

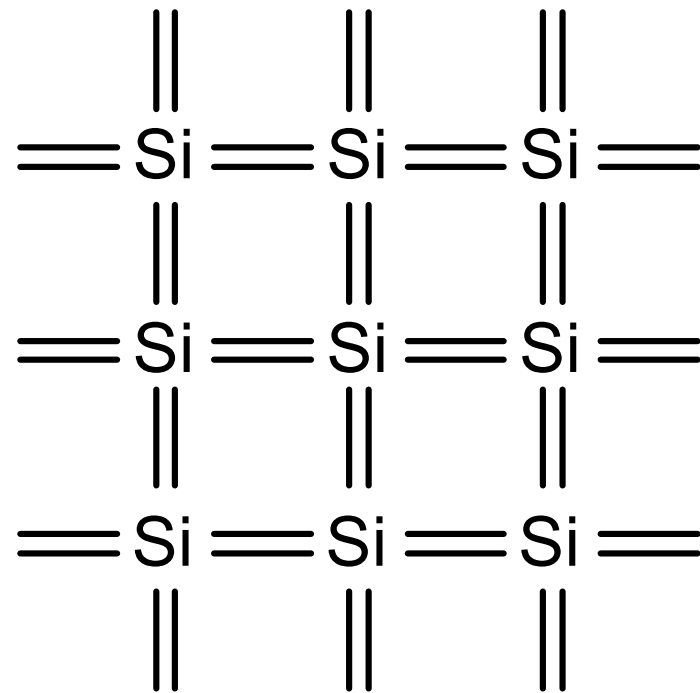
Week_1: Introduction

Introduction

- ❑ Integrated circuits: many transistors on one chip.
- ❑ *Very Large Scale Integration (VLSI)*: bucketloads!
- ❑ *Complementary Metal Oxide Semiconductor*
 - Fast, cheap, low power transistors
- ❑ Today: How to build your own simple CMOS chip
 - CMOS transistors
 - Building logic gates from transistors
 - Transistor layout and fabrication
- ❑ Rest of the course: How to build a good CMOS chip

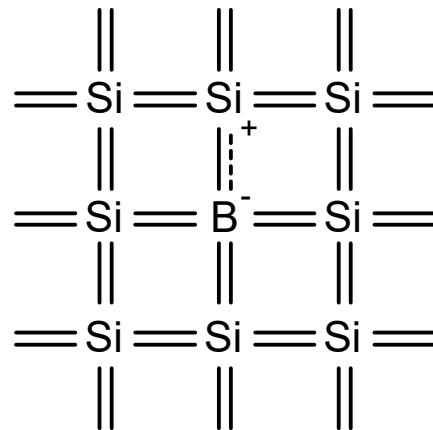
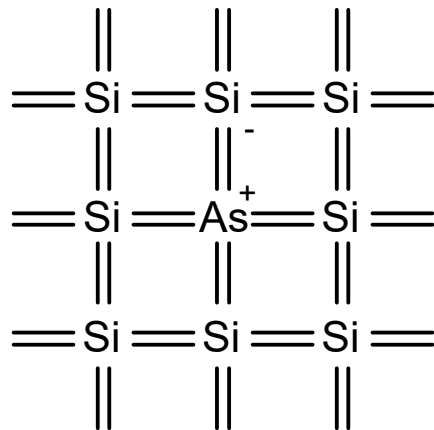
Silicon Lattice

- ❑ Transistors are built on a silicon substrate
- ❑ Silicon is a Group IV material
- ❑ Forms crystal lattice with bonds to four neighbors



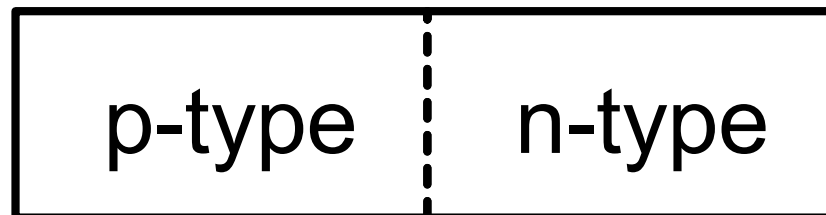
Dopants

- ❑ Silicon is a semiconductor
- ❑ Pure silicon has no free carriers and conducts poorly
- ❑ Adding dopants increases the conductivity
- ❑ Group V: extra electron (n-type)
- ❑ Group III: missing electron, called hole (p-type)

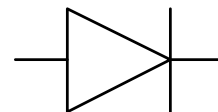


p-n Junctions

- ❑ A junction between p-type and n-type semiconductor forms a diode.
- ❑ Current flows only in one direction

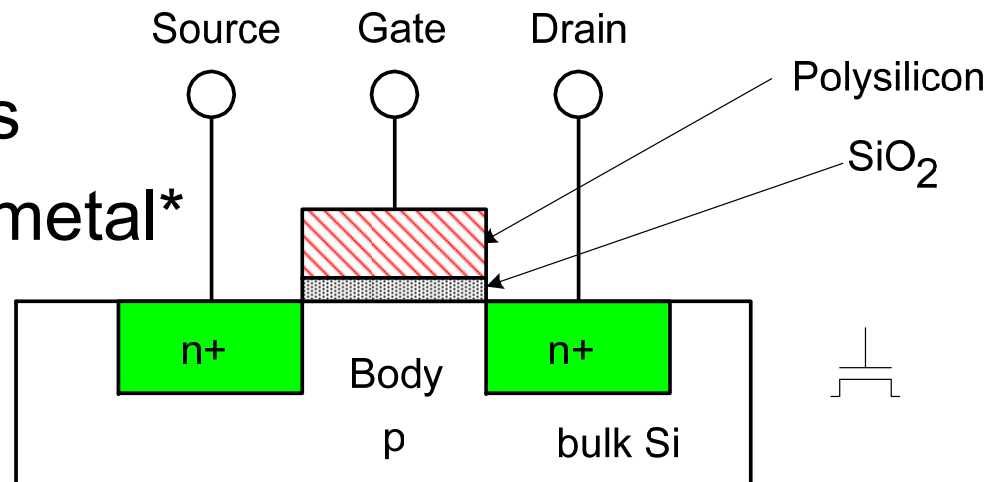


anode cathode



nMOS Transistor

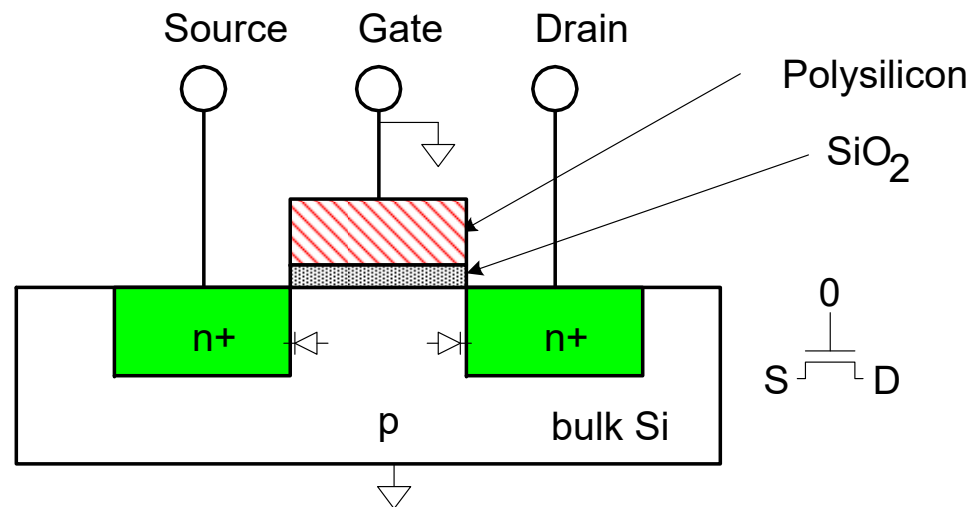
- ❑ Four terminals: gate, source, drain, body
- ❑ Gate – oxide – body stack looks like a capacitor
 - Gate and body are conductors
 - SiO_2 (oxide) is a very good insulator
 - Called metal – oxide – semiconductor (MOS) capacitor
 - Even though gate is no longer made of metal*



* Metal gates are returning today!

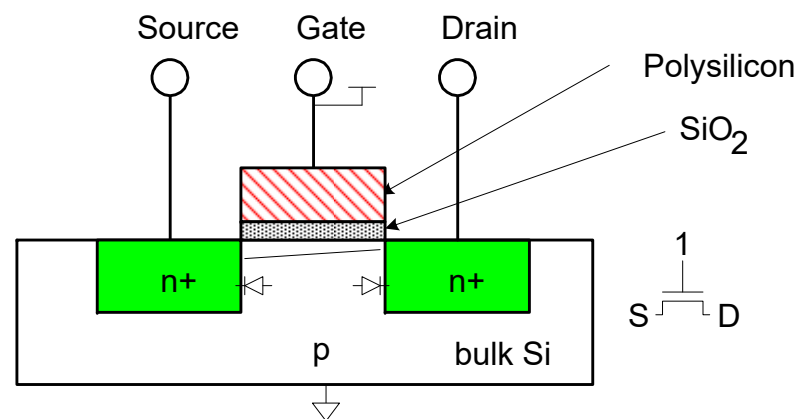
nMOS Operation

- ❑ Body is usually tied to ground (0 V)
- ❑ When the gate is at a low voltage:
 - P-type body is at low voltage
 - Source-body and drain-body diodes are OFF
 - No current flows, transistor is OFF



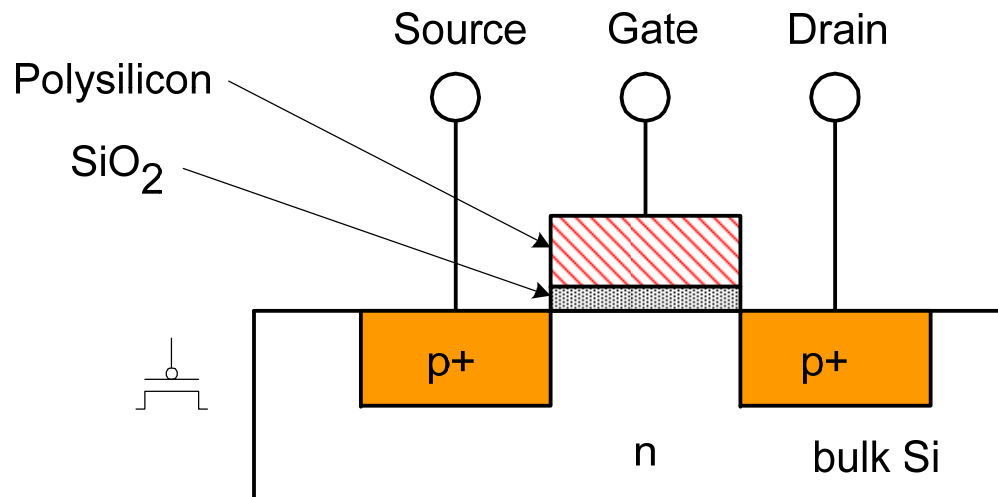
nMOS Operation Cont.

- When the gate is at a high voltage:
 - Positive charge on gate of MOS capacitor
 - Negative charge attracted to body
 - Inverts a channel under gate to n-type
 - Now current can flow through n-type silicon from source through channel to drain, transistor is ON



pMOS Transistor

- Similar, but doping and voltages reversed
 - Body tied to high voltage (V_{DD})
 - Gate low: transistor ON
 - Gate high: transistor OFF
 - Bubble indicates inverted behavior

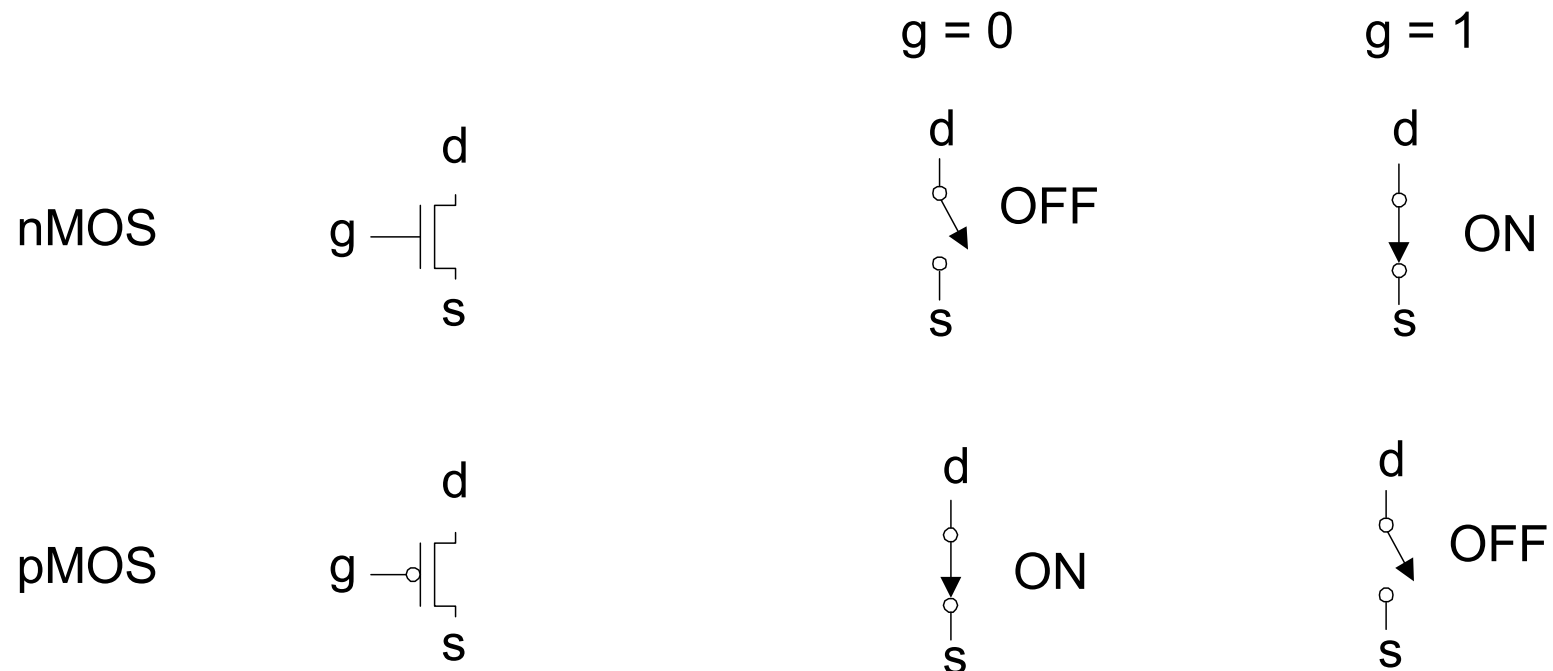


Power Supply Voltage

- $GND = 0\text{ V}$
- In 1980's, $V_{DD} = 5\text{V}$
- V_{DD} has decreased in modern processes
 - High V_{DD} would damage modern tiny transistors
 - Lower V_{DD} saves power
- $V_{DD} = 3.3, 2.5, 1.8, 1.5, 1.2, 1.0, \dots$

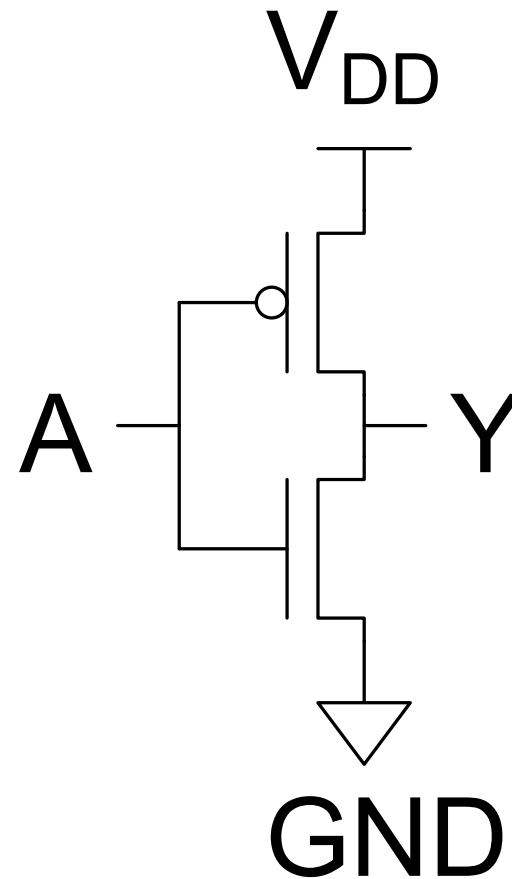
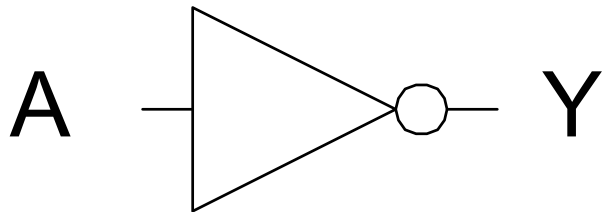
Transistors as Switches

- We can view MOS transistors as electrically controlled switches
- Voltage at gate controls path from source to drain



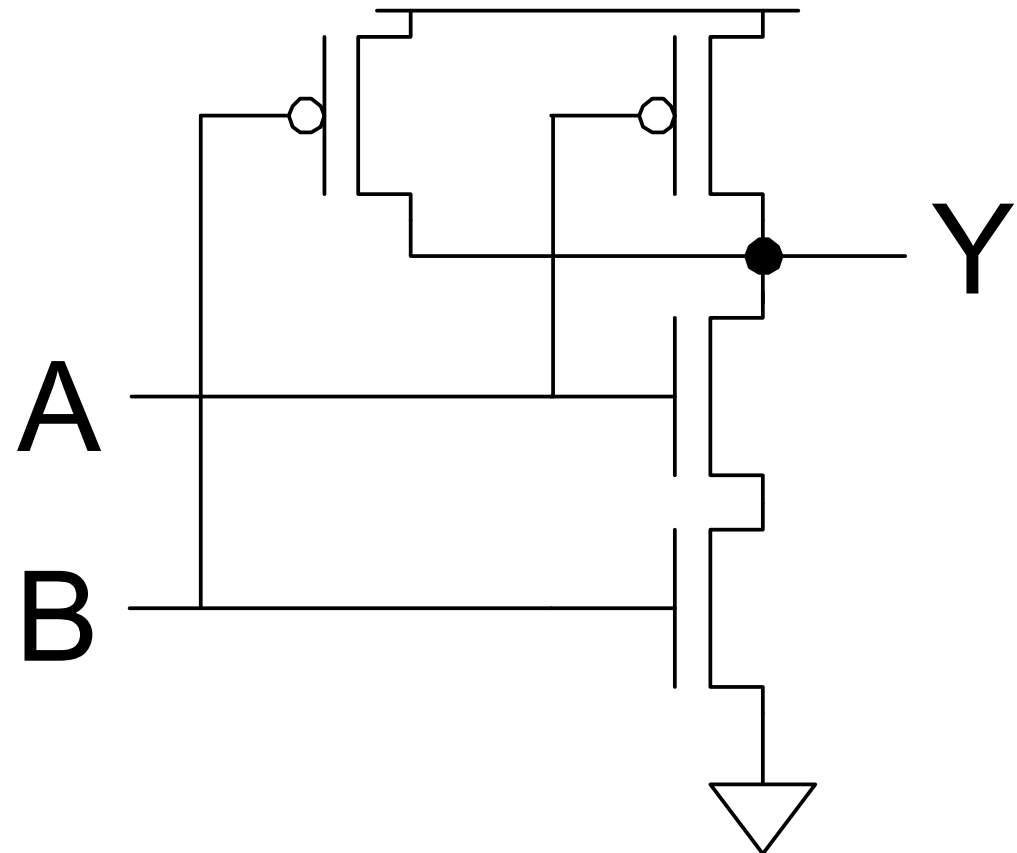
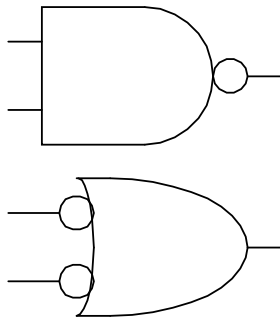
CMOS Inverter

A	Y



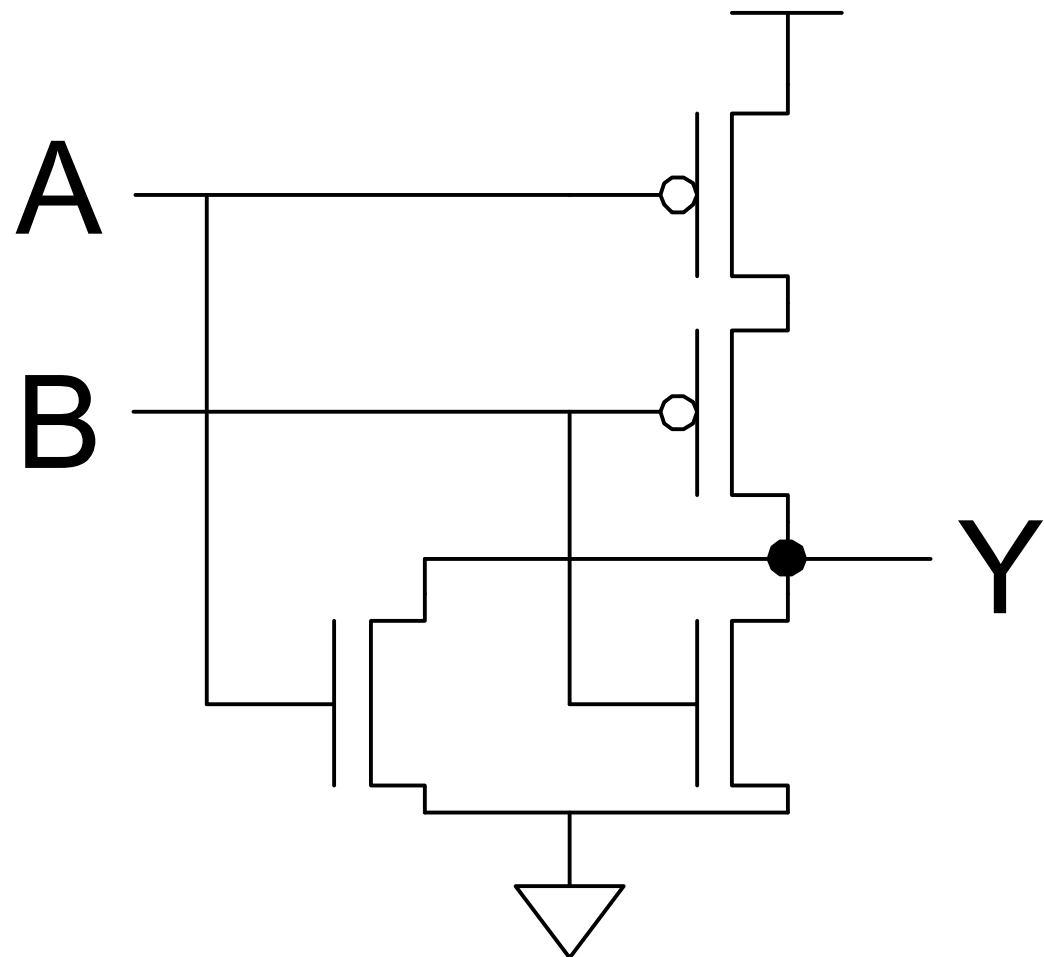
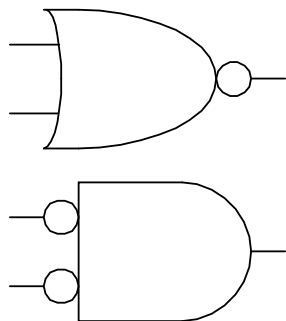
CMOS NAND Gate

A	B	Y
0	0	
0	1	
1	0	
1	1	



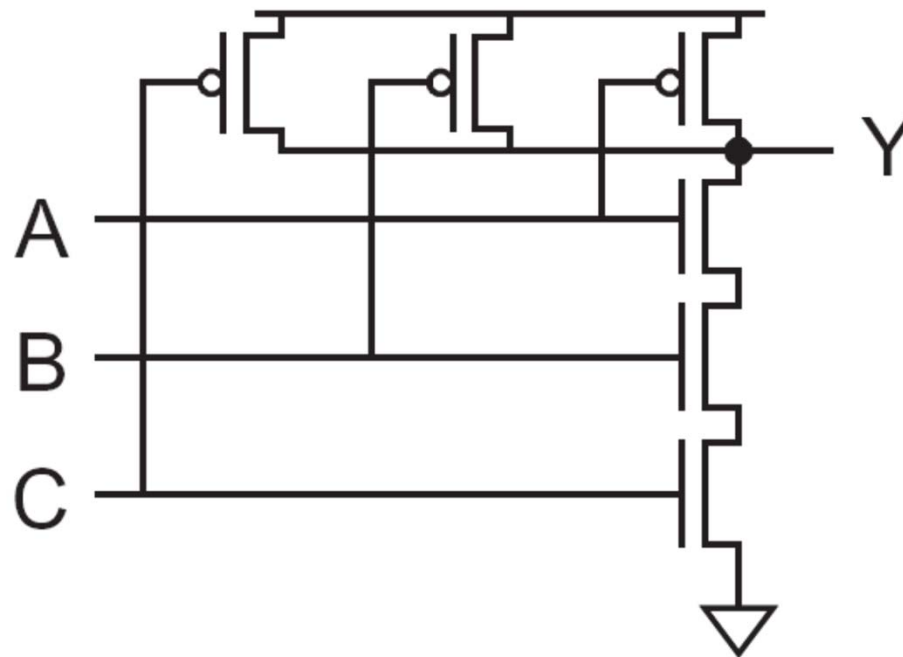
CMOS NOR Gate

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0



3-input NAND Gate

- ❑ Y pulls low if ALL inputs are 1
- ❑ Y pulls high if ANY input is 0

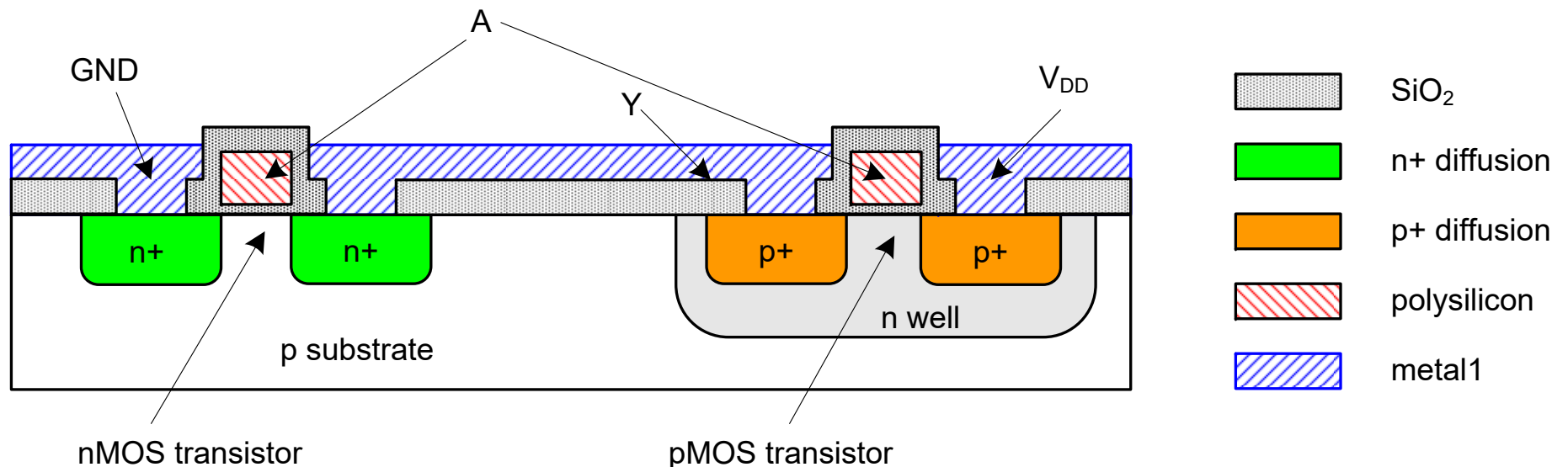


CMOS Fabrication

- ❑ CMOS transistors are fabricated on silicon wafer
- ❑ Lithography process similar to printing press
- ❑ On each step, different materials are deposited or etched
- ❑ Easiest to understand by viewing both top and cross-section of wafer in a simplified manufacturing process

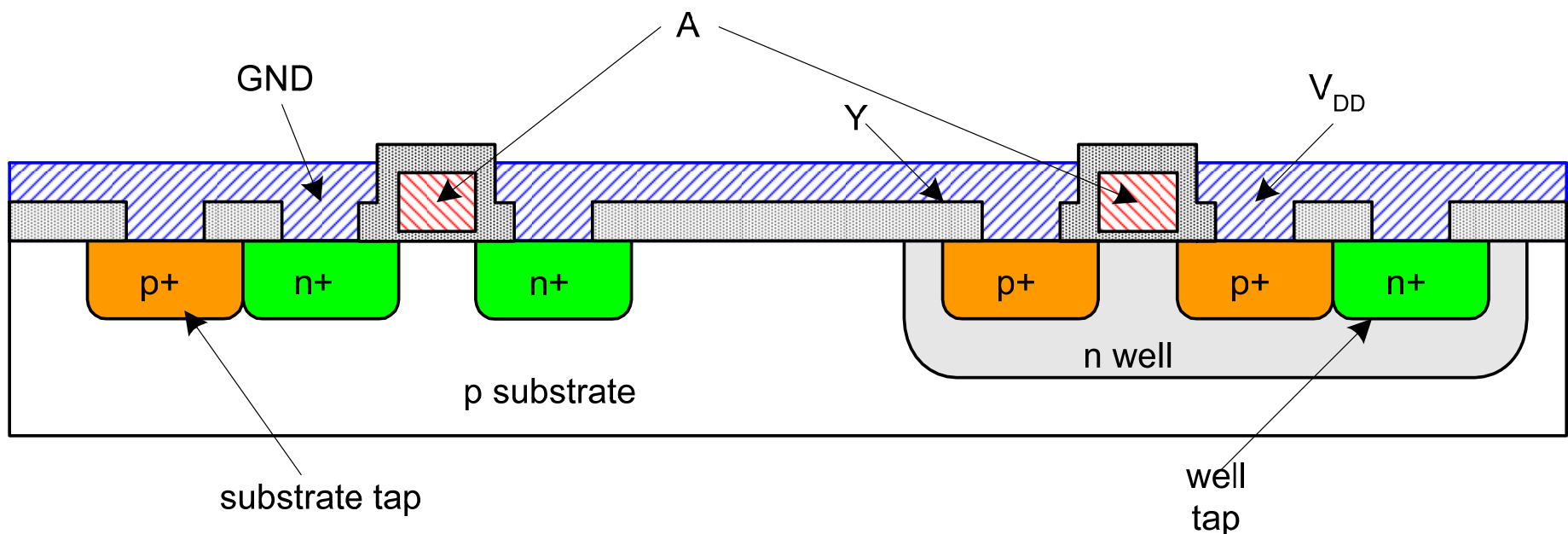
Inverter Cross-section

- Typically use p-type substrate for nMOS transistors
- Requires n-well for body of pMOS transistors



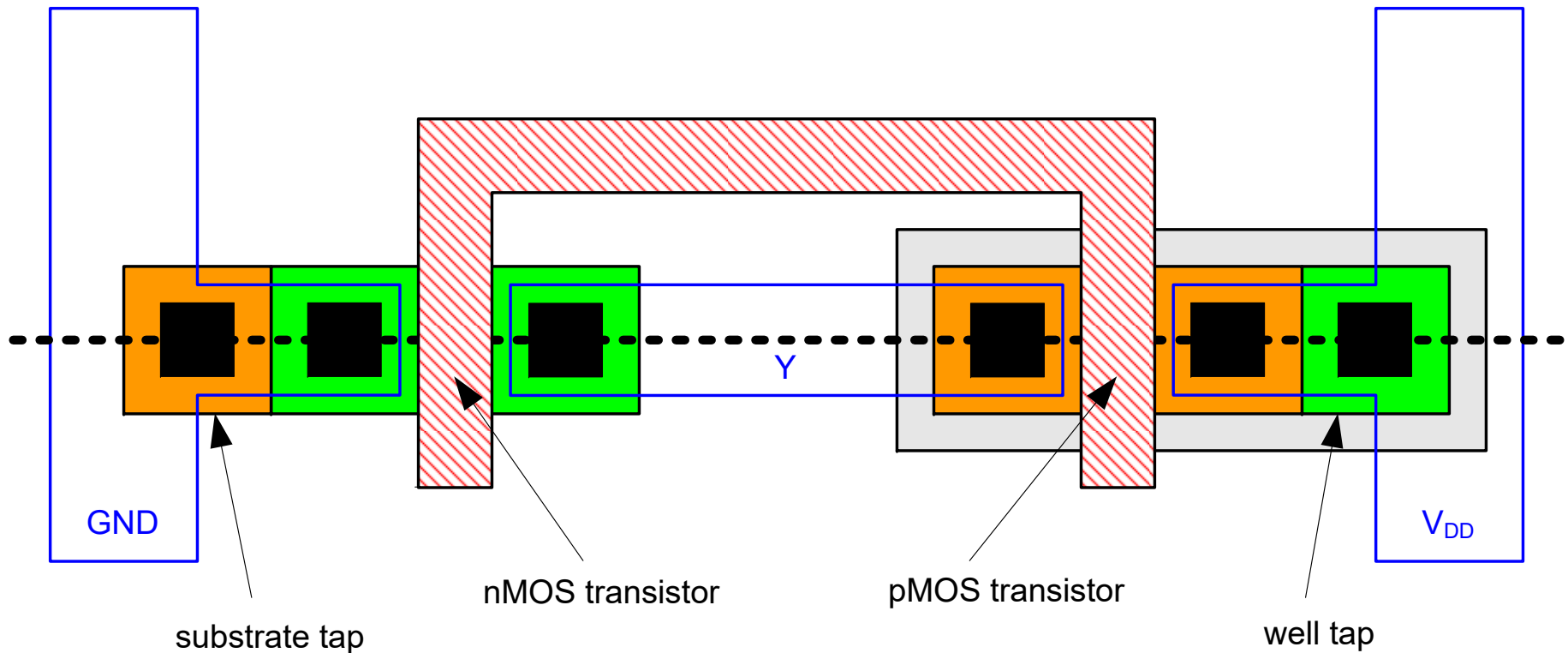
Well and Substrate Taps

- ❑ Substrate must be tied to GND and n-well to V_{DD}
- ❑ Metal to lightly-doped semiconductor forms poor connection called Shottky Diode
- ❑ Use heavily doped well and substrate contacts / taps



