

# **EEC 216 Lecture #3: Power Estimation, Interconnect, & Architecture**

**Rajeevan Amirtharajah  
University of California, Davis**

# Outline

---

- **Announcements**
- **Review: PDP, EDP, Intersignal Correlations, Glitching, Top Level Power Estimation**
- **Behavioral Level Power Estimation**
- **Architectural Level Power Estimation**
- **Interconnect Power**
- **Low Power Architecture**

# Outline

---

- Announcements
- **Review: PDP, EDP, Intersignal Correlations, Glitching, Top Level Power Estimation**
- Behavioral Level Power Estimation
- Architectural Level Power Estimation
- Interconnect Power
- Low Power Architecture

# Review: Power-Delay Product

---

$$PDP = P_{av} t_{pd}$$

- **Product of average power and propagation delay is generally a constant (fixed technology and topology)**
- **PDP = Energy consumed by gate per switching event (Watts x seconds = Joules)**
- **Energy is nice because it is a physical quantity, easy to relate to actual device**
- **Can be distorting: to minimize energy, use  $V_{DD} = 0$  V**
- **Most useful when  $t_{pd}$  is constrained by application**
  - DSP, multimedia applications, real-time operation

# Review: Energy-Delay Product

---

$$EDP = PDP \times t_{pd} = P_{av} t_{pd}^2$$

- **Weight performance more heavily than PDP**
  - Enables more flexible power-performance tradeoff
- **Higher voltages decrease delay but increase energy**
- **Lower voltages decrease energy but increase delay**
- **Therefore there exists an optimum supply voltage**
- **Useful when application allows power and performance (e.g., clock frequency) to both vary**
  - Good for evaluating logic styles, microprocessors, etc.

# Review: Intersignal Correlations

---

- **Activity factor assumes independent, uniformly distributed input data, NOT very good assumptions typically**
  - Switching activity strong function of input statistics
  - Must use conditional probabilities when evaluating circuits with reconvergent fanout
- **Several techniques can be applied to reduce activity factor of logic internal nodes**
  - Logic restructuring: rearrange gates, for example in chains instead of trees
  - Input reordering: put high activity signals late in path
  - Consider parallelizing structures instead of time multiplexing them

# Review: Glitches

---

- **Glitches due to mismatches in nonzero propagation delay of logic gates**
  - Internal nodes may make spurious transitions before settling on final output
  - Reduce glitching by balancing logic delays (can also speed up circuits)
  - Add transparent latches to inputs of complex combinational logic blocks to eliminate glitching when outputs unused

# Review: Top Level Power Estimation

---

- **System Level Power Estimation**
  - Allows designer to analyze power impact of system partitioning among software, FPGAs, ASICs, etc.
  - Spreadsheet analysis using library of models for entire components
  - Models created from measurements, low-level power estimation
- **Instruction Level Power Estimation**
  - Run assembly instructions on target processor and measure power
  - Create database of power cost for individual instructions, pairs, maybe entire frequently used traces
  - Add in cache misses, pipeline stalls, etc.

# Outline

---

- Announcements
- Review: PDP, EDP, Intersignal Correlations, Glitching, Top Level Power Estimation
- **Behavioral Level Power Estimation**
- Architectural Level Power Estimation
- Interconnect Power
- Low Power Architecture

# Outline

---

- Announcements
- Review: PDP, EDP, Intersignal Correlations, Glitching, Top Level Power Estimation
- Behavioral Level Power Estimation
- Architectural Level Power Estimation
- **Interconnect Power**
- Low Power Architecture

# Interconnect Modeling

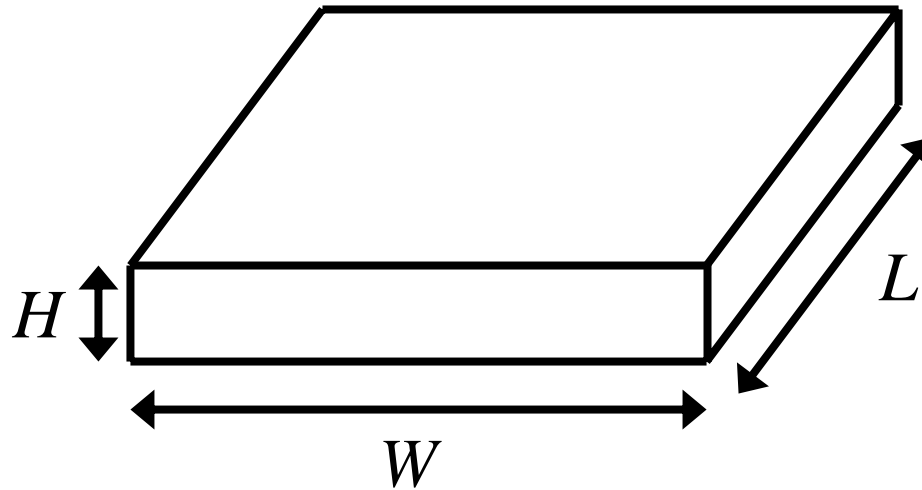
---

- **Early days of CMOS, wires could be treated as ideal for most digital applications, not so anymore!**
- **On-chip wires have resistance, capacitance, and inductance**
  - Similar to MOSFET charging, energy depends on capacitance
  - Resistance might impact adiabatic charging, static current dissipation
  - Ignore inductance for now
- **Interconnect modeling is whole field of research itself!**

# Resistance

---

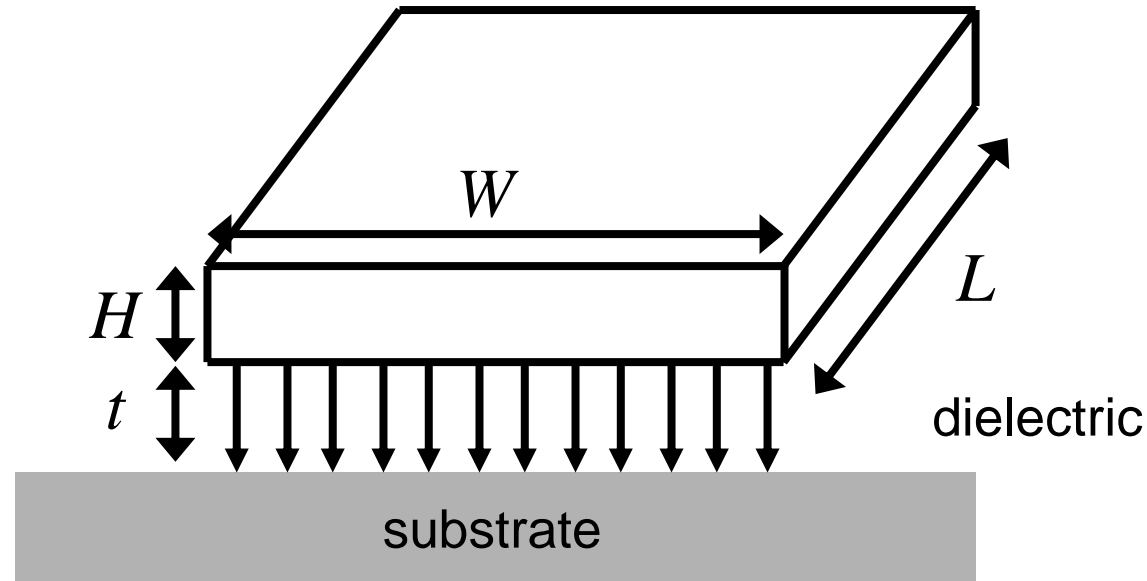
- Resistance proportional to length and inversely proportional to cross section
- Depends on material constant resistivity  $\rho$  ( $\Omega\text{-m}$ )



$$R = \frac{\rho L}{A} = \frac{\rho L}{HW} = R_{sq} \frac{L}{W} \quad R_{sq} = \frac{\rho}{H}$$

# Parallel-Plate Capacitance

- Width large compared to dielectric thickness, height small compared to width: E field lines orthogonal to substrate

