CS152
Computer Architecture and Engineering
Lecture 5: Technology & Delay Modeling

February 1, 1995

Dave Patterson (patterson@cs) and
Shing Kong (shing.kong@eng.sun.com)

Slides available on http://http.cs.berkeley.edu/~patterson

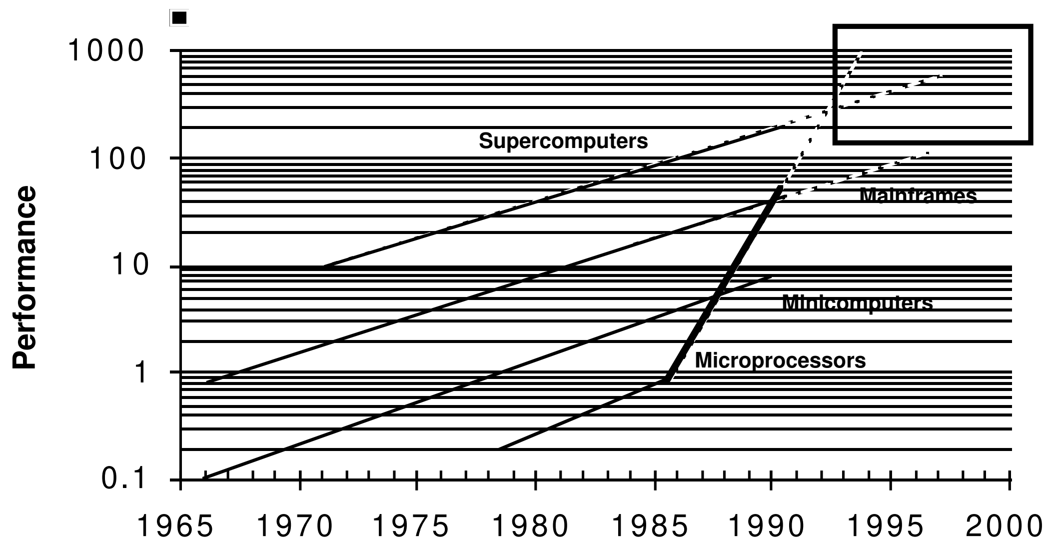
Recap of Last Lecture

- Use general purpose registers with a load-store architecture
  - Provide at least 16 GP registers and separate FP registers
- Support the following addressing modes:
  - Displacement (with an address offset size of 12 to 16 bits)
  - Immediate with size 8 to 16 bits
  - Register deferred
- All addressing modes apply to all data transfer instructions
- Use fixed instruction encoding if interested in performance
  Use variable instruction encoding if interested in code size
- Support these data sizes and types: 8-bit, 16-bit, 32-bit integers
  32-bit and 64-bit IEEE floating point numbers
- Support these simple instructions: load, store, add, subtract, move register-register, and, shift, compare equal, compare not equal, branch, jump, call, and return
- Aim for a minimalist instruction set

Outline of Today's Lecture

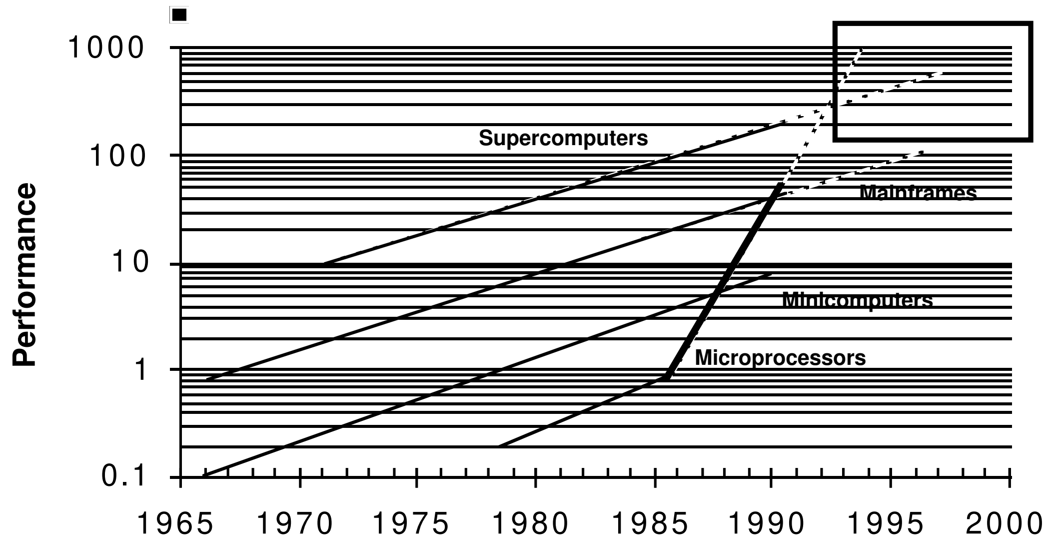
- Recap of Last Lecture and Introduction of Today's Lecture (4 min.)
- Performance and Technology trends (16 minutes)
- Administrative Matters and Questions (5 minutes)
- Delay Modeling and Gate Characterization (25 minutes)
- Questions and Break (5 minutes)
- Clocking Methodologies and Timing Considerations (25 minutes)

Performance and Technology Trends



- Feature Size: shrinks 10% / yr.
  - Switching speed improves 1.2 / yr.
- Density: improves 1.2x / yr.
- Die Area: 1.2x / yr.
- Technology Power: 1.2 x 1.2 x 1.2 = 1.7 x / year

Performance and Technology Trends (Continue)



- **Compute Power:**
  - **Prior to 1985: less than 1.7 x / year**
  - **After 1985: Microprocessor improves greater than 1.7 x / year**
- **The lesson of RISC is to keep the ISA as simple as possible:**
  - **Shorter design cycle => fully exploit the advancing technology.**
  - **Advance pipeline techniques**
  - **Bigger and more sophisticated on-chip caches**

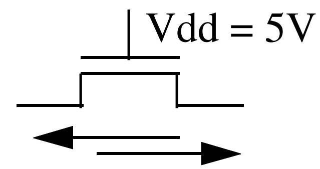
Basic Technology: CMOS

° **CMOS: Complementary Metal Oxide Semiconductor**

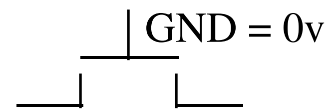
- **NMOS (N-Type Metal Oxide Semiconductor) transistors**
- **PMOS (P-Type Metal Oxide Semiconductor) transistors**

° **NMOS Transistor**

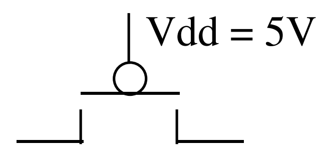
- **Apply a HIGH (Vdd, 5V) to its gate turns the transistor into a "conductor"**



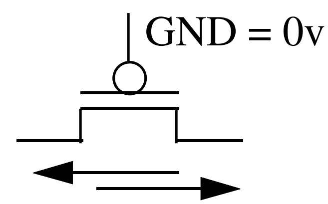
- **Apply a LOW (GND, 0v) to its gate shuts off the conduction path**



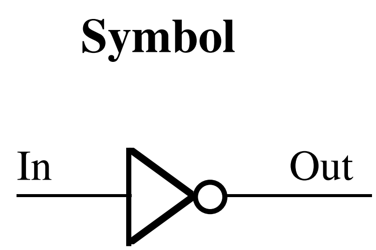
° **PMOS Transistor**

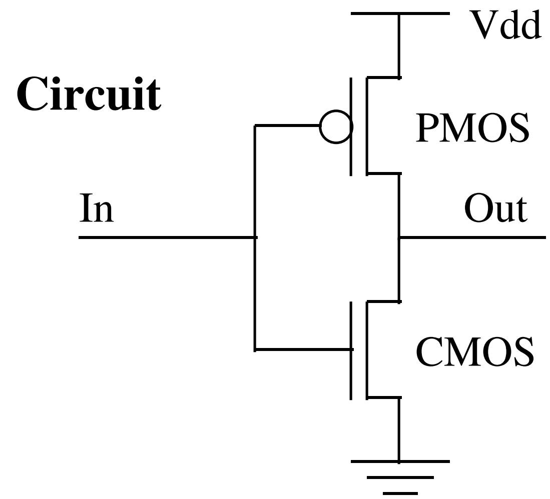


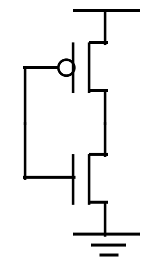
- **Apply a HIGH (Vdd, 5V) to its gate shuts off the conduction path**
- **Apply a LOW (GND, 0v) to its gate turns the transistor into a "conductor"**



