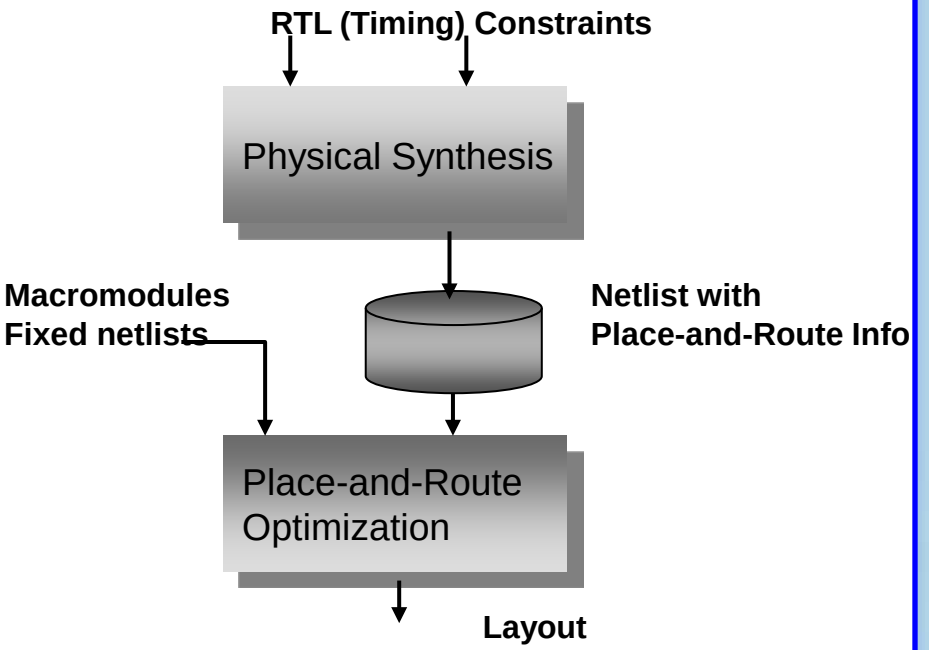


White lines indicate timing violations

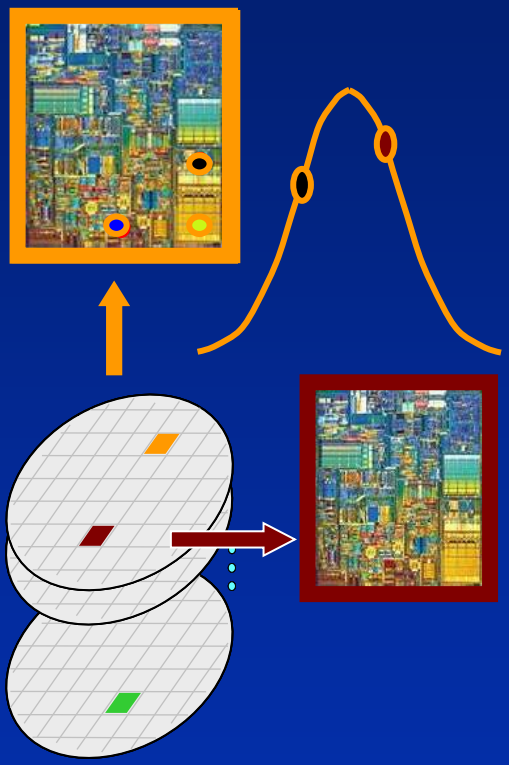
Source: Synopsys

Addressing Timing Closure

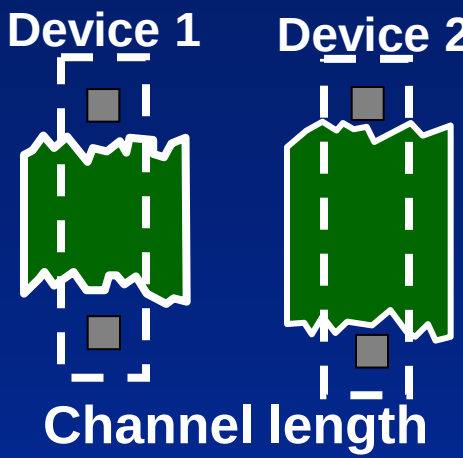
- Tighter integration of logic synthesis and physical design
  - Placement during technology mapping
  - Constant-delay synthesis