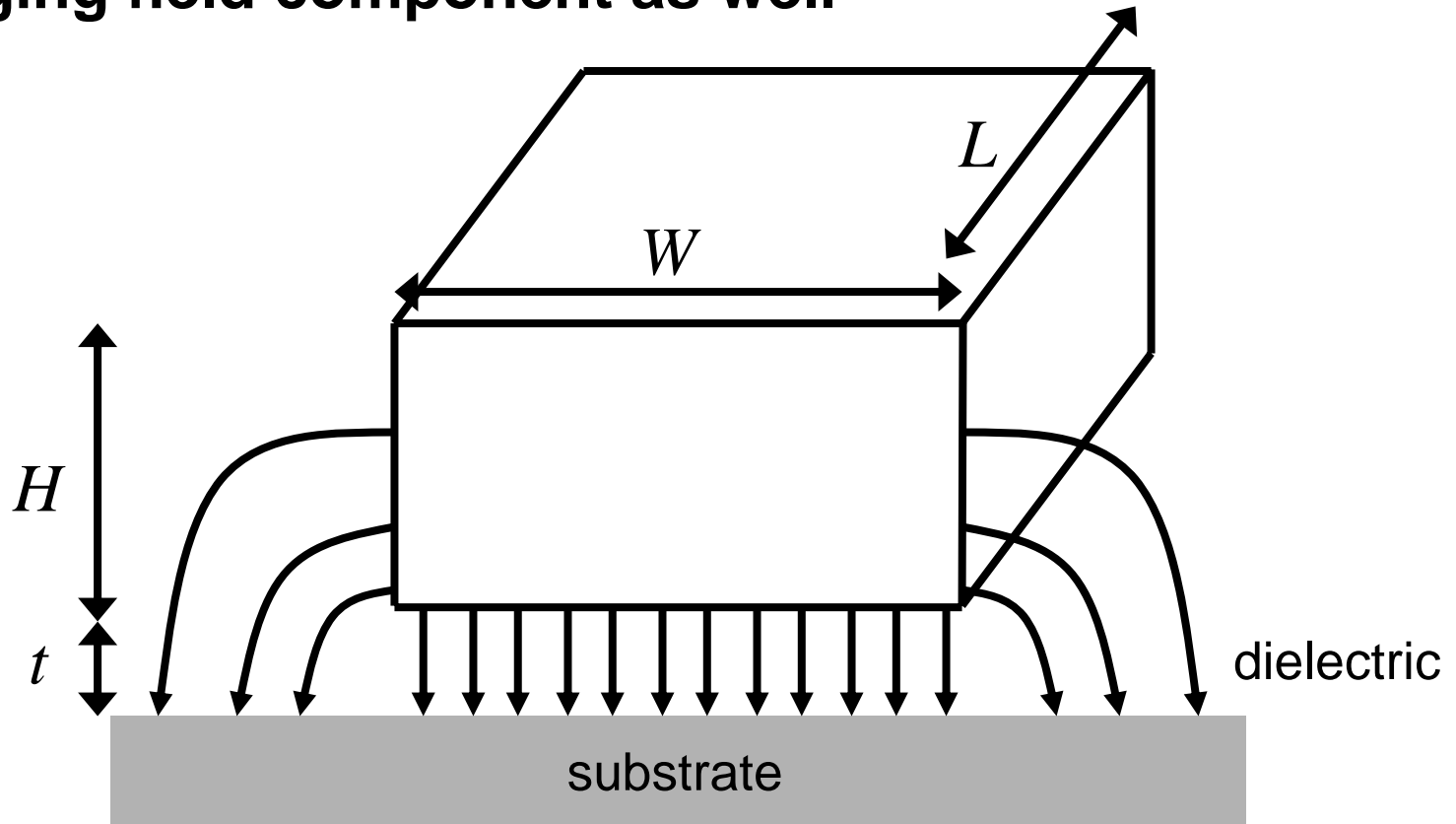


$$C = \frac{\epsilon_r}{t} WL$$

# Fringing Field Capacitance

---

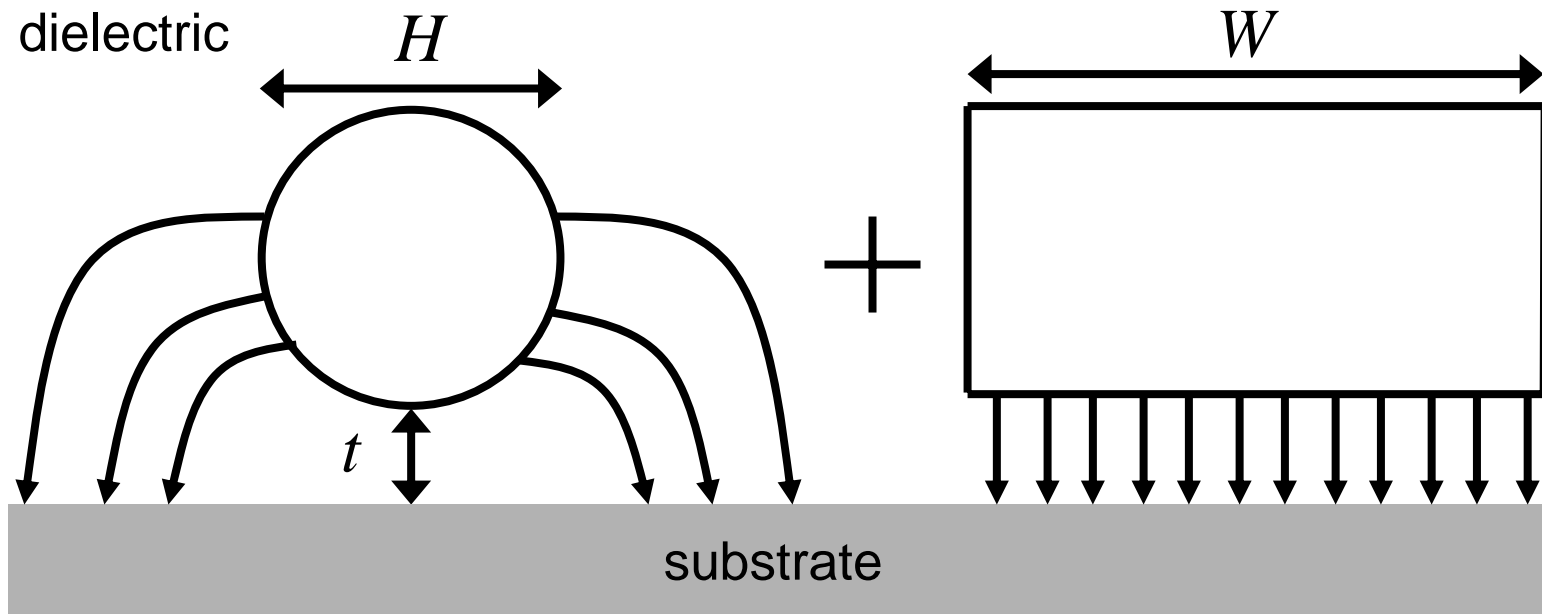
- When height comparable to width, must account for fringing field component as well



# Total Capacitance Model

---

- When height comparable to width, must account for fringing field component as well
- Model as a cylindrical conductor above substrate



# Corrected Total Capacitance Model

---

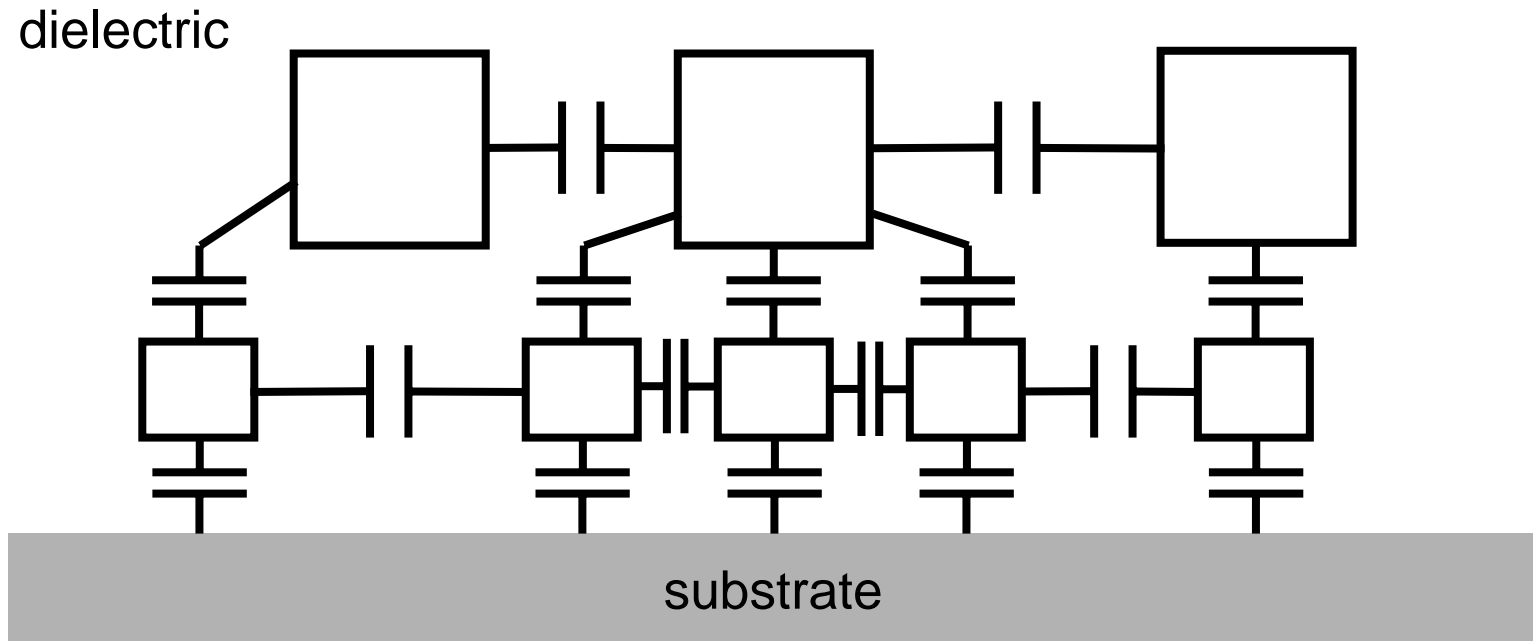
- **Total capacitance per unit length is parallel-plate (area) term plus fringing-field term:**

$$C = C_{pp} + C_{fringe} = \frac{\epsilon_r W}{t} + \frac{2\pi\epsilon_r}{\log(2t/H + 1)}$$

- **Model is simple and works fairly well**
  - More sophisticated numerical models also available
- **Process models often give both area and fringing (also known as sidewall) capacitance numbers per unit length of wire for each interconnect layer**