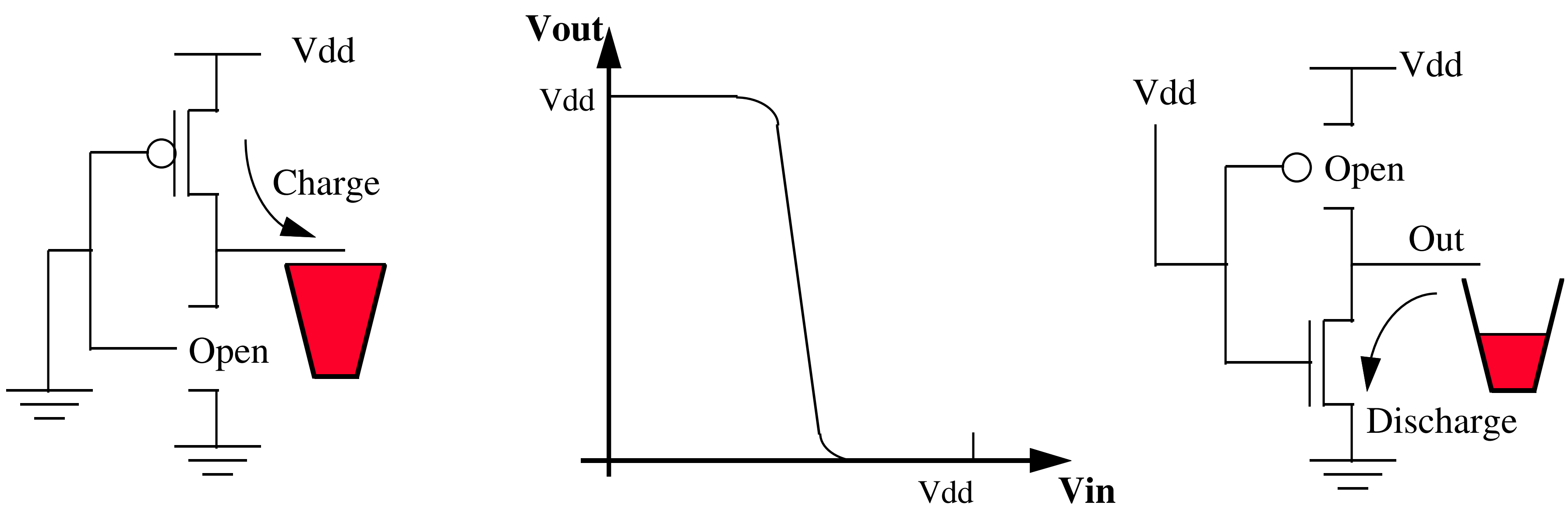
Basic Components: CMOS Inverter





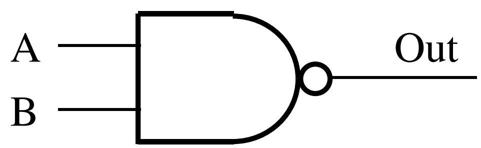


- Inverter Operation

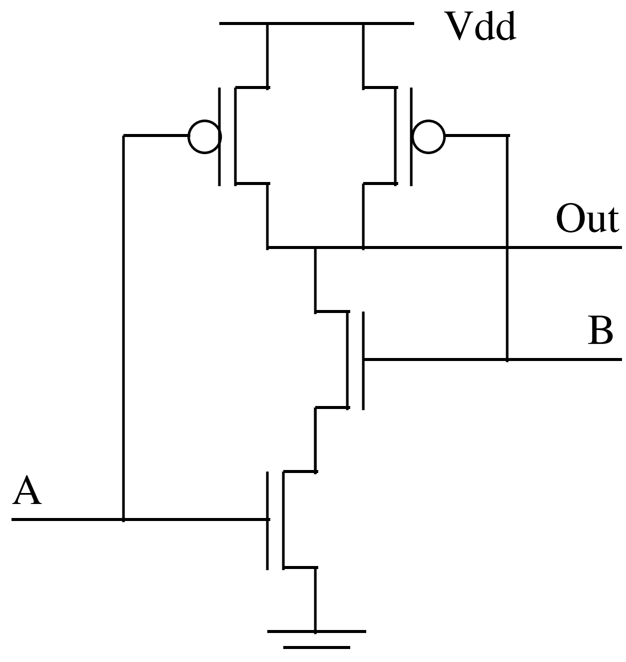


Basic Components: CMOS Logic Gates

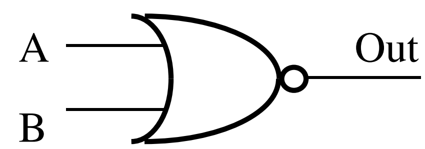
NAND Gate



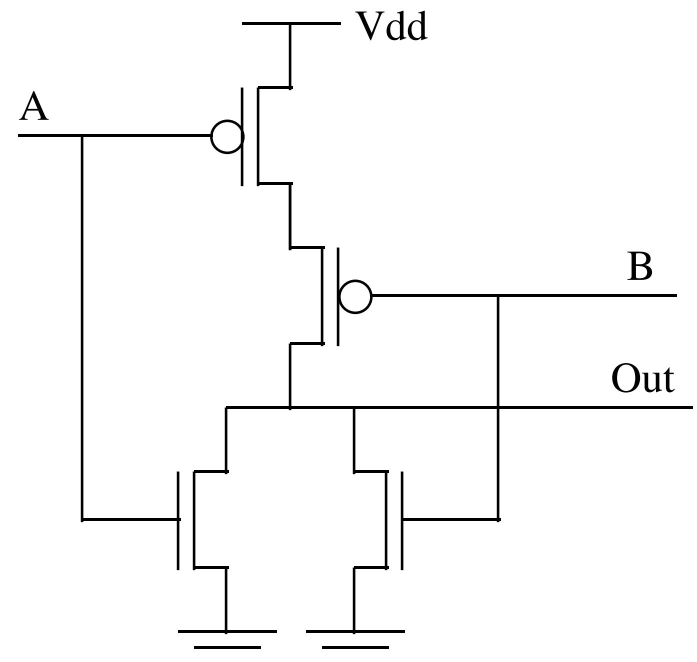
A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0



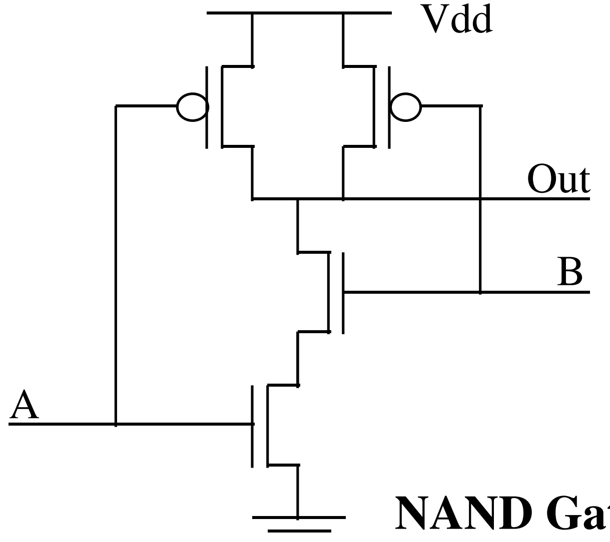
NOR Gate



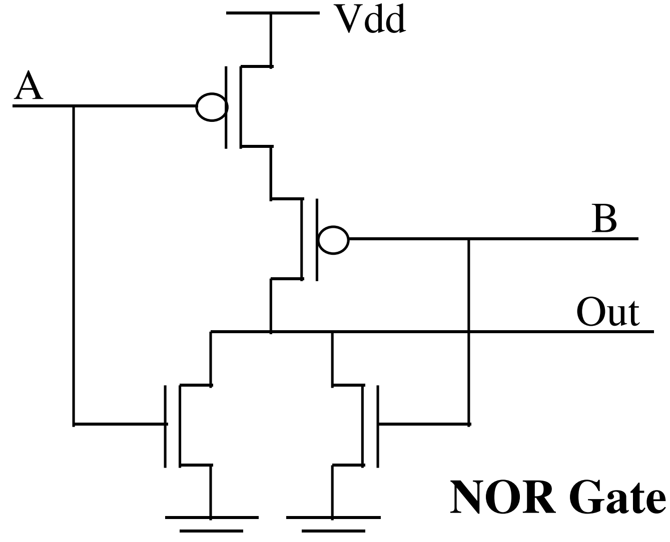
A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0