Inverter Mask Set

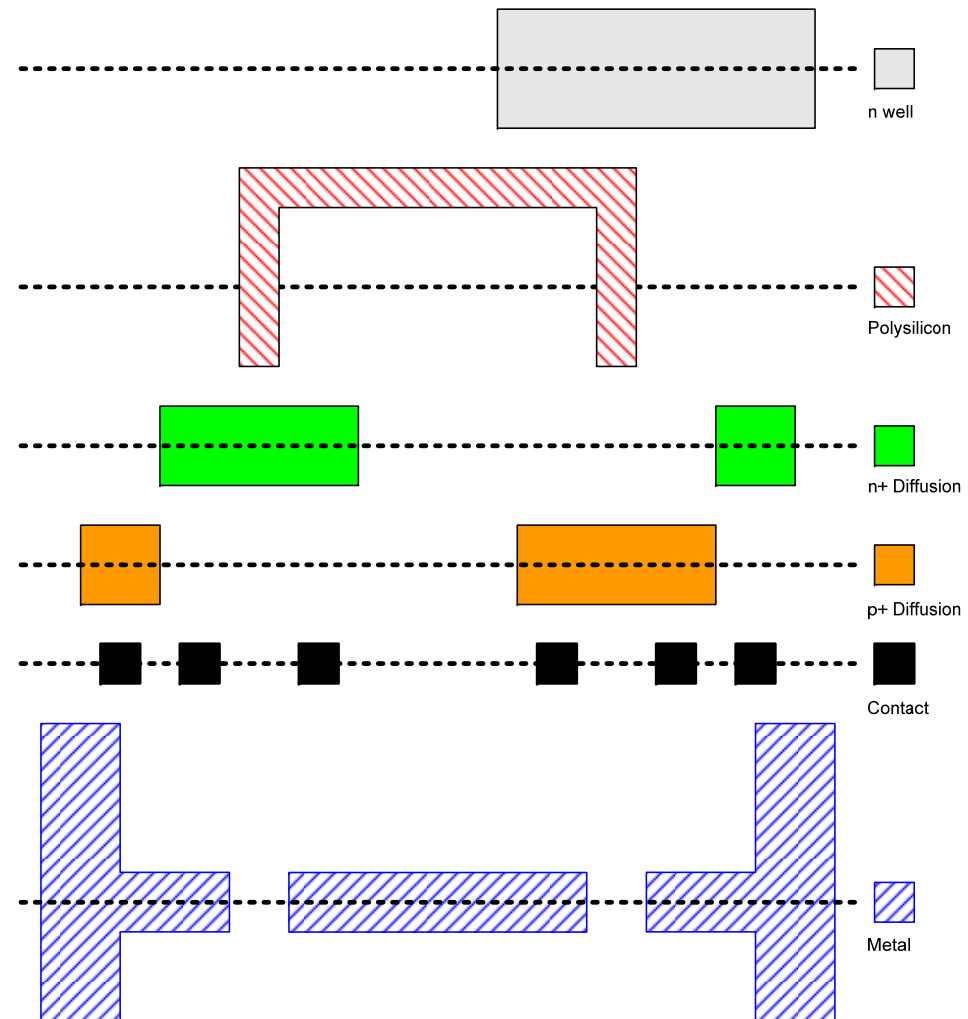
- ❑ Transistors and wires are defined by *masks*
- ❑ Cross-section taken along dashed line

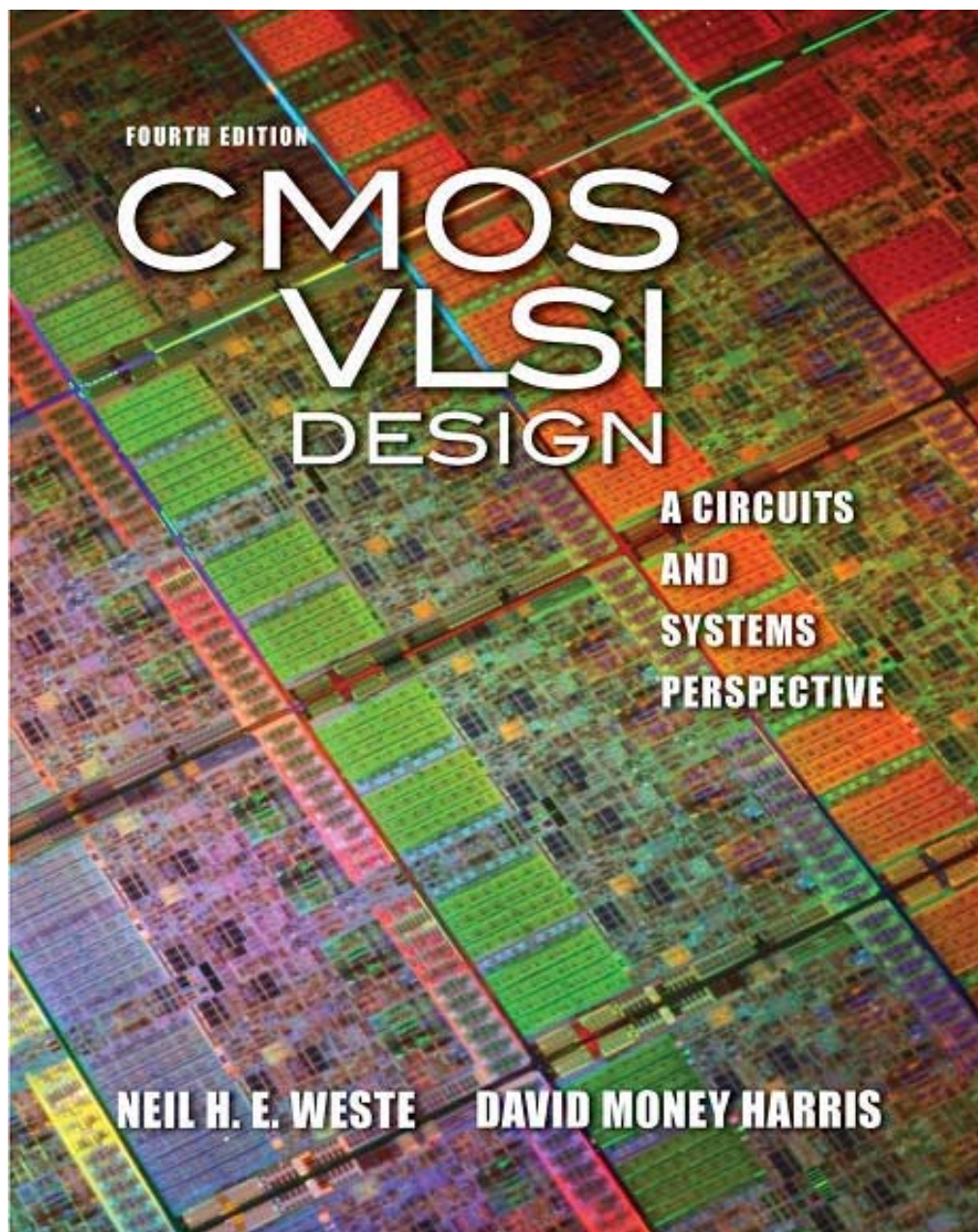


Detailed Mask Views

□ Six masks

- n-well
- Polysilicon
- n+ diffusion
- p+ diffusion
- Contact
- Metal





Week_2: Introduction

Fabrication

- ❑ Chips are built in huge factories called fabs
- ❑ Contain clean rooms as large as football fields



Courtesy of International
Business Machines Corporation.
Unauthorized use not permitted.

Fabrication Steps

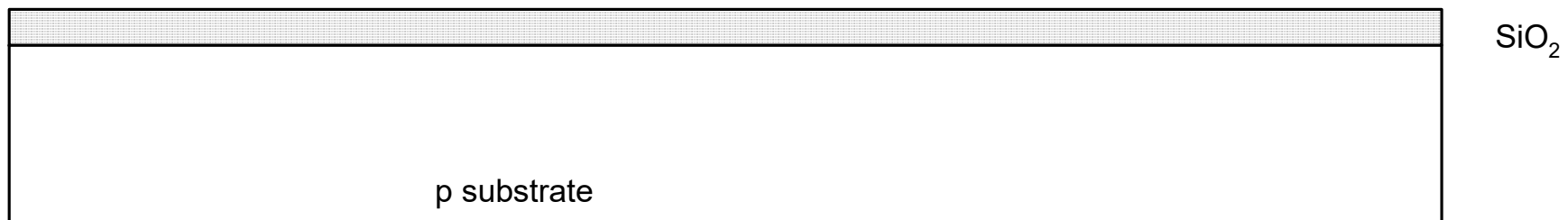
- ❑ Start with blank wafer
- ❑ Build inverter from the bottom up
- ❑ First step will be to form the n-well
 - Cover wafer with protective layer of SiO_2 (oxide)
 - Remove layer where n-well should be built
 - Implant or diffuse n dopants into exposed wafer
 - Strip off SiO_2



p substrate

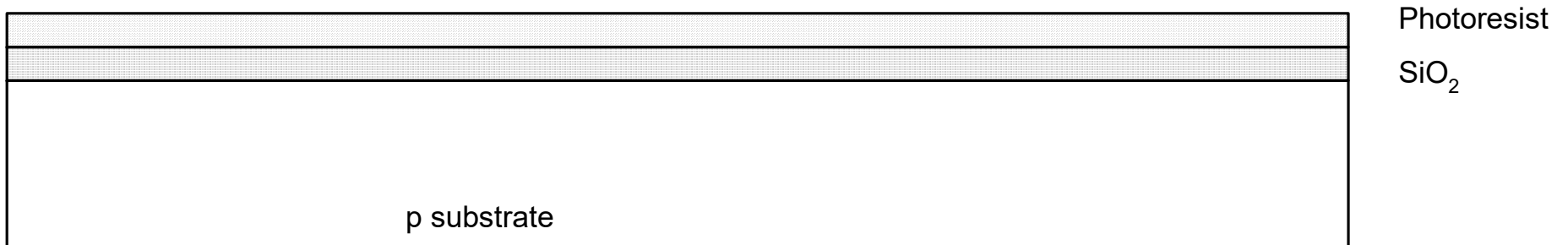
Oxidation

- Grow SiO_2 on top of Si wafer
 - 900 – 1200 C with H_2O or O_2 in oxidation furnace



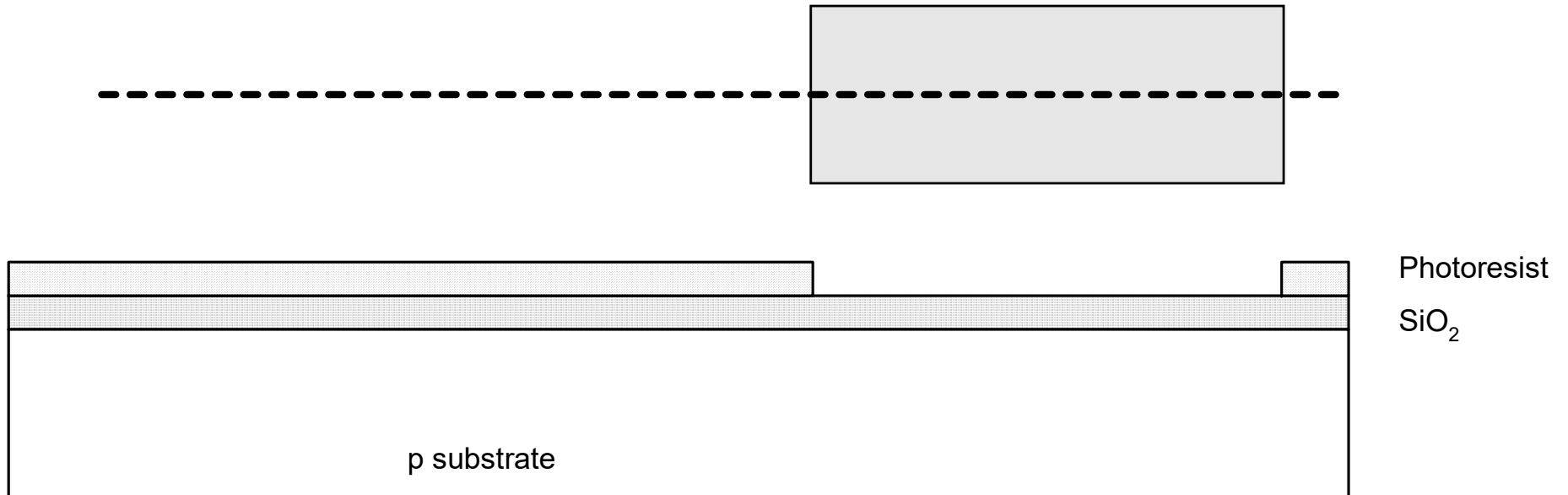
Photoresist

- Spin on photoresist
 - Photoresist is a light-sensitive organic polymer
 - Softens where exposed to light



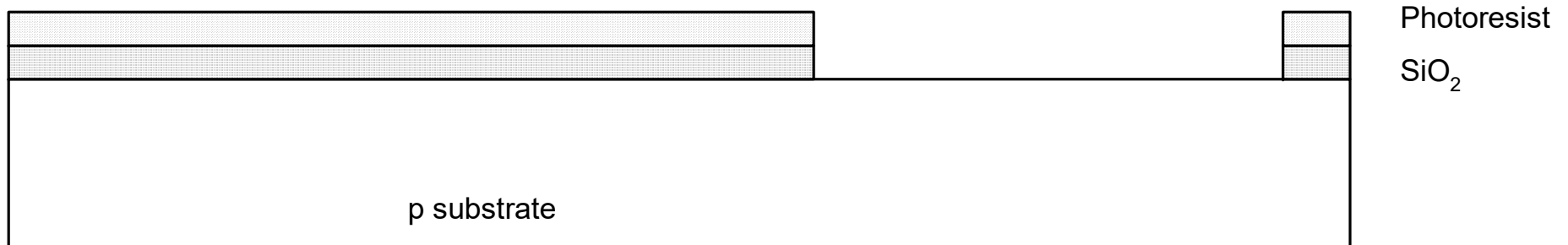
Lithography

- ❑ Expose photoresist through n-well mask
- ❑ Strip off exposed photoresist



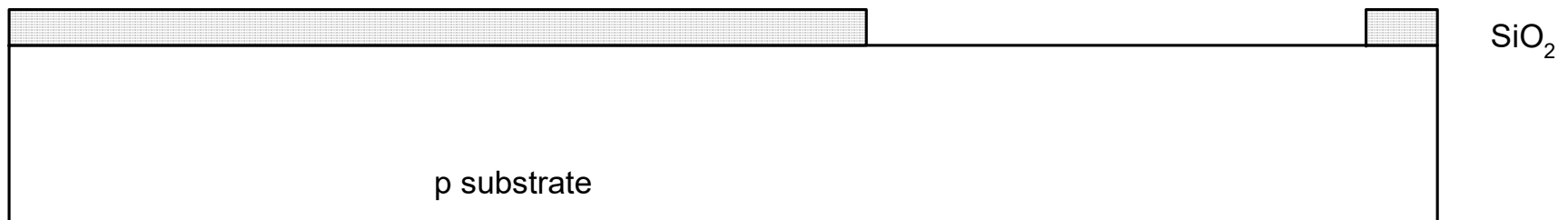
Etch

- ❑ Etch oxide with hydrofluoric acid (HF)
 - Seeps through skin and eats bone; nasty stuff!!!
- ❑ Only attacks oxide where resist has been exposed



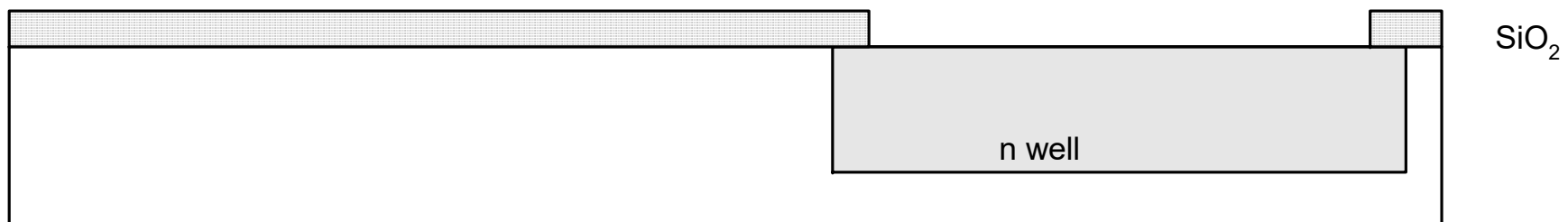
Strip Photoresist

- ❑ Strip off remaining photoresist
 - Use mixture of acids called piranha etch
- ❑ Necessary so resist doesn't melt in next step



n-well

- ❑ n-well is formed with diffusion or ion implantation
- ❑ Diffusion
 - Place wafer in furnace with arsenic gas
 - Heat until As atoms diffuse into exposed Si
- ❑ Ion Implantation
 - Blast wafer with beam of As ions
 - Ions blocked by SiO_2 , only enter exposed Si



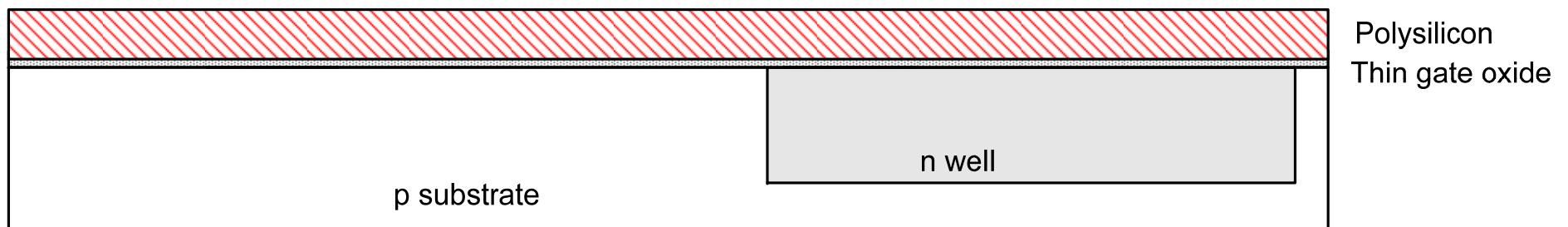
Strip Oxide

- ❑ Strip off the remaining oxide using HF
- ❑ Back to bare wafer with n-well
- ❑ Subsequent steps involve similar series of steps



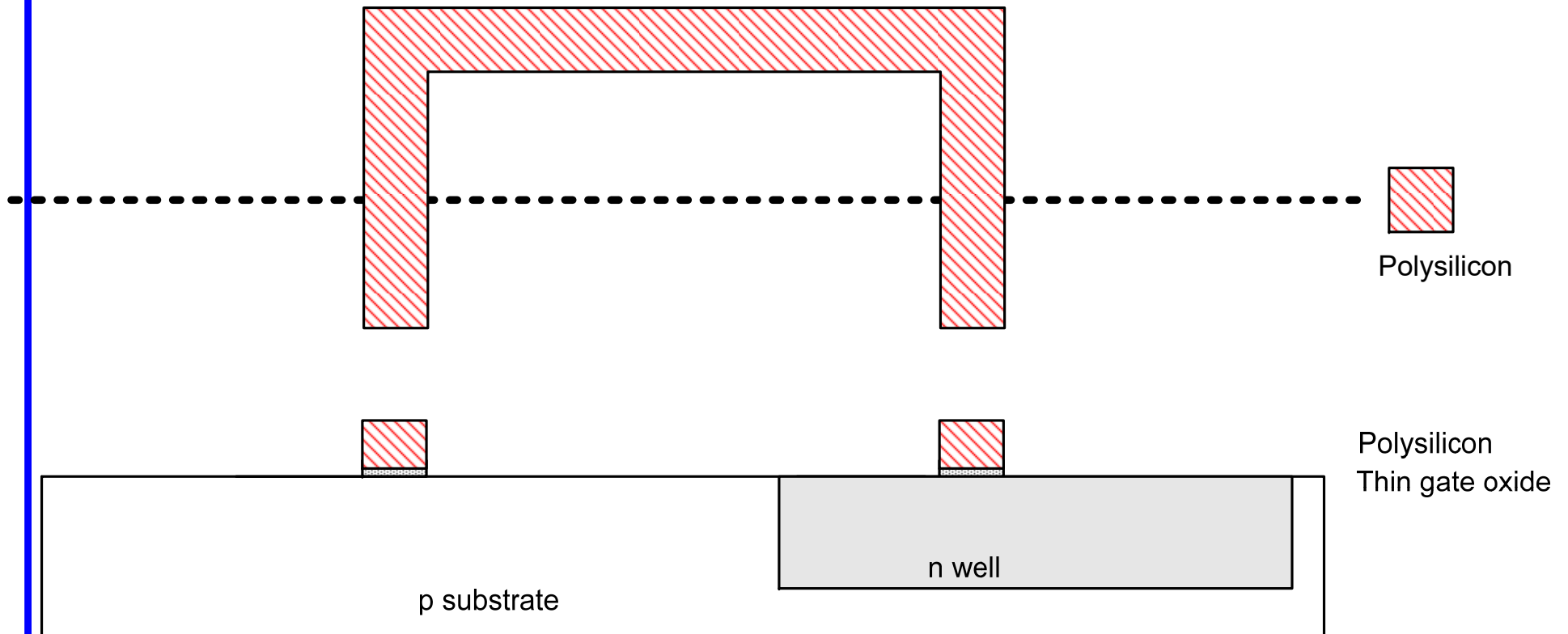
Polysilicon

- ❑ Deposit very thin layer of gate oxide
 - $< 20 \text{ \AA}$ (6-7 atomic layers)
- ❑ Chemical Vapor Deposition (CVD) of silicon layer
 - Place wafer in furnace with Silane gas (SiH_4)
 - Forms many small crystals called polysilicon
 - Heavily doped to be good conductor



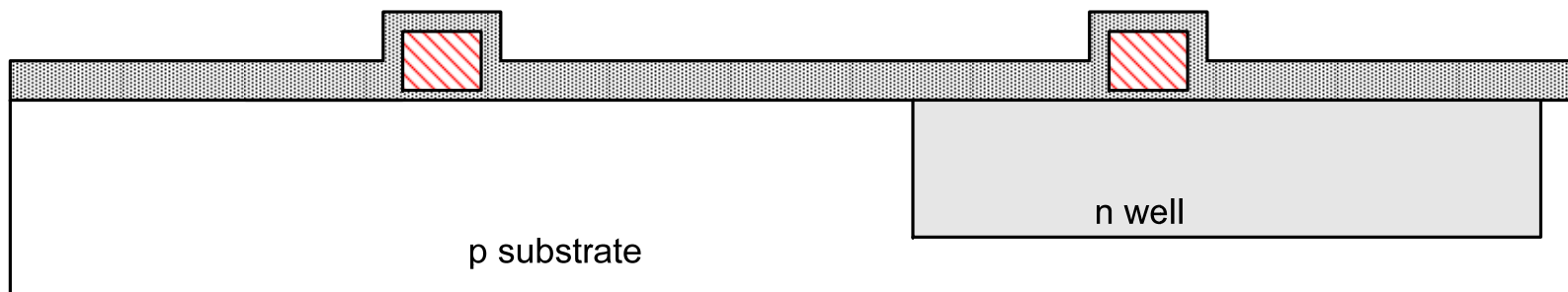
Polysilicon Patterning

- Use same lithography process to pattern polysilicon



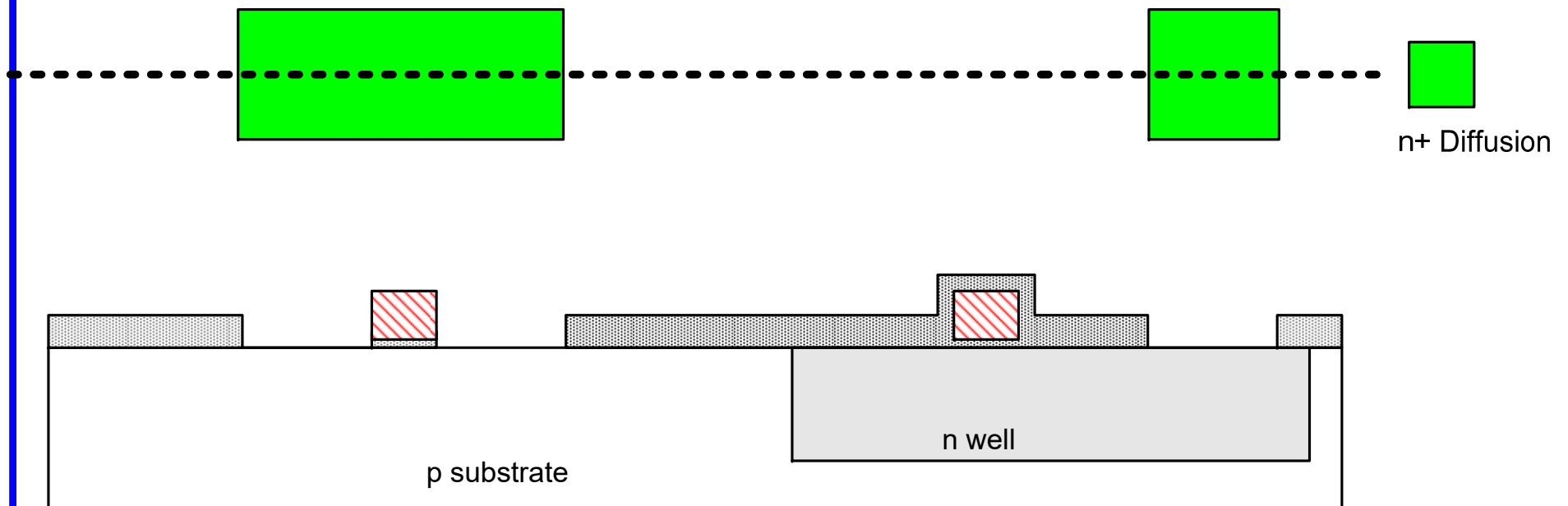
Self-Aligned Process

- ❑ Use oxide and masking to expose where n+ dopants should be diffused or implanted
- ❑ N-diffusion forms nMOS source, drain, and n-well contact



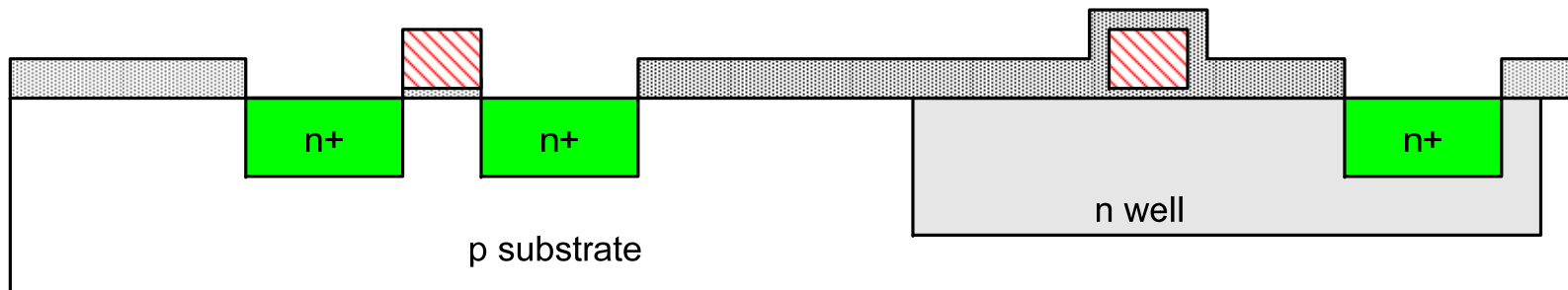
N-diffusion

- ❑ Pattern oxide and form n+ regions
- ❑ *Self-aligned process* where gate blocks diffusion
- ❑ Polysilicon is better than metal for self-aligned gates because it doesn't melt during later processing



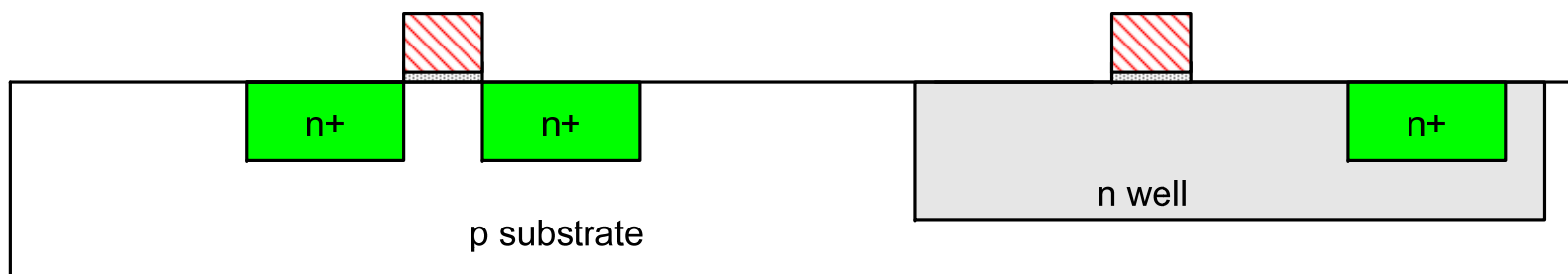
N-diffusion cont.

- ❑ Historically dopants were diffused
- ❑ Usually ion implantation today
- ❑ But regions are still called diffusion



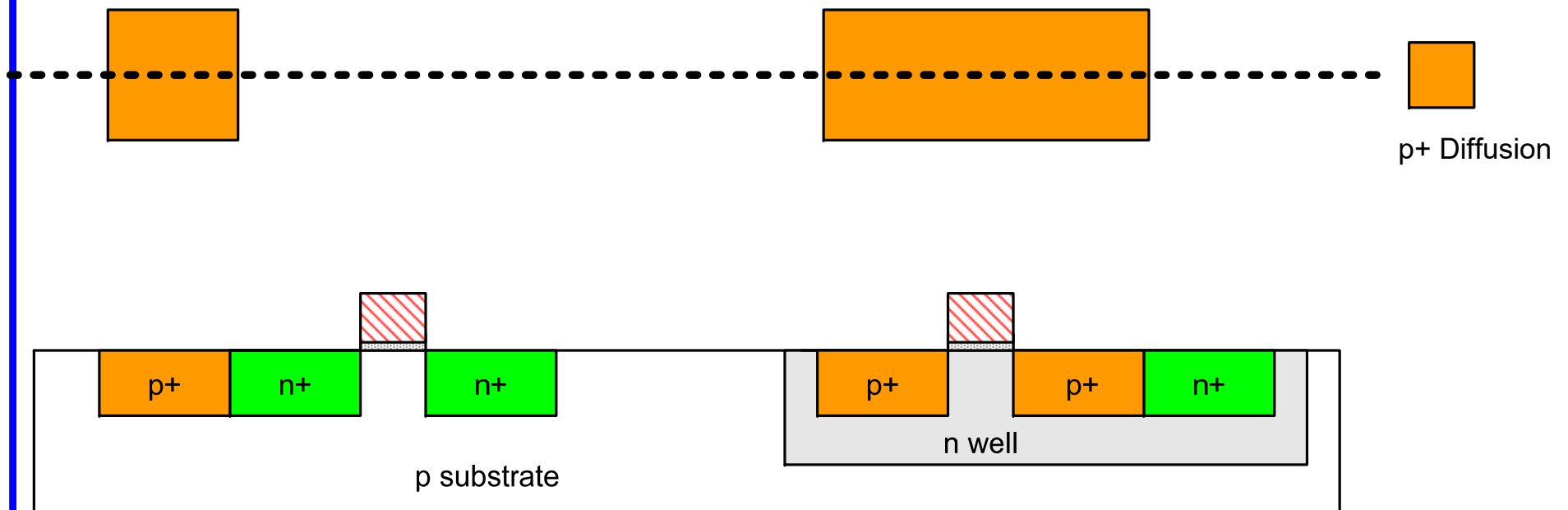
N-diffusion cont.