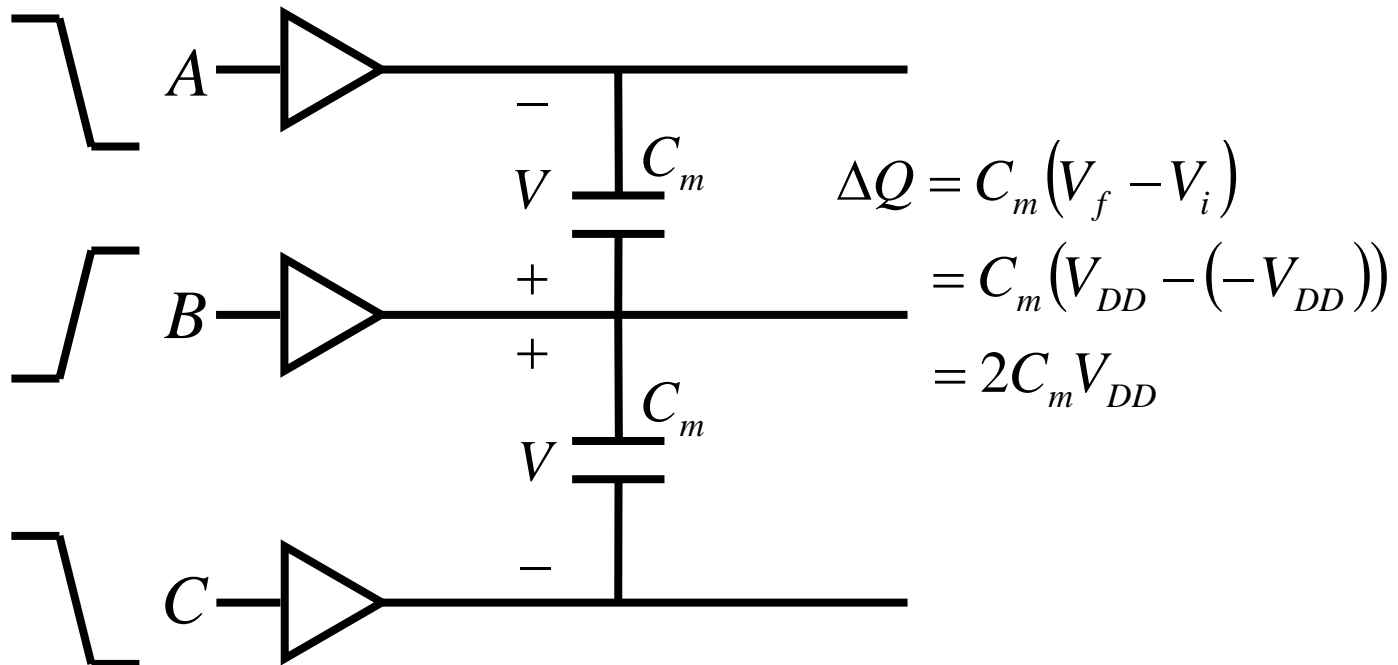
# Capacitive Coupling

- Fringing fields can terminate on adjacent conductors as well as substrate
- Mutual capacitance between wires implies crosstalk, affects data dependency of power



# Miller Capacitance

- Amount of charge moved onto mutual capacitance depends on switching of surrounding wires
- When adjacent wires move in opposite direction, capacitance is effectively doubled (Miller effect)



# Data Dependent Switched Capacitance 1

---

- When adjacent wires move in same direction, mutual capacitance is effectively eliminated

$$A \uparrow B \uparrow C \uparrow \text{ OR } A \downarrow B \downarrow C \downarrow \quad C_{eff} = 0$$

$$A \downarrow B \uparrow C \downarrow \text{ OR } A \uparrow B \downarrow C \uparrow \quad C_{eff} = 4C_m$$

$$A \downarrow B \uparrow C \uparrow \text{ OR } A \downarrow B \downarrow C \uparrow$$
$$A \uparrow B \uparrow C \downarrow \text{ OR } A \uparrow B \downarrow C \downarrow \quad C_{eff} = 2C_m$$

# Data Dependent Switched Capacitance 2

---

- When adjacent wires are static, mutual capacitance is effectively to ground

$$\begin{array}{ccc} 0 B \uparrow 0 & \text{OR} & 1 B \downarrow 1 \\ 1 B \uparrow 0 & \text{OR} & 0 B \downarrow 1 \\ 0 B \uparrow 1 & \text{OR} & 1 B \downarrow 0 \\ 1 B \uparrow 1 & \text{OR} & 0 B \downarrow 0 \end{array} \quad C_{eff} = 2C_m$$

- Remember: it is the *charging* of capacitance where we account for energy from supply, not discharging

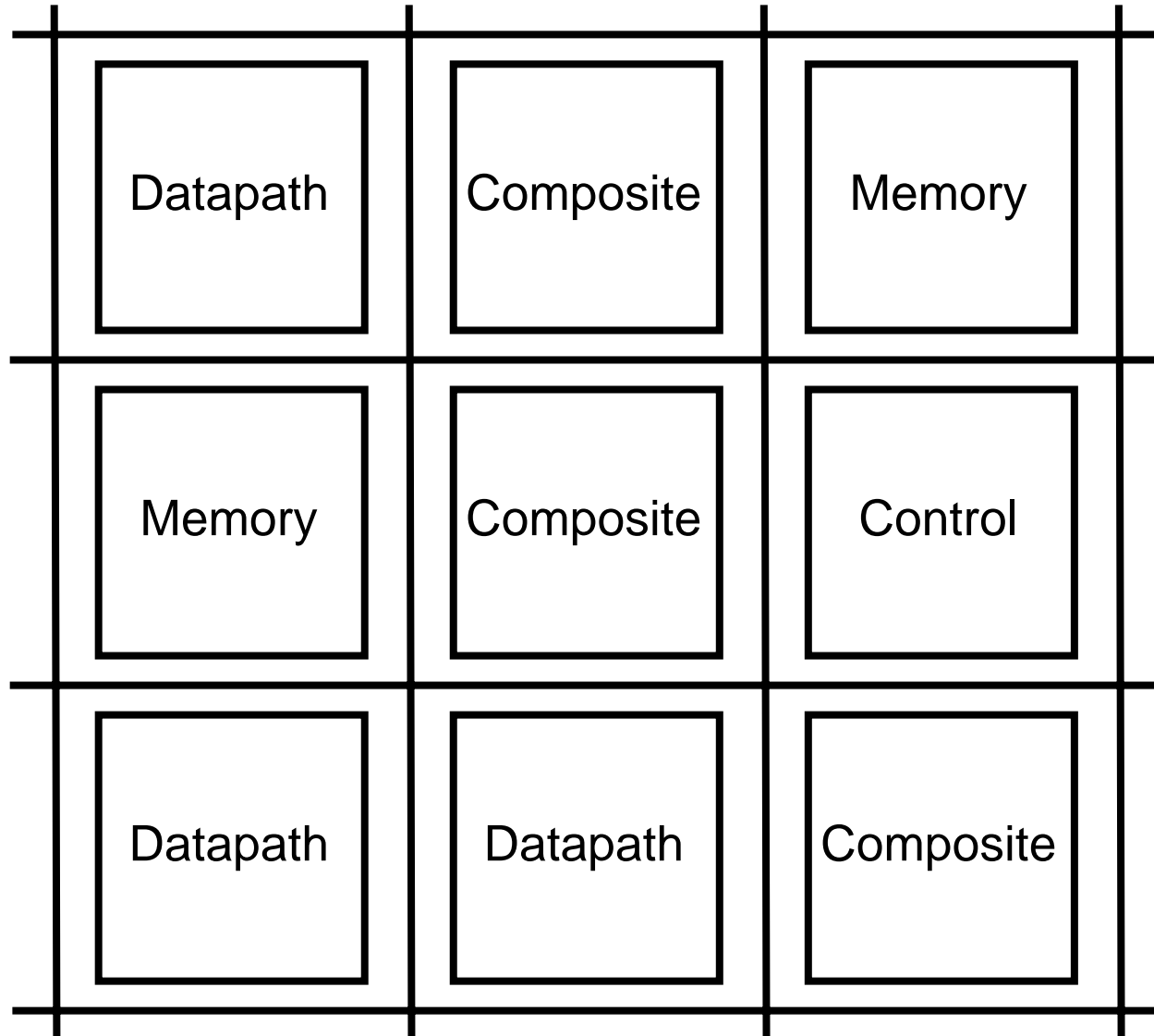
# Wire Length Estimation Flow

---

- **Final piece of architectural power estimation: incorporate interconnect power**
- **Given hierarchical RTL description, estimate wire lengths before design actually placed and routed**
- **Depth-first traversal of hierarchy, where leaf nodes are blocks already well characterized for area and wire length by dedicated analysis**
- **Example: consider four block types**
  - Primitives: memory, datapath, control
  - Composite block made up of primitives

# Hierarchical Chip Floorplan

---



# Memory and Control Blocks

---

- **Assume given blocks are already characterized or can easily be done**
  - Typical for foundry to provide memory hard or soft layout macro for synthesis flow: user given interconnect lengths and area
  - Random control logic usually small, can be quickly synthesized, placed, and routed
- **Fairly straightforward to turn complexity metrics (number of memory cells or gate counts) into area estimates**
  - Turn into power models as discussed earlier

# Composite Blocks Wire Length

---

- **Best approach is to estimate length based on early floorplan**
- **Alternative is to use empirical observations**
  - Studies have shown that “good” cell placement differs from random placement by constant fudge factor  $k$  ( $k$  is often quoted as being  $3/5$ )
  - Average wire length for random placement on a square of area  $A$  array is  $1/3$  length of a side
- **Average wire length of “good” placement on square array:**

$$L = k \frac{\sqrt{A}}{3} = \frac{\sqrt{A}}{5}$$

# Composite Blocks Area

---

- Area of composite block equals sum of areas of constituent blocks  $A_B$  and area of wires  $A_w$

$$A = A_w + A_B = A_w + \sum_{i \in \{blocks\}} A_i$$

- Routing area depends on total number of wires  $N_w$ , average wire pitch  $W_p$ , and average length  $L$

$$A_w = N_w W_p L$$

- Using formula on preceding slide, can solve quadratic for length  $L$ :

$$L = \frac{k^2 N_w W_p + \sqrt{\left(k^2 N_w W_p\right)^2 + 36k^2 A_B}}{18}$$

# Datapath Blocks Wire Length

---

- **Datapaths often laid out in linear (bit slice) pattern**
  - Tile  $N$  bit slices to create an  $N$ -bit datapath
  - Interconnect length thus proportional to datapath length
  - Use another empirical factor to relate “good” placement to random placement for length in  $x$  dimension

$$L_x = k \frac{L_{DP}}{3} = \frac{k}{3} \left( L_R + \sum_{i \in \{blocks\}} L_i \right)$$