Gate Comparison



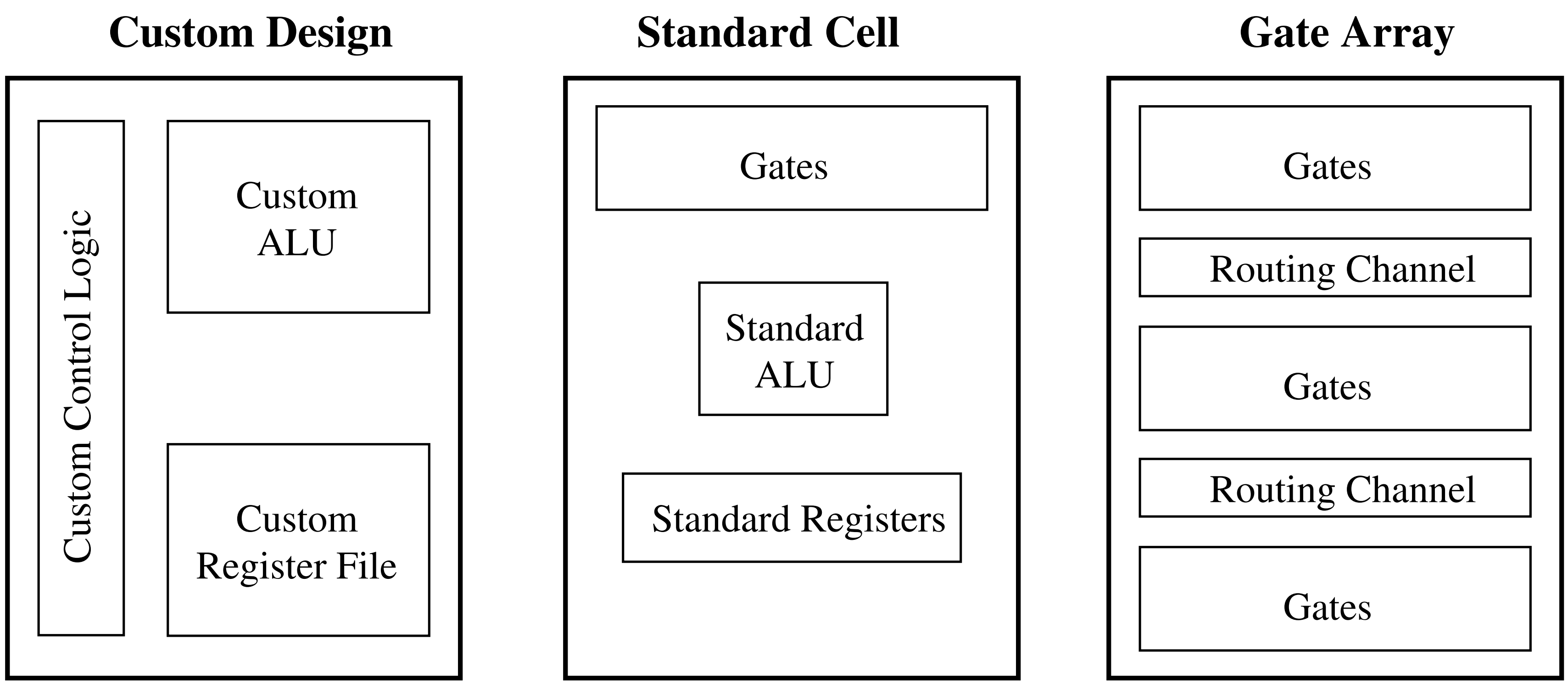
NAND Gate



NOR Gate

- If PMOS transistors is faster:
  - It is OK to have PMOS transistors in series
  - NOR gate is preferred
  - NOR gate is preferred also if H -> L is more critical than L -> H
- If NMOS transistors is faster:
  - It is OK to have NMOS transistors in series
  - NAND gate is preferred
  - NAND gate is preferred also if L -> H is more critical than H -> L

Range of Design Style

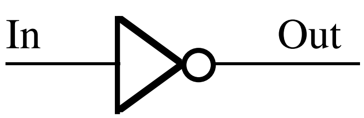


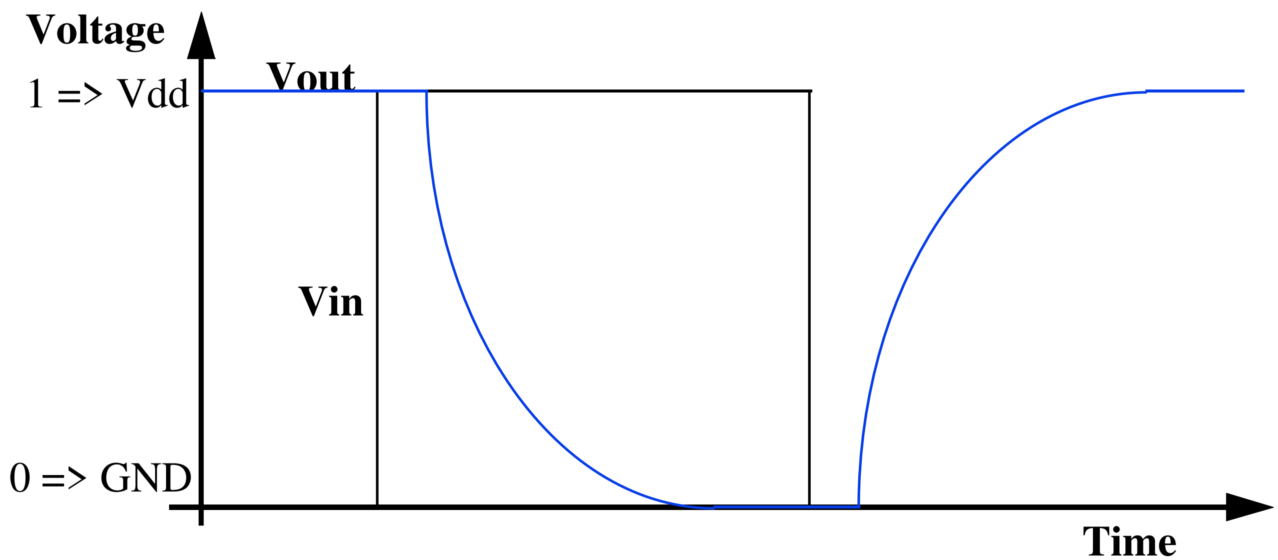
Questions and Administrative Matters

- CS152 Logic and Storage Components

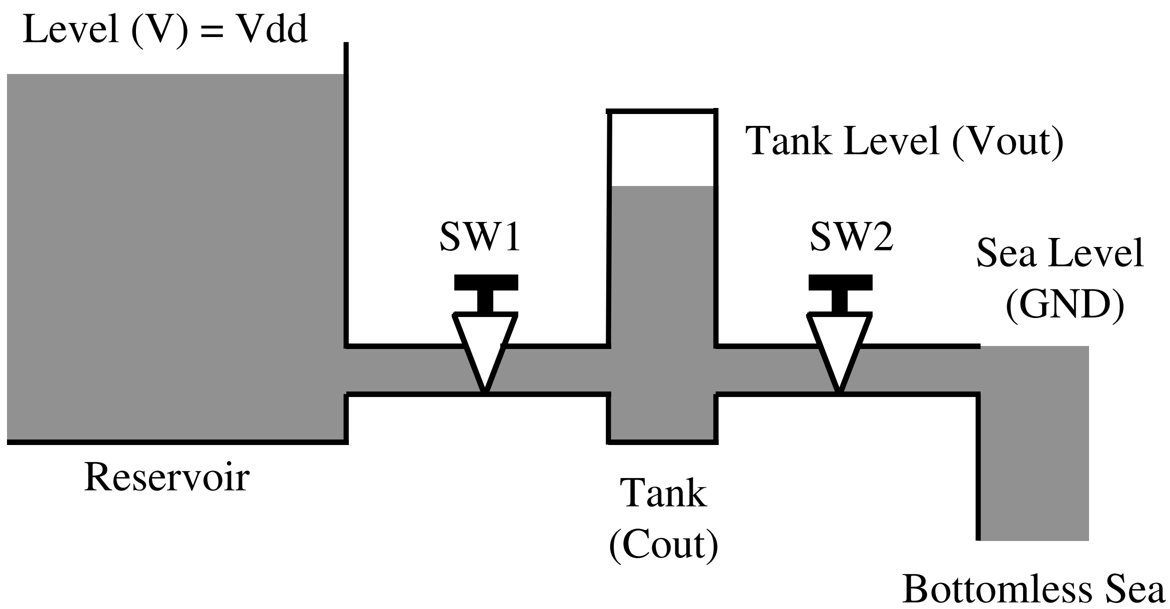
Ideal (CS) versus Reality (EE)

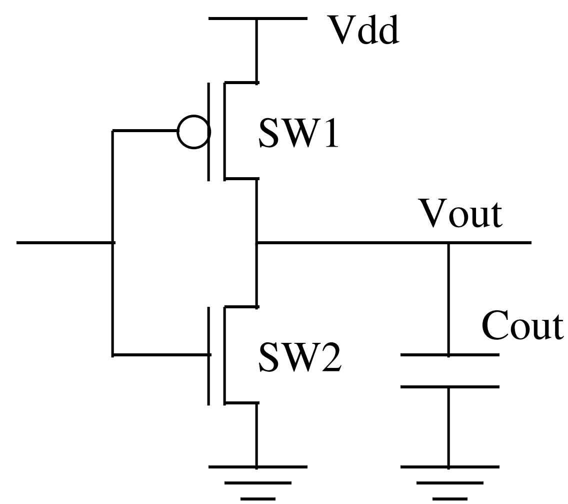
- When input 0 -> 1, output 1 -> 0 but NOT instantly
  - Output goes 1 -> 0: output voltage goes from Vdd (5v) to 0v
- When input 1 -> 0, output 0 -> 1 but NOT instantly
  - Output goes 0 -> 1: output voltage goes from 0v to Vdd (5v)
- Voltage does not like to change instantaneously