- Strip off oxide to complete patterning step



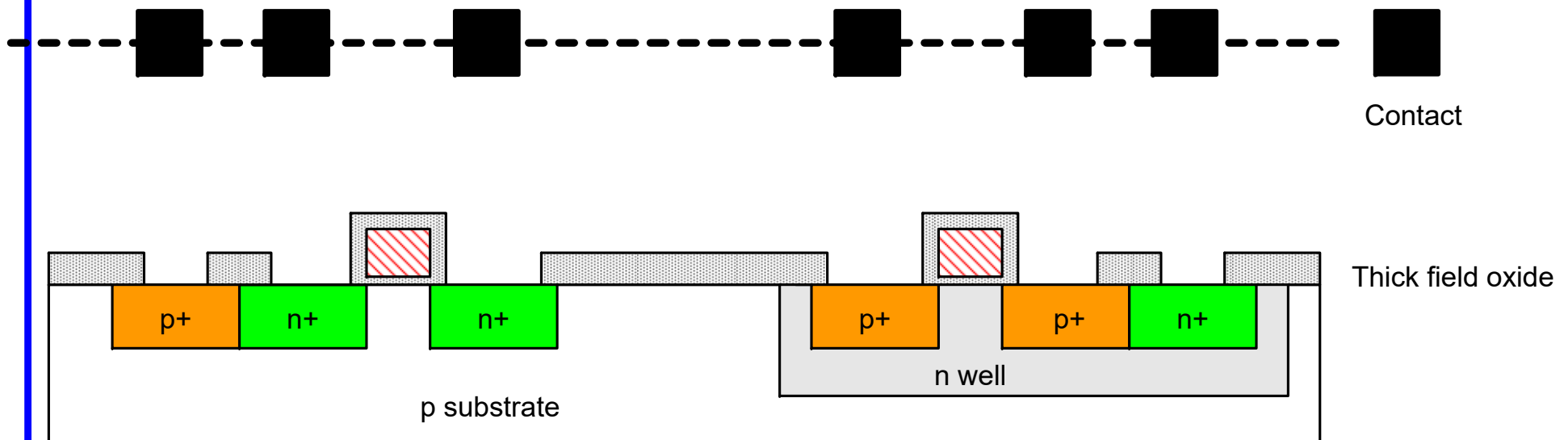
P-Diffusion

- Similar set of steps form p+ diffusion regions for pMOS source and drain and substrate contact



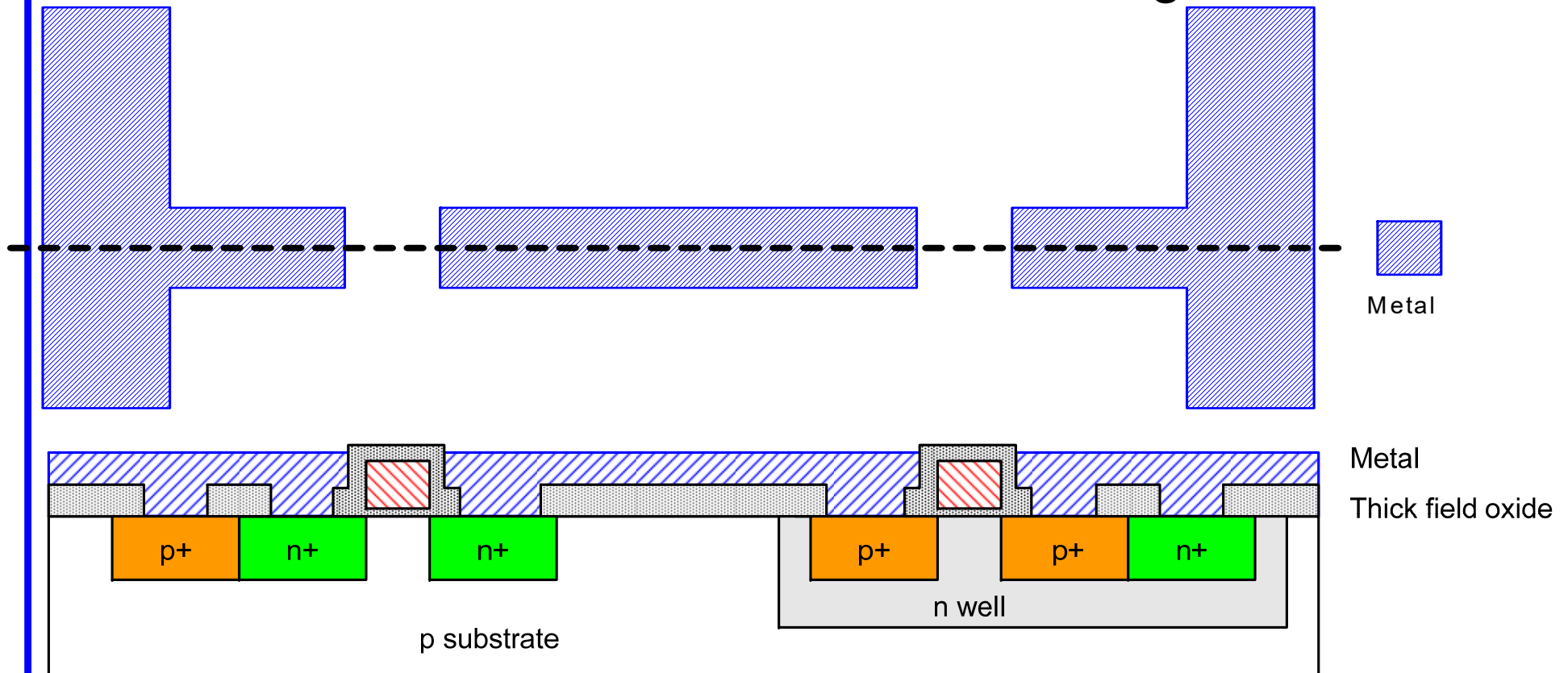
Contacts

- ❑ Now we need to wire together the devices
- ❑ Cover chip with thick field oxide
- ❑ Etch oxide where contact cuts are needed



Metalization

- ❑ Sputter on aluminum over whole wafer
- ❑ Pattern to remove excess metal, leaving wires

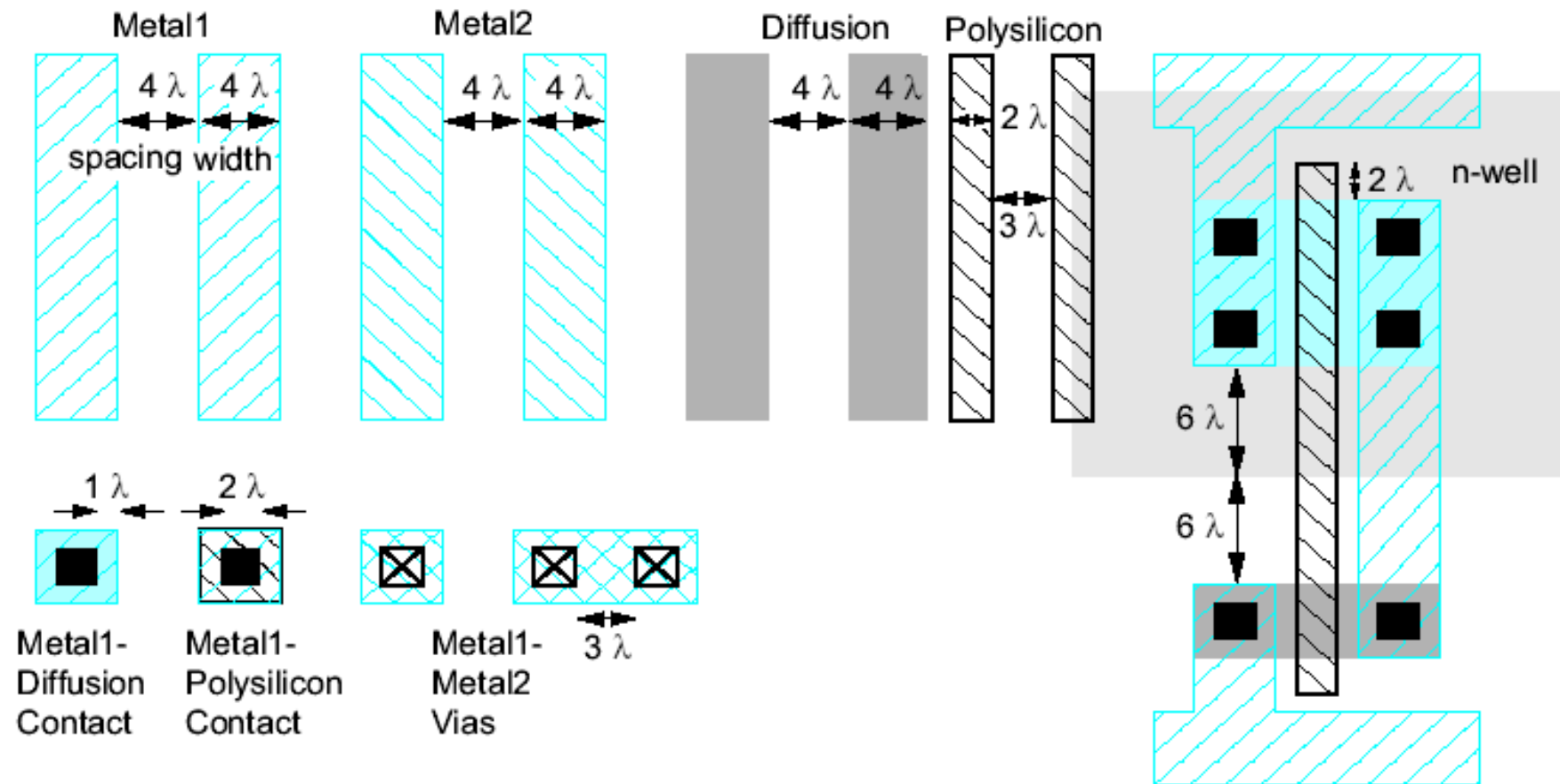


Layout

- ❑ Chips are specified with set of masks
- ❑ Minimum dimensions of masks determine transistor size (and hence speed, cost, and power)
- ❑ Feature size f = distance between source and drain
 - Set by minimum width of polysilicon
- ❑ Feature size improves 30% every 3 years or so
- ❑ Normalize for feature size when describing design rules
- ❑ Express rules in terms of $\lambda = f/2$
 - E.g. $\lambda = 0.3 \mu\text{m}$ in $0.6 \mu\text{m}$ process

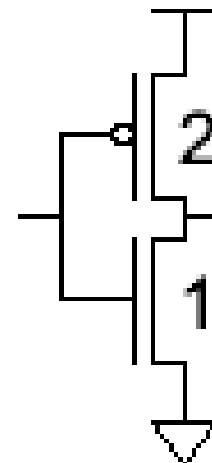
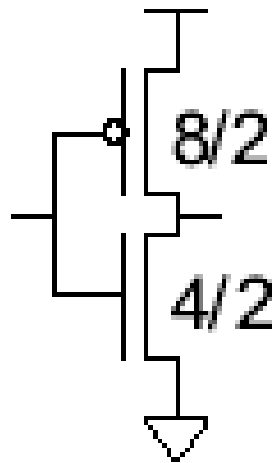
Simplified Design Rules

- Conservative rules to get you started



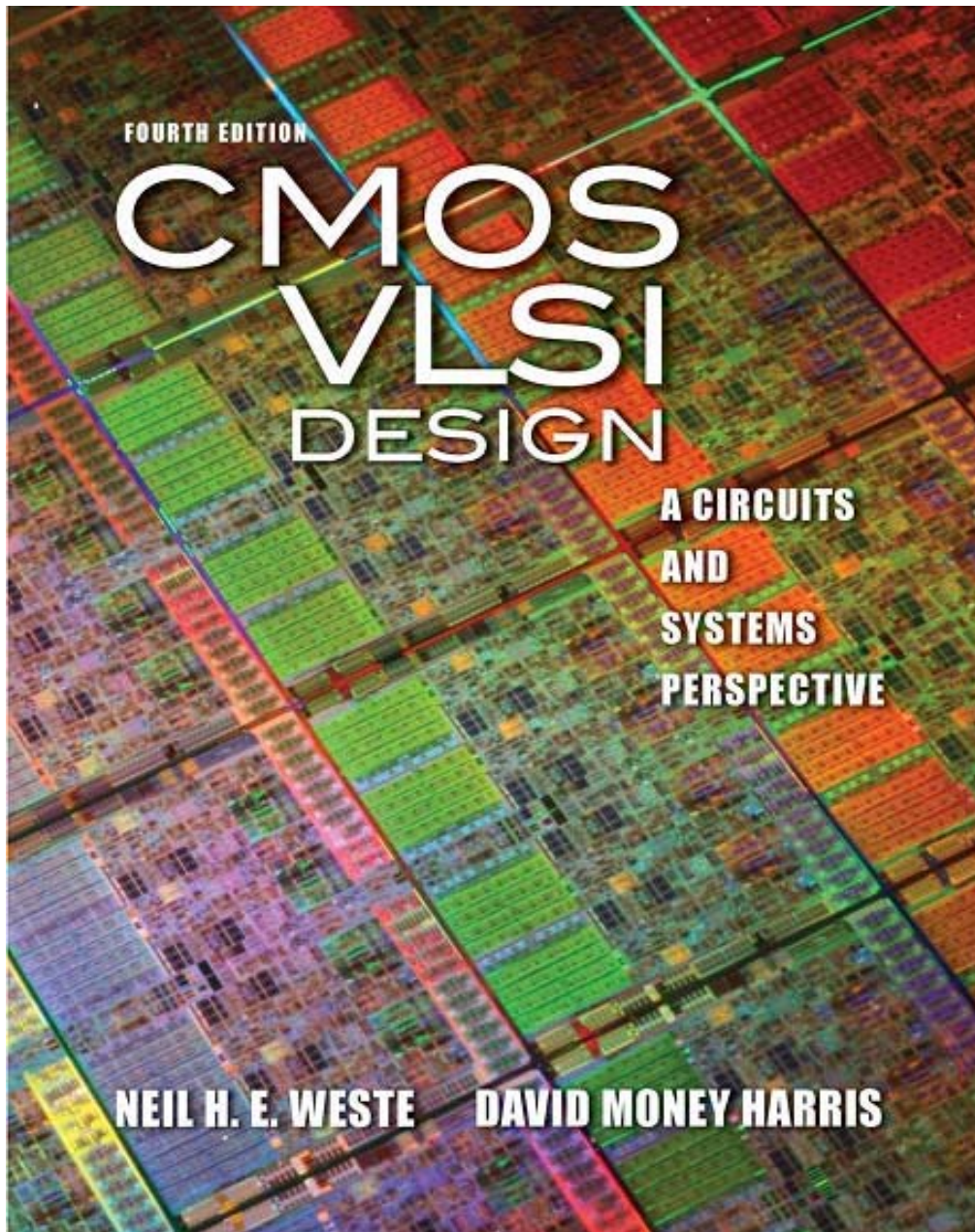
Inverter Layout

- Transistor dimensions specified as Width / Length
 - Minimum size is $4\lambda / 2\lambda$, sometimes called 1 unit
 - In $f = 0.6 \mu\text{m}$ process, this is $1.2 \mu\text{m}$ wide, $0.6 \mu\text{m}$ long



Summary

- ❑ MOS transistors are stacks of gate, oxide, silicon
 - ❑ Act as electrically controlled switches
 - ❑ Build logic gates out of switches
 - ❑ Draw masks to specify layout of transistors
-
- ❑ Now you know everything necessary to start designing schematics and layout for a simple chip!



Lecture_3: Circuits & Layout

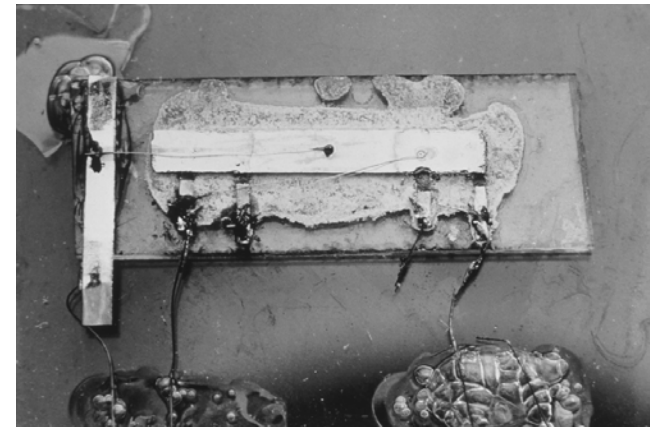
Outline

- A Brief History
- CMOS Gate Design
- Pass Transistors
- CMOS Latches & Flip-Flops
- Standard Cell Layouts
- Stick Diagrams

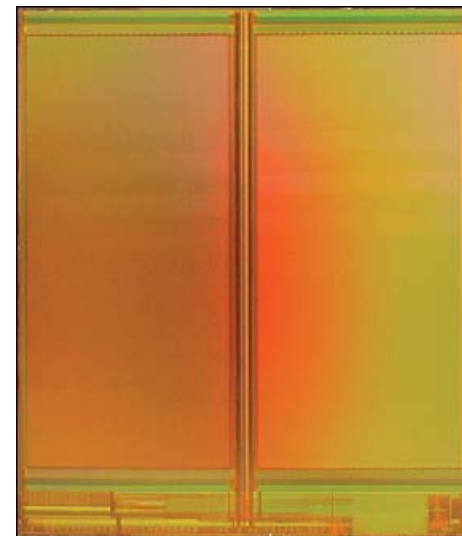
A Brief History

- ❑ 1958: First integrated circuit
 - Flip-flop using two transistors
 - Built by Jack Kilby at Texas Instruments

- ❑ 2010
 - Intel Core i7 μ processor
 - 2.3 billion transistors
 - 64 Gb Flash memory
 - > 16 billion transistors



Courtesy Texas Instruments



[Trinh09]
© 2009 IEEE

Growth Rate

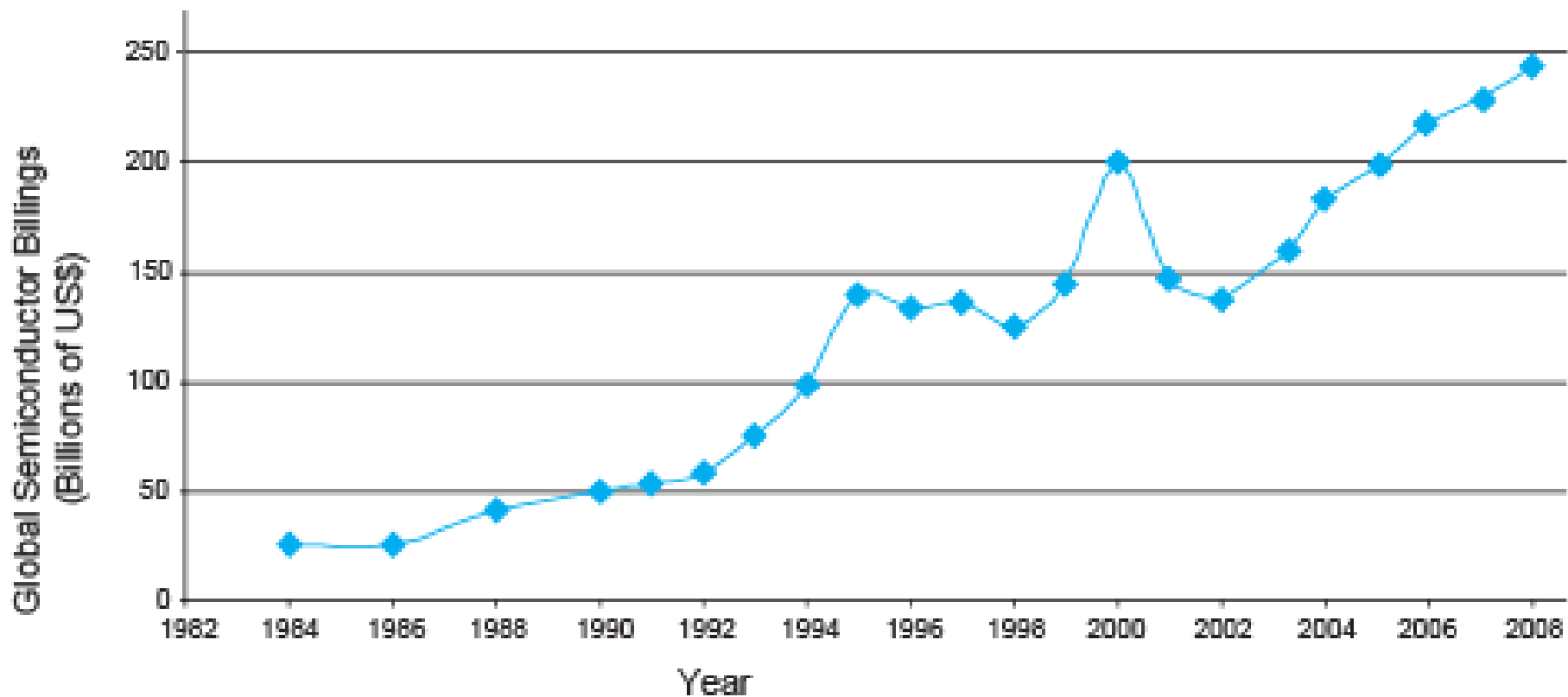
- ❑ 53% compound annual growth rate over 50 years
 - No other technology has grown so fast so long
- ❑ Driven by miniaturization of transistors
 - Smaller is cheaper, faster, lower in power!
 - Revolutionary effects on society



[Moore65]
Electronics Magazine

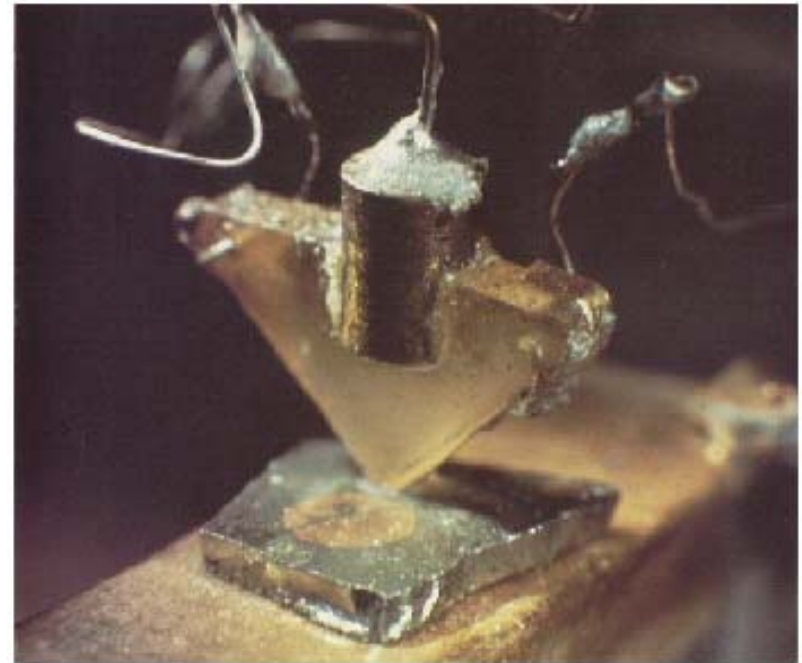
Annual Sales

- $>10^{19}$ transistors manufactured in 2008
 - 1 billion for every human on the planet



Invention of the Transistor

- ❑ Vacuum tubes ruled in first half of 20th century
Large, expensive, power-hungry, unreliable
- ❑ 1947: first point contact transistor
 - John Bardeen and Walter Brattain at Bell Labs
 - See *Crystal Fire*
by Riordan, Hoddeson



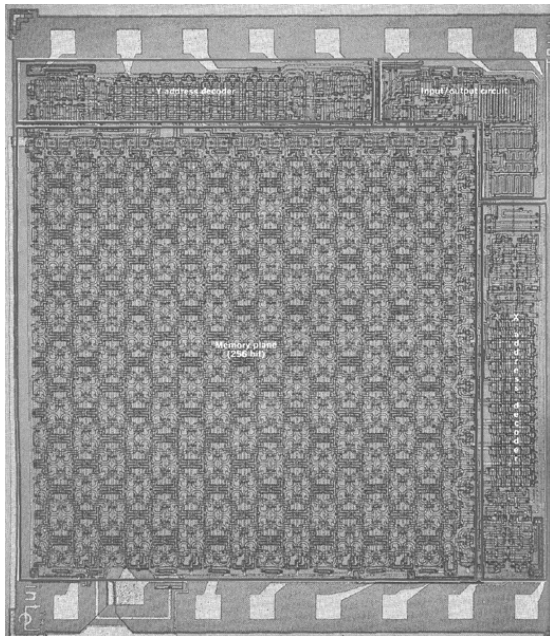
AT&T Archives.
Reprinted with
permission.

Transistor Types

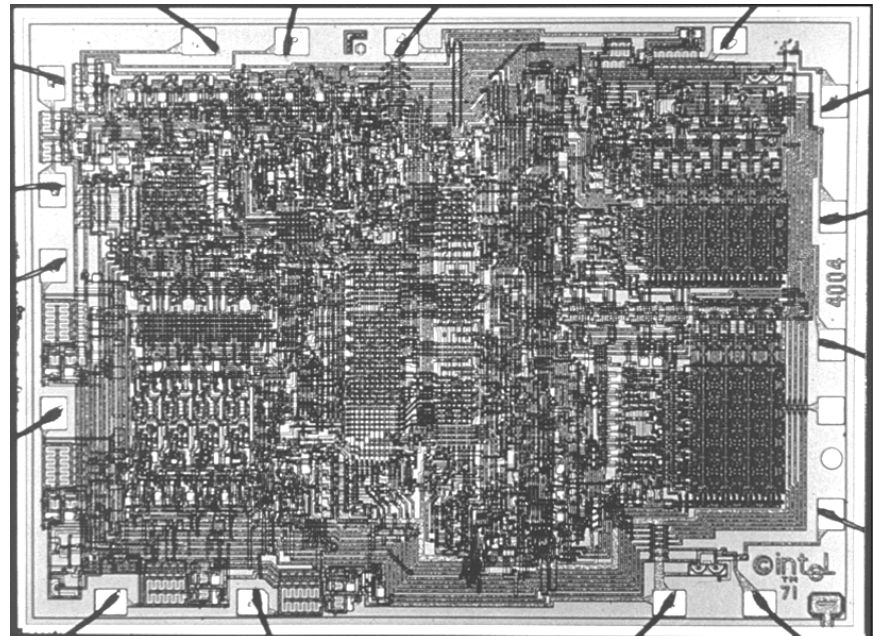
- Bipolar transistors
 - npn or pnp silicon structure
 - Small current into very thin base layer controls large currents between emitter and collector
 - Base currents limit integration density
- Metal Oxide Semiconductor Field Effect Transistors
 - nMOS and pMOS MOSFETS
 - Voltage applied to insulated gate controls current between source and drain
 - Low power allows very high integration

MOS Integrated Circuits

- 1970's processes usually had only nMOS transistors
 - Inexpensive, but consume power while idle



[Vadasz69]
© 1969 IEEE.



Intel
Museum.
Reprinted
with
permission.

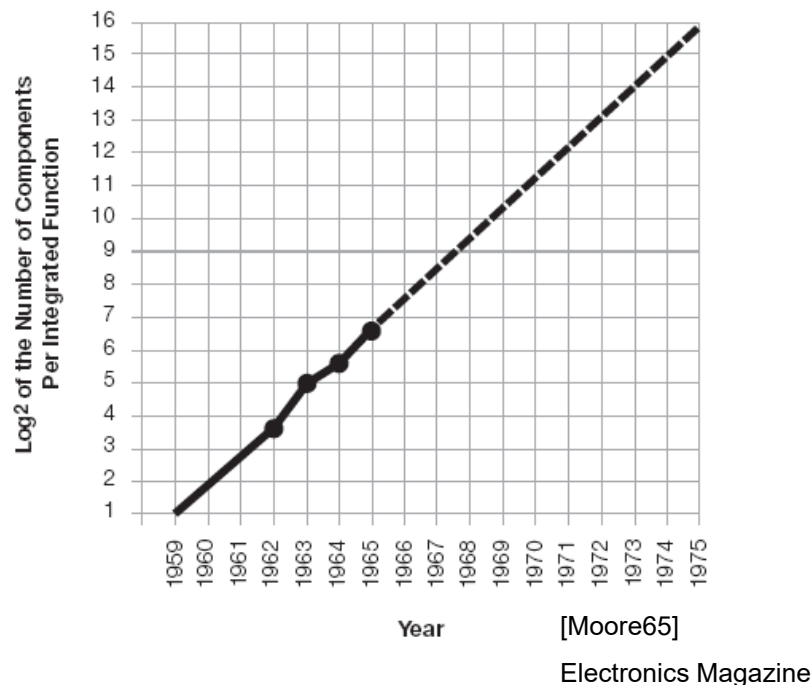
Intel 1101 256-bit SRAM

Intel 4004 4-bit μ Proc

- 1980s-present: CMOS processes for low idle power

Moore's Law: Then

- 1965: Gordon Moore plotted transistor on each chip
 - Fit straight line on semilog scale
 - Transistor counts have doubled every 26 months



Integration Levels

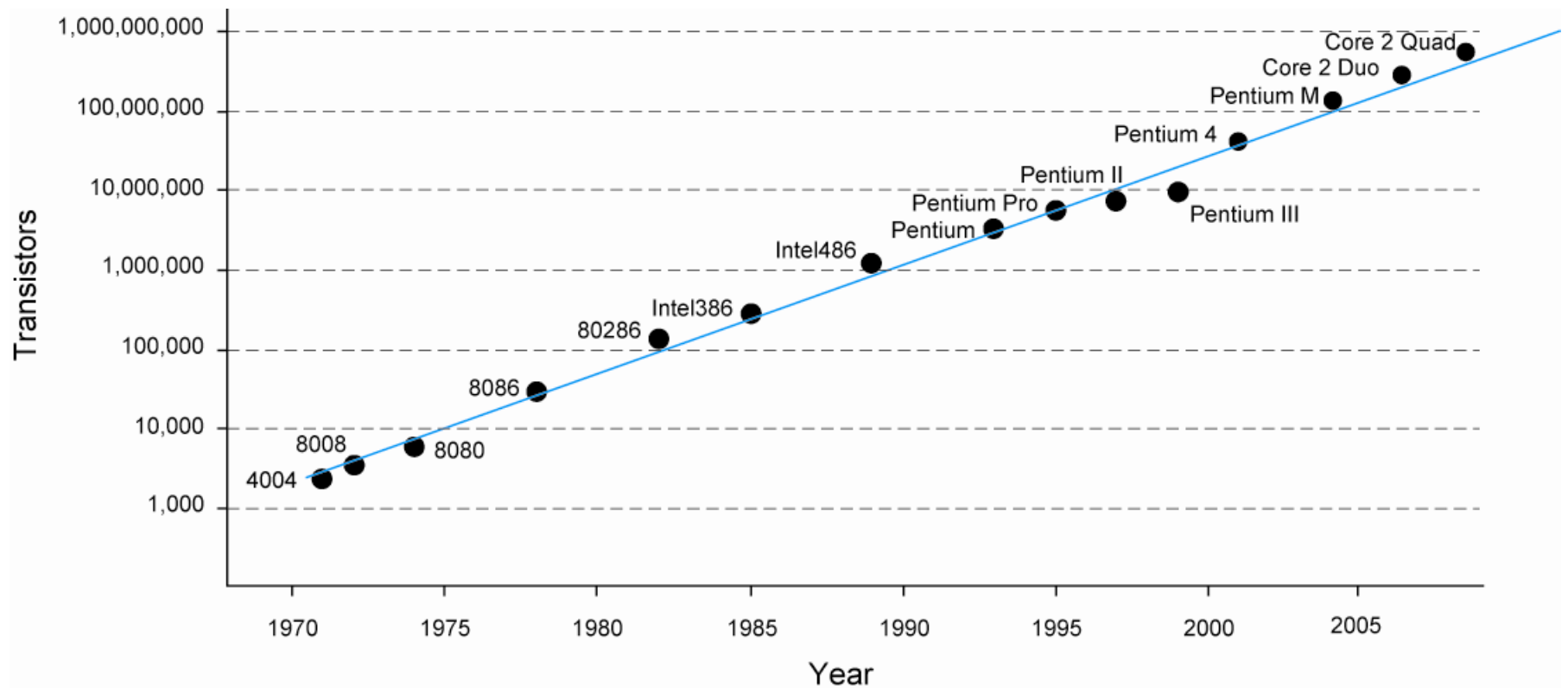
SSI: 10 gates

MSI: 1000 gates

LSI: 10,000 gates

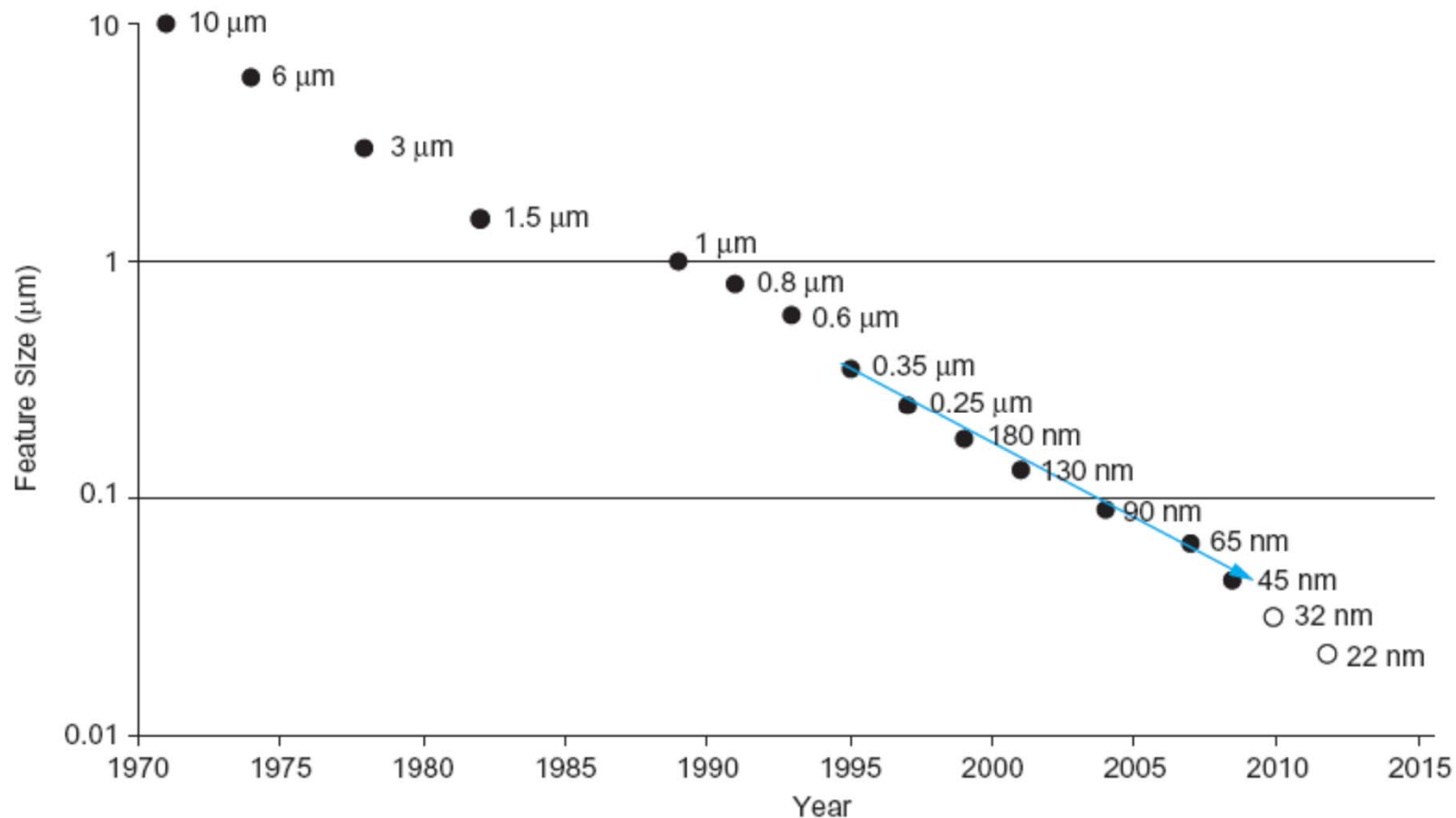
VLSI: > 10k gates

And Now...