- **Routing channel length  $L_R$  estimated from wiring pitch and number of I/Os on each block**

$$L_R = W_p \sum_{i \in \{blocks\}} N_{IO_i}$$

# Datapath Blocks Area

---

- **Similar approach used to estimate vertical routing in feedthroughs within a bit slice of width  $W_{BS}$ :**

$$L_y = 2k \frac{W_{BS}}{3}$$

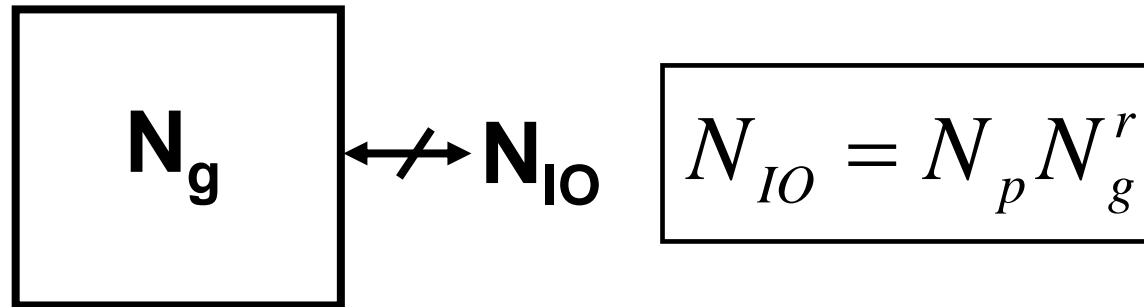
- **Total average interconnect length is  $L = L_x + L_y$**
- **Approximate area as datapath width x datapath length for  $N$  bit slices:**

$$A = W_{DP} L_{DP} = N W_{BS} L_{DP}$$

- **Incorporate all of these approximations plus equations for physical capacitance into our RTL-level power estimates**

# Rent's Rule

---



- Empirical rule relating number of I/Os in and out of a module to number of gates within the module
- $N_p$  is average number of pins per gate (~2.5)
- Rent's exponent  $r$  between 0.65 and 0.7
- Numbers can be used to characterize various design styles (memories, gate arrays, microprocessors)
- Extended to derive average wire lengths

# Outline

---

- Announcements
- Review: PDP, EDP, Intersignal Correlations, Glitching, Top Level Power Estimation
- Behavioral Level Power Estimation
- Architectural Level Power Estimation
- Interconnect Power
- **Low Power Architecture**

# Low Power Architecture Outline

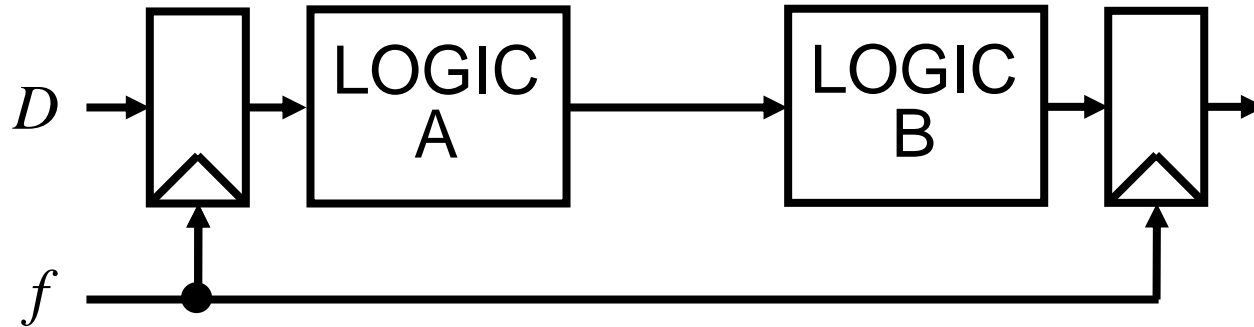
---

- **Clock Gating**
- **Power Down Modes**
- **Parallelization**
- **Pipelining**
- **Bit Serial vs. Bit Parallel Datapaths**

# Example Pipeline

---

- Use the following example logic pipeline: two combinational logic blocks between registers



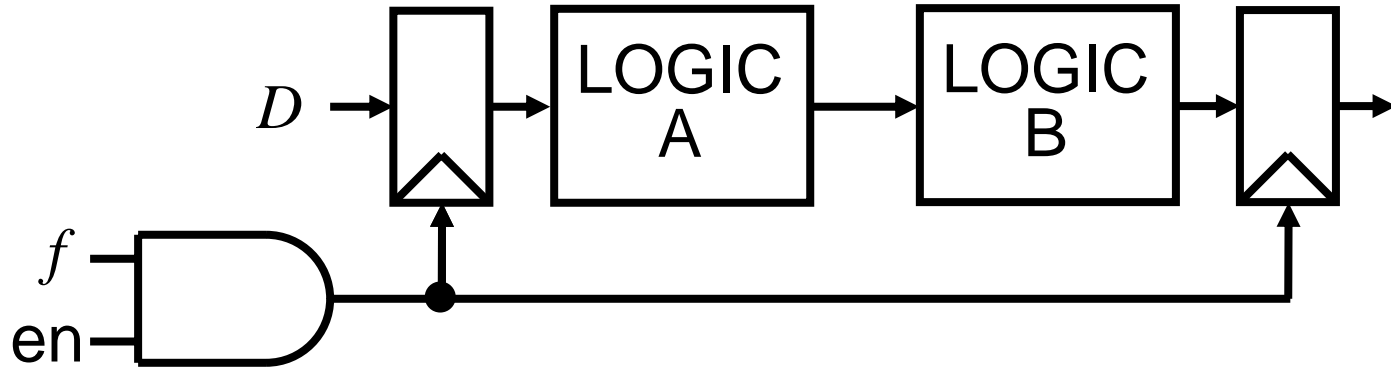
- Define reference dynamic power:

$$P_0 = C_0 V_{DD}^2 f$$

- Consider various architectural transformations to reduce power, primarily through voltage scaling and duty cycle

# Clock Gated Pipeline

- Use enable signal to turn off clock when not in use

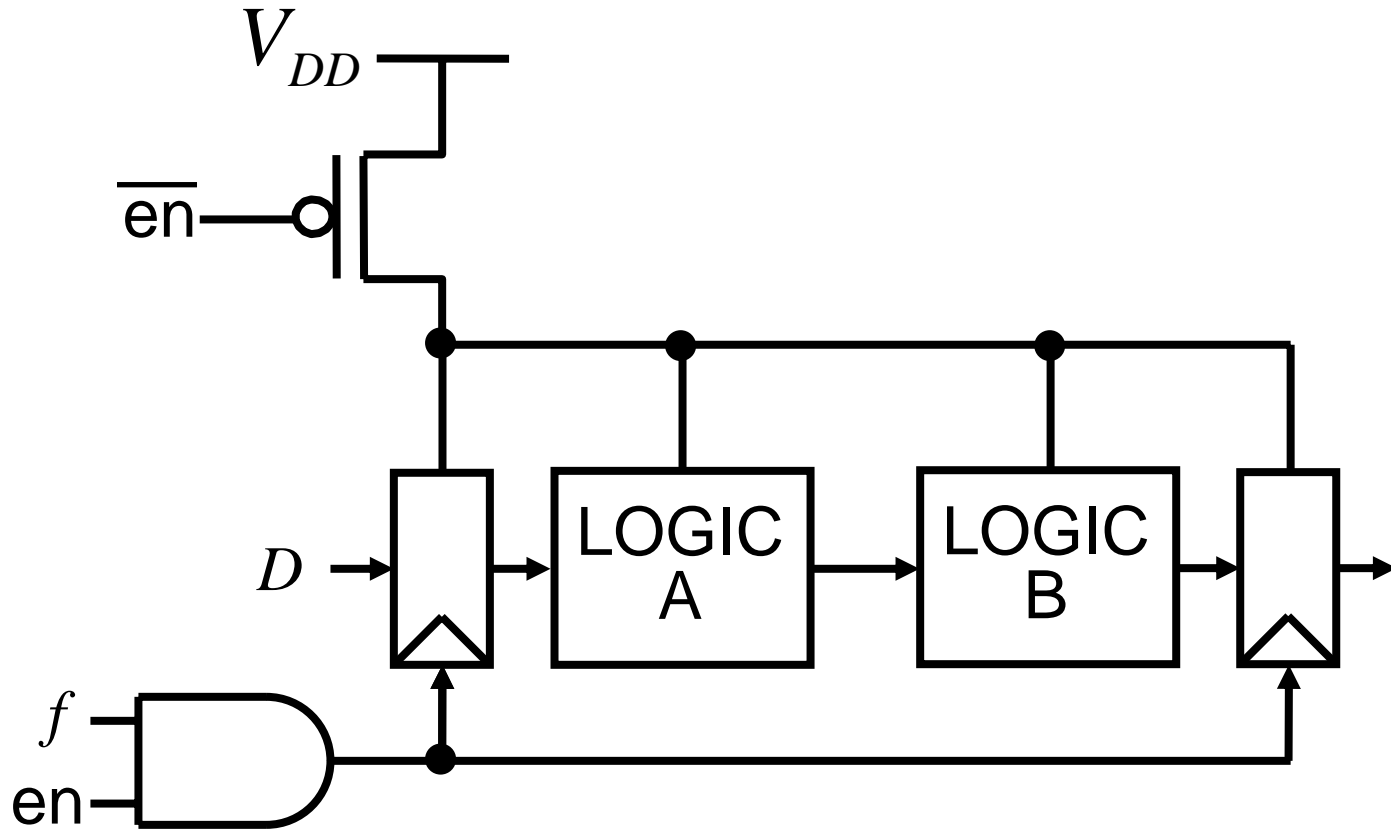


- Dynamic power reduction proportional to duty cycle  $D_C$  (% time the system in use):

$$P_G = D_C C_0 V_{DD}^2 f$$

- Must take care in implementation: glitches on enable signal result in false clocking