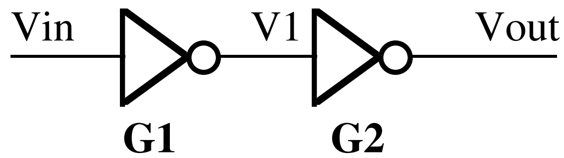
Fluid Timing Model

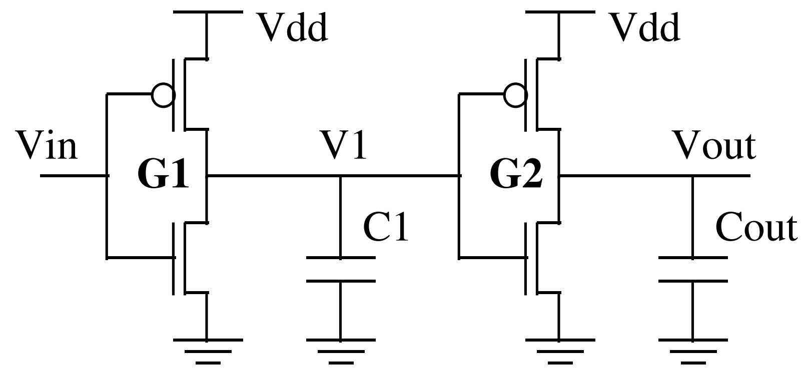


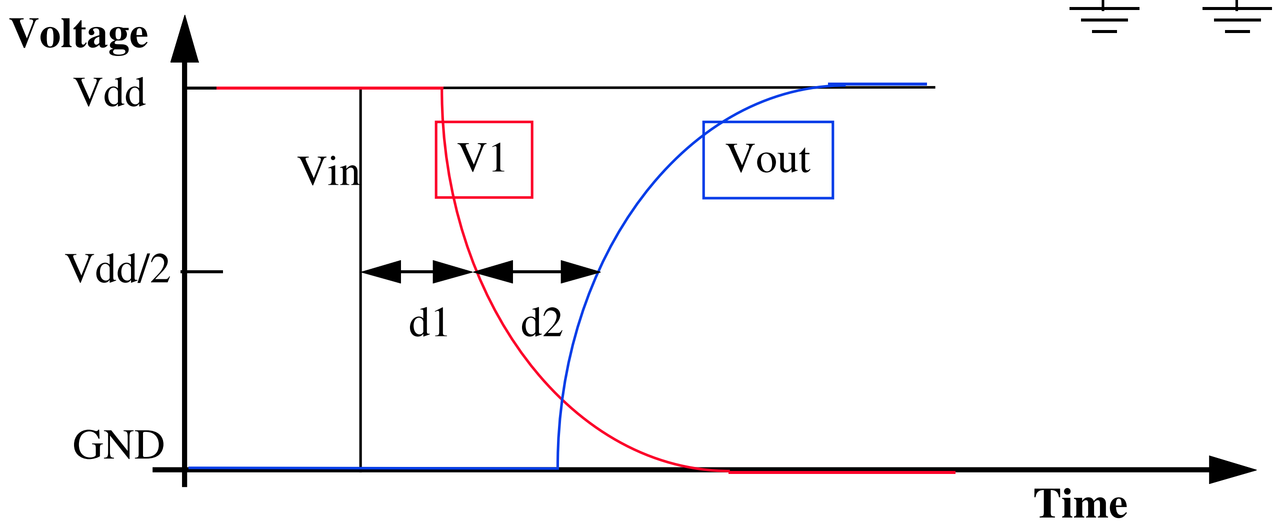


- Water <-> Electrical Charge   Tank Capacity <-> Capacitance (C)
- Water Level <-> Voltage   Water Flow <-> Charge Flowing (Current)
- Size of Pipes <-> Strength of Transistors (G)
- Time to fill up the tank ~ C / G

Series Connection

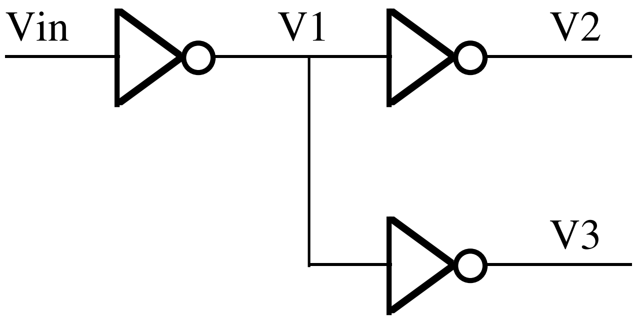


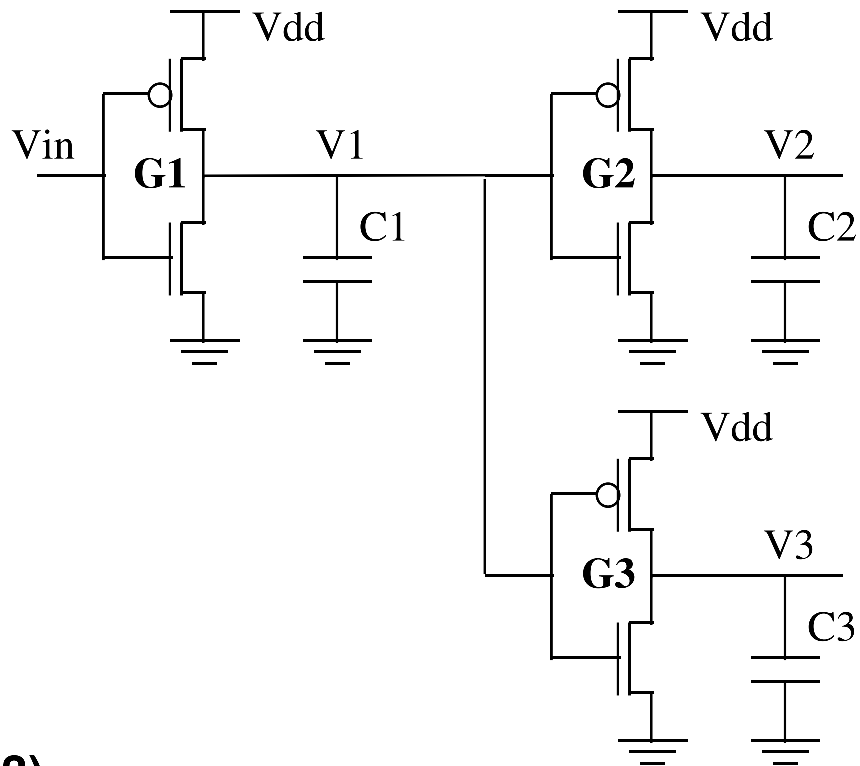




- Total Propagation Delay = Sum of individual delays = d1 + d2
- Capacitance C1 has two components:
  - Capacitance of the wire connecting the two gates
  - Input capacitance of the second inverter

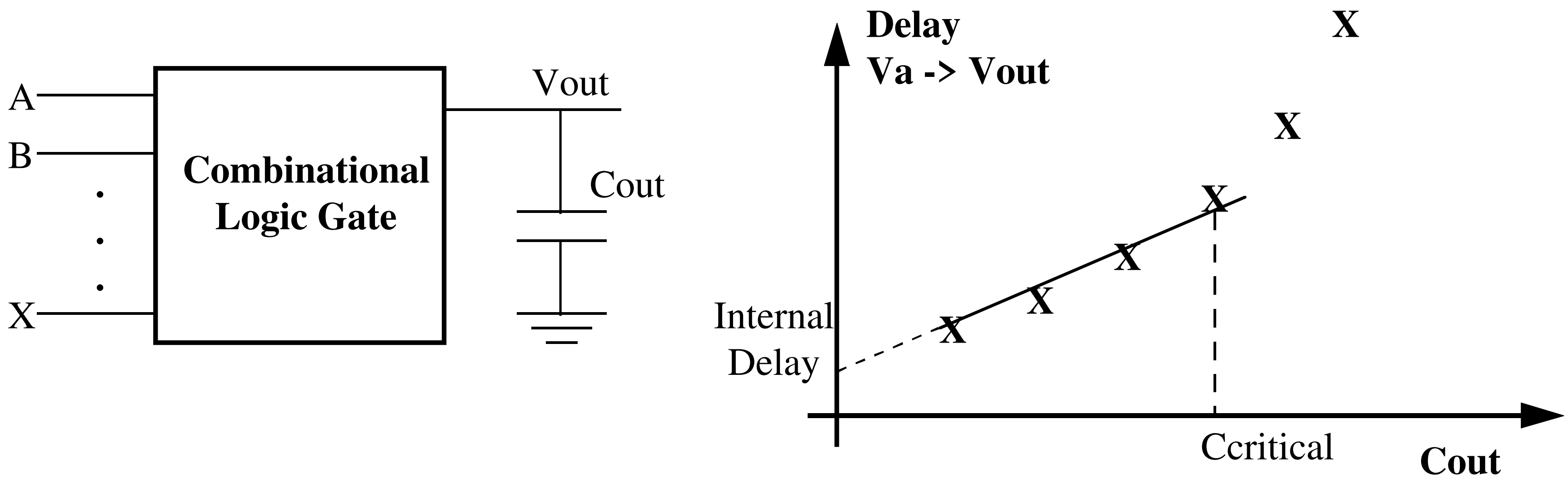
Parallel Connection





- Delay (Vin -> V2) ! = Delay (Vin -> V3)
  - Delay (Vin -> V2) = Delay (Vin -> V1) + Delay (V1 -> V2)
  - Delay (Vin -> V3) = Delay (Vin -> V1) + Delay (V1 -> V3)
- Critical Path = The longest among the N parallel paths
- C1 = Wire C + Cin of Gate 2 + Cin of Gate 3