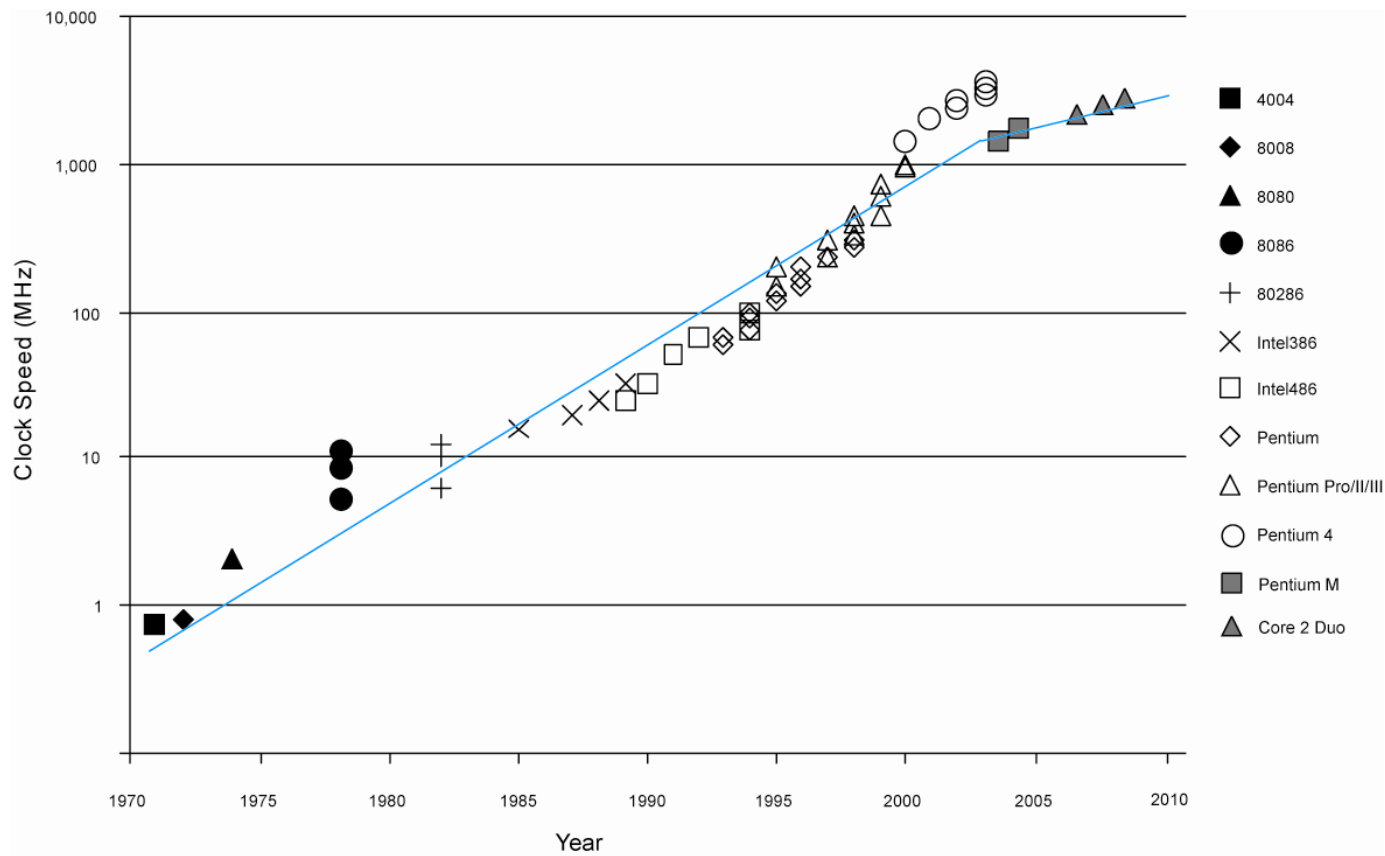
Feature Size

- Minimum feature size shrinking 30% every 2-3 years



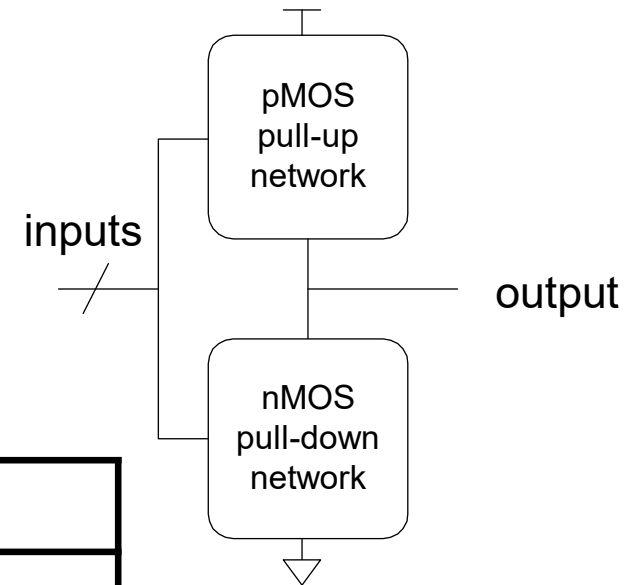
Corollaries

- Many other factors grow exponentially
 - Ex: clock frequency, processor performance



Complementary CMOS

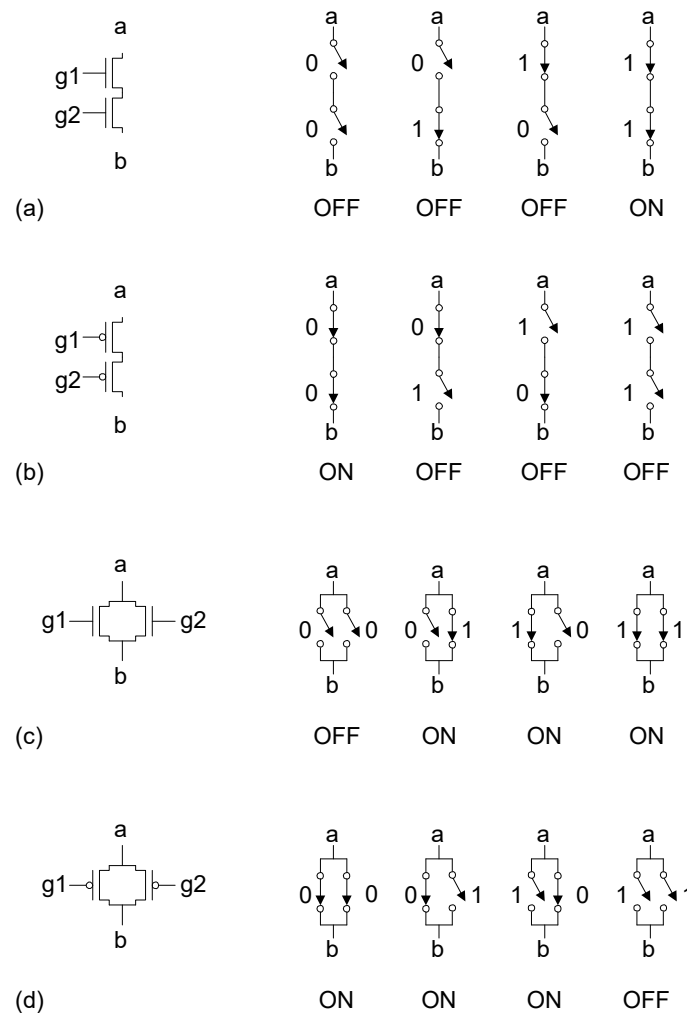
- Complementary CMOS logic gates
 - nMOS *pull-down network*
 - pMOS *pull-up network*
 - a.k.a. static CMOS



	Pull-up OFF	Pull-up ON
Pull-down OFF	Z (float)	1
Pull-down ON	0	X (crowbar)

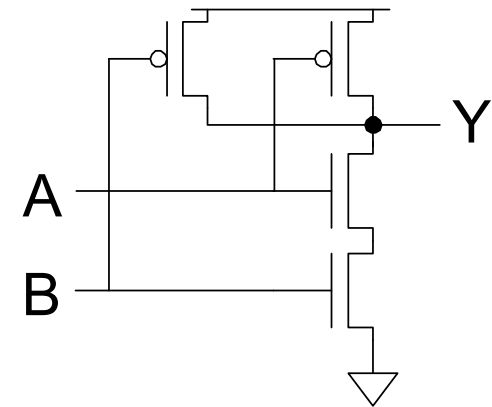
Series and Parallel

- nMOS: 1 = ON
- pMOS: 0 = ON
- *Series*: both must be ON
- *Parallel*: either can be ON



Conduction Complement

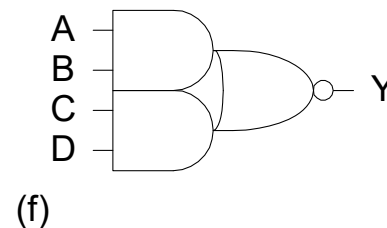
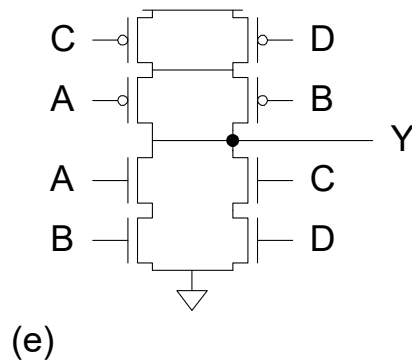
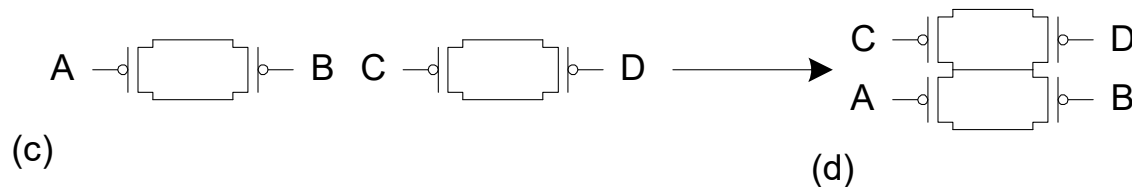
- ❑ Complementary CMOS gates always produce 0 or 1
- ❑ Ex: NAND gate
 - Series nMOS: $Y=0$ when both inputs are 1
 - Thus $Y=1$ when either input is 0
 - Requires parallel pMOS
- ❑ Rule of *Conduction Complements*
 - Pull-up network is a complement of pull-down
 - Parallel \rightarrow series, series \rightarrow parallel



Compound Gates

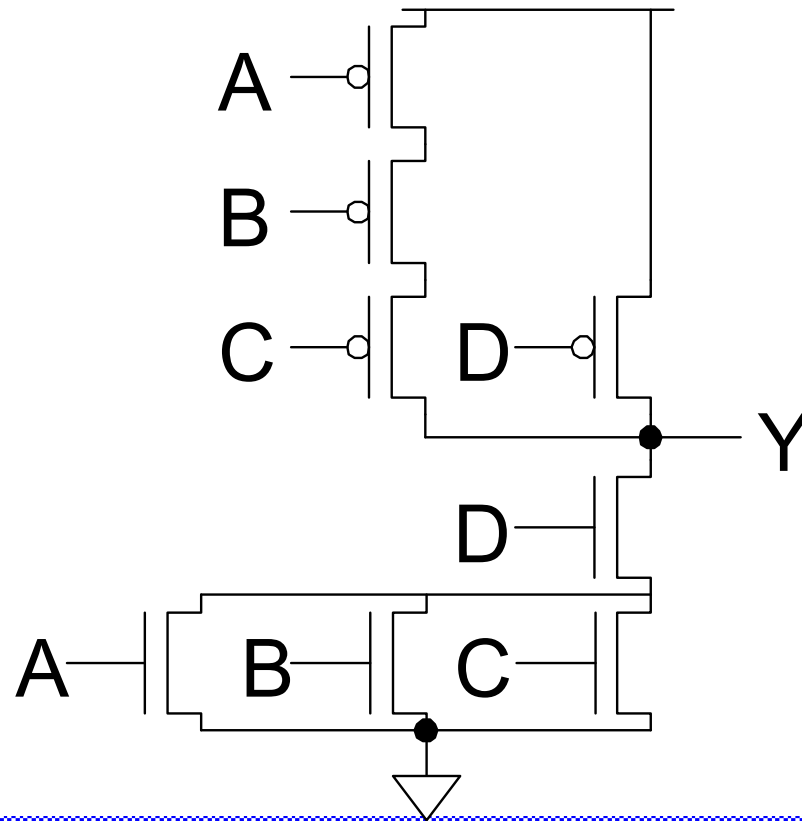
□ *Compound gates can do any inverting function*

$$Y = \overline{A.B + C.D} \quad (\text{AND} - \text{AND} - \text{OR} - \text{INVERT}, \text{AOI22})$$



Example: O3AI

$$Y = \overline{(A + B + C).D}$$

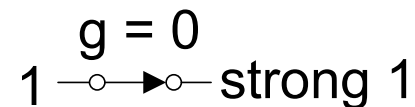
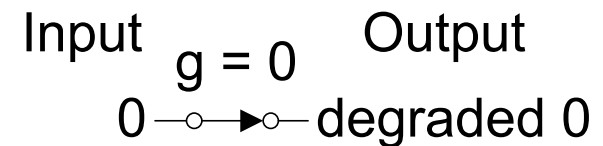
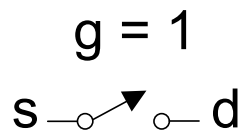
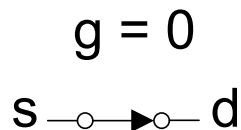
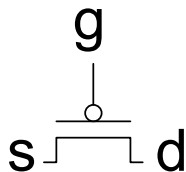
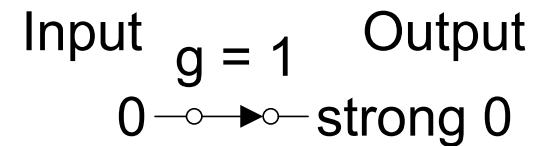
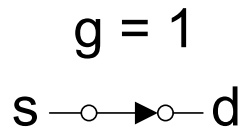
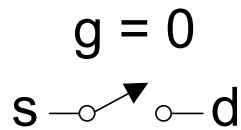
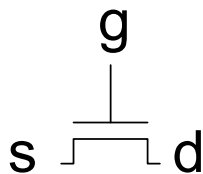


Signal Strength

- ❑ *Strength* of signal
 - How close it approximates ideal voltage source
- ❑ V_{DD} and GND rails are strongest 1 and 0
- ❑ nMOS pass strong 0
 - But degraded or weak 1
- ❑ pMOS pass strong 1
 - But degraded or weak 0
- ❑ Thus, nMOS are best for pull-down network

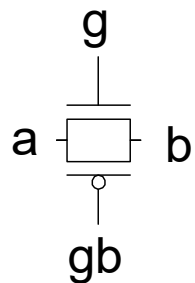
Pass Transistors

- Transistors can be used as switches

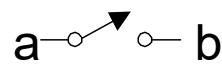


Transmission Gates

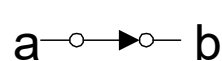
- ❑ Pass transistors produce degraded outputs
- ❑ *Transmission gates* pass both 0 and 1 well



$$g = 0, gb = 1$$



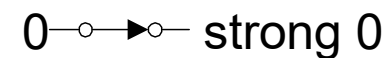
$$g = 1, gb = 0$$



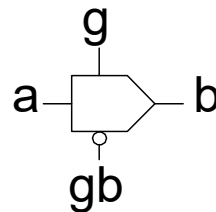
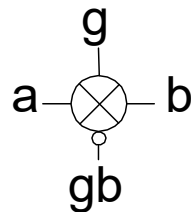
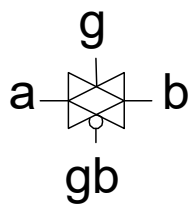
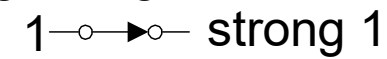
Input

Output

$$g = 1, gb = 0$$



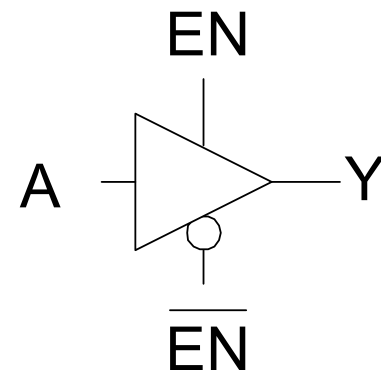
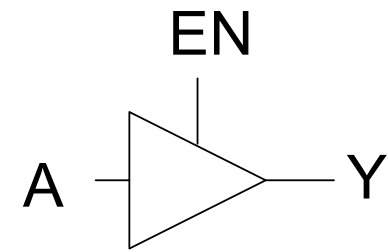
$$g = 1, gb = 0$$



Tristates

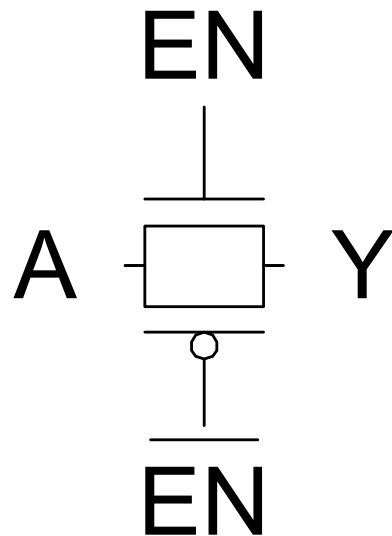
- ❑ *Tristate buffer* produces Z when not enabled

EN	A	Y
0	0	
0	1	
1	0	
1	1	



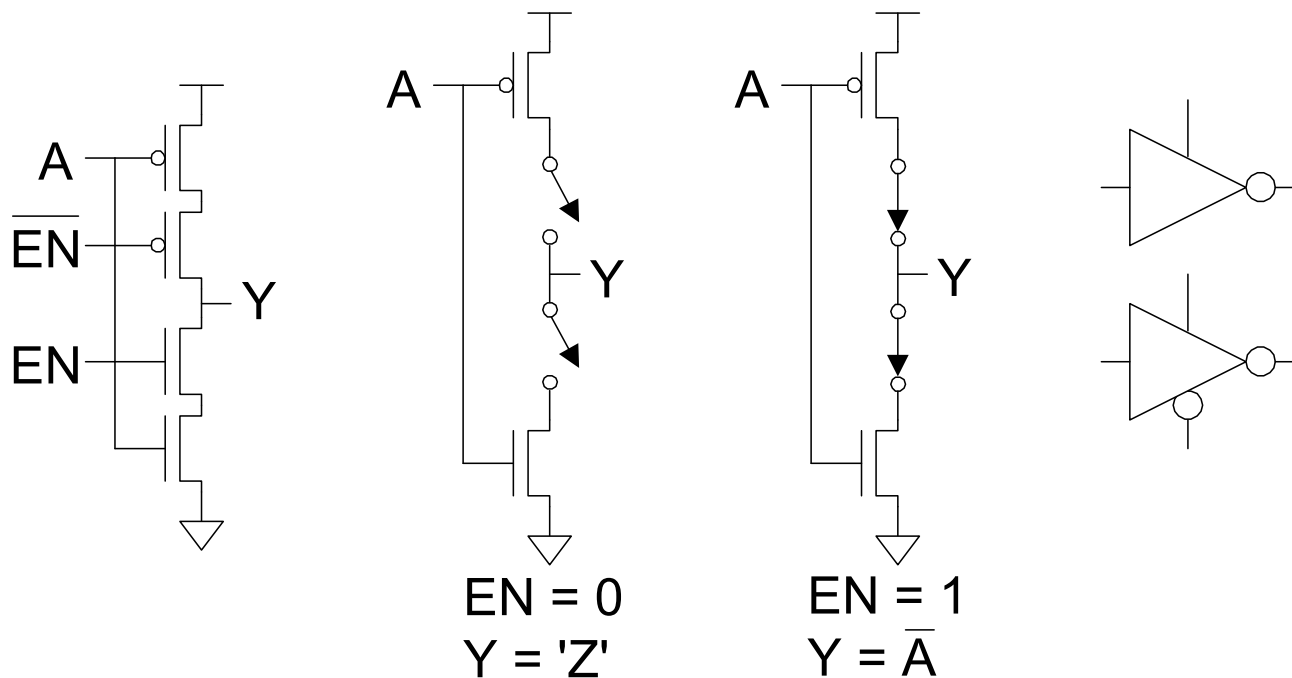
Nonrestoring Tristate

- Transmission gate acts as tristate buffer
 - Only two transistors
 - But *nonrestoring*
 - Noise on A is passed on to Y



Tristate Inverter

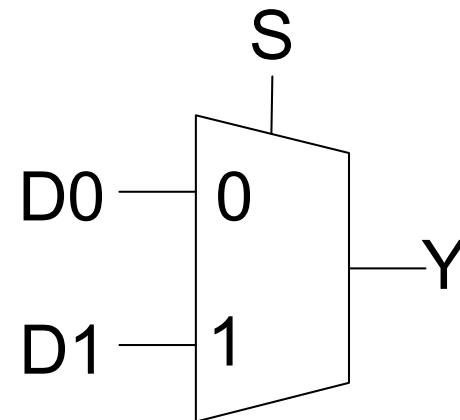
- Tristate inverter produces restored output
 - Violates conduction complement rule
 - Because we want a Z output



Multiplexers

- 2:1 multiplexer chooses between two inputs

S	D1	D0	Y
0	X	0	
0	X	1	
1	0	X	
1	1	X	

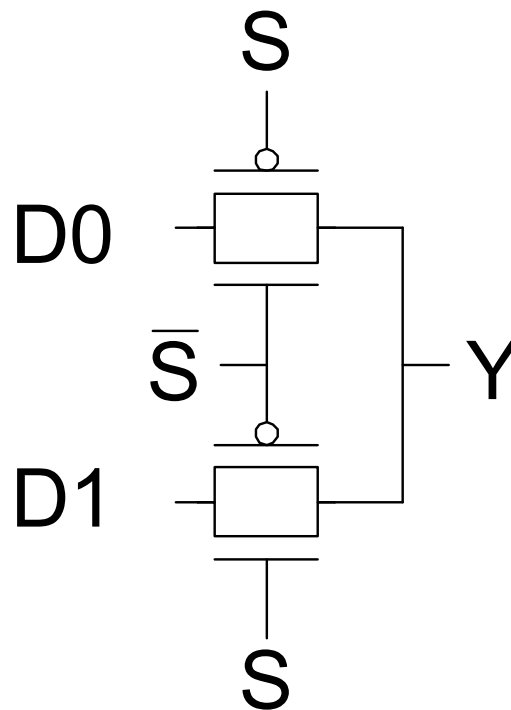


Gate-Level Mux Design

- $Y = SD_1 + \bar{S}D_0$ (too many transistors)
- How many transistors are needed?

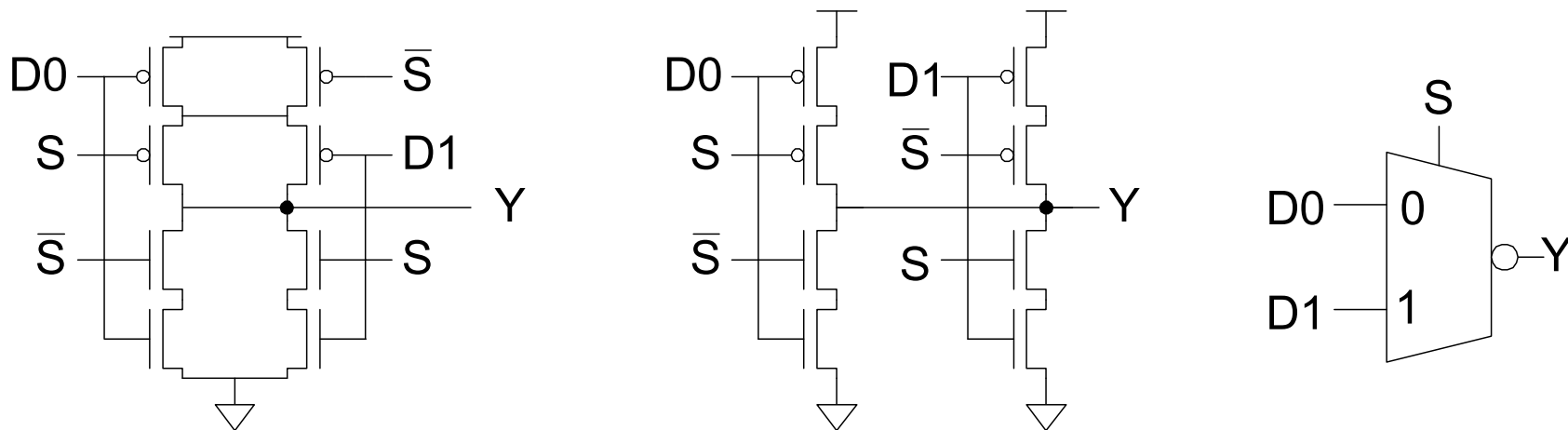
Transmission Gate Mux

- Nonrestoring mux uses two transmission gates
 - Only 4 transistors



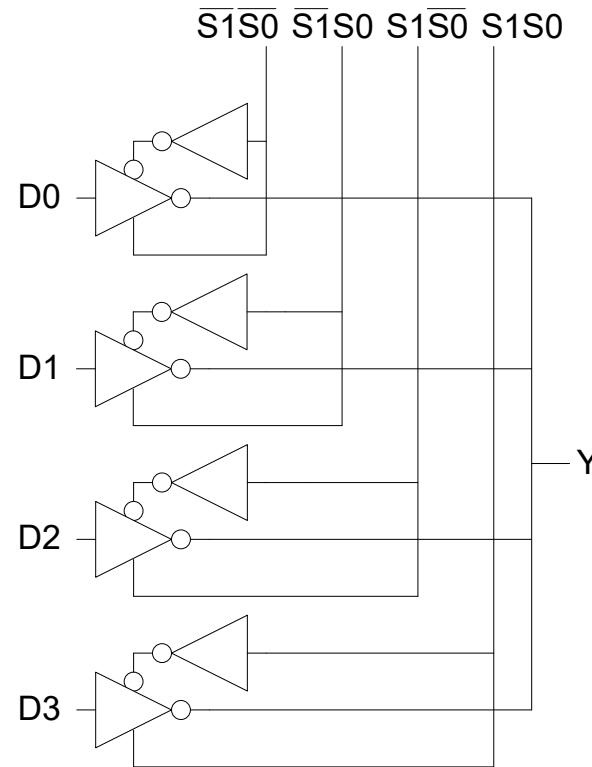
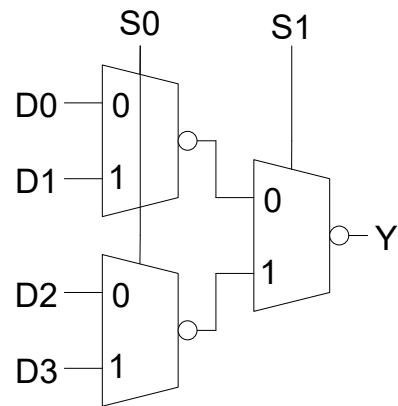
Inverting Mux

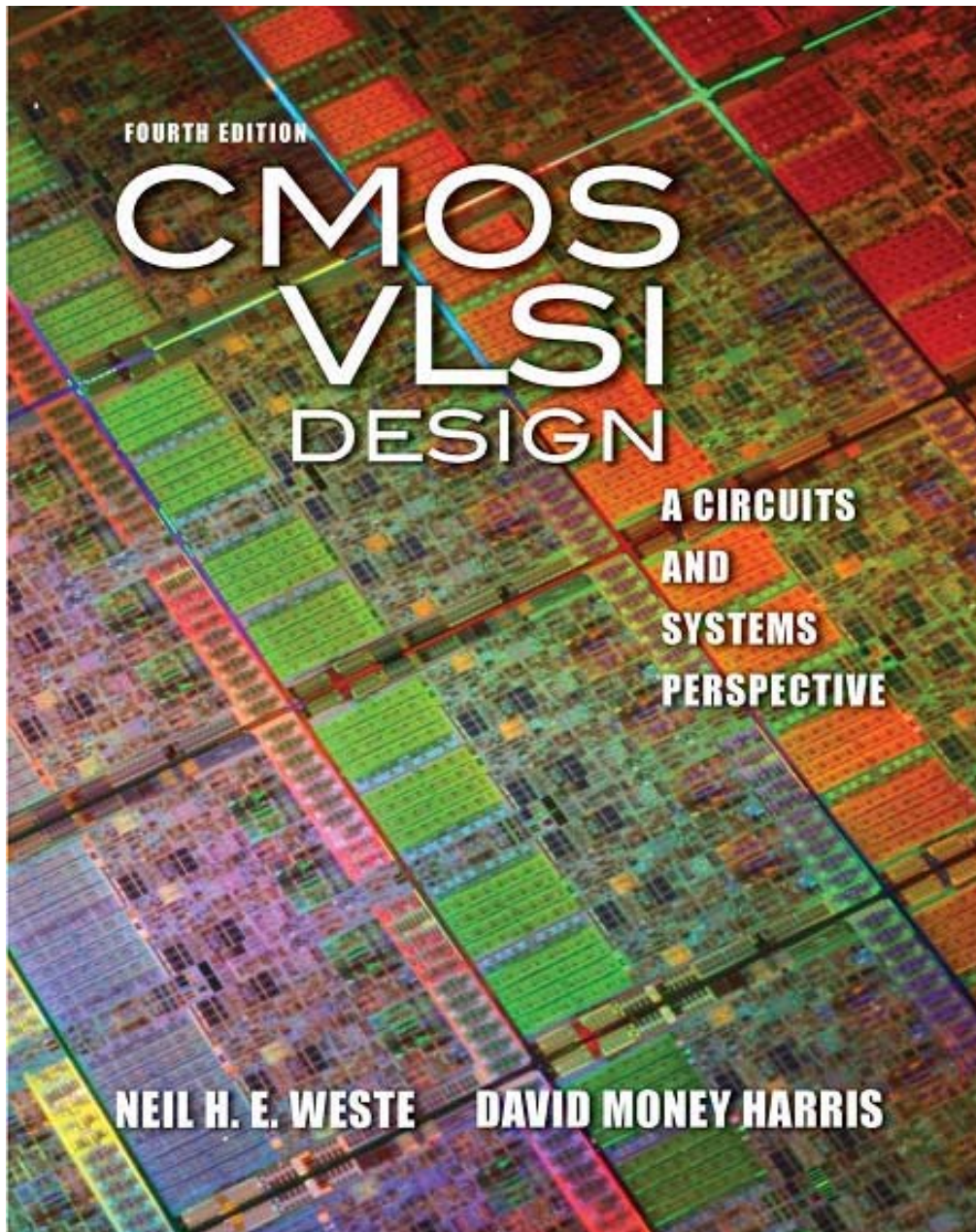
- ❑ Inverting multiplexer
 - Use compound AOI22
 - Or pair of tristate inverters
 - Essentially the same thing
- ❑ Noninverting multiplexer adds an inverter



4:1 Multiplexer

- 4:1 mux chooses one of 4 inputs using two selects
 - Two levels of 2:1 muxes
 - Or four tristates

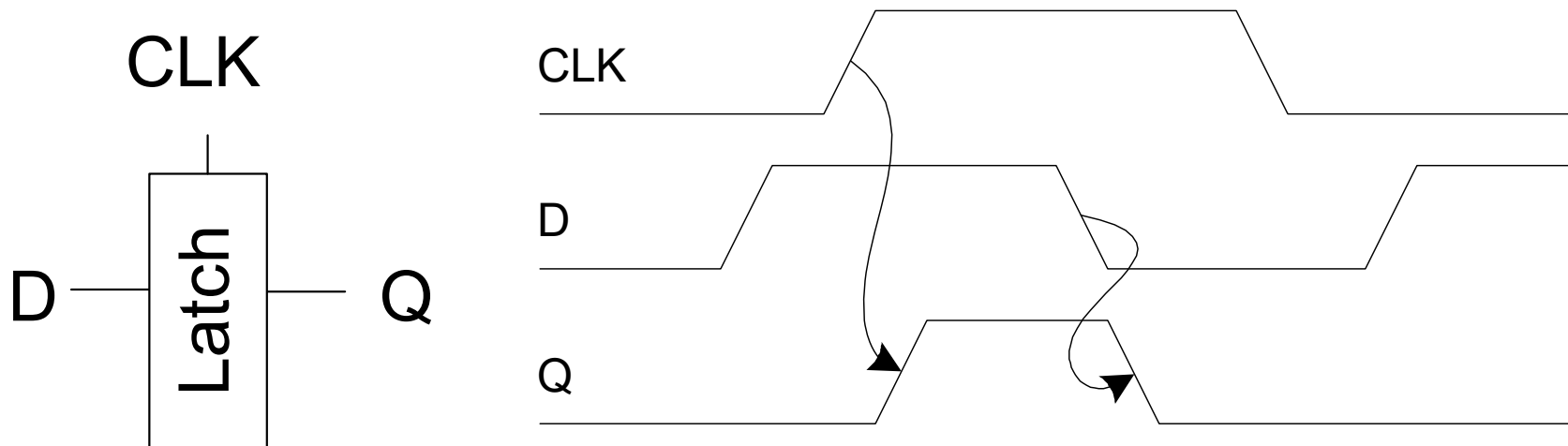




Lecture_4 Circuits & Layout

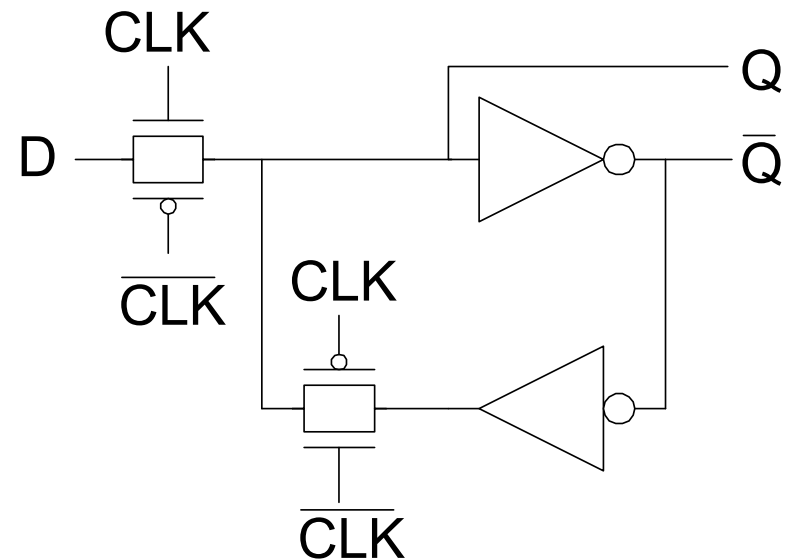
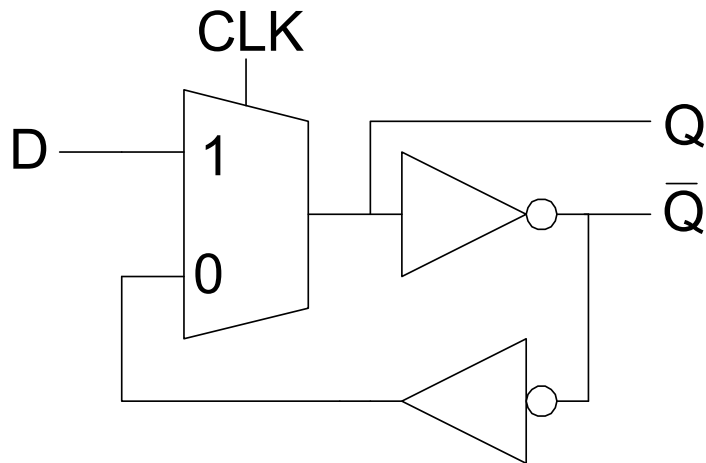
D Latch

- ❑ When $CLK = 1$, latch is *transparent*
 - D flows through to Q like a buffer
- ❑ When $CLK = 0$, the latch is *opaque*
 - Q holds its old value independent of D
- ❑ a.k.a. *transparent latch* or *level-sensitive latch*



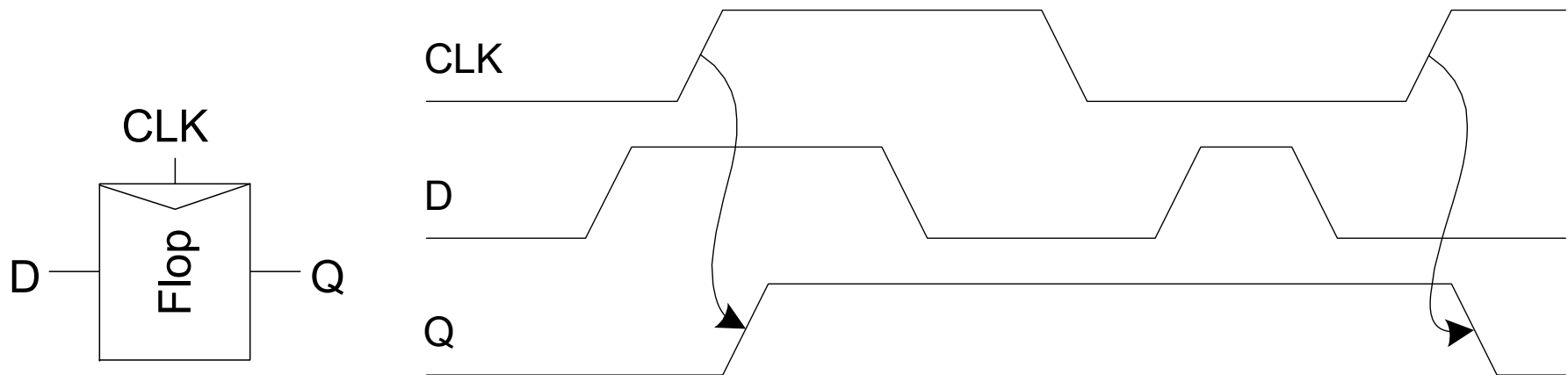
D Latch Design

- ❑ Multiplexer chooses D or old Q



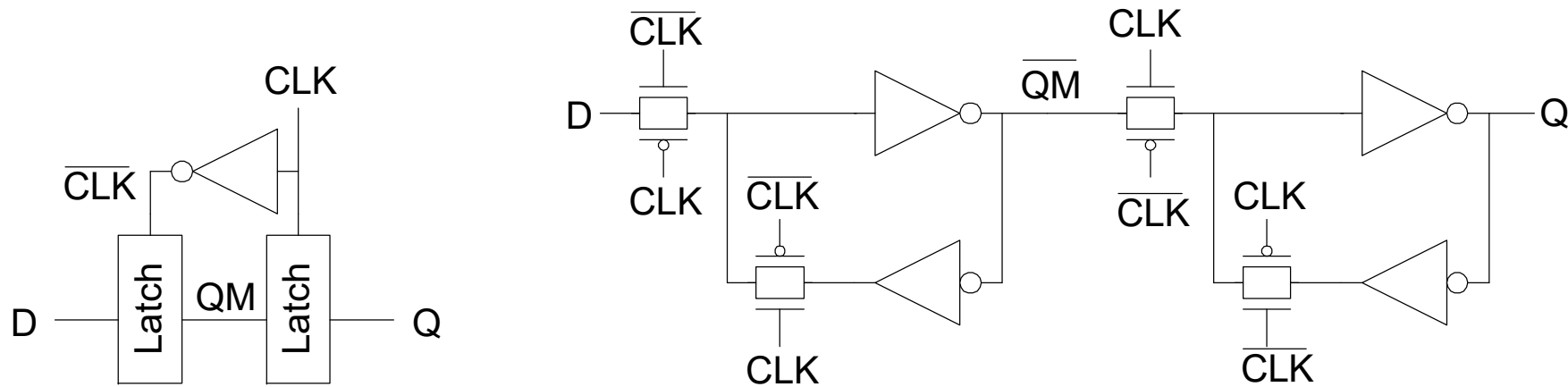
D Flip-flop

- ❑ When CLK rises, D is copied to Q
- ❑ At all other times, Q holds its value
- ❑ a.k.a. *positive edge-triggered flip-flop, master-slave flip-flop*