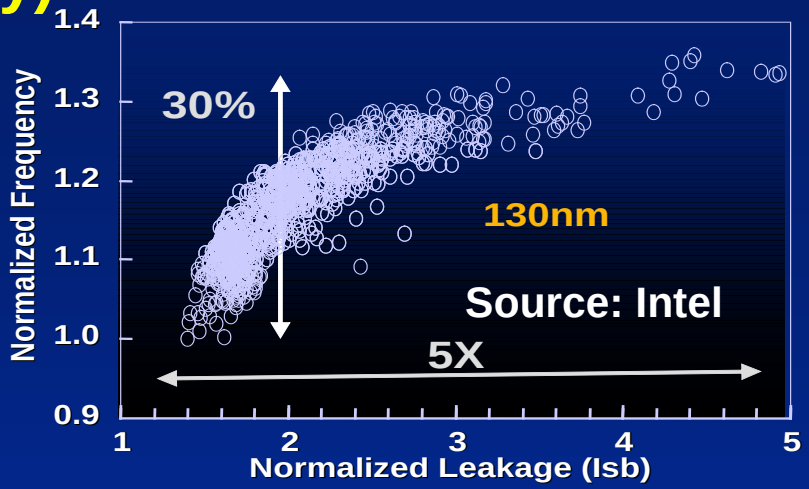


Nanometer Design Challenges :Variation in Process Parameters (Source: Kaushik Roy)