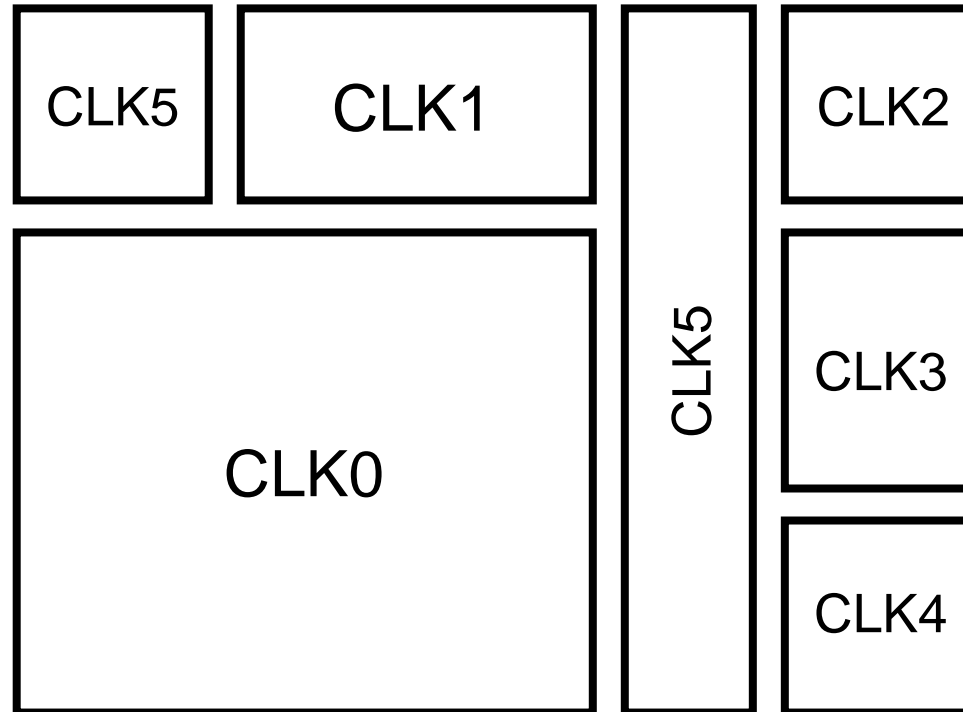
# Clock Gated Pipeline With Power Down



- **Disconnect logic from power supply when clock off**
- **Eliminates leakage, static current for further power reduction**

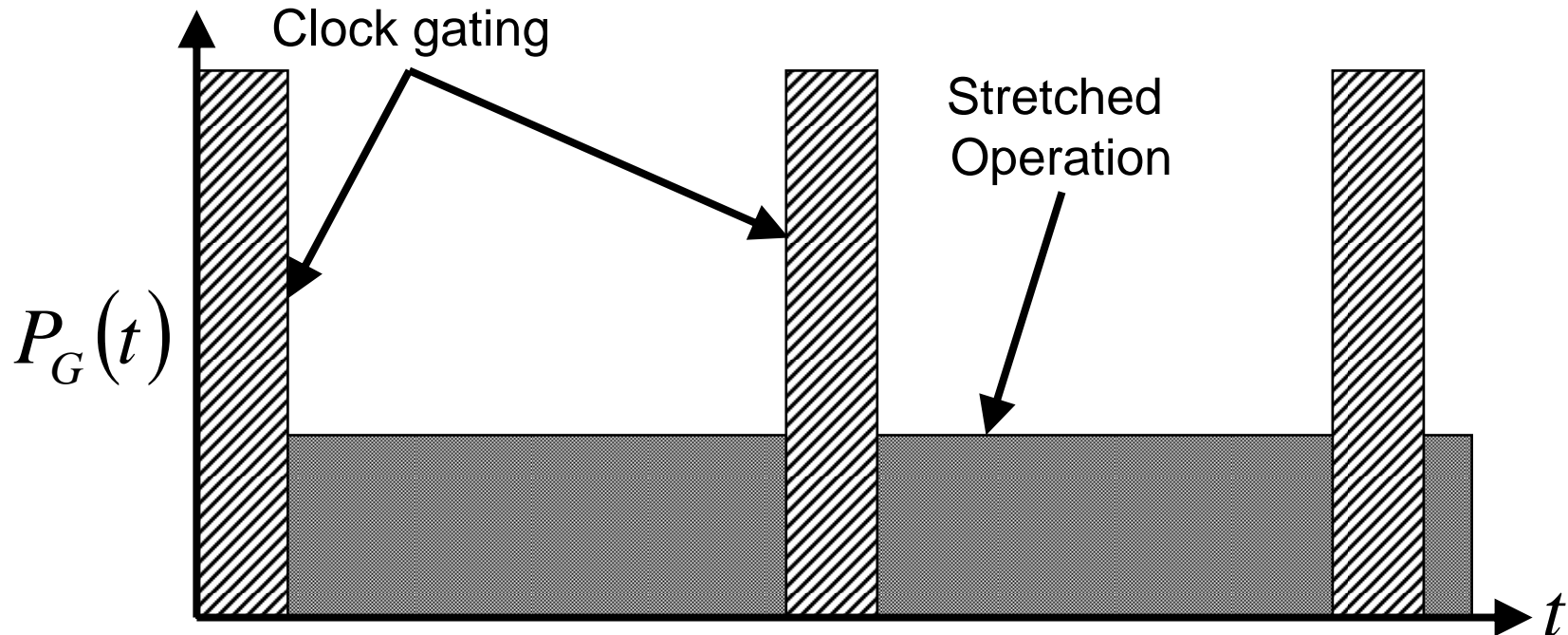
# Partitioning Into Gated Clock Domains

---



- **Generally gate off entire modules or functional units**
- **Globally Asynchronous Locally Synchronous (GALS) domains form natural clock gating partitions**
- **Synchronize on boundaries (clock gating can introduce skew due to logic in clock path)**

# Power Reduction Due to Clock Gating



- **Only linear reduction in average power, peak power stays same (issue for power supply and delivery net)**
- **Decrease frequency to expand computing to fill time allows voltage reduction also: better than linear gain**

# Low Power Architecture Outline

---

- Clock Gating
- **Power Down Modes**
- Parallelization
- Pipelining
- Bit Serial vs. Bit Parallel Datapaths

# An Explosion of Power Down Modes

---

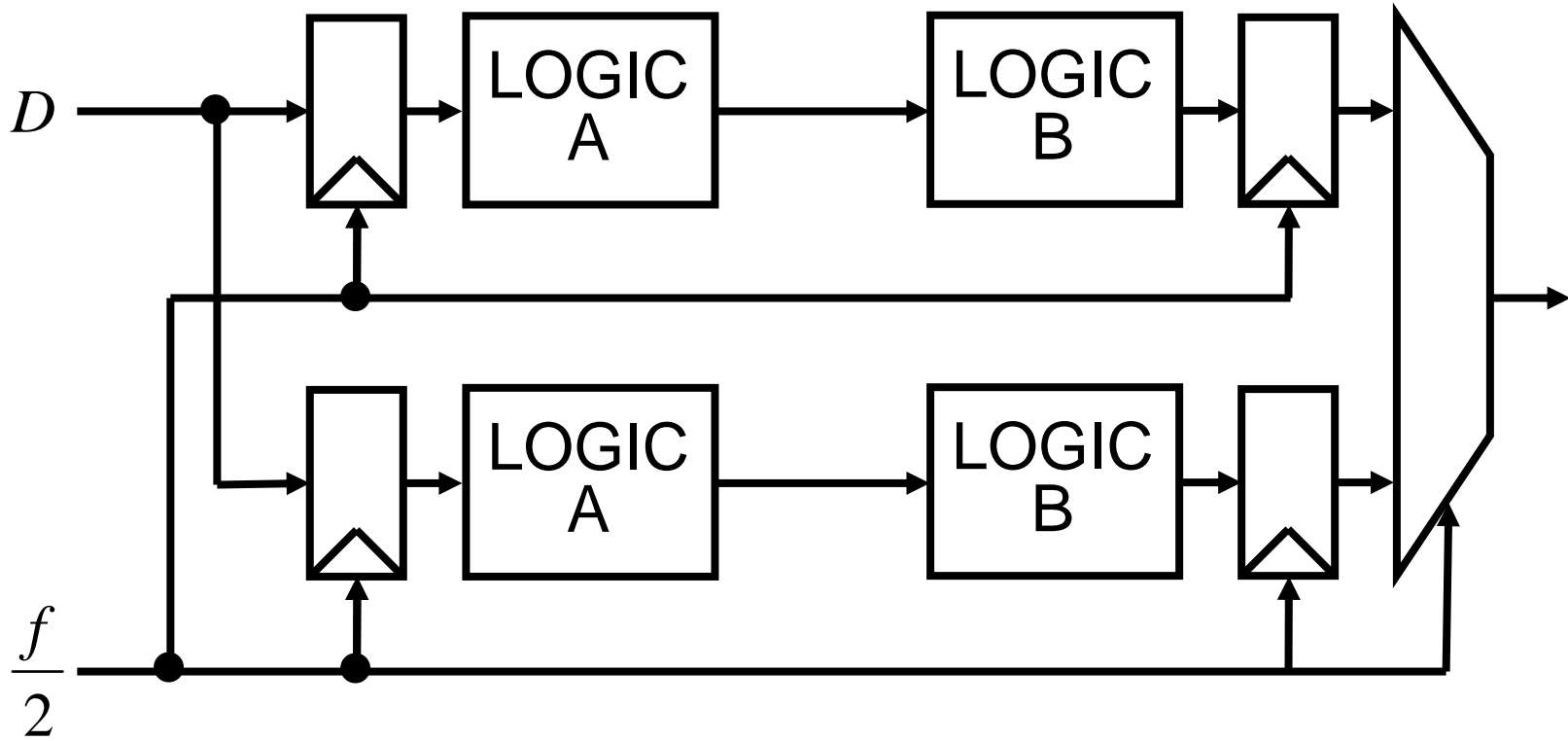
- **Although suboptimal for power reduction, clock gating is often used in practice**
  - Power electronics necessary to generate optimum voltage, more cost and complexity
- **Tradeoff between power reduction and startup delay to return to operation (“light sleep vs. deep sleep”)**
  - Gate clocks off to individual modules, fastest to start up again
  - Turn off clock generating PLLs, phase-lock transient potentially lasts several ms
  - Ground power supply, must charge VDD net before even turning on PLL