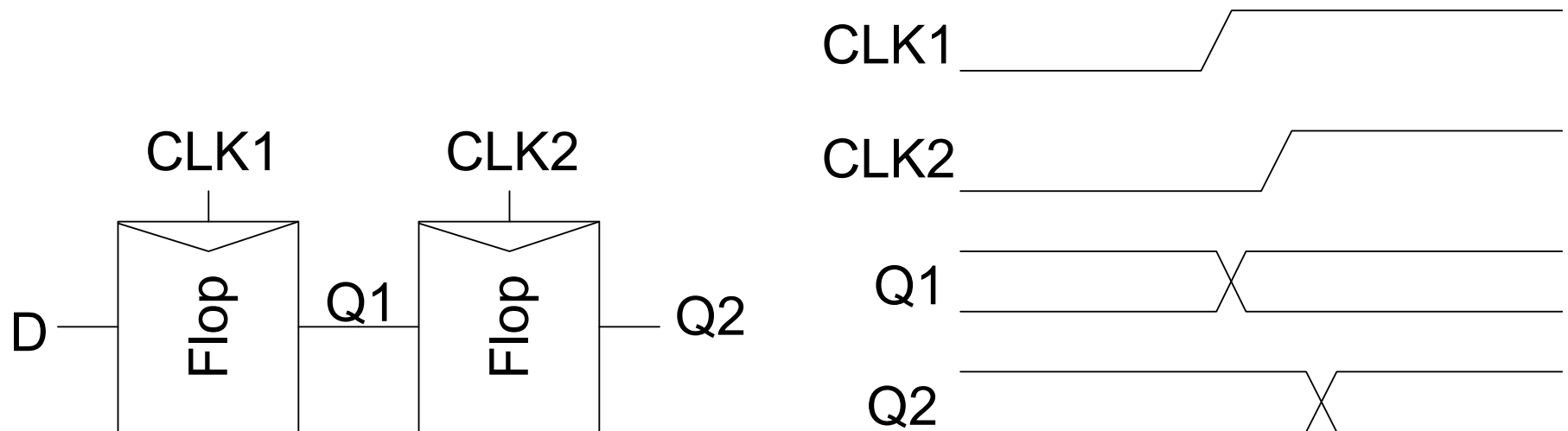
D Flip-flop Design

- Built from master and slave D latches



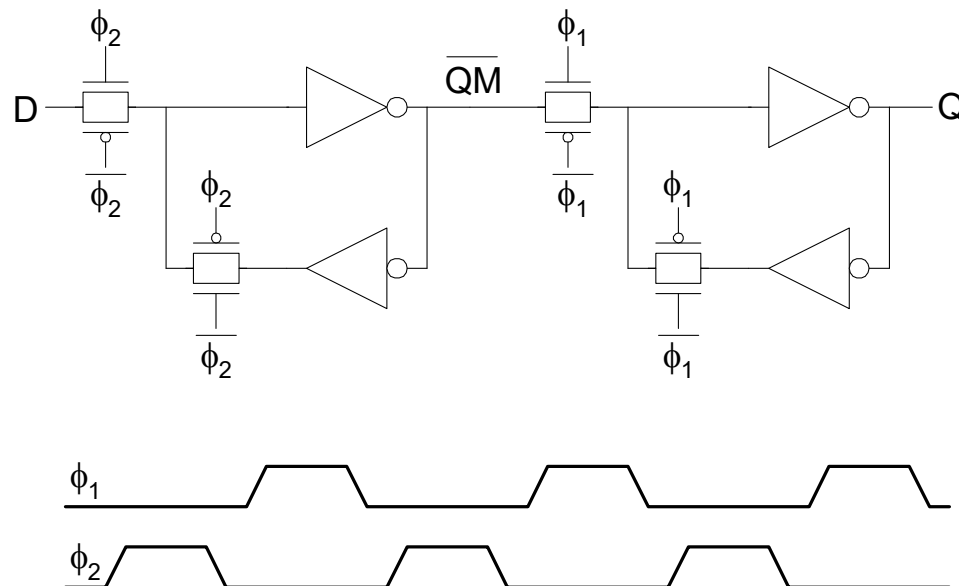
Race Condition

- ❑ Back-to-back flops can malfunction from clock skew
 - Second flip-flop fires late
 - Sees first flip-flop change and captures its result
 - Called *hold-time failure* or *race condition*



Nonoverlapping Clocks

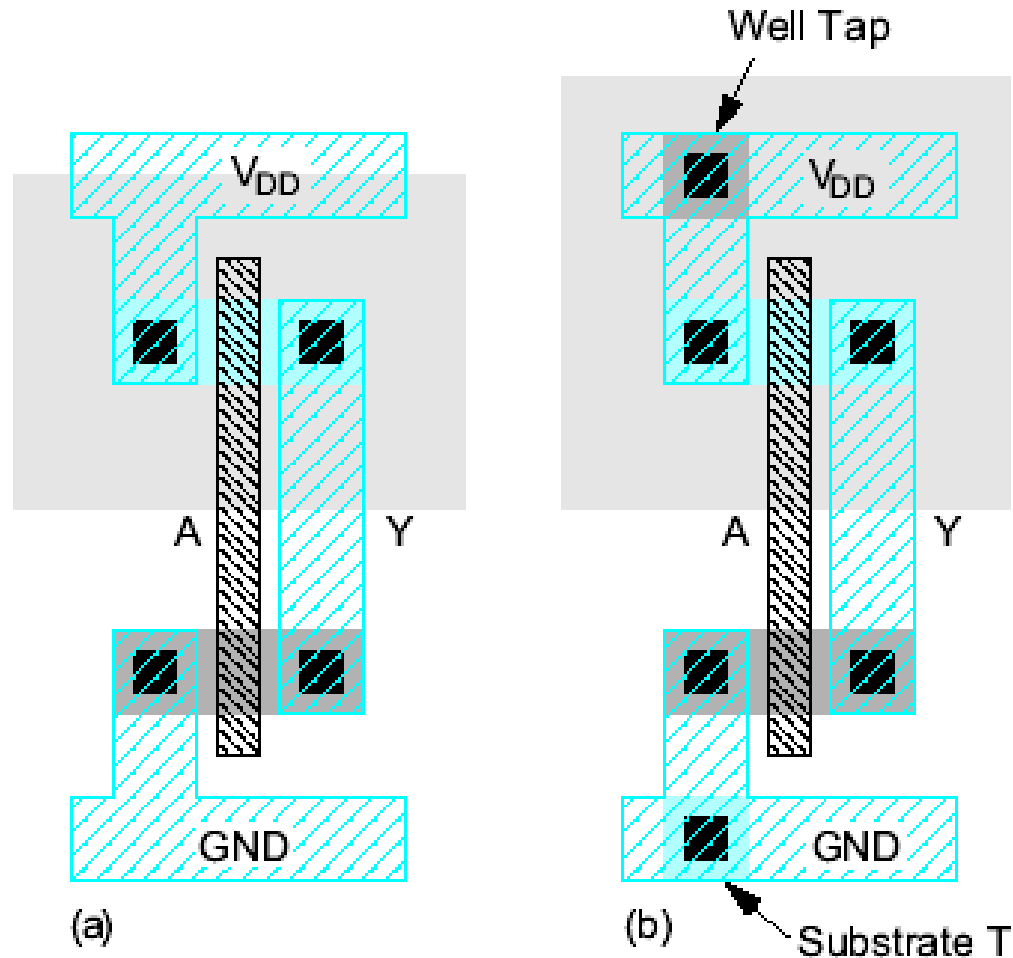
- ❑ Nonoverlapping clocks can prevent races
 - As long as nonoverlap exceeds clock skew
- ❑ We will use them in this class for safe design
 - Industry manages skew more carefully instead



→ Gate Layout

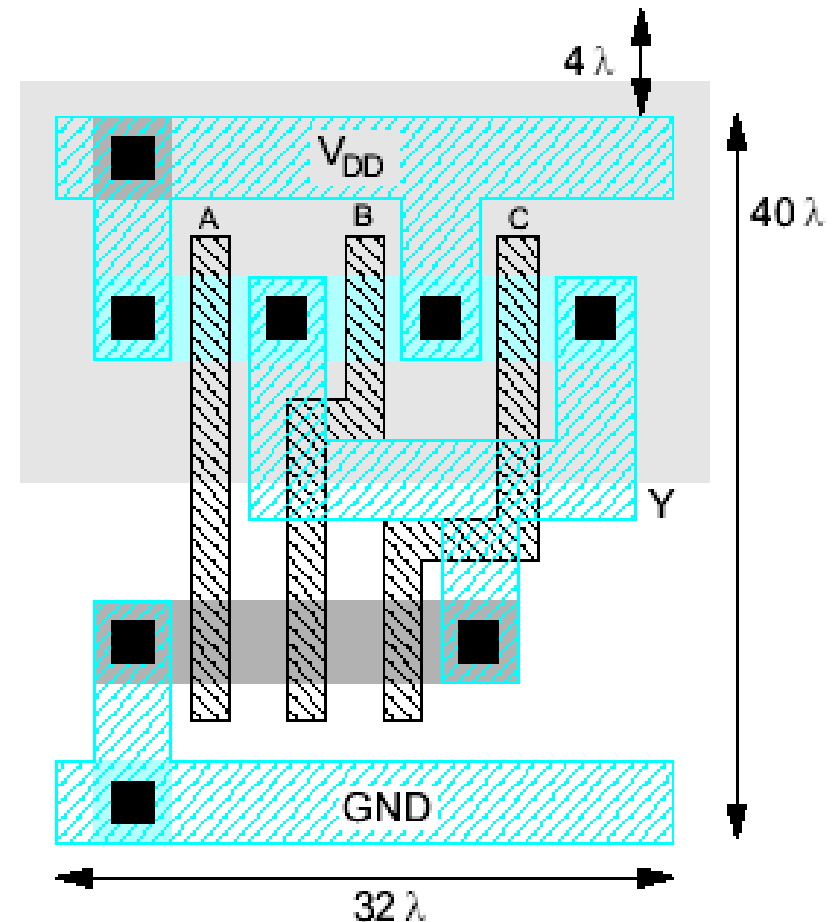
- Layout can be very time consuming
 - Design gates to fit together nicely
 - Build a library of standard cells
- Standard cell design methodology
 - V_{DD} and GND should abut (standard height)
 - Adjacent gates should satisfy design rules
 - nMOS at bottom and pMOS at top
 - All gates include well and substrate contacts

Example: Inverter



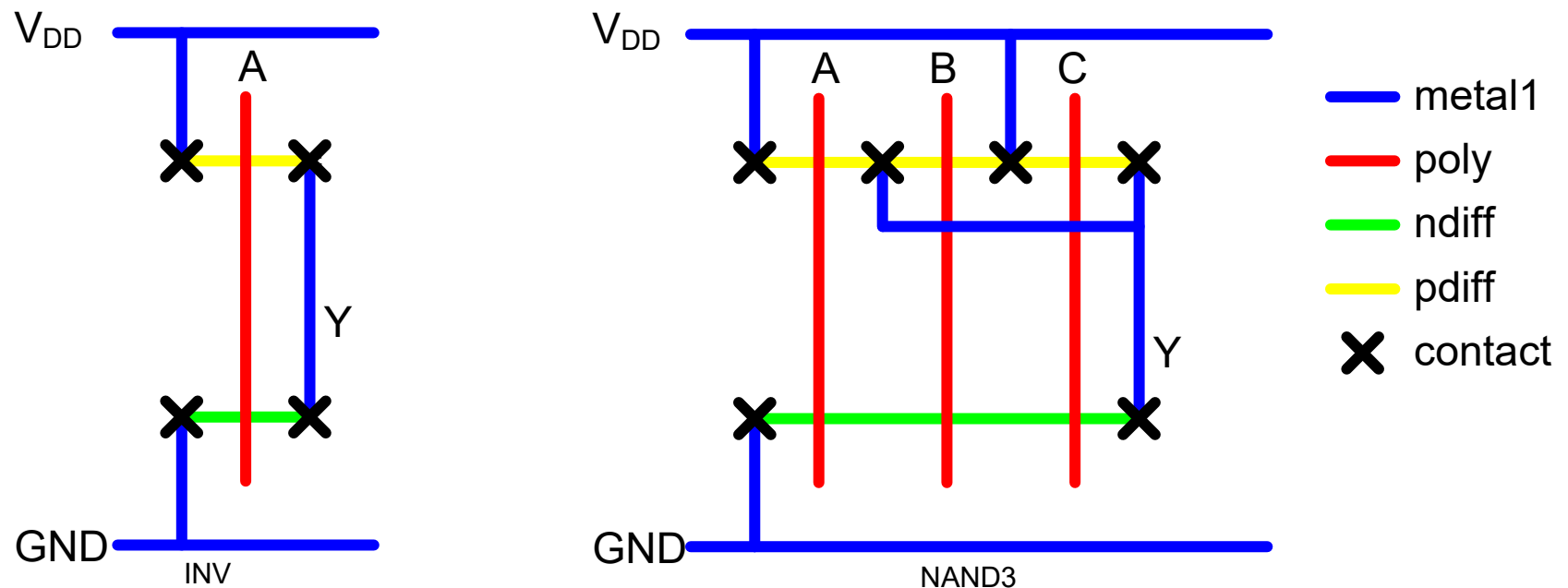
Example: NAND3

- ❑ Horizontal N-diffusion and p-diffusion strips
- ❑ Vertical polysilicon gates
- ❑ Metal1 V_{DD} rail at top
- ❑ Metal1 GND rail at bottom
- ❑ 32λ by 40λ



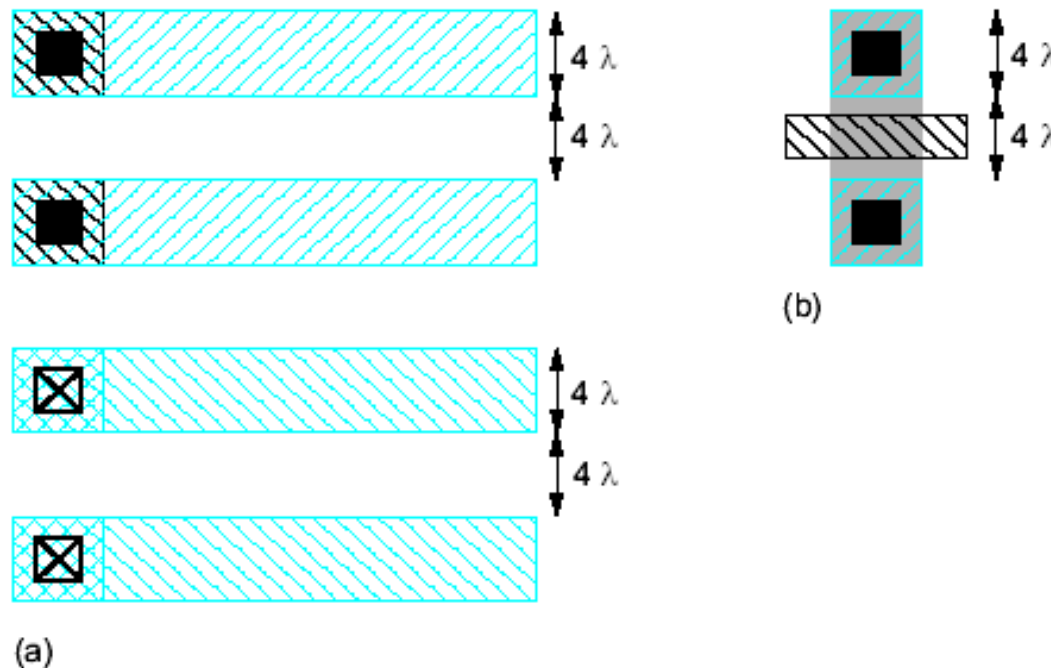
Stick Diagrams

- *Stick diagrams* help plan layout quickly
 - Need not be to scale
 - Draw with color pencils or dry-erase markers



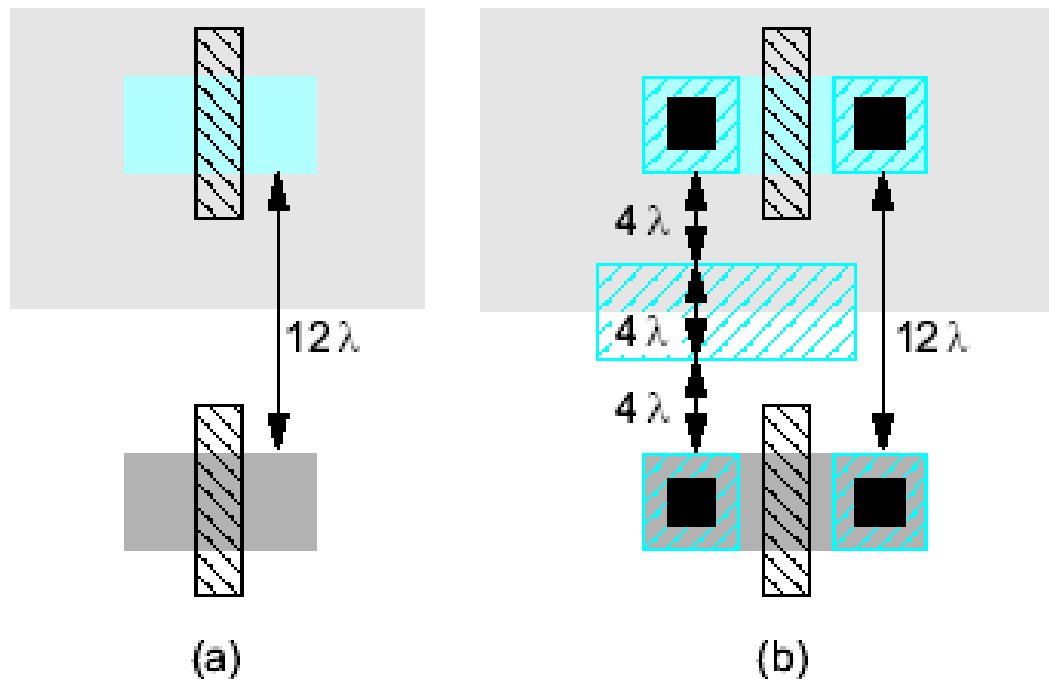
Wiring Tracks

- A *wiring track* is the space required for a wire
 - 4λ width, 4λ spacing from neighbor = 8λ pitch
- Transistors also consume one wiring track



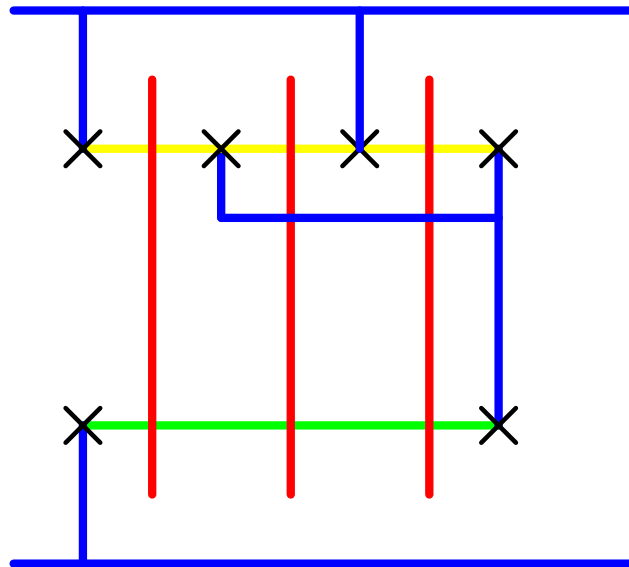
Well spacing

- Wells must surround transistors by 6λ
 - Implies 12λ between opposite transistor flavors
 - Leaves room for one wire track



Area Estimation

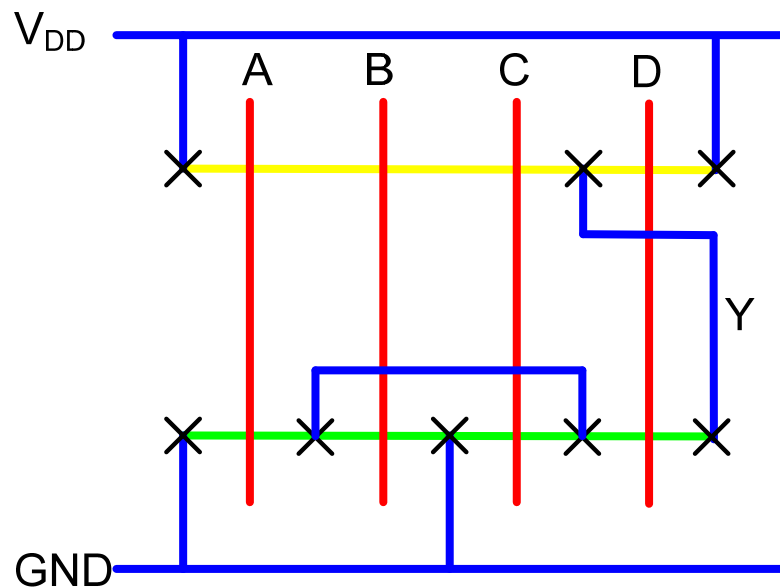
- Estimate area by counting wiring tracks
 - Multiply by 8 to express in λ



Example: O3AI

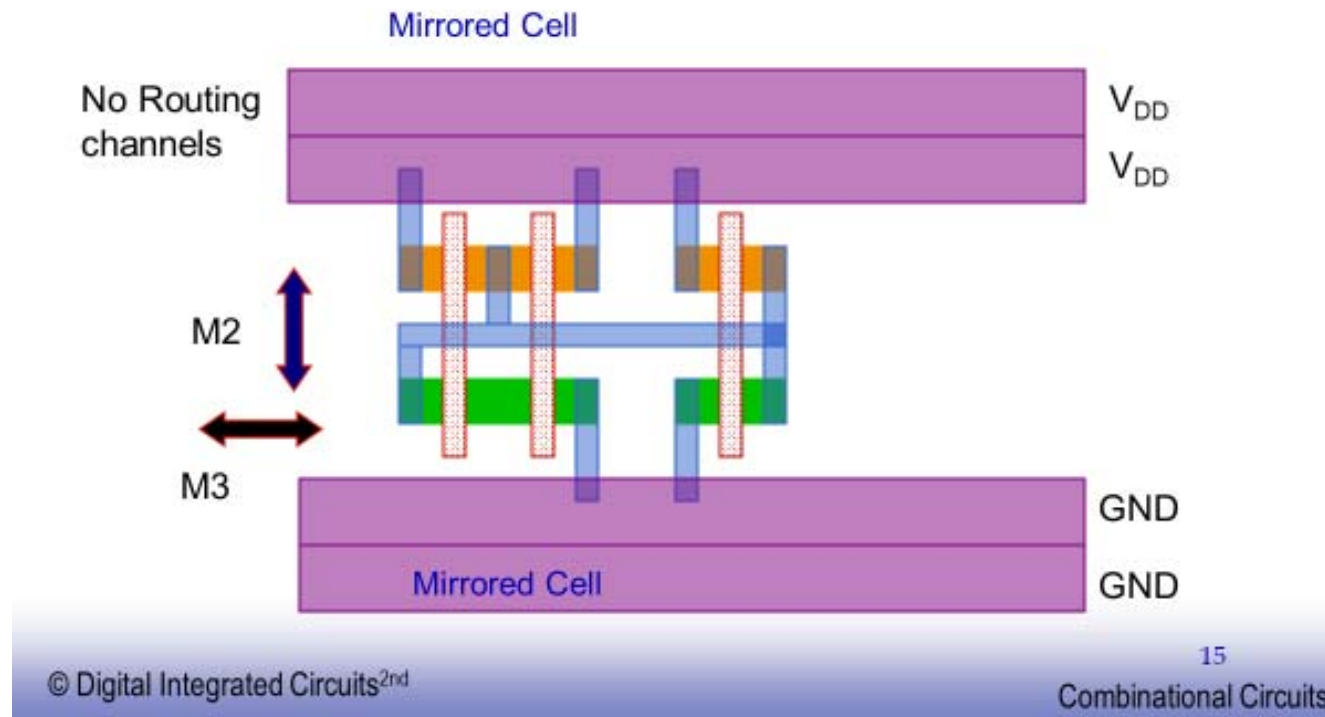
- Sketch a stick diagram for O3AI and estimate area

$$Y = \overline{(A + B + C)} \square D$$



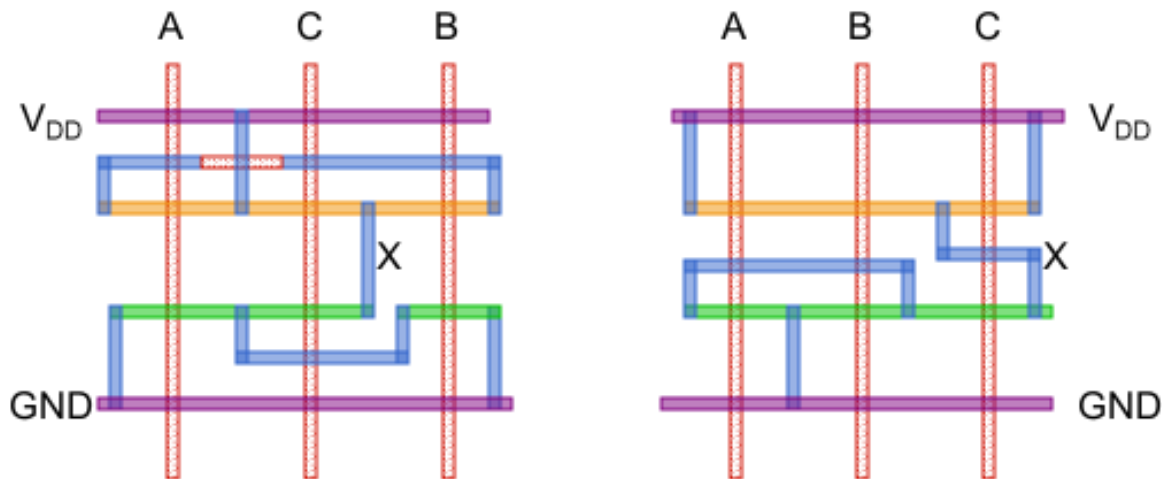
Euler's Path

Standard Cell Layout Methodology – 1990s



Euler's Path

Two Versions of $C \cdot (A + B)$

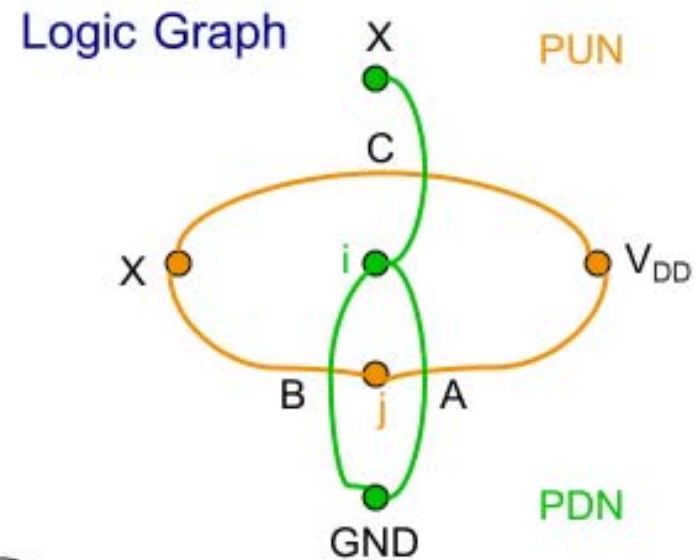
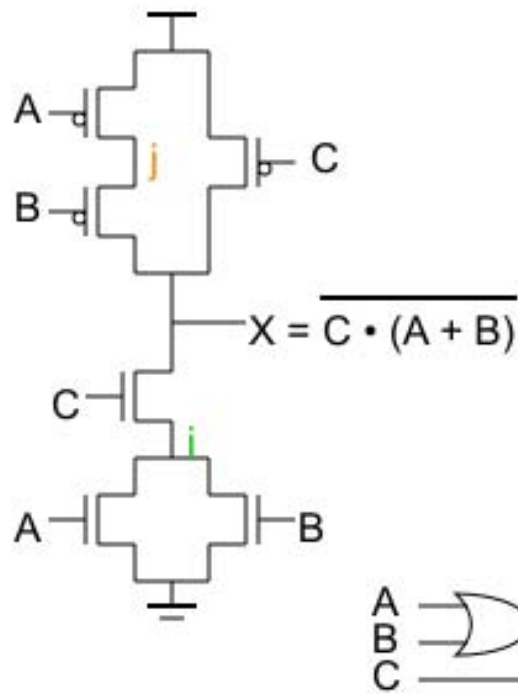


© Digital Integrated Circuits^{2nd}

21
Combinational Circuits

Euler's Path

Stick Diagrams



Euler's Path

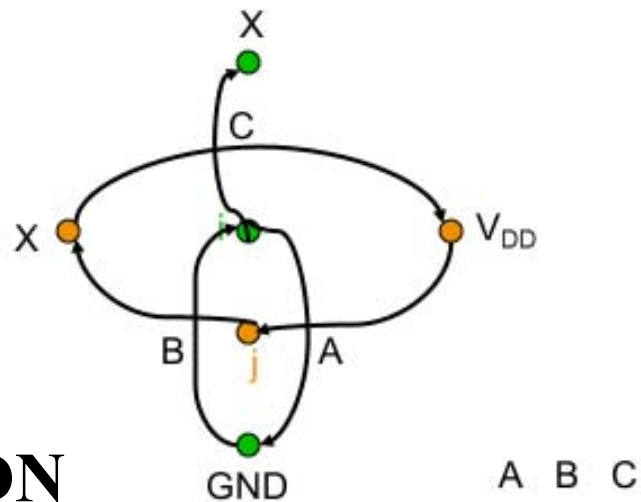
Consistent Euler Path

A B C

Has a PUN and PDN

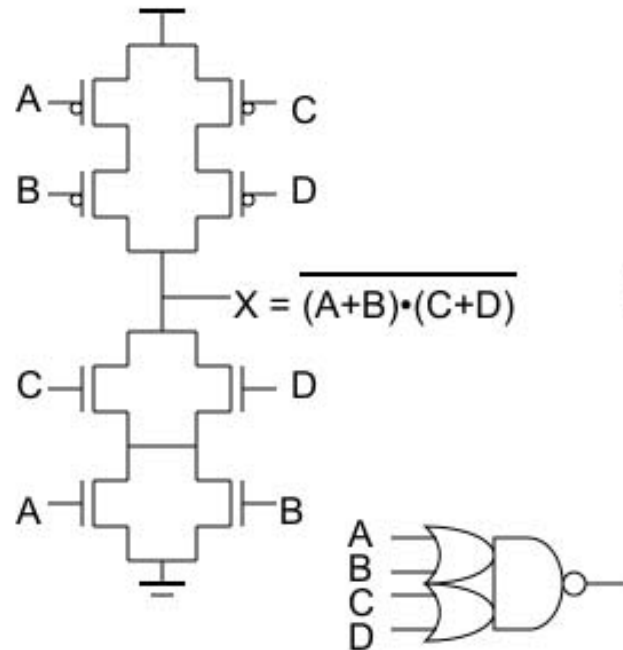
B C A

Has a PUN but no PDN



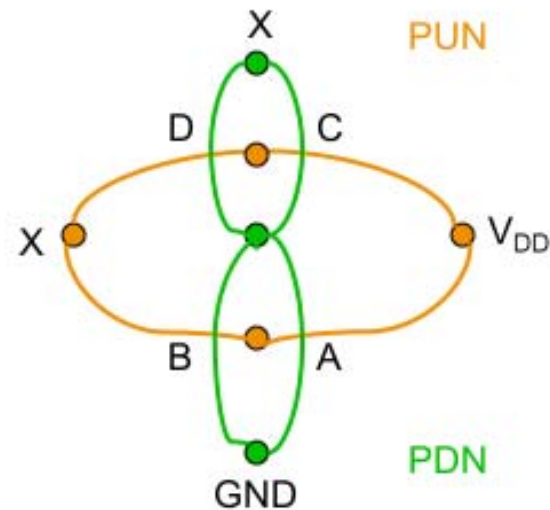
OAI22 Logic Graph

OAI22 Logic Graph



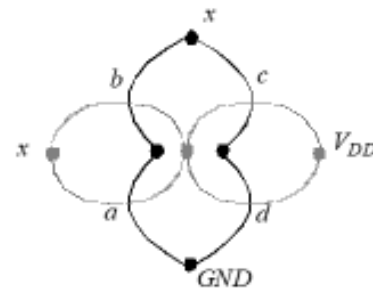
A B C D PDN bot not PUN

A B D C PDN and PUN

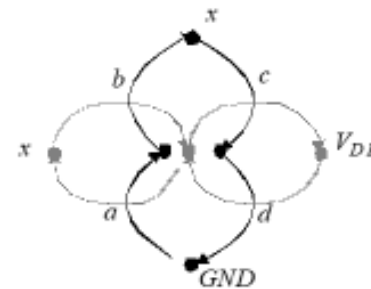


Example: $x = ab + cd$

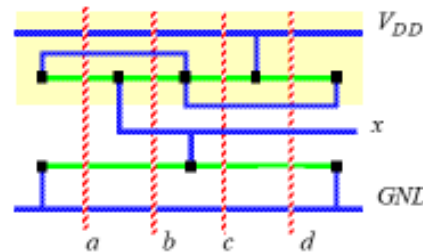
Example: $x = ab + cd$



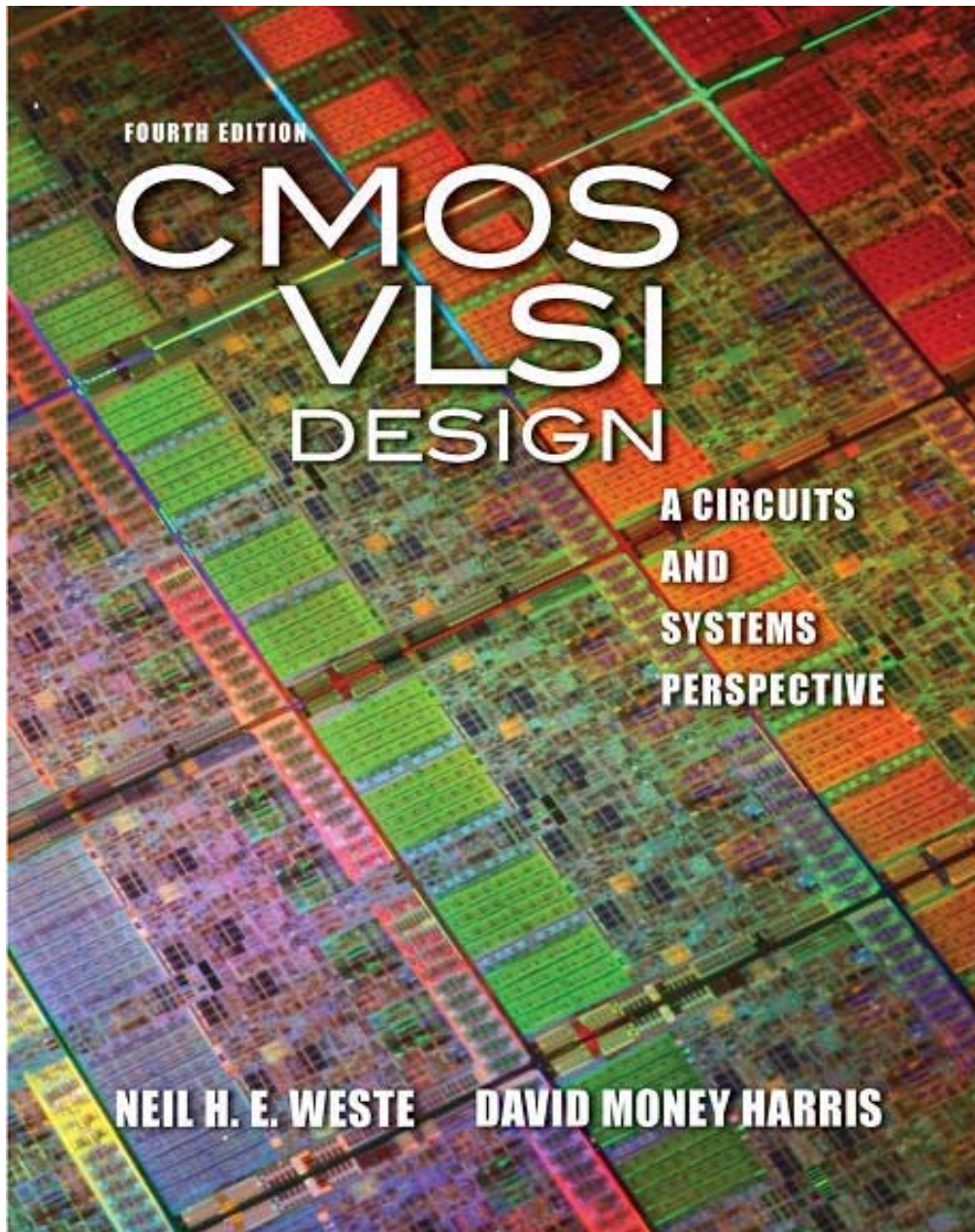
(a) Logic graphs for $\overline{(ab+cd)}$



(b) Euler Paths $\{a b c d\}$



(c) stick diagram for ordering $\{a b c d\}$



Lecture_5: Logical Effort

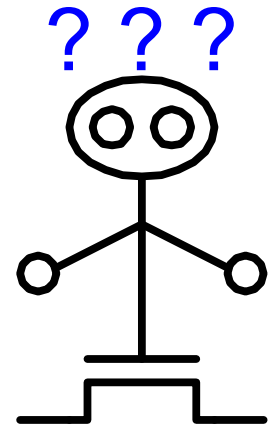
Outline

- Logical Effort
- Delay in a Logic Gate
- Multistage Logic Networks
- Choosing the Best Number of Stages
- Example
- Summary

Introduction

- ❑ Chip designers face a bewildering array of choices
 - What is the best circuit topology for a function?
 - How many stages of logic give least delay?
 - How wide should the transistors be?