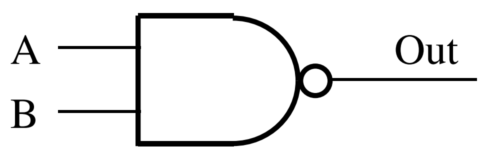
General Cell Delay Model



- Tied inputs B ... X such that a change in A will causes Vout to change
  - Example: If this is a AND gate, tie Input B, C, ... X to "1"
- Do not use this gate to drive any Cout > Ccritical
- Keep the model simple by using a linear equation:
  - Delay (A -> Out) = Internal Delay + (Load Dependent Delay) x Cout

Characterize a Gate

- Input capacitance for each input
- For each input-to-output path:
  - For each output transition type (H->L, L->H, H->Z, L->Z ... etc.)
    - Internal delay (ns)
    - Load dependent delay (ns / fF)
- Example: 2-input NAND Gate

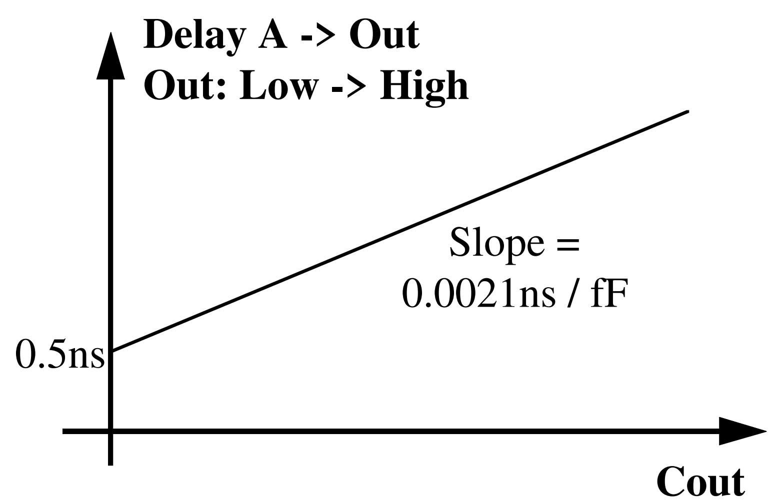


For A and B: Input Load = 61 fF

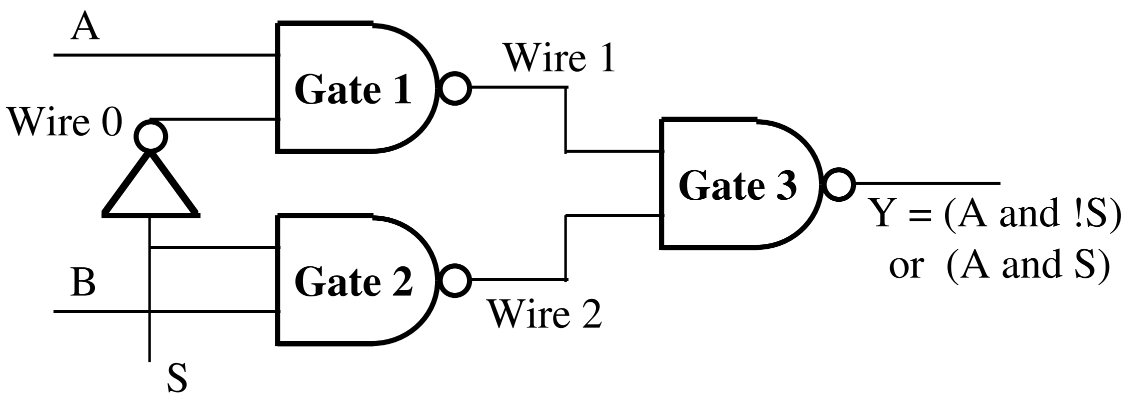
For either A -> Out or B -> Out:

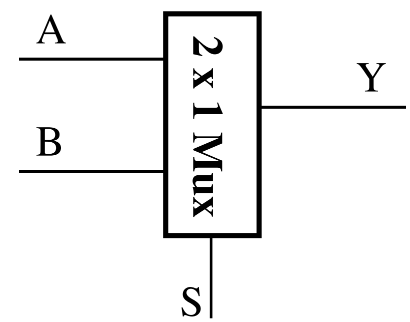
TPlh = 0.5ns   Tplhf = 0.0021ns / fF

TPhl = 0.1ns   TPhlf = 0.0020ns / fF



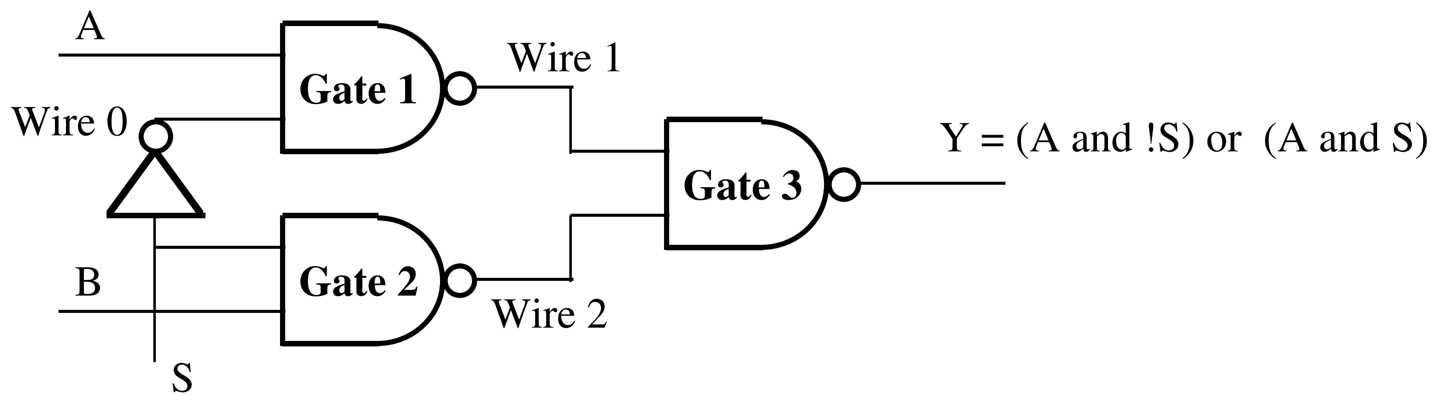
A Specific Example: 2 to 1 MUX





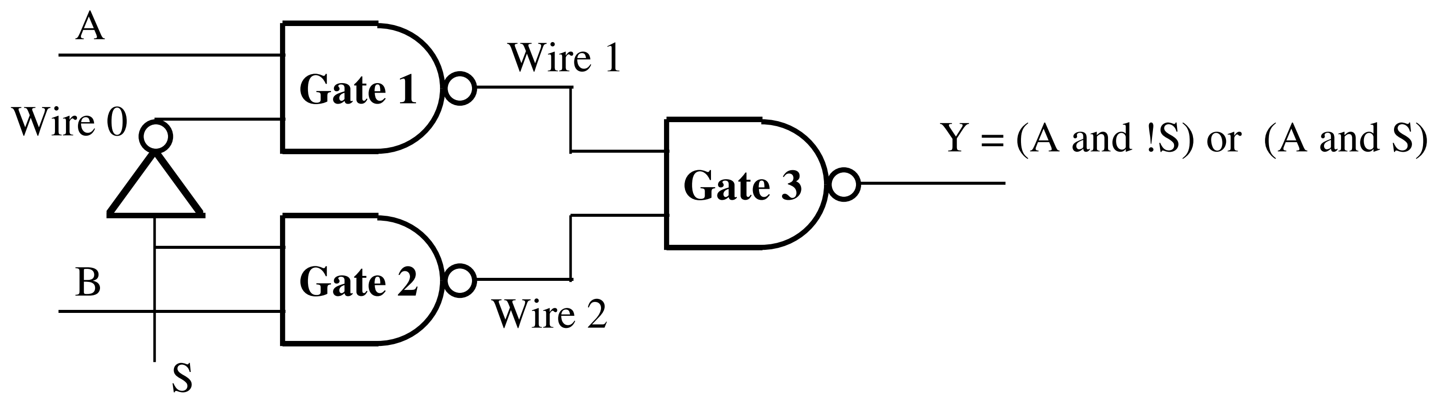
- **Input Load**
  - **A, B: I.L. (NAND) = 61 fF**
  - **S: I.L. (INV) + I.L. (NAND) = 50 fF + 61 fF = 111 fF**
- **Load Dependent Delay: Same as Gate 3**
  - **TAYlhf = 0.021 ns / fF   TAYhlf = 0.020 ns / fF**
  - **TBYlhf = 0.021 ns / fF   TBYhlf = 0.020 ns / fF**
  - **TSYlhf = 0.021 ns / fF   TSYlhf = 0.020 ns / fF**