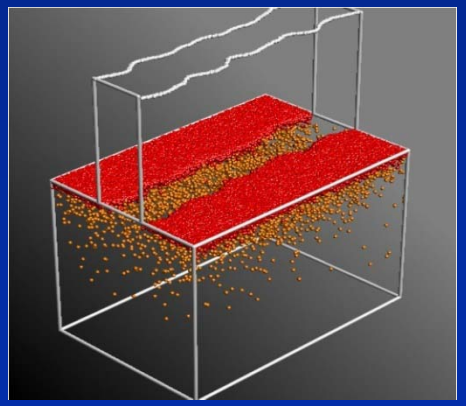


Inter and Intra-die Variations

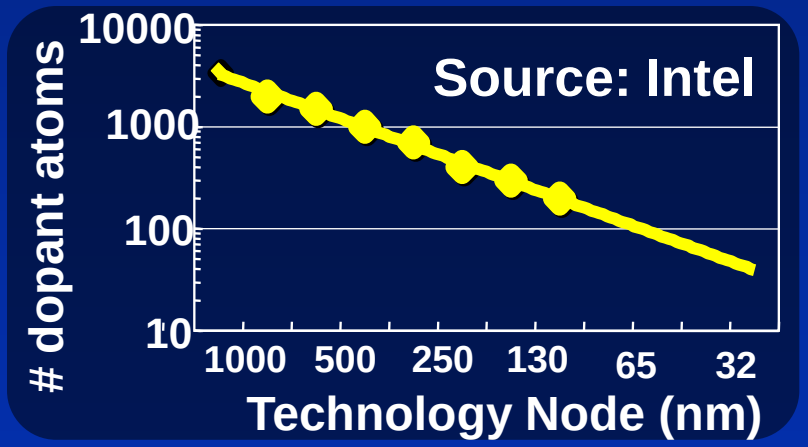




Delay and Leakage Spread



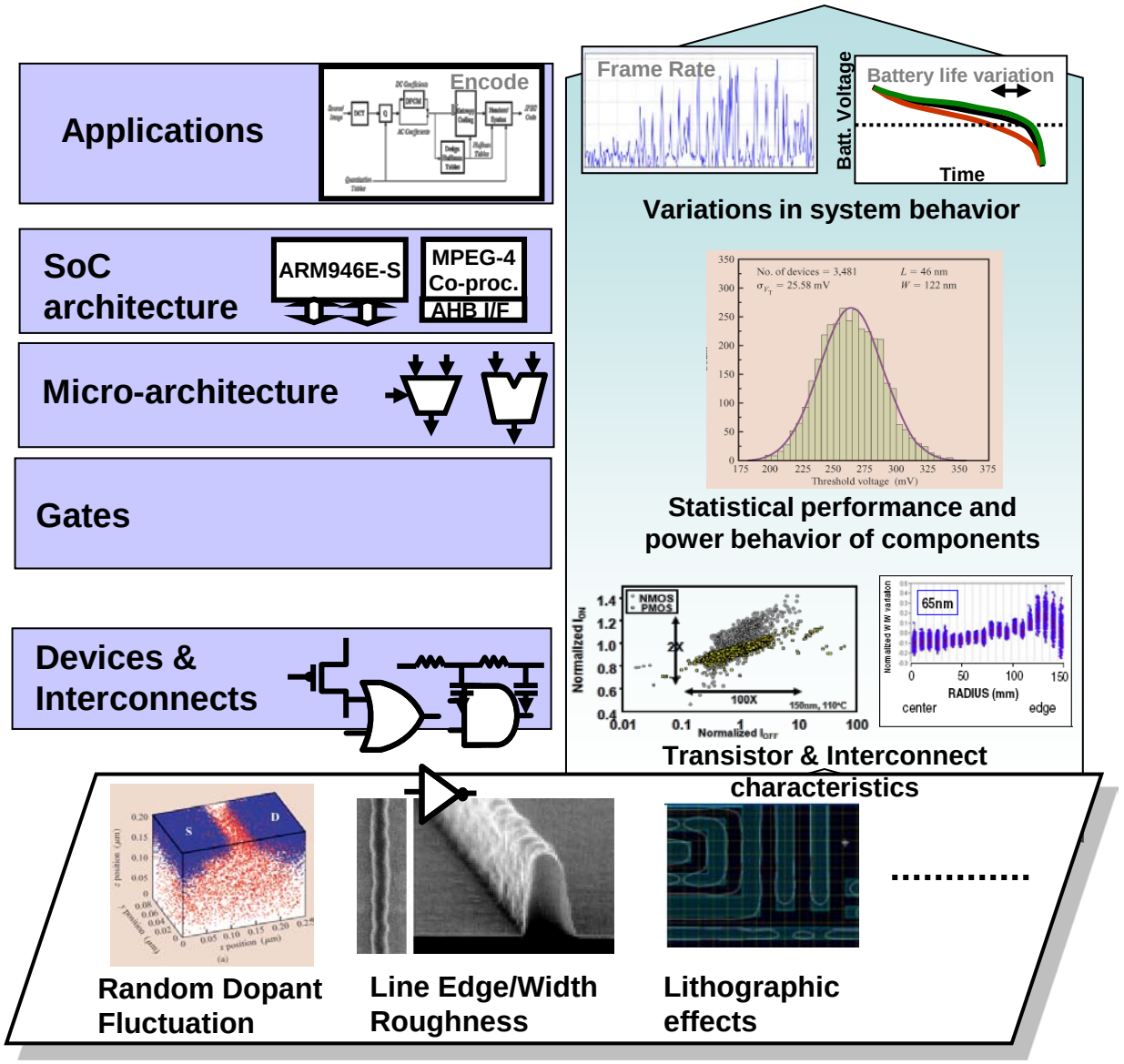
Random dopant fluctuation



Device parameters are no longer deterministic

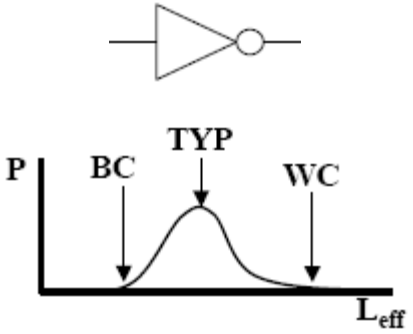
Impact of Process Variations

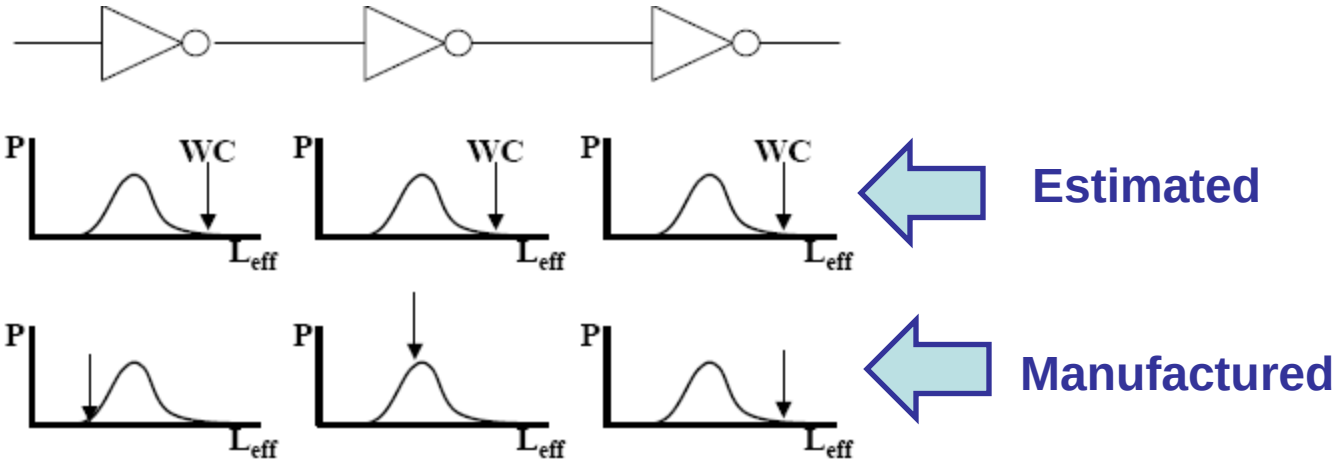
- Need to consider statistical models of delay, power!