- ❑ Logical effort is a method to make these decisions
 - Uses a simple model of delay
 - Allows back-of-the-envelope calculations
 - Helps make rapid comparisons between alternatives
 - Emphasizes remarkable symmetries



Example

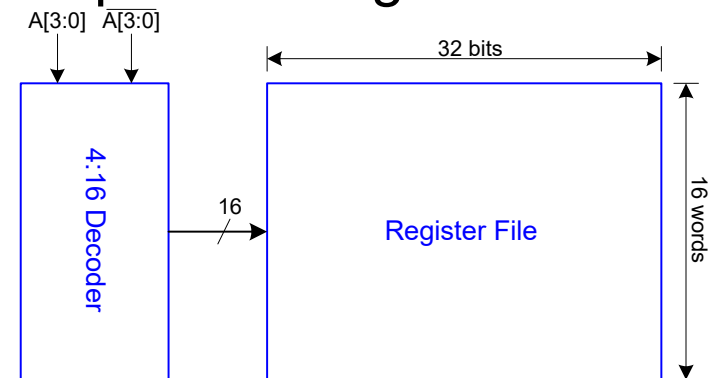
- ❑ Ben Bitdiddle is the memory designer for the Motoroil 68W86, an embedded automotive processor. Help Ben design the decoder for a register file.

- ❑ Decoder specifications:

- 16 word register file
- Each word is 32 bits wide
- Each bit presents load of 3 unit-sized transistors
- True and complementary address inputs $A[3:0]$
- Each input may drive 10 unit-sized transistors

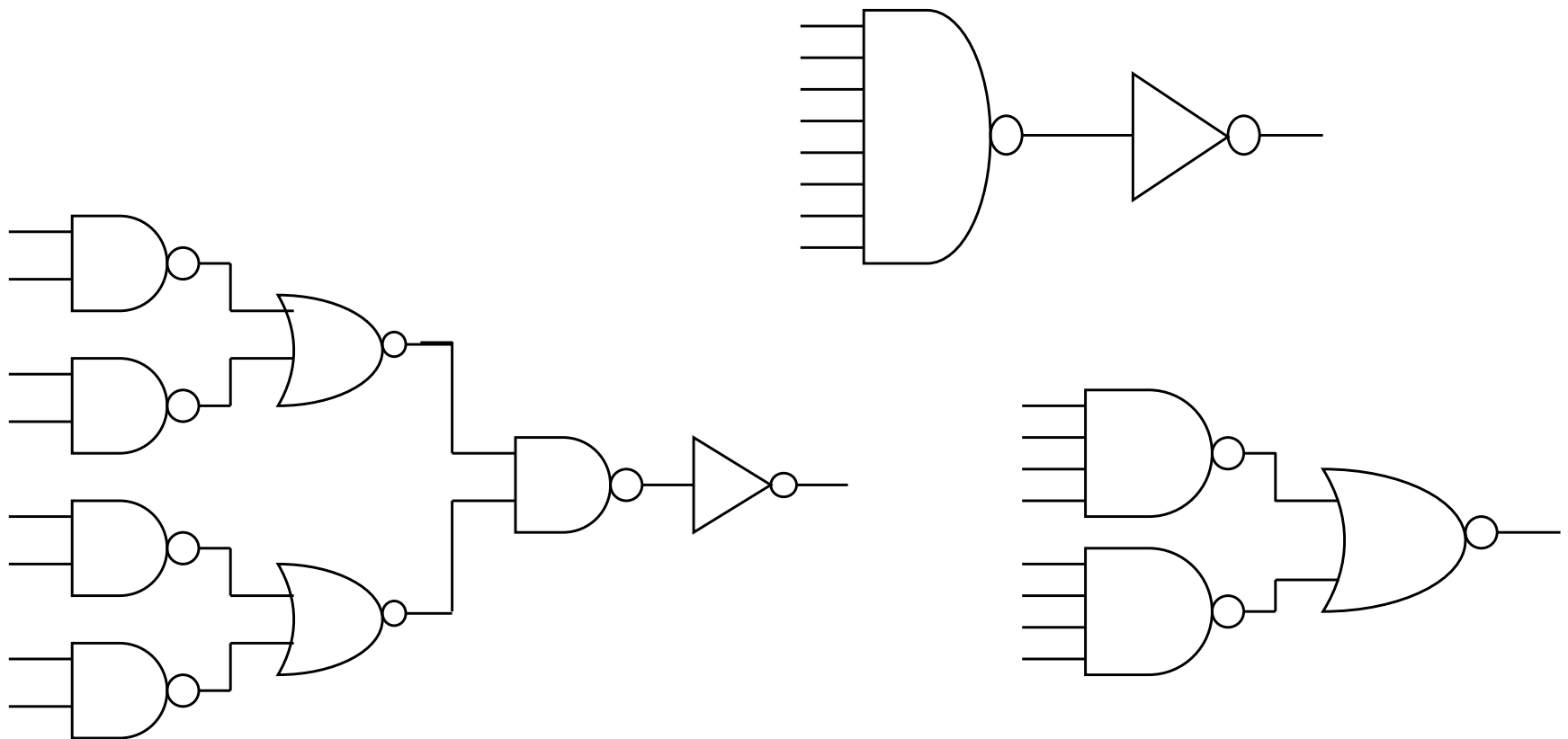
- ❑ Ben needs to decide:

- How many stages to use?
- How large should each gate be?
- How fast can decoder operate?



Alternative Logic Structures

$$F = ABCDEFGH$$



Delay in a Logic Gate

- ❑ Express delays in process-independent unit $d = \frac{d_{abs}}{\tau}$
- ❑ Delay has two components: $d = f + p$
- ❑ f : effort delay = gh (a.k.a. stage effort)
 - Again, has two components
- ❑ g : logical effort
 - Measures relative ability of gate to deliver current
 - $g \equiv 1$ for inverter
- ❑ h : electrical effort = C_{out} / C_{in}
 - Ratio of output to input capacitance
 - Sometimes called fanout
- ❑ p : parasitic delay
 - Represents delay of gate driving no load
 - Set by internal parasitic capacitance

$$\tau = 3RC$$

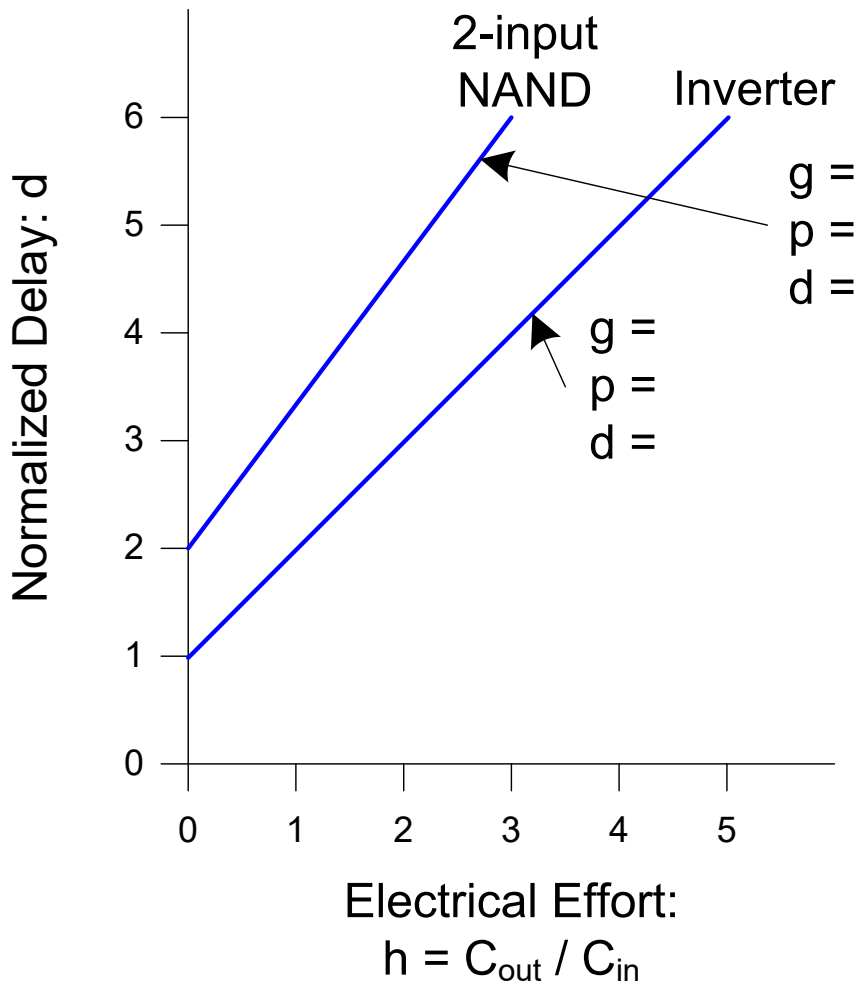
$\approx 3 \text{ ps in } 65 \text{ nm process}$
 $60 \text{ ps in } 0.6 \mu\text{m process}$

$$p = \text{fan-in}$$

$$g = \frac{C_{gatenorm}}{C_{invnorm}} = \frac{C_{gate}}{3}$$

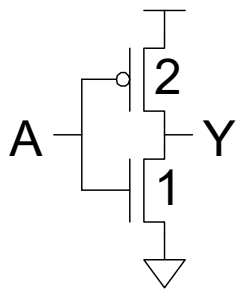
Delay Plots

$$d = f + p$$
$$= gh + p$$

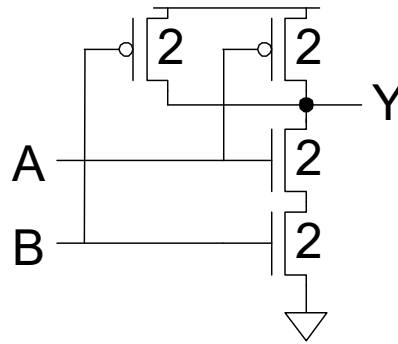


Computing Logical Effort

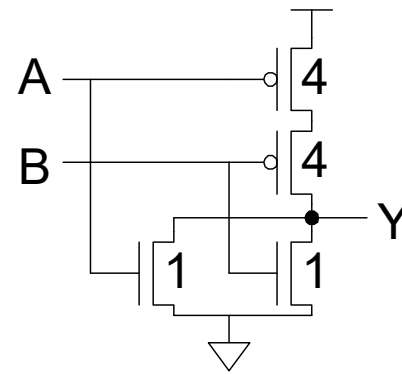
- ❑ DEF: *Logical effort is the ratio of the input capacitance of a gate to the input capacitance of an inverter delivering the same output current.*
- ❑ Measure from delay vs. fanout plots
- ❑ Or estimate by counting transistor widths



$$C_{in} = 3$$
$$g = 3/3$$



$$C_{in} = 4$$
$$g = 4/3$$



$$C_{in} = 5$$
$$g = 5/3$$

Catalog of Gates

□ Logical effort of common gates

Gate type	Number of inputs				
	1	2	3	4	n
Inverter	1				
NAND		$4/3$	$5/3$	$6/3$	$(n+2)/3$
NOR		$5/3$	$7/3$	$9/3$	$(2n+1)/3$
Tristate / mux	2	2	2	2	2
XOR, XNOR		4, 4	6, 12, 6	8, 16, 16, 8	

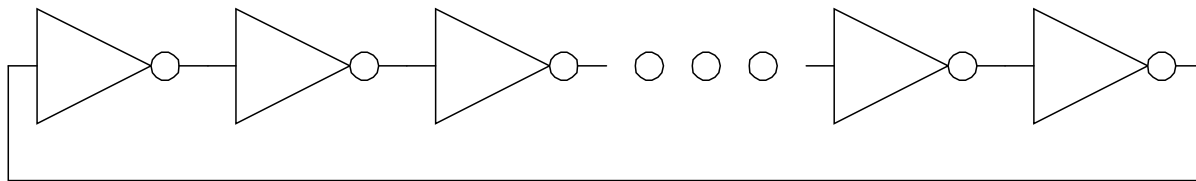
Catalog of Gates

- Parasitic delay of common gates
 - In multiples of p_{inv} (≈ 1)

Gate type	Number of inputs				
	1	2	3	4	n
Inverter	1				
NAND		2	3	4	n
NOR		2	3	4	n
Tristate / mux	2	4	6	8	2n
XOR, XNOR		4	6	8	

Example: Ring Oscillator

- Estimate the frequency of an N-stage ring oscillator



Logical Effort: $g =$

Electrical Effort: $h =$

Parasitic Delay: $p =$

Stage Delay: $d =$

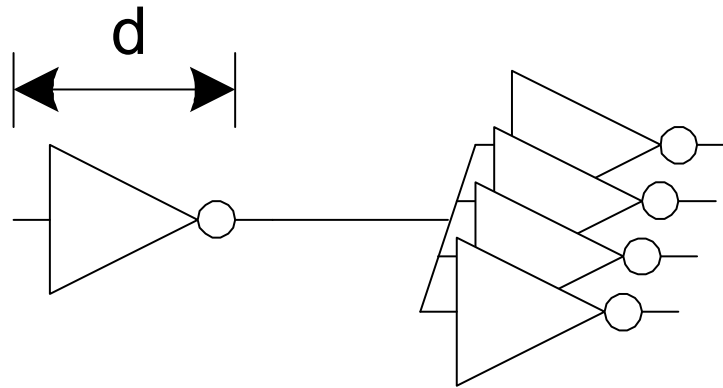
Frequency: $f_{osc} =$

31 stage ring oscillator in
0.6 μm process has
frequency of ~ 200 MHz

$$f_{osc} = \frac{1}{4Nt_{inv}} \text{ Hz}$$

Example: FO4 Inverter

- Estimate the delay of a fanout-of-4 (FO4) inverter



Logical Effort: $g =$

Electrical Effort: $h =$

Parasitic Delay: $p =$

Stage Delay: $d =$

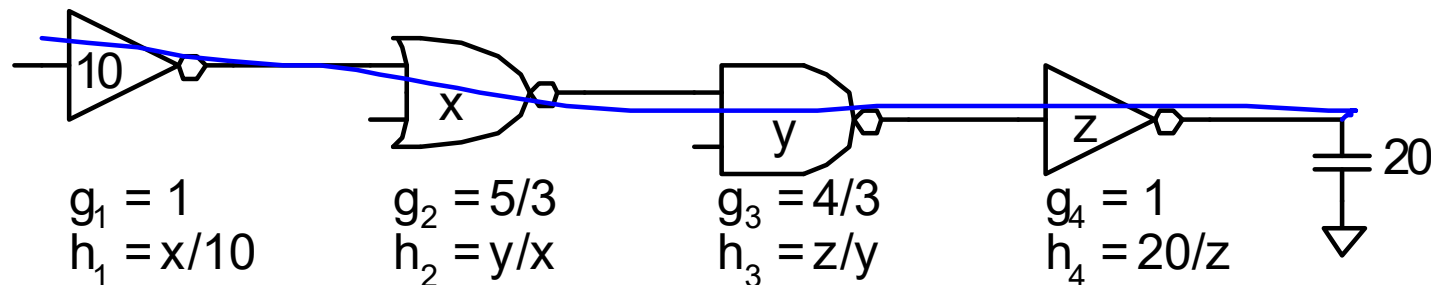
Multistage Logic Networks

❑ Logical effort generalizes to multistage networks

❑ *Path Logical Effort* $G = \prod g_i$

❑ *Path Electrical Effort* $H = \frac{C_{\text{out-path}}}{C_{\text{in-path}}}$

❑ *Path Effort Delay* $F = \prod f_i = \prod g_i h_i$



Multistage Logic Networks

❑ Logical effort generalizes to multistage networks

❑ *Path Logical Effort* $G = \prod g_i$

❑ *Path Electrical Effort* $H = \frac{C_{out-path}}{C_{in-path}}$

❑ *Path Effort Delay* $F = \prod f_i = \prod g_i h_i$

❑ Can we write $F = GH$?

Paths that Branch

❑ No! Consider paths that branch:

G =

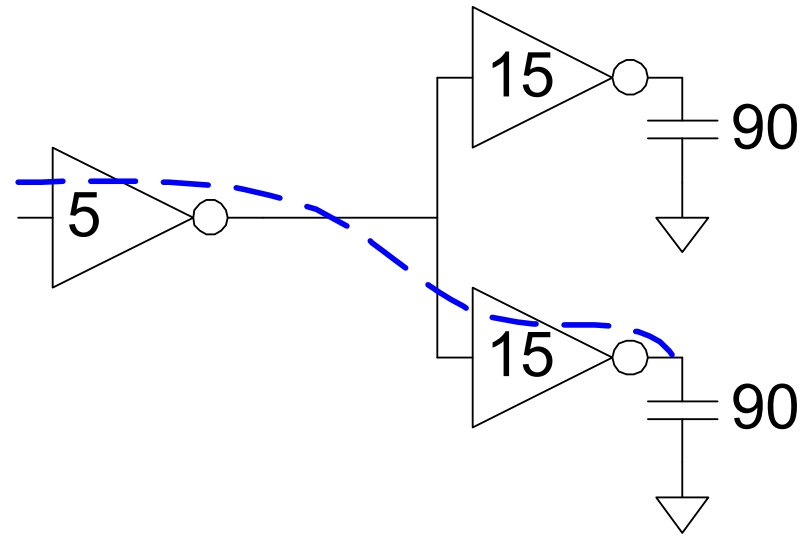
H =

GH =

h_1 =

h_2 =

F =



Branching Effort

- Introduce *branching effort*
 - Accounts for branching between stages in path

$$b = \frac{C_{\text{on path}} + C_{\text{off path}}}{C_{\text{on path}}}$$

$$B = \prod b_i$$

Note:

$$\prod h_i = BH$$

- Now we compute the path effort delay.
 - $F = GBH$

Multistage Delays

- Path Effort Delay $D_F = \sum f_i$
- Path Parasitic Delay $P = \sum p_i$
- Path Delay $D = \sum d_i = D_F + P$

Designing Fast Circuits

$$D = \sum d_i = D_F + P$$

- Delay is smallest when each stage bears same effort

$$\hat{f} = g_i h_i = F^{\frac{1}{N}}$$

- Thus minimum delay of N stage path is



- This is a **key** result of logical effort
 - Find fastest possible delay
 - Doesn't require calculating gate sizes

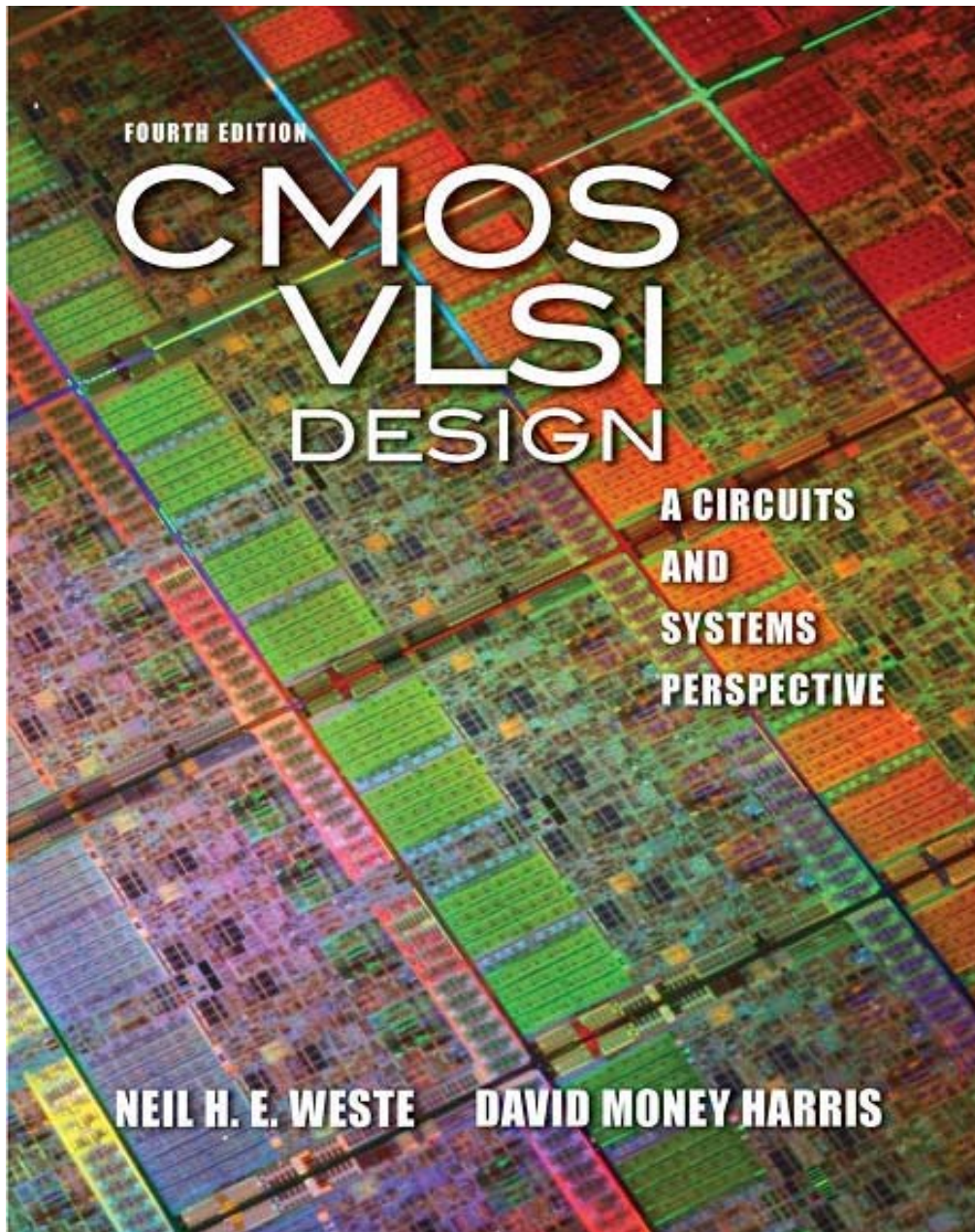
Gate Sizes

- How wide should the gates be for least delay?

$$\hat{f} = gh = g \frac{C_{out}}{C_{in}}$$

$$\Rightarrow C_{in_i} = \frac{g_i C_{out_i}}{\hat{f}}$$

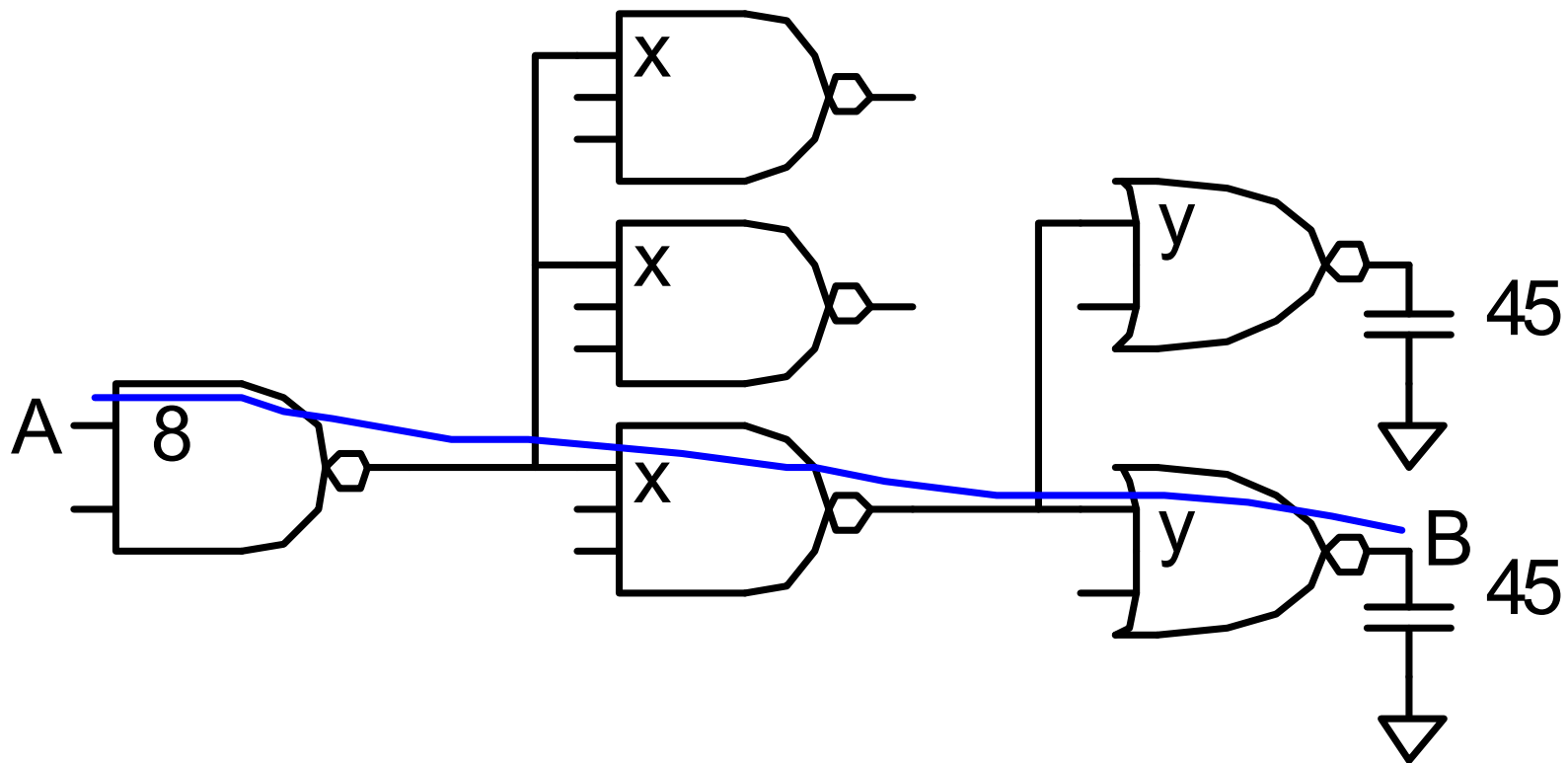
- Working backward, apply capacitance transformation to find input capacitance of each gate given the load it drives.
- Check work by verifying input cap spec is met.



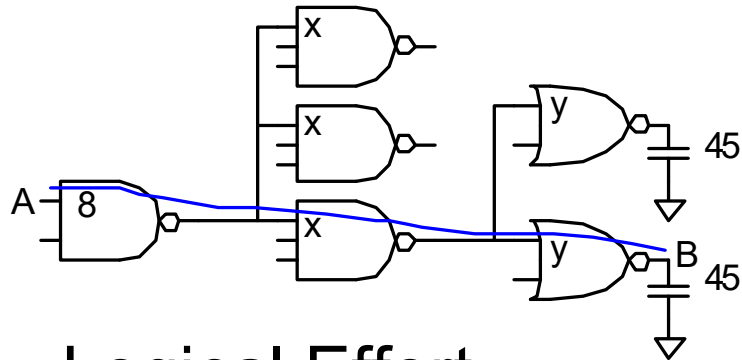
Lecture_6: Logical Effort

Example: 3-stage path

- Select gate sizes x and y for least delay from A to B



Example: 3-stage path



Logical Effort $G =$

Electrical Effort $H =$

Branching Effort $B =$

Path Effort $F =$

Best Stage Effort $\hat{f} =$

Parasitic Delay $P =$

Delay $D =$

Example: 3-stage path

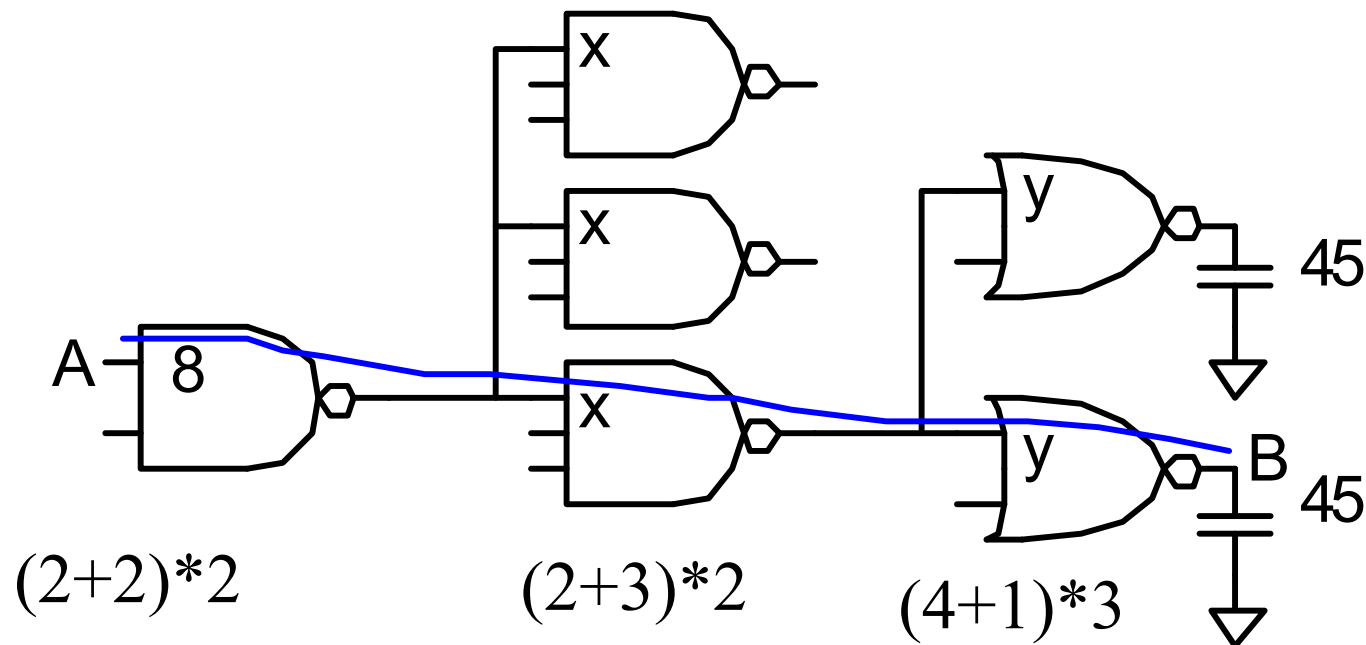
- Work backward for sizes

$y =$

$x =$

$2y/x$ for branching

$(3x/C_{in}).4/3 = 5$ gives $C_{in}=8$

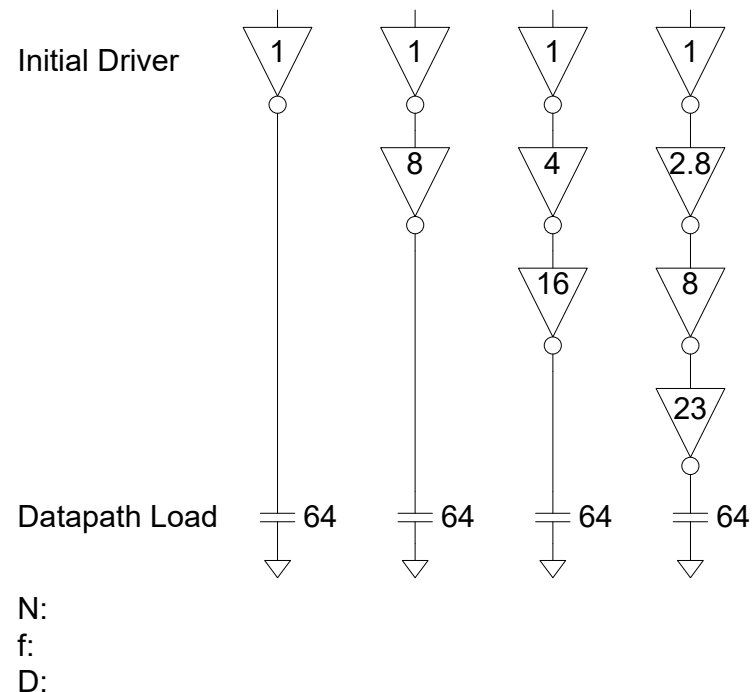


Best Number of Stages

- How many stages should a path use?
 - Minimizing number of stages is not always fastest
- Example: drive 64-bit datapath with unit inverter

D =

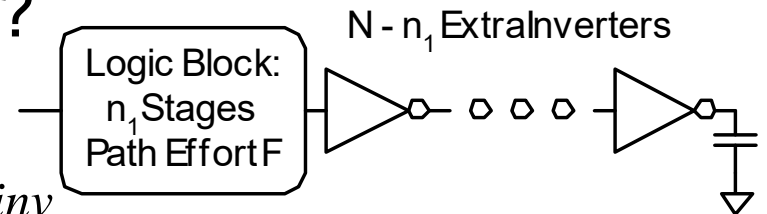
$g_i = 1$ then h_i is always
Equal to $F \frac{1}{N} = f$



Derivation

- Consider adding inverters to end of path
 - How many give least delay?

$$D = NF^{\frac{1}{N}} + \sum_{i=1}^{n_1} p_i + (N - n_1) p_{inv}$$



$$\frac{\partial D}{\partial N} = -\frac{1}{N} F^{\frac{1}{N}} \ln F + F^{\frac{1}{N}} + p_{inv} = 0$$

- Define best stage effort $\rho = F^{\frac{1}{N}}$

$$p_{inv} + \rho(1 - \ln \rho) = 0$$

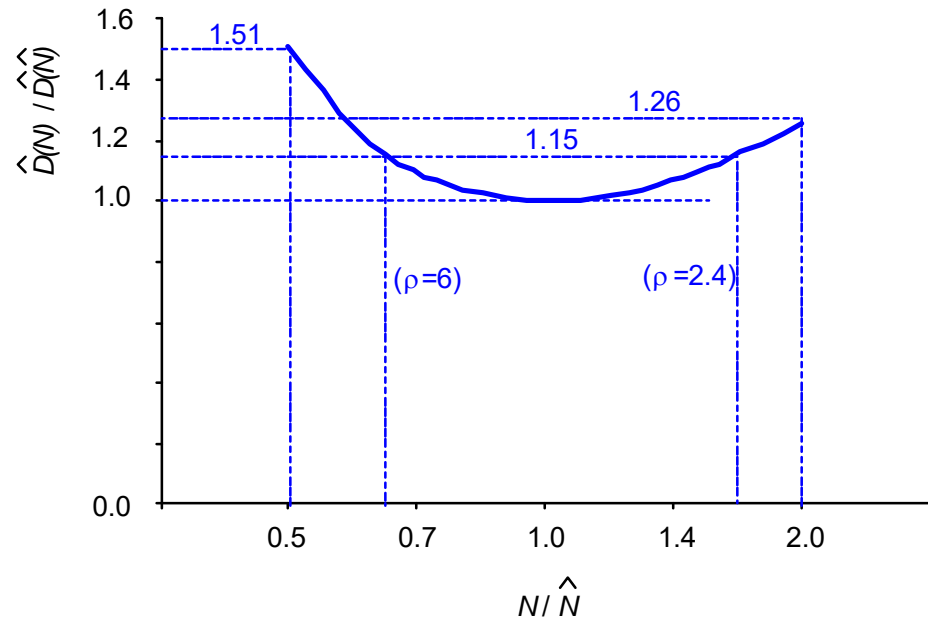
$$\frac{d}{dx} (ax^{\frac{1}{x}}) = -\frac{ax^{\frac{1}{x}} \ln a}{x^2}$$

Best Stage Effort

- $p_{inv} + \rho(1 - \ln \rho) = 0$ has no closed-form solution
- Neglecting parasitic ($p_{inv} = 0$), we find $\rho = 2.718$ (e)
- For $p_{inv} = 1$, solve numerically for $\rho = 3.59$

Sensitivity Analysis

- How sensitive is delay to using exactly the best number of stages?