2 to 1 MUX: Internal Delay Calculation



- **Internal Delay:**
  - **A to Y: I.D. G1 + (Wire 1 C + G3 Input C) * L.D.D G1 + I.D. G3**
  - **B to Y: I.D. G2 + (Wire 2 C + G3 Input C) * L.D.D. G2 + I.D. G3**
  - **S to Y (Worst Case): I.D. Inv + (Wire 0 C + G1 Input C) * L.D.D. Inv + Internal Delay A to Y**
- **We can approximate the effect of "Wire 1 C" by:**
  - **Assume Wire 1 has the same C as all the gate C attache to it.**
  - **Total C Gate 1 need to drive: 2.0 x Input C of Gate 3**

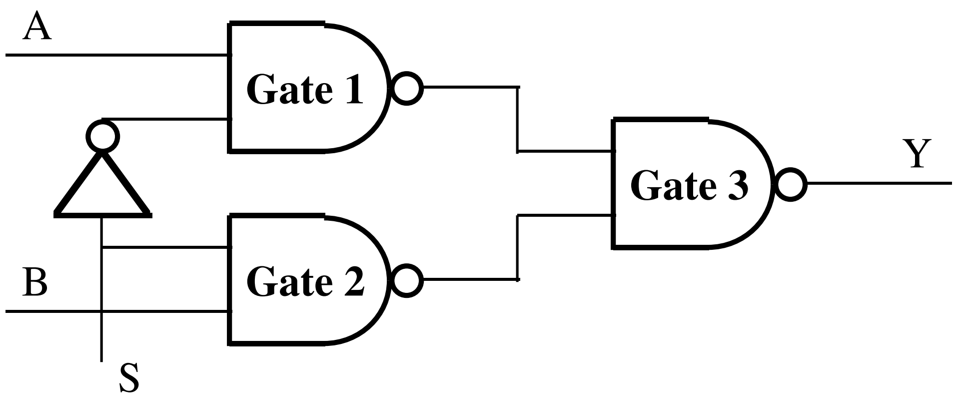
2 to 1 MUX: Internal Delay Calculation (continue)

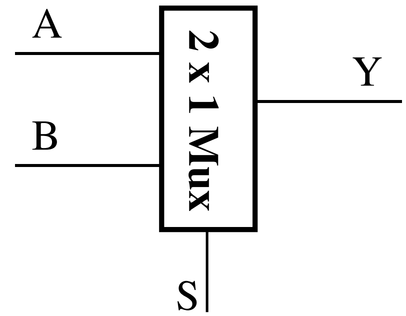


- **Internal Delay:**
  - **A to Y: I.D. G1 + (Wire 1 C + G3 Input C) * L.D.D G1 + I.D. G3**
  - **B to Y: I.D. G2 + (Wire 2 C + G3 Input C) * L.D.D. G2 + I.D. G3**
  - **S to Y (Worst Case): I.D. Inv + (Wire 0 C + G1 Input C) * L.D.D. Inv + Internal Delay A to Y**

- **Specific Example:**
  - **TAYlh = TPhl G1 + (2.0 * 61 fF) * TPhl G1 + TPlh G3**
    = 0.1ns + 122 fF * 0.0020ns/fF + 0.5ns = 0.844ns

Abstraction: 2 to 1 MUX

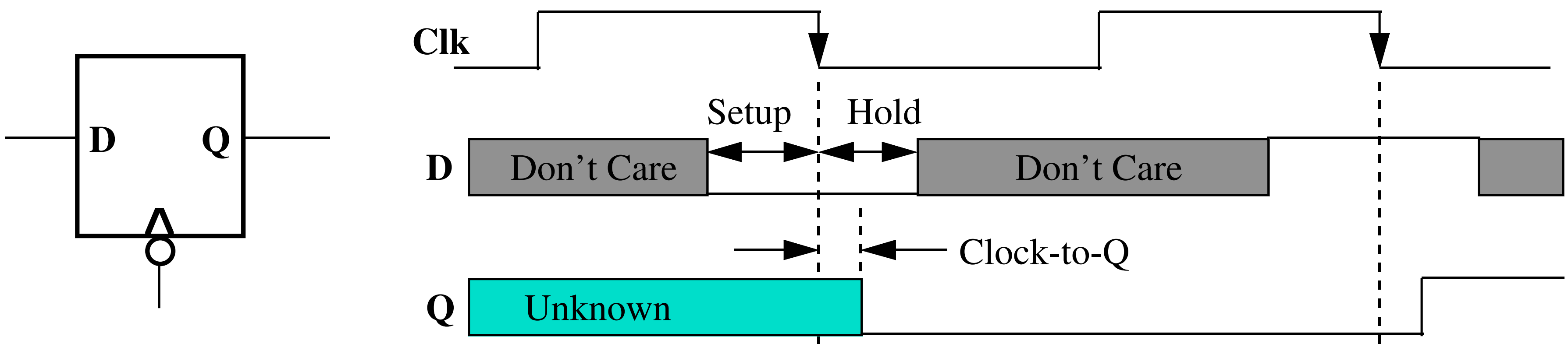




- Input Load: A = 61 fF, B = 61 fF, S = 111 fF
- Load Dependent Delay:
  - TAYlhf = 0.021 ns / fF TAYhlf = 0.020 ns / fF
  - TBYlhf = 0.021 ns / fF TBYhlf = 0.020 ns / fF
  - TSYlhf = 0.021 ns / fF TSYlhf = 0.020 ns / f F
- Internal Delay:
  - TAYlh = TPhl G1 + (2.0 * 61 fF) * TPhl G1 + TPlh G3

    = 0.1ns + 122 fF * 0.0020ns/fF + 0.5ns = 0.844ns
  - Fun Exercises: TAYhl, TBYlh, TSYlh, TSYlh

Storage Element's Timing Model



- Setup Time: Input must be stable BEFORE the trigger clock edge
- Hold Time: Input must REMAIN stable after the trigger clock edge
- Clock-to-Q time:
  - Output cannot change instantaneously at the trigger clock edge
  - Similar to delay in logic gates, two components:
    - Internal Clock-to-Q
    - Load dependent Clock-to-Q

Break (5 Minutes)

CS152 Logic Elements

- NAND2, NAND3, NAND 4
- NOR2, NOR3, NOR4
- INV1x (normal inverter)
- INV4x (inverter with big output drive)

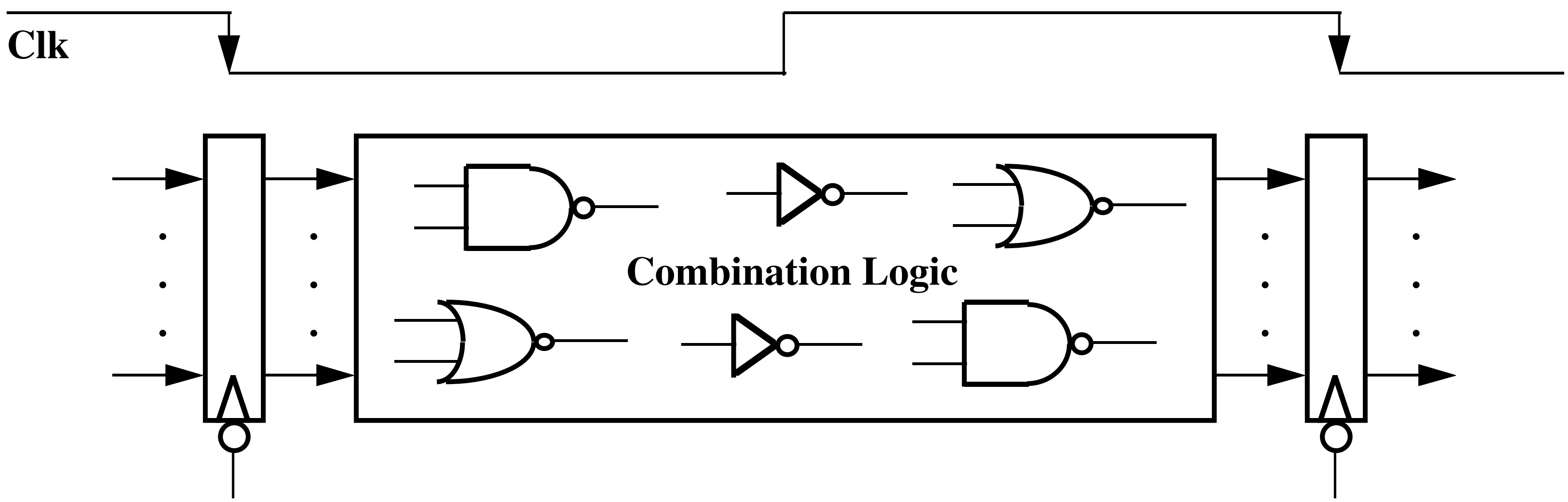
CS152 Logic Elements (Continue)