# Low Power Architecture Outline

---

- Clock Gating
- Power Down Modes
- **Parallelization**
- Pipelining
- Bit Serial vs. Bit Parallel Datapaths

# Parallelization Driven Voltage Scaling



- **Parallelize computation up to N times**
- **Reduce clock frequency by factor N**
- **Reduce voltage to meet relaxed frequency constraint**

# Tradeoffs of Parallelization

---

- **Amount of parallelism in application may be limited**
- **Extra capacitance overhead of multiple datapaths**
  - N times higher input loading
  - N-to-1 selector on output
  - Lower clock frequency somewhat offset by higher clock load
- **Consumes more area, devices, more leakage power especially in deep submicron**
- **Voltage reduction typically results in dramatic power gains**
  - Chandrakasan92: ~3X power reduction

# Low Power Architecture Outline

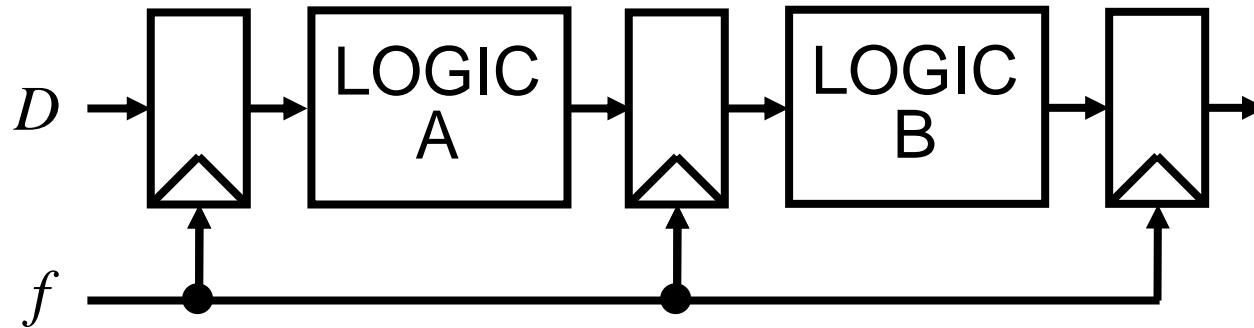
---

- Clock Gating
- Power Down Modes
- Parallelization
- **Pipelining**
- Bit Serial vs. Bit Parallel Datapaths

# Pipeline Driven Voltage Scaling

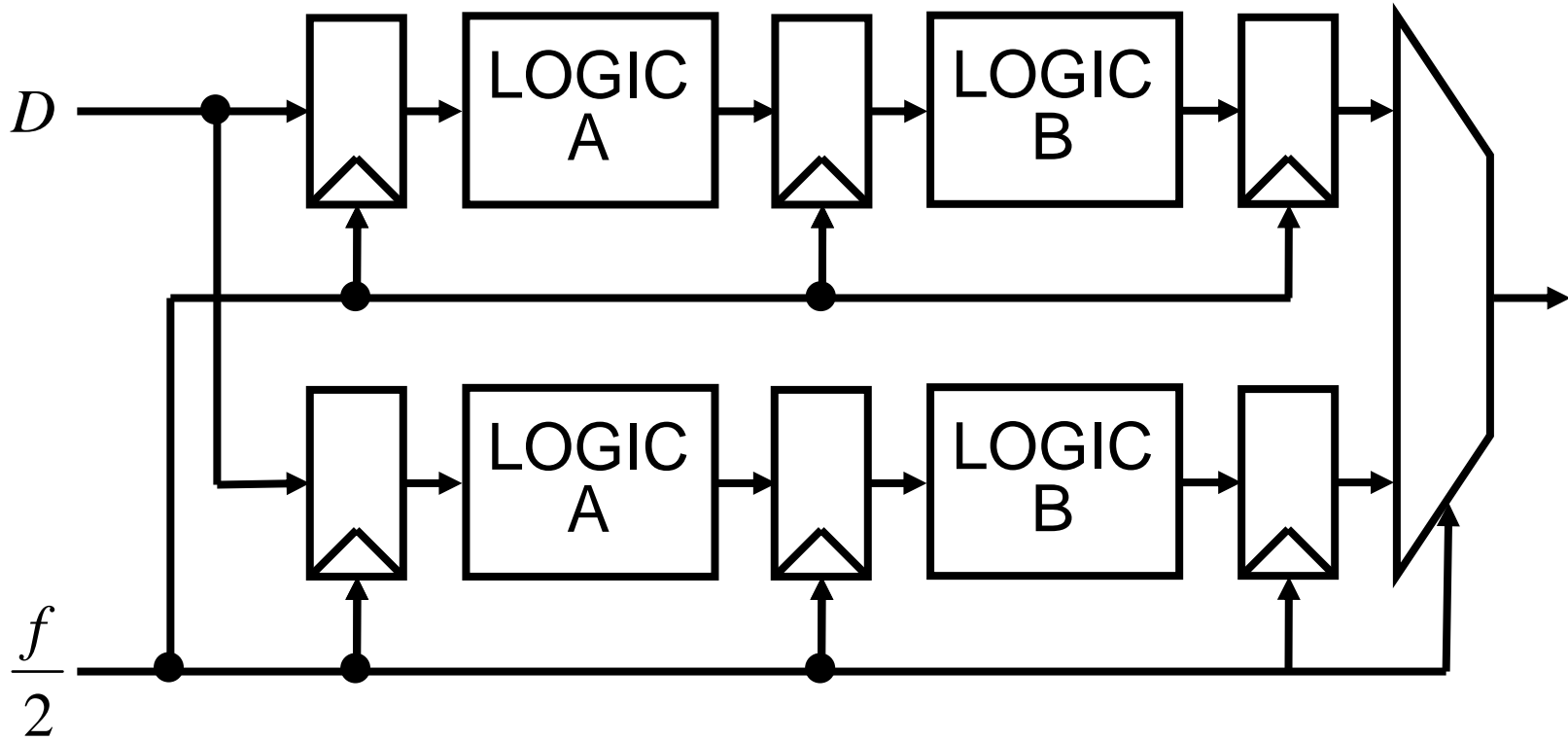
---

- Pipeline at finer granularity to relax critical path constraint



- Clock frequency stays the same
- Reduce voltage to meet relaxed frequency constraint
- Increased clock load offsets power reduction somewhat
- Can't pipeline beyond single gate granularity

# Parallel / Pipeline Driven Voltage Scaling



- **Combine parallelism and pipelining for lowest voltage**
- **Reduce clock frequency by parallelism factor N**
- **Largest increase in area, capacitance, leakage**

# Low Power Architecture Outline

---

- Clock Gating
- Power Down Modes
- Parallelization
- Pipelining
- **Bit Serial vs. Bit Parallel Datapaths**

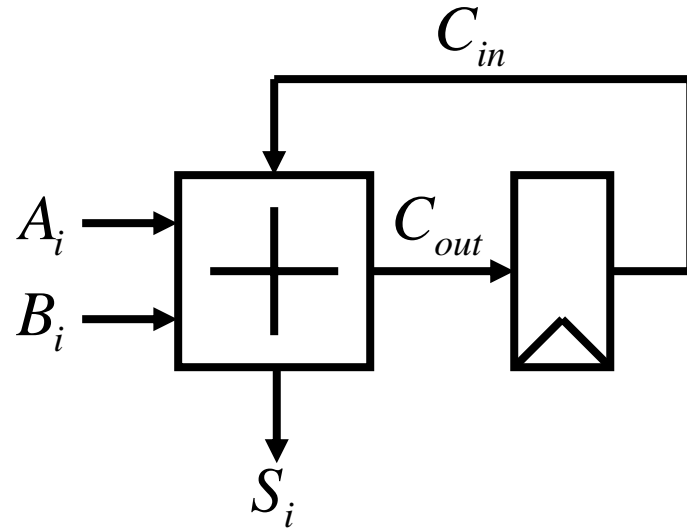
# Bit Serial vs. Bit Parallel Computation