- $2.4 < \rho < 6$ gives delay within 15% of optimal

- We can be sloppy!

- I like $\rho = 4$

$$\rho = 4 = F\overline{N} \Rightarrow N = \log_4 F$$

Example, Revisited

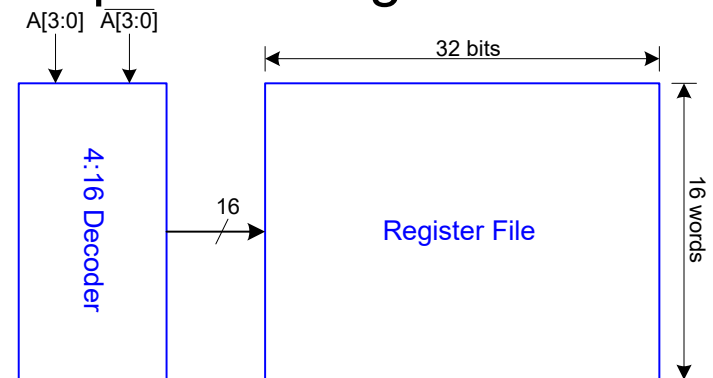
- ❑ Ben Bitdiddle is the memory designer for the Motoroil 68W86, an embedded automotive processor. Help Ben design the decoder for a register file.

- ❑ Decoder specifications:

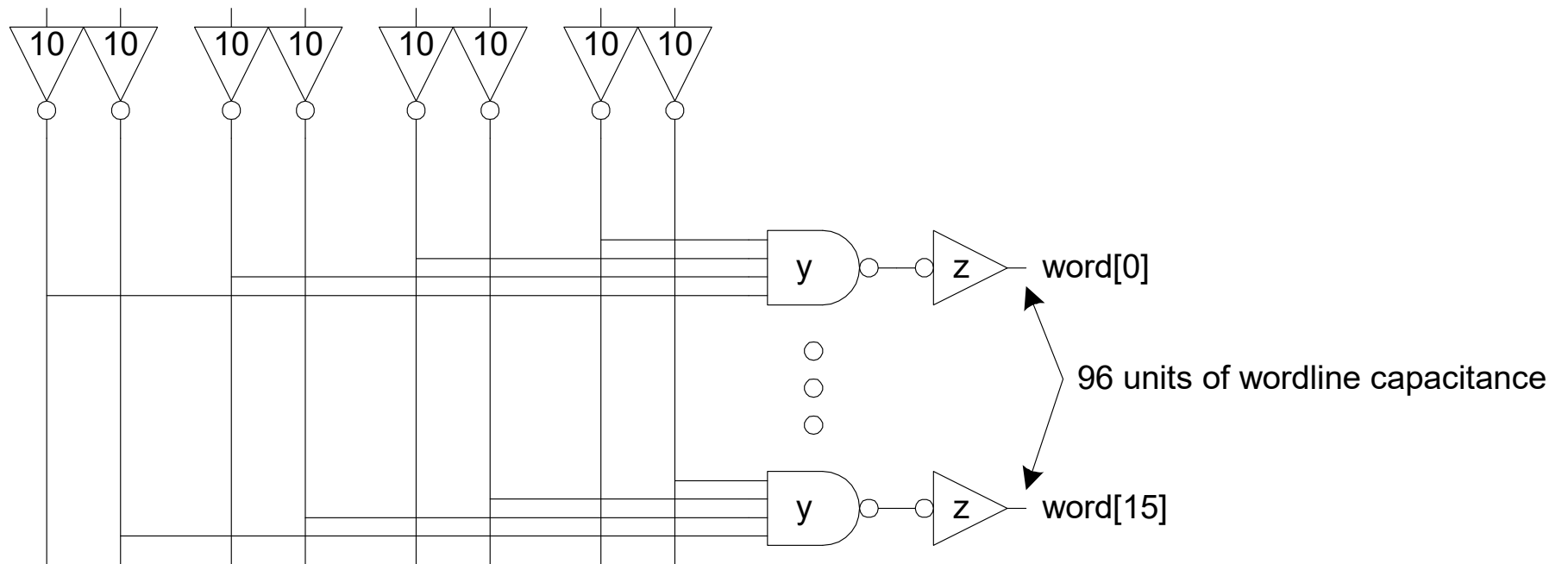
- 16 word register file
- Each word is 32 bits wide
- Each bit presents load of 3 unit-sized transistors
- True and complementary address inputs $A[3:0]$
- Each input may drive 10 unit-sized transistors

- ❑ Ben needs to decide:

- How many stages to use?
- How large should each gate be?
- How fast can decoder operate?



A[3] $\overline{A[3]}$ A[2] $\overline{A[2]}$ A[1] $\overline{A[1]}$ A[0] $\overline{A[0]}$



Number of Stages

- ❑ Decoder effort is mainly electrical and branching

Electrical Effort: $H =$

Branching Effort: $B =$

- ❑ If we neglect logical effort (assume $G = 1$)

Path Effort: $F =$

Number of Stages: $N =$

- ❑ Try a -stage design

Gate Sizes & Delay

Logical Effort: $G =$

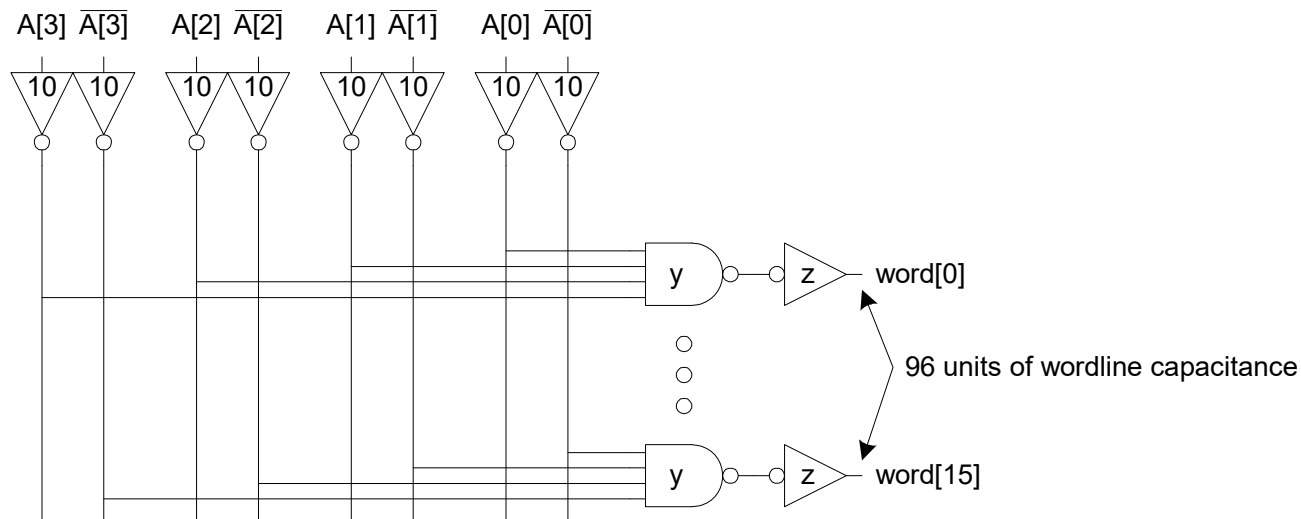
Path Effort: $F =$

Stage Effort: $\hat{f} =$

Path Delay: $D =$

Gate sizes: $z =$

$y =$



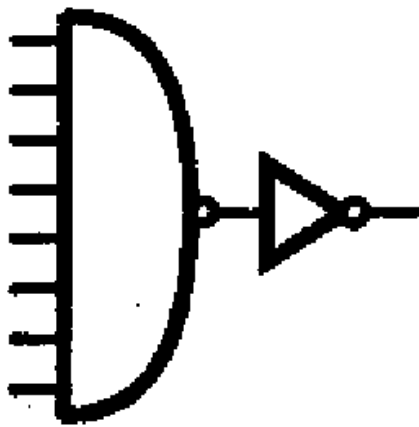
G ,H and B Calculations

- $G = 1(\text{INV}10) * 6/3 (\text{NAND}4) * 1(\text{INV}z) = 6/3 = 2$
- $H = 3 * 32/10 = 9.6$
- B, each input is connected to 8 words because the input variables A[0-3] and their complements are available.
 - So, path branching is (1+7)/1 one ON path and seven OFF paths.
 - So, B is equal to 8
- Then $F = GHB = 6/3 * 9.6 * 8 = 153.6 \sim 154$

Which is the best!!

Low C_L

Large C_L

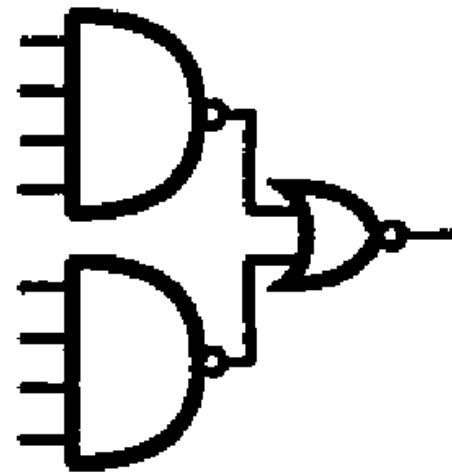


$$g=10/3 \quad g=1$$

(a)

$$G = 10/3$$

$$P = 8 + 1 = 9$$

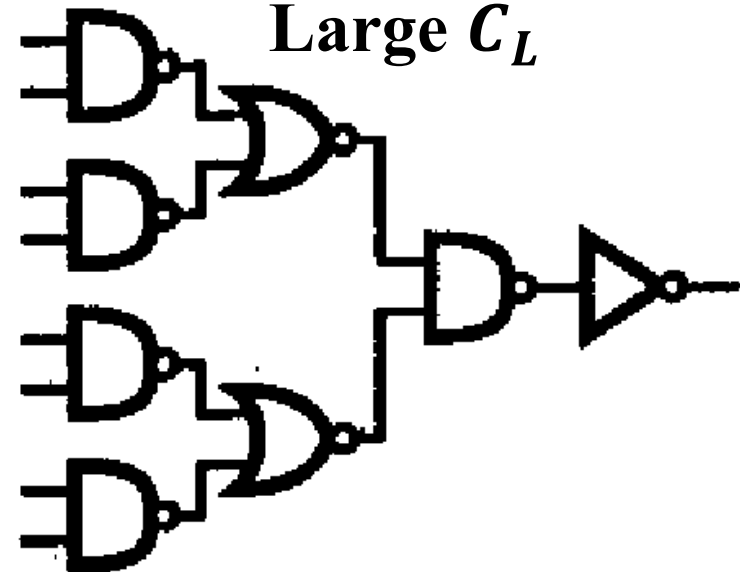


$$g=2 \quad g=5/3$$

(b)

$$G = 10/3$$

$$P = 4 + 2 = 6$$



$$g=4/3 \quad g=5/3 \quad g=4/3 \quad g=1$$

(c)

$$G = 80/27$$

$$P = 2 + 2 + 2 + 1 = 7$$

Comparison

- ❑ Compare many alternatives with a spreadsheet
- ❑ $D = N(76.8 G)^{1/N} + P$

Design	N	G	P	D
NOR4	1	3	4	234
NAND4-INV	2	2	5	29.8
NAND2-NOR2	2	20/9	4	30.1
INV-NAND4-INV	3	2	6	22.1
NAND4-INV-INV-INV	4	2	7	21.1
NAND2-NOR2-INV-INV	4	20/9	6	20.5
NAND2-INV-NAND2-INV	4	16/9	6	19.7
INV-NAND2-INV-NAND2-INV	5	16/9	7	20.4
NAND2-INV-NAND2-INV-INV-INV	6	16/9	8	21.6

Review of Definitions

Term	Stage	Path
number of stages	1	N
logical effort	g	$G = \prod g_i$
electrical effort	$h = \frac{C_{out}}{C_{in}}$	$H = \frac{C_{out-path}}{C_{in-path}}$
branching effort	$b = \frac{C_{on-path} + C_{off-path}}{C_{on-path}}$	$B = \prod b_i$
effort	$f = gh$	$F = GBH$
effort delay	f	$D_F = \sum f_i$
parasitic delay	p	$P = \sum p_i$
delay	$d = f + p$	$D = \sum d_i = D_F + P$

Method of Logical Effort

1) Compute path effort

$$F = GBH$$

2) Estimate best number of stages

$$N = \log_4 F$$

3) Sketch path with N stages

4) Estimate least delay

$$D = NF^{\frac{1}{N}} + P$$

5) Determine best stage effort

$$\hat{f} = F^{\frac{1}{N}}$$

6) Find gate sizes

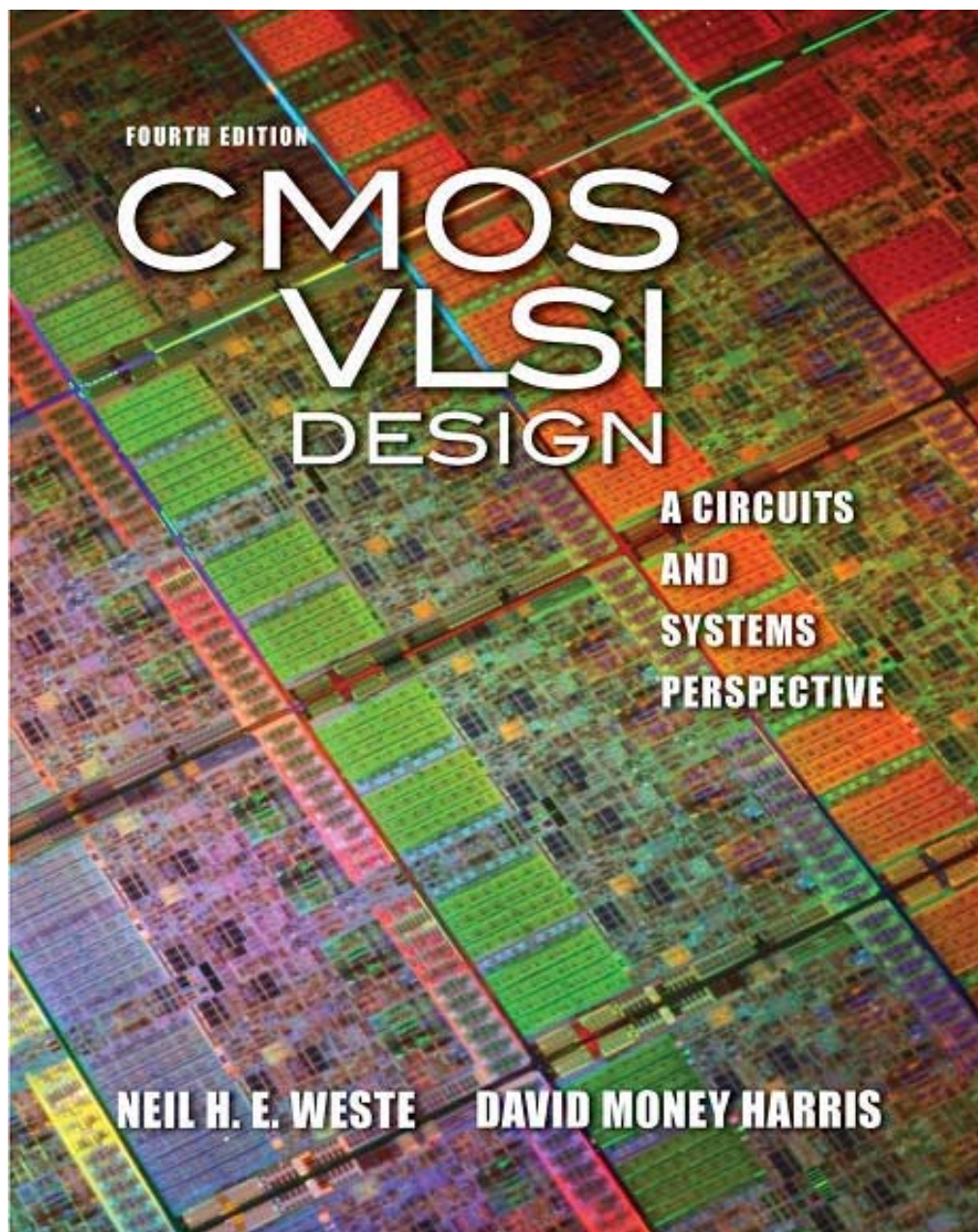
$$C_{in_i} = \frac{g_i C_{out_i}}{\hat{f}}$$

Limits of Logical Effort

- ❑ Chicken and egg problem
 - Need path to compute G
 - But don't know number of stages without G
- ❑ Simplistic delay model
 - Neglects input rise time effects
- ❑ Interconnect
 - Iteration required in designs with wire
- ❑ Maximum speed only
 - Not minimum area/power for constrained delay

Summary

- Logical effort is useful for thinking of delay in circuits
 - Numeric logical effort characterizes gates
 - NANDs are faster than NORs in CMOS
 - Paths are fastest when effort delays are ~ 4
 - Path delay is weakly sensitive to stages, sizes
 - But using fewer stages doesn't mean faster paths
 - Delay of path is about $\log_4 F$ FO4 inverter delays
 - Inverters and NAND2 best for driving large caps
- Provides language for discussing fast circuits
 - But requires practice to master



Lecture_7: Power

Outline

- Power and Energy
- Dynamic Power
- Static Power

Power and Energy

- ❑ Power is drawn from a voltage source attached to the V_{DD} pin(s) of a chip.
- ❑ Instantaneous Power: $P(t) =$
- ❑ Energy: $E =$
- ❑ Average Power: $P_{avg} =$

Power in Circuit Elements

$$P_{VDD}(t) = I_{DD}(t)V_{DD}$$



$$P_R(t) = \frac{V_R^2(t)}{R} = I_R^2(t)R$$



$$\begin{aligned} E_C &= \int_0^{\infty} I(t)V(t) dt = \int_0^{\infty} C \frac{dV}{dt} V(t) dt \\ &= C \int_0^{V_C} V(t) dV = \frac{1}{2} CV_C^2 \end{aligned}$$



Charging a Capacitor

- When the gate output rises
 - Energy stored in capacitor is

$$E_C = \frac{1}{2} C_L V_{DD}^2$$

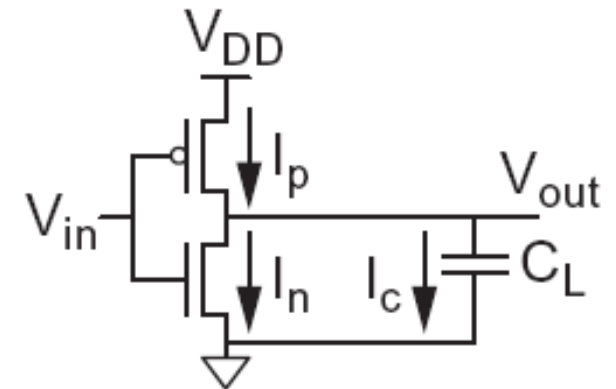
- But energy drawn from the supply is

$$E_{VDD} = \int_0^{\infty} I(t) V_{DD} dt = \int_0^{\infty} C_L \frac{dV}{dt} V_{DD} dt$$

$$= C_L V_{DD} \int_0^{V_{DD}} dV = C_L V_{DD}^2$$

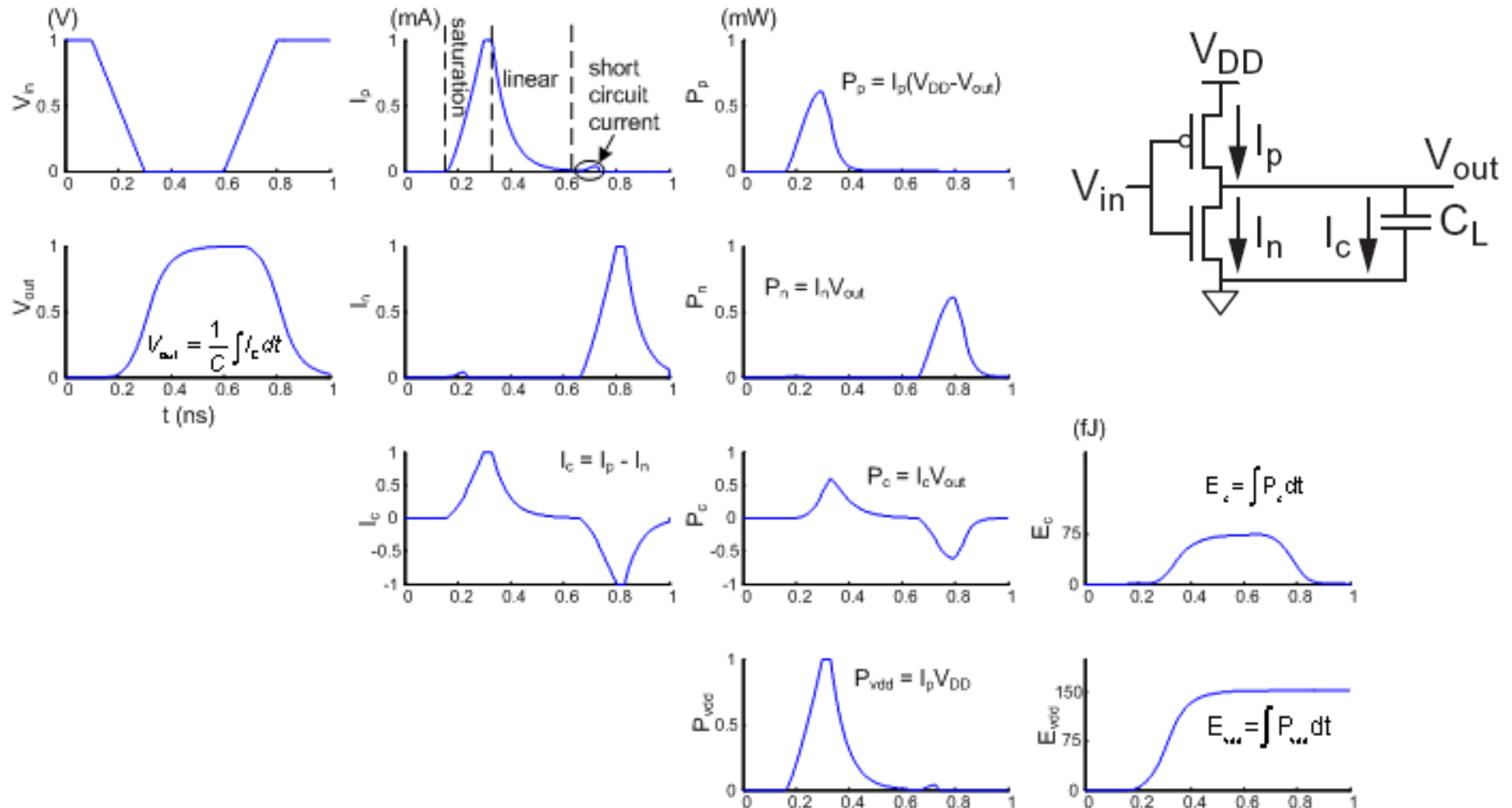
- Half the energy from V_{DD} is dissipated in the pMOS transistor as heat, other half stored in capacitor

- When the gate output falls
 - Energy in capacitor is dumped to GND
 - Dissipated as heat in the nMOS transistor



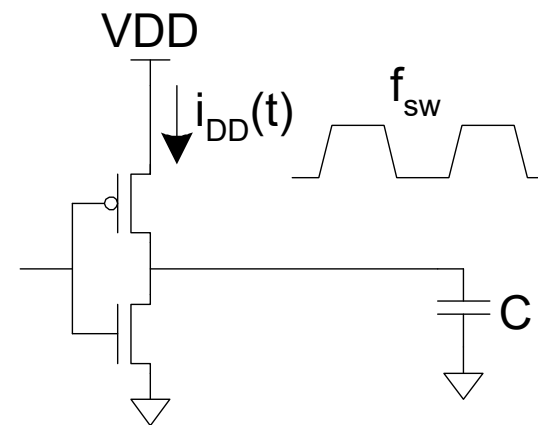
Switching Waveforms

□ Example: $V_{DD} = 1.0 \text{ V}$, $C_L = 150 \text{ fF}$, $f = 1 \text{ GHz}$



Switching Power

$$\begin{aligned} P_{\text{switching}} &= \frac{1}{T} \int_0^T i_{DD}(t) V_{DD} dt \\ &= \frac{V_{DD}}{T} \int_0^T i_{DD}(t) dt \\ &= \frac{V_{DD}}{T} [T f_{\text{sw}} C V_{DD}] \\ &= C V_{DD}^2 f_{\text{sw}} \end{aligned}$$



Activity Factor

- Suppose the system clock frequency = f
- Let $f_{sw} = \alpha f$, where α = activity factor
 - If the signal is a clock, $\alpha = 1$
 - If the signal switches once per cycle, $\alpha = 1/2$

- Dynamic power:

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

Short Circuit Current

- ❑ When transistors switch, both nMOS and pMOS networks may be momentarily ON at once
- ❑ Leads to a blip of “short circuit” current.
- ❑ $< 10\%$ of dynamic power if rise/fall times are comparable for input and output
- ❑ We will generally ignore this component

Power Dissipation Sources

- $P_{\text{total}} = P_{\text{dynamic}} + P_{\text{static}}$
- Dynamic power: $P_{\text{dynamic}} = P_{\text{switching}} + P_{\text{shortcircuit}}$
 - Switching load capacitances
 - Short-circuit current
- Static power: $P_{\text{static}} = (I_{\text{sub}} + I_{\text{gate}} + I_{\text{junct}} + I_{\text{contention}})V_{\text{DD}}$
 - Subthreshold leakage
 - Gate leakage
 - Junction leakage
 - Contention current

Dynamic Power Example

- 1 billion transistor chip
 - 50M logic transistors
 - Average width: 12λ
 - Activity factor = 0.1
 - 950M memory transistors
 - Average width: 4λ
 - Activity factor = 0.02
 - 1.0 V 65 nm process, $L_{eff} = 50nm$
 - $C = 1 \text{ fF}/\mu\text{m}$ (gate) + $0.8 \text{ fF}/\mu\text{m}$ (diffusion)
- Estimate dynamic power consumption @ 1 GHz.
Neglect wire capacitance and short-circuit current.

Solution

$$C_{logic} = (50 \times 10^6)(12 \times 0.025\mu m)(1.0 + 0.8) \left(\frac{pF}{\mu m} \right) = 27nF$$

$$C_{mem} = (950 \times 10^6)(4 \times 0.025\mu m)(1.0 + 0.8) \left(\frac{pF}{\mu m} \right) = 171nF$$

$$P_{dynamic} = [0.1C_{logic} + 0.02C_{mem}](1.0)^2(1.0Ghz) = 6.1W$$

$$f = 50nm \text{ and } \lambda = 25nm = 0.025\mu m$$

Dynamic Power Reduction

- $P_{\text{switching}} = \alpha C V_{DD}^2 f$
- Try to minimize:
 - Activity factor
 - Capacitance
 - Supply voltage
 - Frequency

Activity Factor Estimation

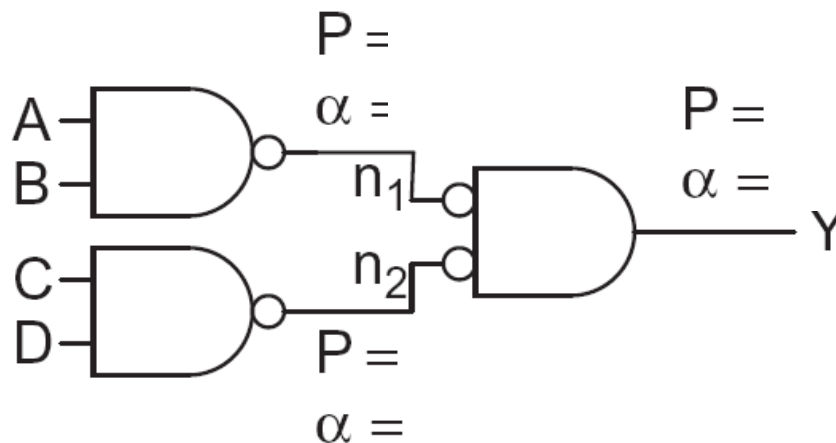
- Let $P_i = \text{Prob}(\text{node } i = 1)$
 - $\bar{P}_i = 1 - P_i$, $\text{Prob}(\text{node } i = 0)$
- $\alpha_i = \bar{P}_i \times P_i$
- Completely random data has $P = 0.5$ and $\alpha = 0.25$
- Data is often not completely random
 - Structured data, e.g. upper bits of 64-bit unsigned integer representing bank account balances are usually 0
- Data propagating through ANDs and ORs has lower activity factor
 - Depends on design, but typically $\alpha \approx 0.1$

Switching Probability

Gate	P_Y
AND2	$P_A P_B$
AND3	$P_A P_B P_C$
OR2	$1 - \bar{P}_A \bar{P}_B$
NAND2	$1 - P_A P_B$
NOR2	$\bar{P}_A \bar{P}_B$
XOR2	$P_A \bar{P}_B + \bar{P}_A P_B$

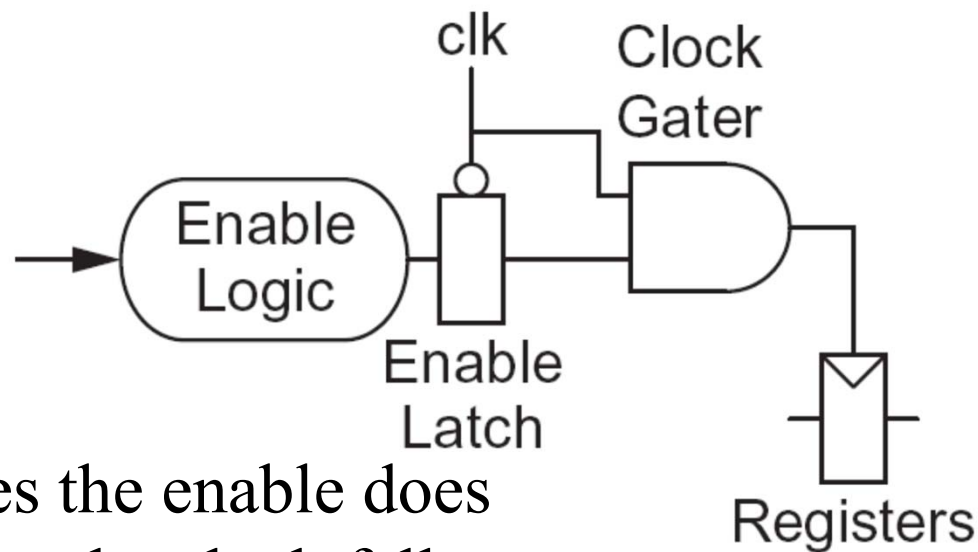
Example

- ❑ A 4-input AND is built out of two levels of gates
- ❑ Estimate the activity factor at each node if the inputs have $P = 0.5$

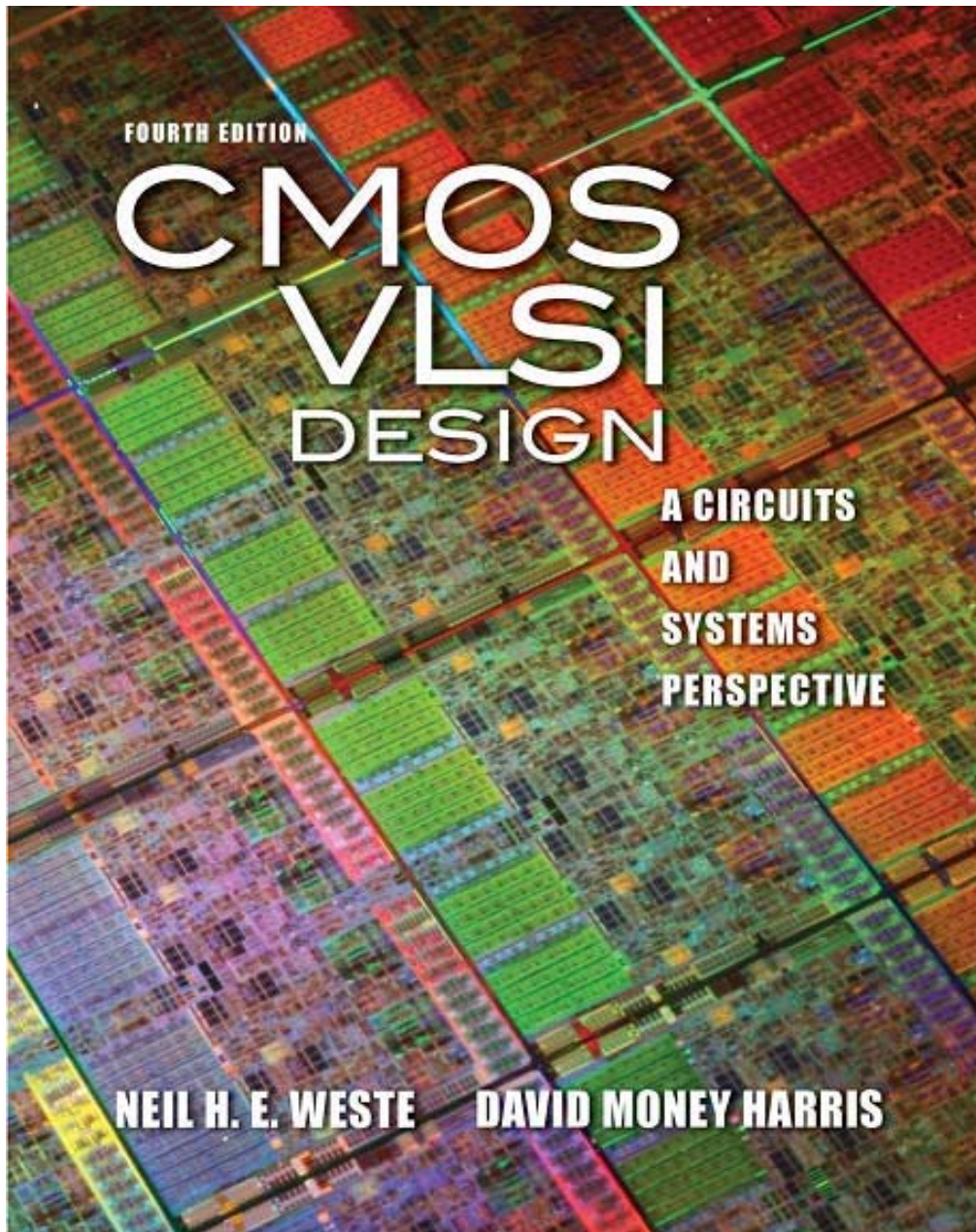


Clock Gating

- ❑ The best way to reduce the activity is to turn off the clock to registers in unused blocks
 - Saves clock activity ($\alpha = 1$)
 - Eliminates all switching activity in the block
 - Requires determining if block will be used