- XOR2
- XNOR2
- PWR: Source of 1's
- GND: Source of 0's

CS152 Storage Element

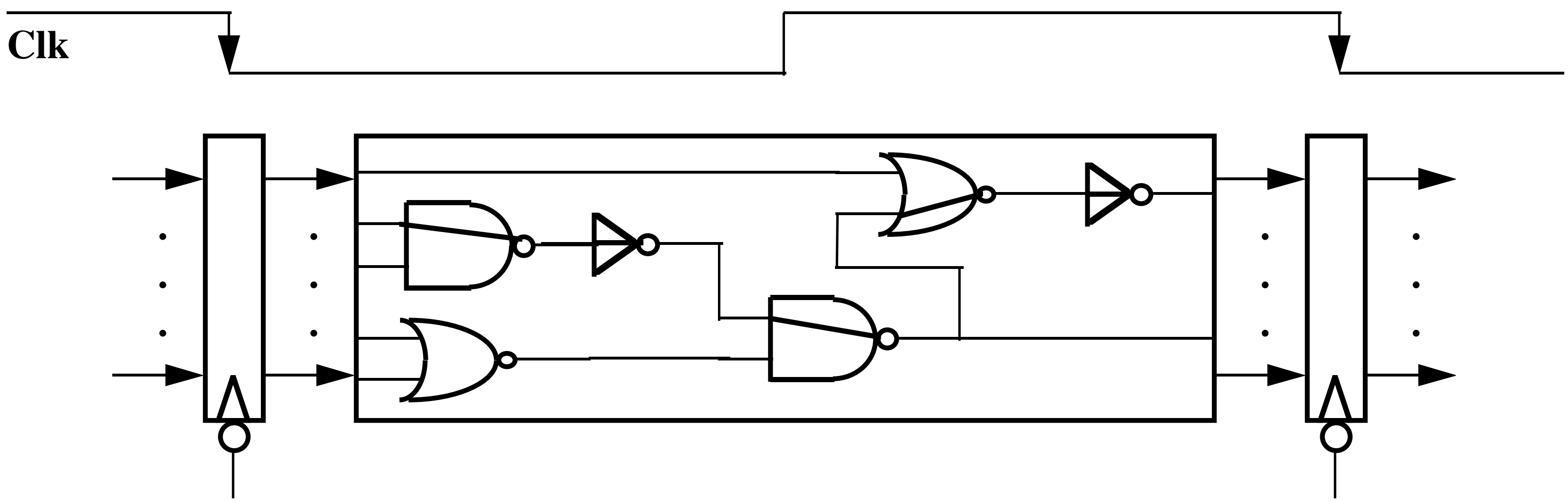
- D flip flop with negative edge triggered

Clocking Methodology



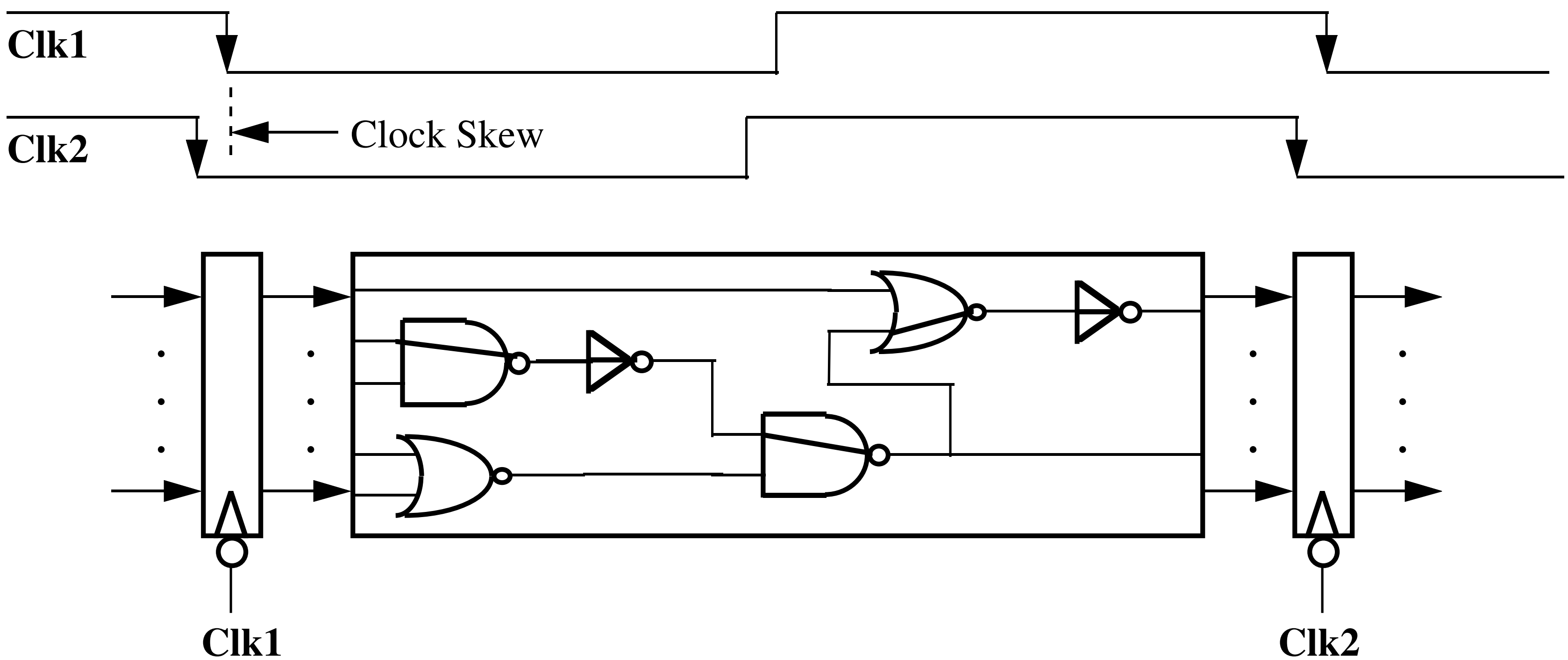
- All storage elements are clocked by the same clock edge
- The combination logic block's:
  - Inputs are updated at each clock tick
  - All outputs MUST be stable before the next clock tick

Critical Path & Cycle Time



- Critical path: the slowest path between any two storage devices
- Cycle time is a function of the critical path
- More specifically, the cycle time must be greater than:
  - Clock-to-Q + Longest Path through the Combination Logic + Setup

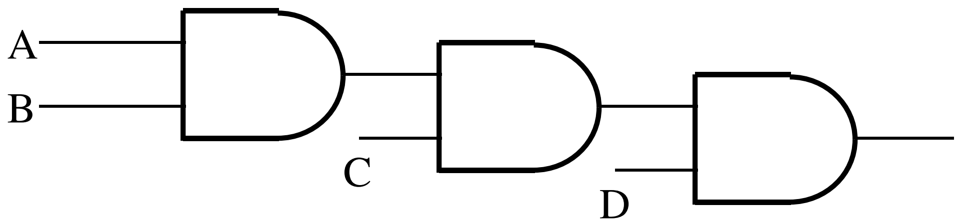
Clock Skew's Effect on Cycle Time

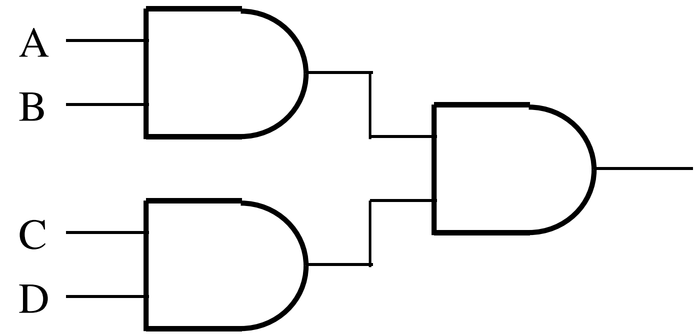


- The worst case scenario for cycle time consideration:
  - The input register sees CLK1
  - The output register sees CLK2
- Cycle Time = CLK-to-Q + Longest Delay Path + Setup + Clock Skew