---

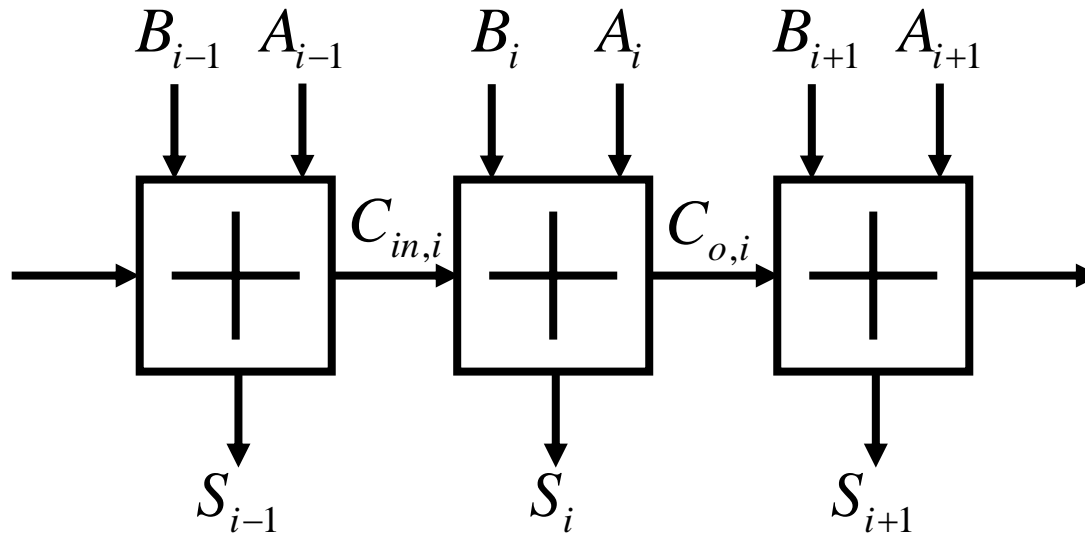
- **So far, we've talked about serial versus parallel implementations at the functional unit level**
- **Can also consider serial versus parallel implementations at the bit level**
  - Historically, datapaths have almost always operated on words (several bits in parallel)
  - In the past, heavily area constrained designs have used serial techniques where one output bit is produced per clock cycle
  - Multipliers in older CMOS processes ( $> 2 \mu\text{m}$ ) often implemented serially
- **Rather than area, current and future motivation for bit serial techniques may be leakage power**