Dealing with Variations: Statistical Timing Analysis

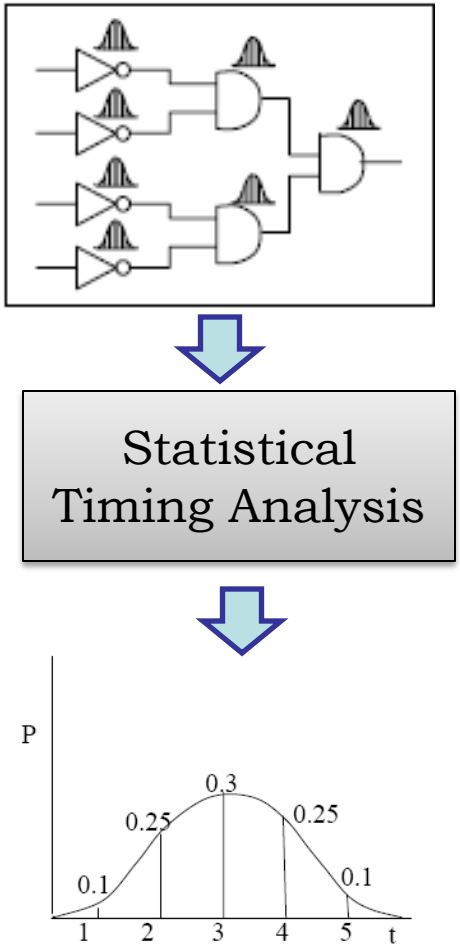
- Traditional approach: Corners
  - Use best-case, worst-case, and typical-case values
- Problem: As spread of distribution increases, this leads to highly conservative estimates, incorrect critical paths, and over-design





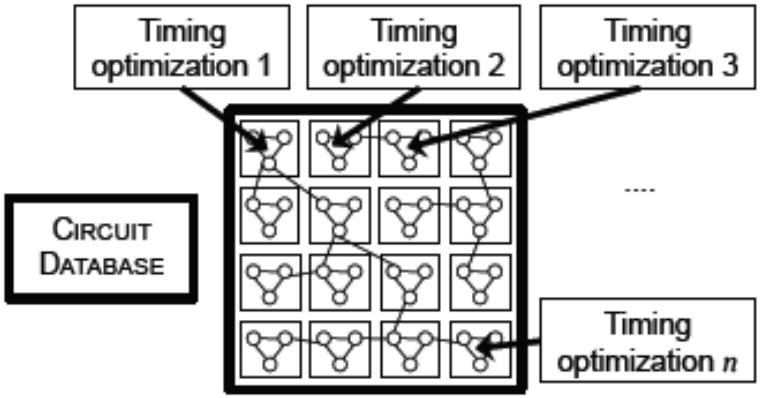
Statistical Timing Analysis and Design

- Model delays of gates as probability distributions
- Statistical Timing Analysis: Compute delay distribution for entire circuit
- Statistical Optimization: Improve design considering overall delay distribution rather than worst-case delay



Parallelizing EDA

- EDA tools cope with increase in complexity by leveraging better algorithms and faster computers
  - Performance via parallelism is the new paradigm
- Focus on parallelizing the core algorithms in EDA
  - Graph algorithms (sis
  - Backtrack / B&B (SAT, ATPG, ESPRESSO)
  - Dynamic Programming (
  - Linear Algebra (Placement & Routing, Circuit Simulation)



Optimization Agents acting on a repository: A model for parallel EDA

Catanzaro et. al., "Parallelizing CAD: A Timely Research Agenda for EDA," DAC 2008.

Summary

- Digital Logic Synthesis is a mature area
  - Strong foundation in algorithms and optimization techniques
- Evolving nature of VLSI (scaling driven trends) keeps it a vibrant and active research area