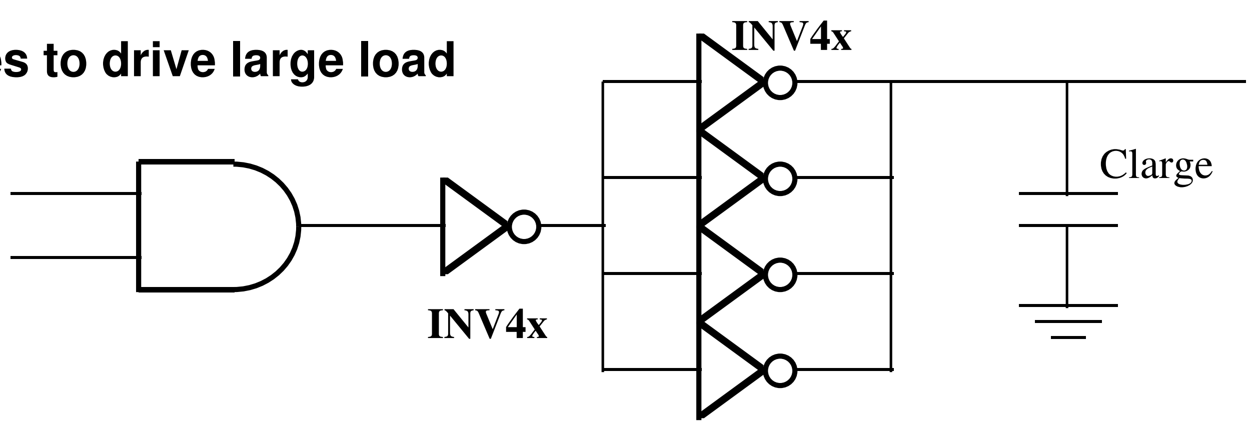
Tricks to Reduce Cycle Time

- Reduce the number of gate levels

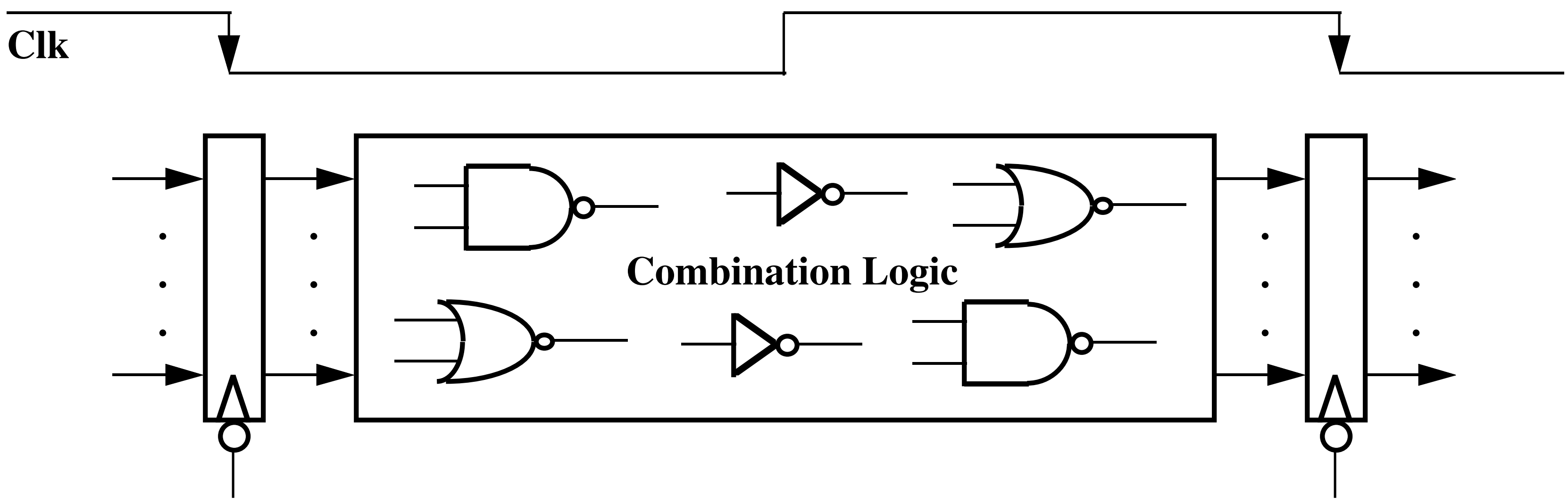




- Pay attention to loading
  - One gate driving many gates is a bad idea
  - Avoid using a small gate to drive a long wire
- Use multiple stages to drive large load

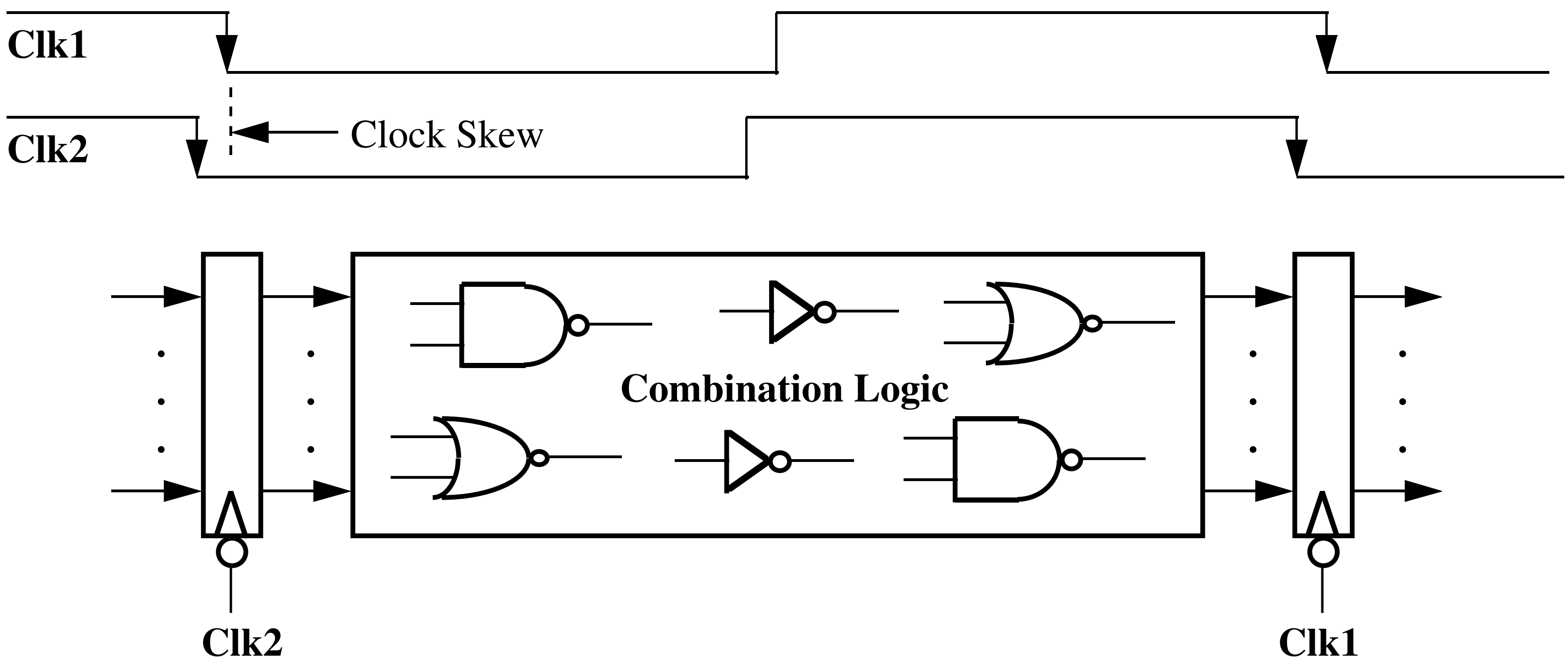


How to Avoid Hold Time Violation?



- Hold time requirement:
    - Input to register must NOT change immediately after the clock tick
- This is usually easy to meet in the “edge trigger” clocking scheme
- CLK-to-Q + Shortest Delay Path must be greater than Hold Time

Clock Skew's Effect on Hold Time



- The worst case scenario for hold time consideration:
  - The input register sees CLK2
  - The output register sees CLK1
- (CLK-to-Q + Shortest Delay Path - Clock Skew) > Hold Time

Summary

- Performance and Technology Trends
  - Keep the design simple to take advantage of the latest technology
  - CMOS inverter and CMOS logic gates
- Delay Modeling and Gate Characterization
  - Delay = Internal Delay + (Load Dependent Delay x Output Load)
- Clocking Methodology and Timing Considerations
  - Simplest clocking methodology
    - All storage elements use the SAME clock edge
  - Cycle Time = CLK-to-Q + Longest Delay Path + Setup + Clock Skew
  - CLK-to-Q + Shortest Delay Path - Clock Skew < Hold Time

To Get More Information

- A Classic Book that Started it All:
  - Carver Mead and Lynn Conway, “Introduction to VLSI Systems,” Addison-Wesley Publishing Company, October 1980.
- A Good VLSI Circuit Design Book
  - Lance Glasser & Daniel Dobberpuhl, “The Design and Analysis of VLSI Circuits,” Addison-Wesley Publishing Company, 1985.
    - Mr. Dobberpuhl is responsible for the DEC Alpha chip design.
- A Book by Dean Hodges:
  - David Hodges & Horace Jackson, “Analysis and Design of Digital Integrated Circuits,” McGraw-Hill Book Company, 1983.