# Bit Serial and Bit Parallel Adders

- **Serial Adder:**

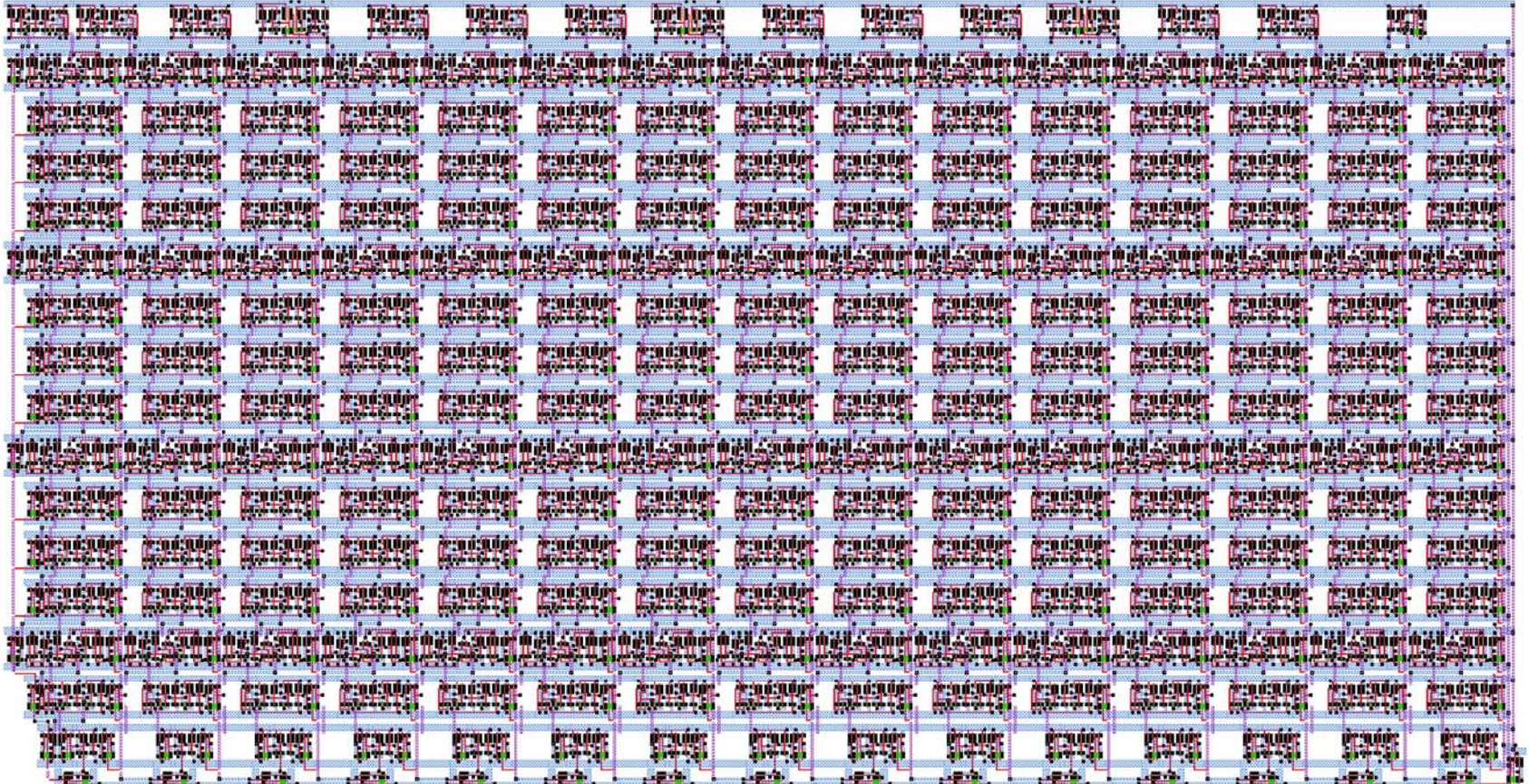


- **Parallel Adder:**



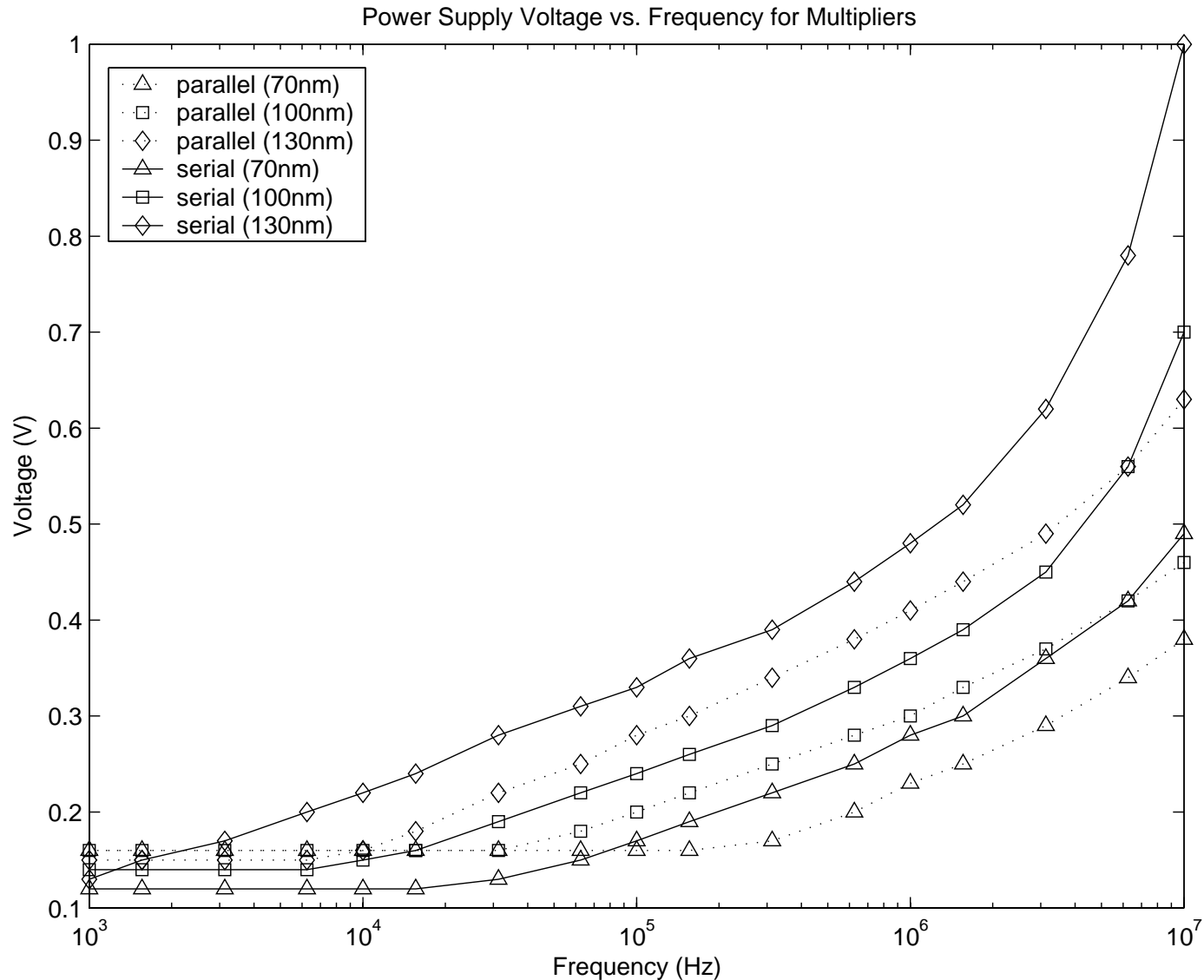
# Parallel Array Multiplier

---

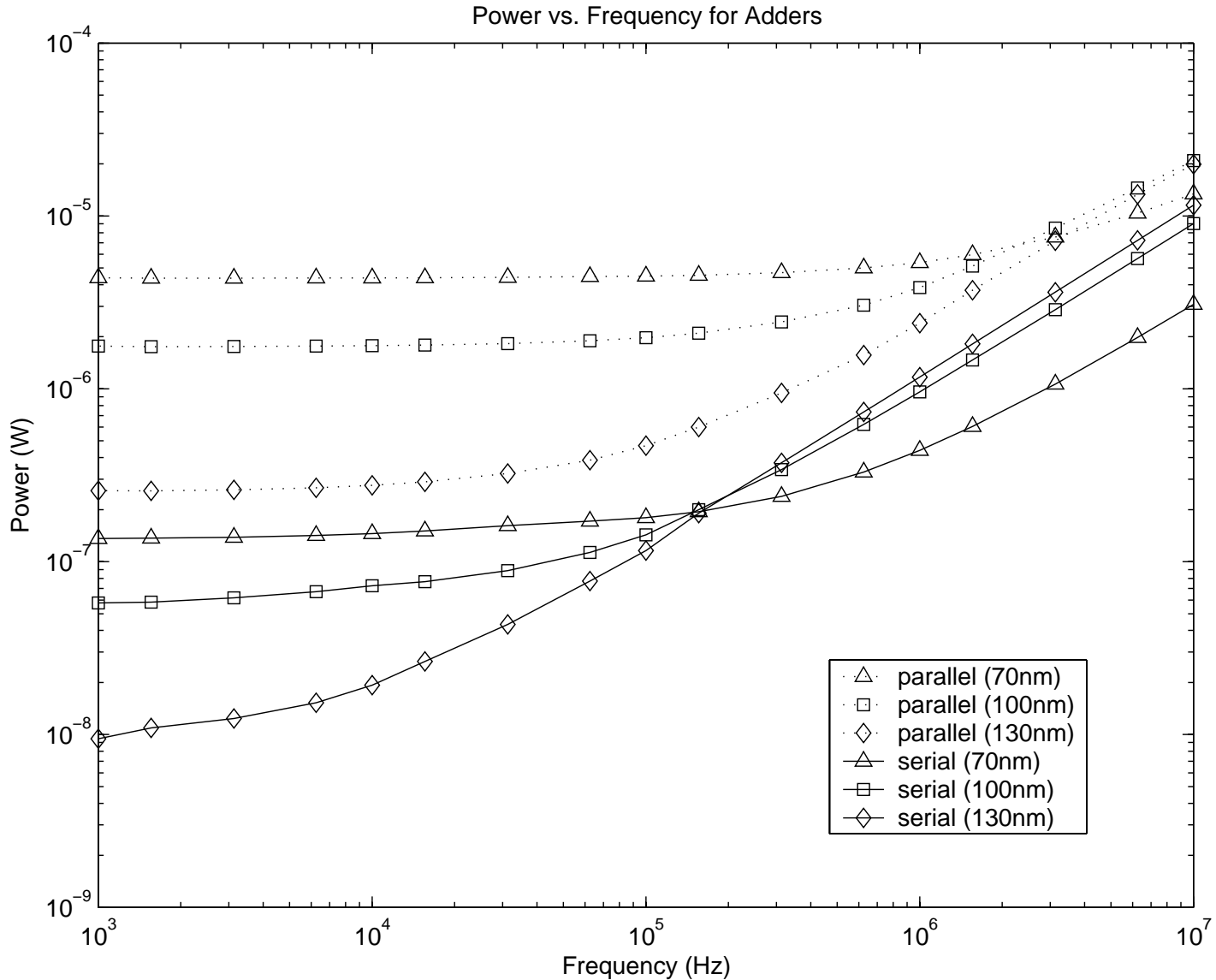


- **16b Parallel multiplier: 32390  $\mu\text{m}^2$ , Serial multiplier: 2743  $\mu\text{m}^2$**
- **32b Parallel adder: 2543  $\mu\text{m}^2$ , Serial adder: 139  $\mu\text{m}^2$**

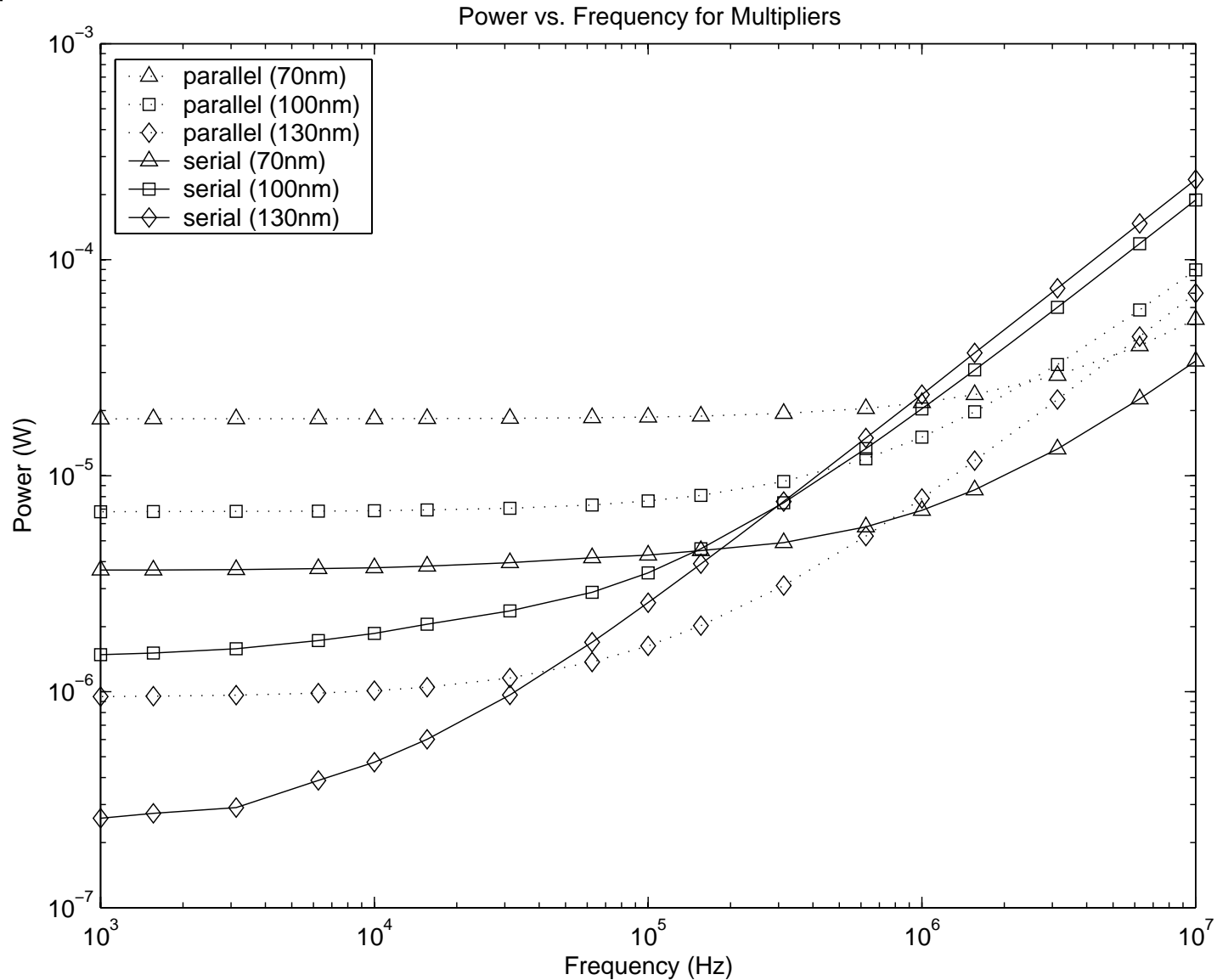
# Supply Voltage – Clock Frequency Tradeoff



# Serial vs. Parallel Adder Power



# Serial vs. Parallel Multiplier Power



# Bit Serial vs. Bit Parallel Arithmetic Power

---

- **At low frequencies, lower leakage of smaller serial implementation results in less power**
- **Similar result for adders, but more impact since array multiplier is quadratically larger than serial version**
- **Are these frequencies interesting?**
  - Below 10 MHz is typical for sensor applications, biomedical DSP, RFID tags

# Next Topic: Low Power Circuit Design

---

- **Logic families**
- **Transistor sizing for low power**
- **Clocking methodologies**

# Announcements

---

# Total Capacitance Model (Incorrect)

---

- Total capacitance per unit length is parallel-plate (area) term plus fringing-field term:

$$C = C_{pp} + C_{fringe} = \frac{\epsilon_r}{t} \left( W - \frac{H}{2} \right) + \frac{2\pi\epsilon_r}{\log(2t/H + 1)}$$

- Model is simple and works fairly well
  - More sophisticated numerical models also available
- Process models often give both area and fringing (also known as sidewall) capacitance numbers per unit length of wire for each interconnect layer