The latch ensures the enable does not change before the clock falls



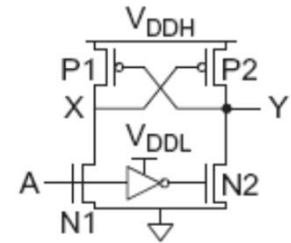
Lecture_8: Power

Capacitance

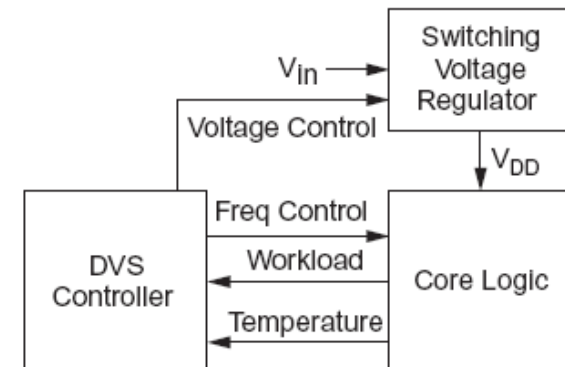
- ❑ Gate capacitance
 - Fewer stages of logic
 - Small gate sizes
- ❑ Wire capacitance
 - Good floorplanning to keep communicating blocks close to each other
 - Drive long wires with inverters or buffers rather than complex gates

Voltage / Frequency

- ❑ Run each block at the lowest possible voltage and frequency that meets performance requirements
- ❑ Voltage Domains
 - Provide separate supplies to different blocks
 - Level converters required when crossing from low to high V_{DD} domains



- ❑ Dynamic Voltage Scaling
 - Adjust V_{DD} and f according to workload



Static Power

- ❑ Static power is consumed even when chip is quiescent.
 - Leakage draws power from nominally OFF devices
 - Ratioed circuits burn power in fight between ON transistors

Static Power Example

- ❑ Revisit power estimation for 1 billion transistor chip
- ❑ Estimate static power consumption
 - Subthreshold leakage
 - Normal V_t : 100 nA/ μm
 - High V_t : 10 nA/ μm
 - High V_t used in all memories and in 95% of logic gates
 - Gate leakage 5 nA/ μm
 - Junction leakage negligible

Solution

$$W_{\text{normal-}V_t} = (50 \times 10^6)(12\lambda)(0.025 \mu\text{m} / \lambda)(0.05) = 0.75 \times 10^6 \mu\text{m}$$

$$W_{\text{high-}V_t} = \left[(50 \times 10^6)(12\lambda)(0.95) + (950 \times 10^6)(4\lambda) \right] (0.025 \mu\text{m} / \lambda) = 109.25 \times 10^6 \mu\text{m}$$

$$I_{\text{sub}} = \left[W_{\text{normal-}V_t} \times 100 \text{ nA}/\mu\text{m} + W_{\text{high-}V_t} \times 10 \text{ nA}/\mu\text{m} \right] / 2 = 584 \text{ mA}$$

$$I_{\text{gate}} = \left[(W_{\text{normal-}V_t} + W_{\text{high-}V_t}) \times 5 \text{ nA}/\mu\text{m} \right] / 2 = 275 \text{ mA}$$

$$P_{\text{static}} = (584 \text{ mA} + 275 \text{ mA})(1.0 \text{ V}) = 859 \text{ mW}$$

Subthreshold Leakage

- For $V_{ds} > 50$ mV

$$I_{sub} \approx I_{off} 10^{\frac{V_{gs} + \eta(V_{ds} - V_{DD}) - k_{\gamma} V_{sb}}{S}}$$

- I_{off} = leakage at $V_{gs} = 0$, $V_{ds} = V_{DD}$

Typical values in 65 nm

$$I_{off} = 100 \text{ nA}/\mu\text{m} \text{ @ } V_t = 0.3 \text{ V}$$

$$I_{off} = 10 \text{ nA}/\mu\text{m} \text{ @ } V_t = 0.4 \text{ V}$$

$$I_{off} = 1 \text{ nA}/\mu\text{m} \text{ @ } V_t = 0.5 \text{ V}$$

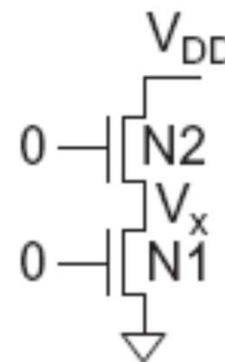
$$\eta = 0.1$$

$$k_{\gamma} = 0.1$$

$$S = 100 \text{ mV/decade}$$

Stack Effect

- Series OFF transistors have less leakage
 - $V_x > 0$, so N2 has negative V_{gs}



$$I_{sub} = \underbrace{I_{off} 10^{\frac{\eta(V_x - V_{DD})}{S}}}_{N1} = \underbrace{I_{off} 10^{\frac{-V_x + \eta((V_{DD} - V_x) - V_{DD}) - k_\gamma V_x}{S}}}_{N2}$$

$$V_x = \frac{\eta V_{DD}}{1 + 2\eta + k_\gamma}$$

$$I_{sub} = I_{off} 10^{\frac{-\eta V_{DD} \left(\frac{1 + \eta + k_\gamma}{1 + 2\eta + k_\gamma} \right)}{S}} \approx I_{off} 10^{\frac{-\eta V_{DD}}{S}}$$

- Leakage through 2-stack reduces ~10x
- Leakage through 3-stack reduces further

Leakage Control

- ❑ Leakage and delay trade off
 - Aim for low leakage in sleep and low delay in active mode
- ❑ To reduce leakage:
 - Increase V_t : *multiple V_t*
 - Use low V_t only in critical circuits
 - Increase V_s : *stack effect*
 - *Input vector control* in sleep
 - Decrease V_b
 - *Reverse body bias* in sleep
 - Or forward body bias in active mode

Gate Leakage

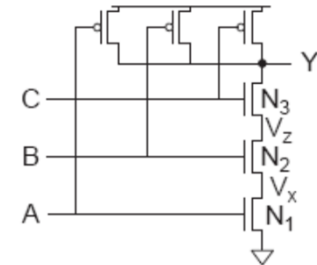
- ❑ Extremely strong function of t_{ox} and V_{gs}
 - Negligible for older processes
 - Approaches subthreshold leakage at 65 nm and below in some processes
- ❑ An order of magnitude less for pMOS than nMOS
- ❑ Control leakage in the process using $t_{\text{ox}} > 10.5 \text{ \AA}$
 - High-k gate dielectrics help
 - Some processes provide multiple t_{ox}
 - e.g. thicker oxide for 3.3 V I/O transistors
- ❑ Control leakage in circuits by limiting V_{DD}

NAND3 Leakage Example

□ 100 nm process

$$I_{gn} = 6.3 \text{ nA} \quad I_{gp} = 0$$

$$I_{offn} = 5.63 \text{ nA} \quad I_{offp} = 9.3 \text{ nA}$$



Input State (ABC)	I_{sub}	I_{gate}	I_{total}	V_x	V_z
000	0.4	0	0.4	stack effect	stack effect
001	0.7	0	0.7	stack effect	$V_{DD} - V_t$
010	0.7	1.3	2.0	intermediate	intermediate
011	3.8	0	3.8	$V_{DD} - V_t$	$V_{DD} - V_t$
100	0.7	6.3	7.0	0	stack effect
101	3.8	6.3	10.1	0	$V_{DD} - V_t$
110	5.6	12.6	18.2	0	0
111	28	18.9	46.9	0	0

Data from [Lee03]

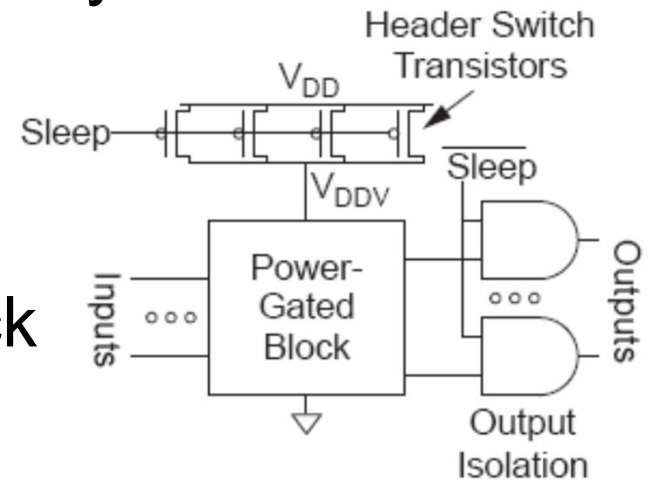
Junction Leakage

- ❑ From reverse-biased p-n junctions
 - Between diffusion and substrate or well
- ❑ Ordinary diode leakage is negligible
- ❑ Band-to-band tunneling (BTBT) can be significant
 - Especially in high- V_t transistors where other leakage is small
 - Worst at $V_{db} = V_{DD}$
- ❑ Gate-induced drain leakage (GIDL) exacerbates
 - Worst for $V_{gd} = -V_{DD}$ (or more negative)

Power Gating

- ❑ Turn OFF power to blocks when they are idle to save leakage

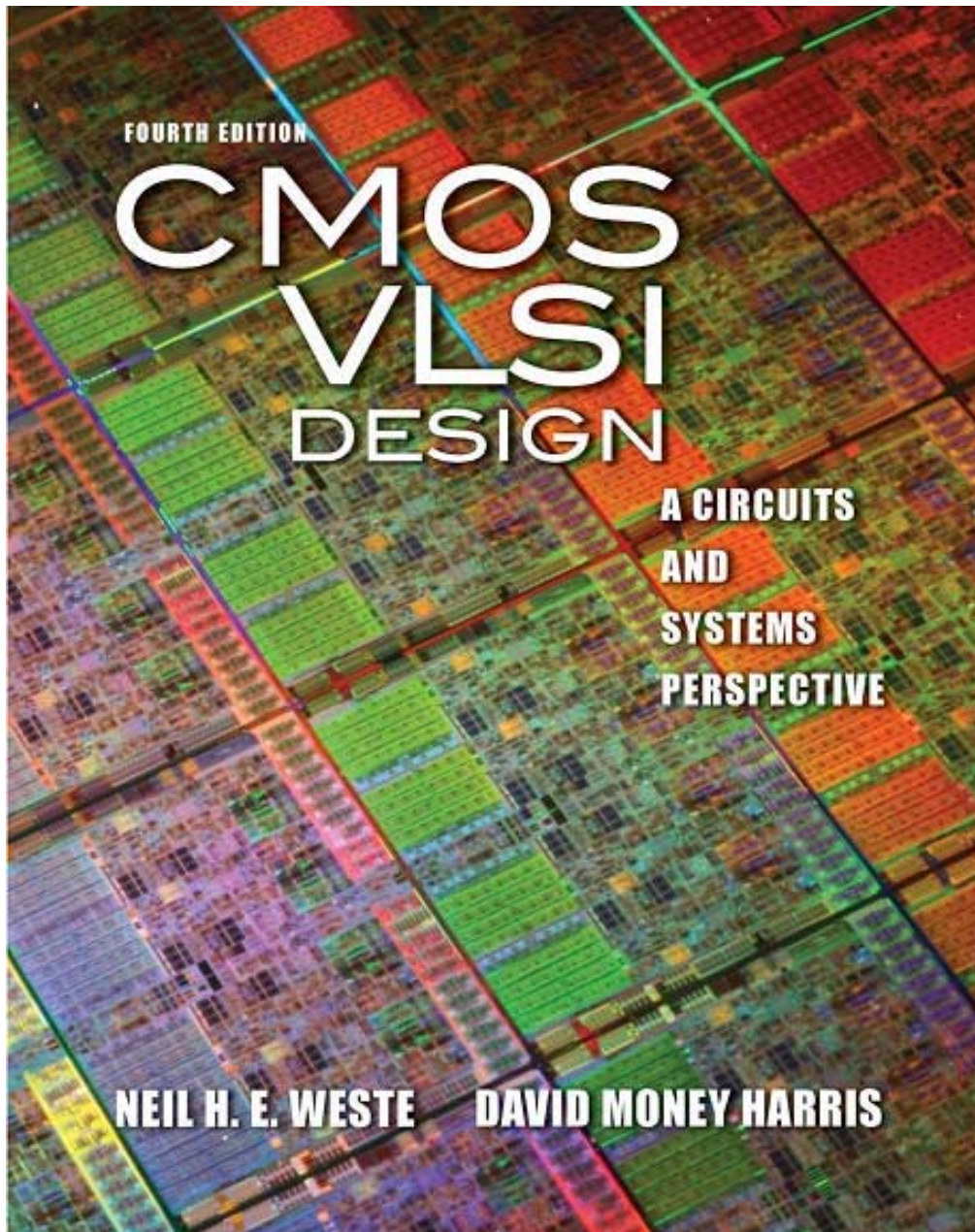
- Use virtual V_{DD} (V_{DDV})
- Gate outputs to prevent invalid logic levels to next block



- ❑ Voltage drop across sleep transistor degrades performance during normal operation

- Size the transistor wide enough to minimize impact

- ❑ Switching wide sleep transistor costs dynamic power
- Only justified when circuit sleeps long enough



Lecture_9: Combinational Circuit Design

Outline

- Bubble Pushing
- Compound Gates
- Logical Effort Example
- Input Ordering
- Asymmetric Gates
- Skewed Gates
- Best P/N ratio

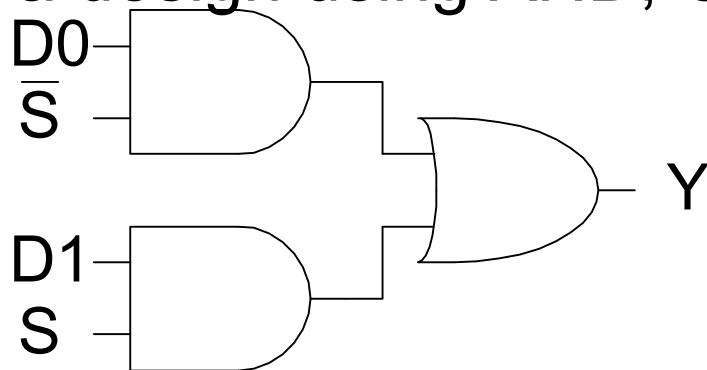
Example 1

```
module mux(input s, d0, d1,  
           output y);
```

```
    assign y = s ? d1 : d0; //Ternary Operator. If s is  
                           //true y = d1 else y = d0
```

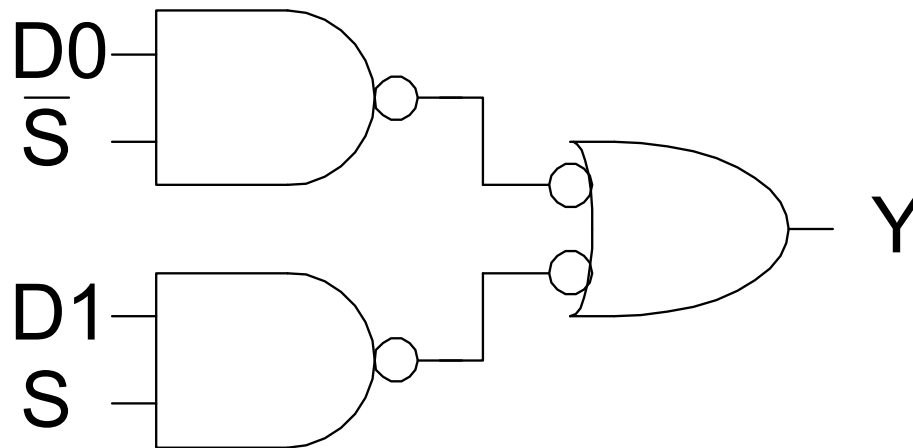
```
Endmodule
```

1) Sketch a design using AND, OR, and NOT gates.



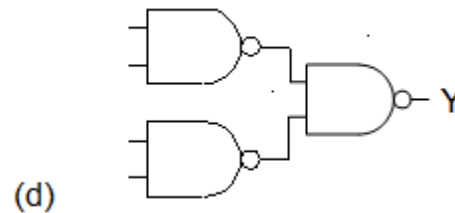
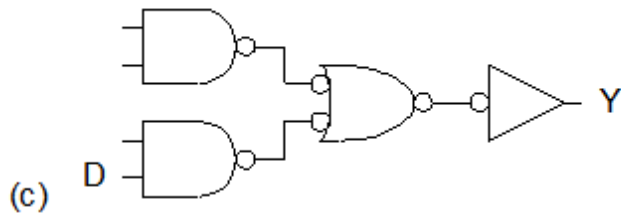
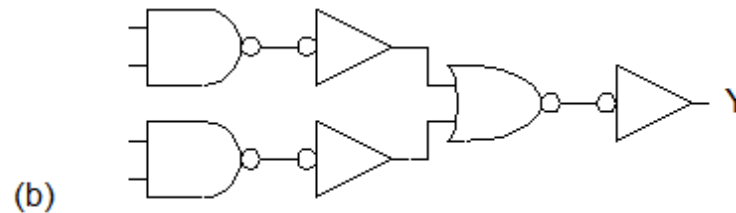
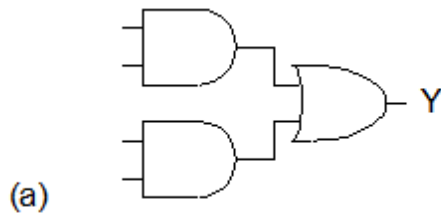
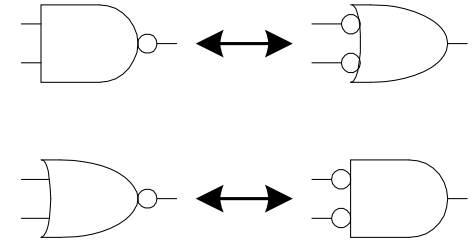
Example 2

2) Sketch a design using NAND, NOR, and NOT gates.
Assume $\sim S$ is available.



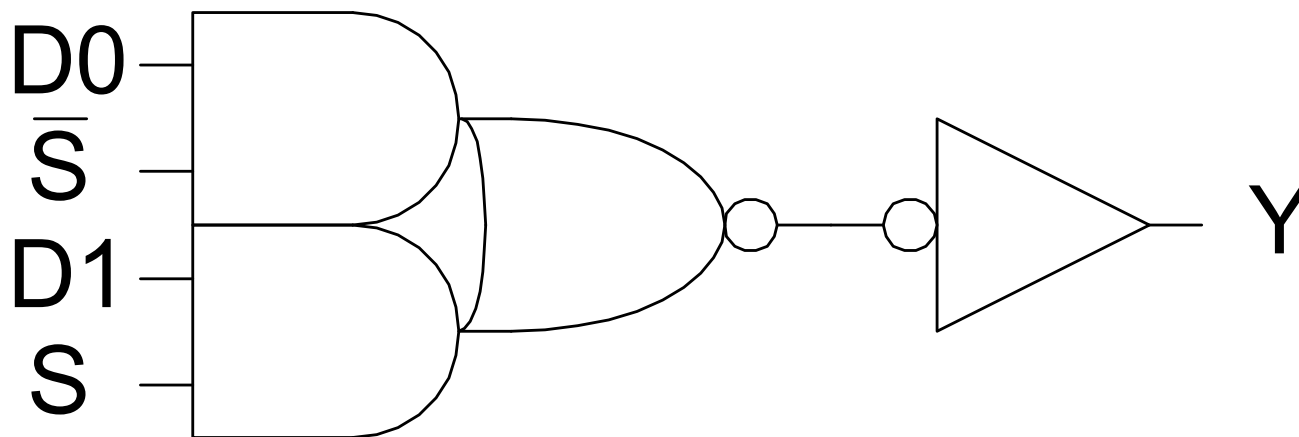
Bubble Pushing

- ❑ Start with network of AND / OR gates
- ❑ Convert to NAND / NOR + inverters
- ❑ Push bubbles around to simplify logic
 - Remember DeMorgan's Law



Example 3

3) Sketch a design using one compound gate and one NOT gate. Assume $\sim S$ is available.

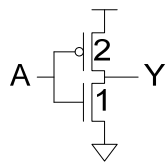
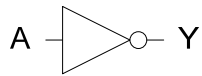


Compound Gates

□ Logical Effort of compound gates

unit inverter

$$Y = \overline{A}$$

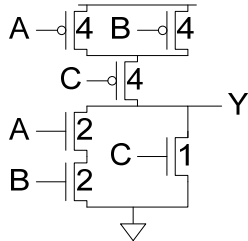
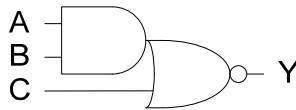


$$g_A = 3/3$$

$$p = 3/3$$

AOI21

$$Y = \overline{A \square B + C}$$



$$g_A = 6/3$$

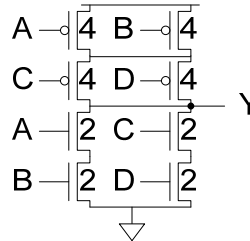
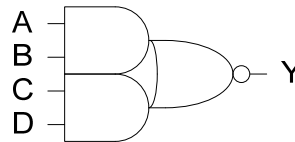
$$g_B = 6/3$$

$$g_C = 5/3$$

$$p = 7/3$$

AOI22

$$Y = \overline{A \square B + C \square D}$$



$$g_A =$$

$$g_B =$$

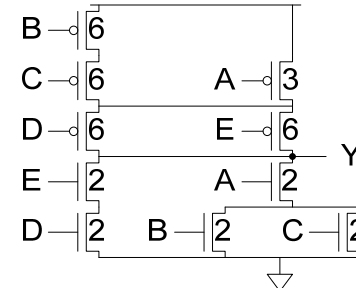
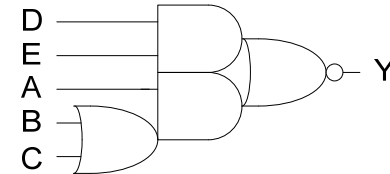
$$g_C =$$

$$g_D =$$

$$p =$$

Complex AOI

$$Y = \overline{A \square (B + C) + D \square E}$$



$$g_A =$$

$$g_B =$$

$$g_C =$$

$$g_D =$$

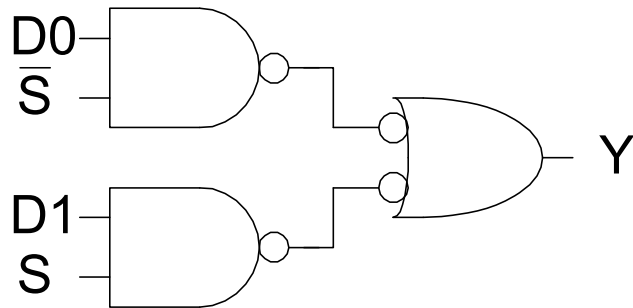
$$g_E =$$

$$p =$$

Example 4

- The multiplexer has a maximum input capacitance of 16 units on each input. It must drive a load of 160 units. Estimate the delay of the two designs.

H =



$P =$

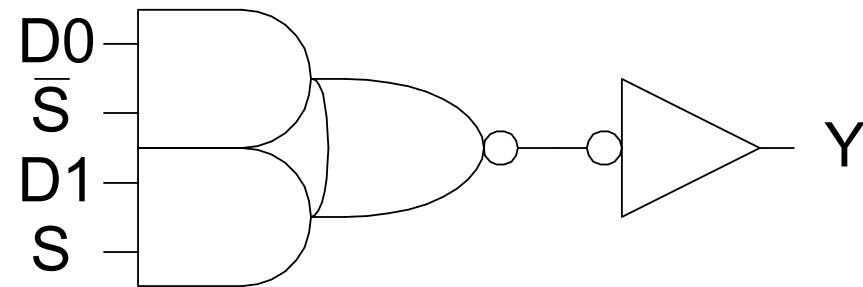
$G =$

$F =$

$\hat{f} =$

$D =$

B =



$P =$

$G =$

$F =$

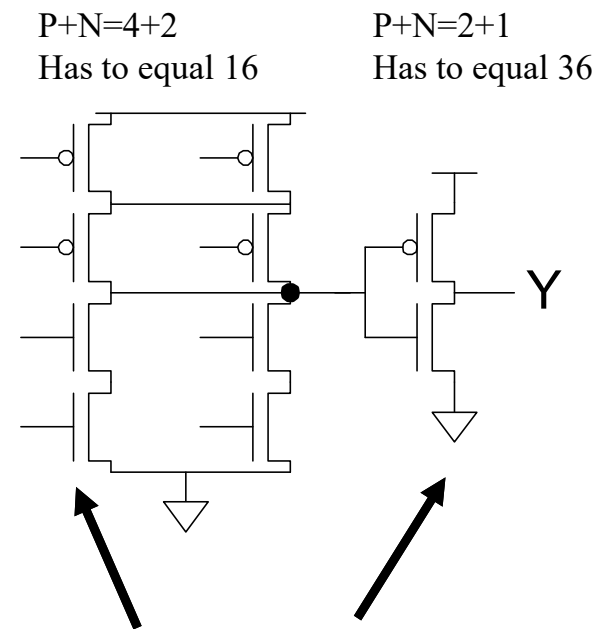
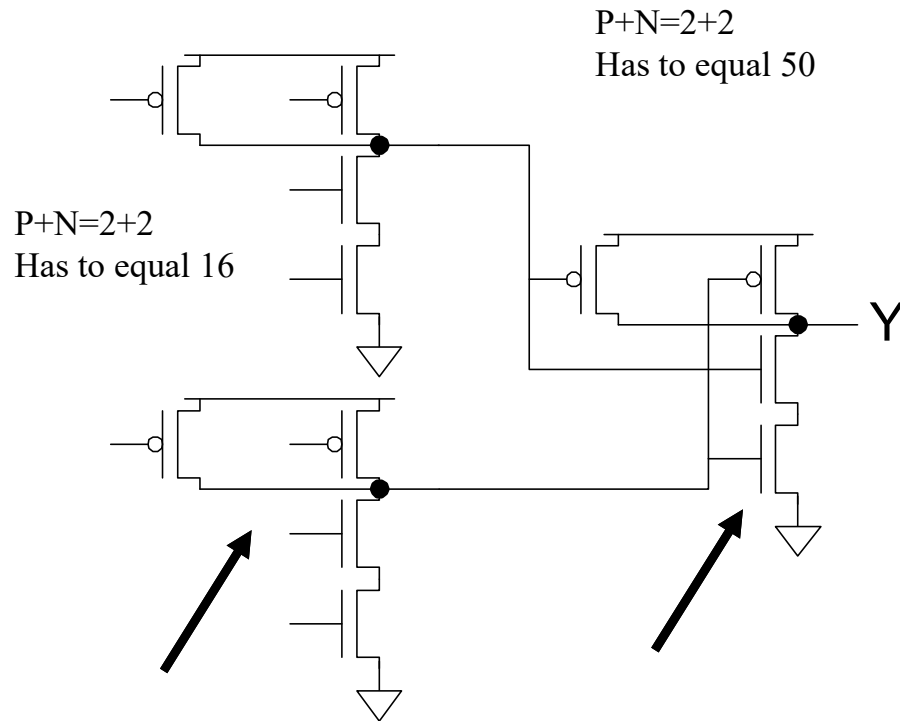
$\hat{f} =$

$D =$

N =

Example 5

- Annotate your designs with transistor sizes that achieve this delay.



Input Order

□ Our parasitic delay model was too simple

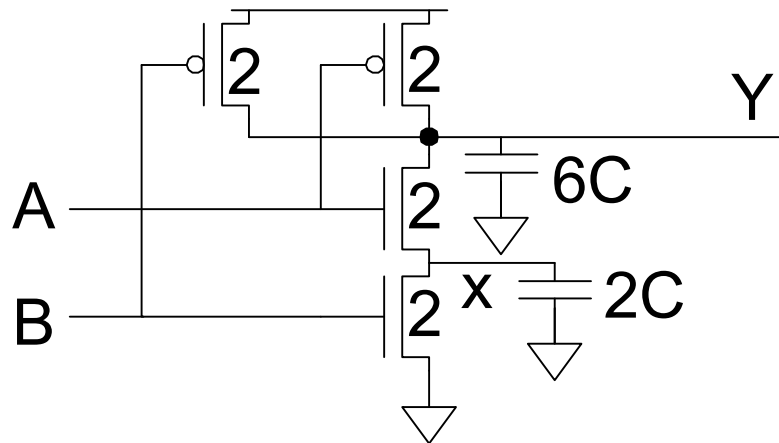
– Calculate parasitic delay for Y falling

• If A arrives latest?

$$t_{pd} = 6C * \left(\frac{R}{2} + \frac{R}{2}\right) / 3RC$$

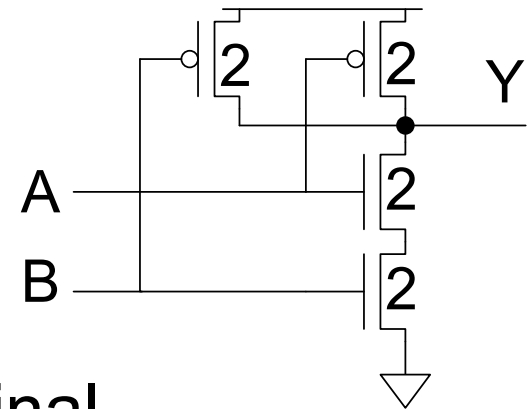
• If B arrives latest?

$$t_{pd} = (2C * R/2 + 6C * \left(\frac{R}{2} + \frac{R}{2}\right)) / 3RC$$



Inner & Outer Inputs

- ❑ *Inner* input is closest to output (A)
- ❑ *Outer* input is closest to rail (B)
- ❑ If input arrival time is known
 - Connect latest input to inner terminal

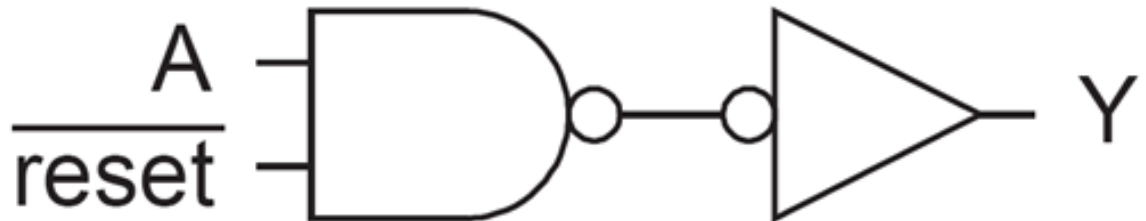


Asymmetric Gates

Buffer

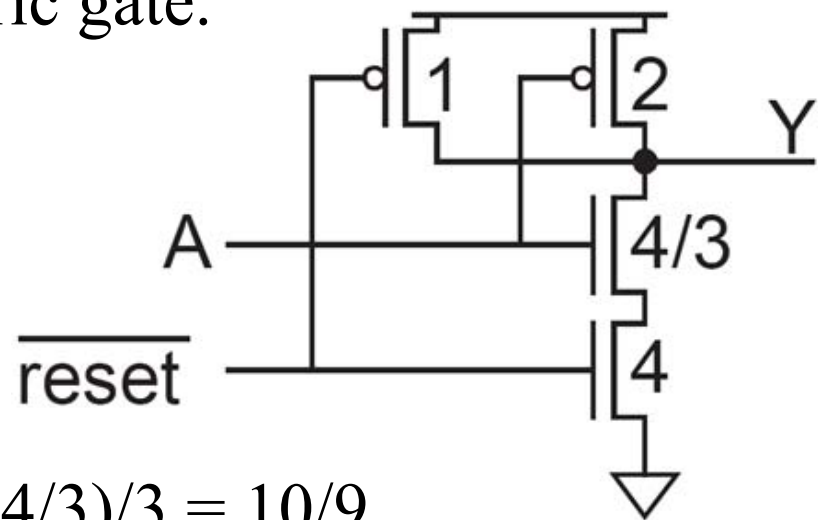
Reset asserted $y=0$

Required to reset less frequently



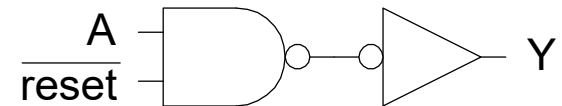
A is most critical, go for Asymmetric gate.

- Make it inner
- Less gate capacitance
- Reset to a wider nMOS, Less R
- Reset narrower pMOS, Less C
- Series nMOS $R = \text{unity}$
- $R/4 + R/(4/3) = R$ and $g_A = (2+4/3)/3 = 10/9$
- As the reset nMOS W gets larger, g_A becomes closer to unity



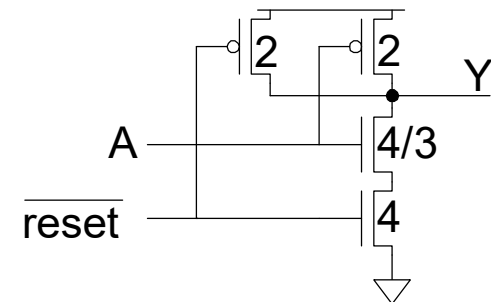
Asymmetric Gates

- ❑ Asymmetric gates favor one input over another
- ❑ Ex: suppose input A of a NAND gate is most critical
 - Use smaller transistor on A (less capacitance)
 - Boost size of noncritical input
 - So total resistance is same



- ❑ $g_A =$
- ❑ $g_B =$
- ❑ $g_{\text{total}} = g_A + g_B =$

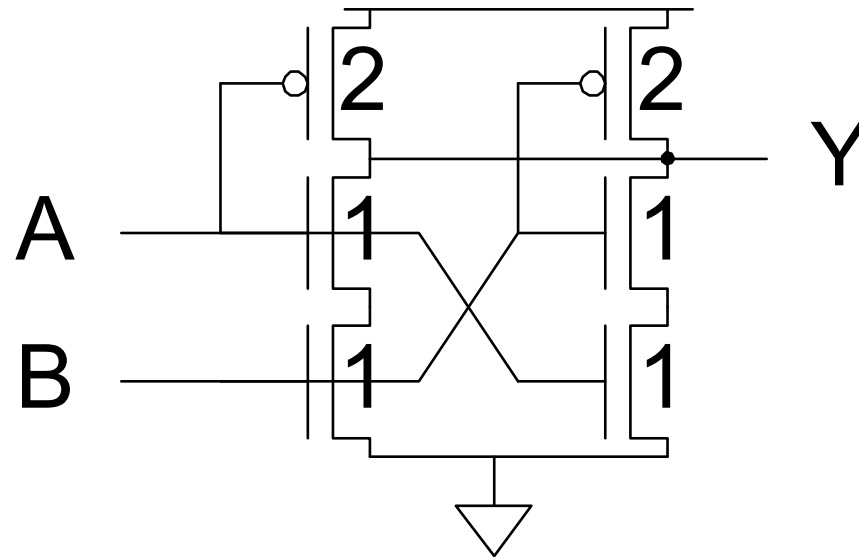
$$R_{PD} = \frac{1}{4} + \frac{3}{4} = 1$$



- ❑ Asymmetric gate approaches $g = 1$ on critical input
- ❑ But total logical effort goes up

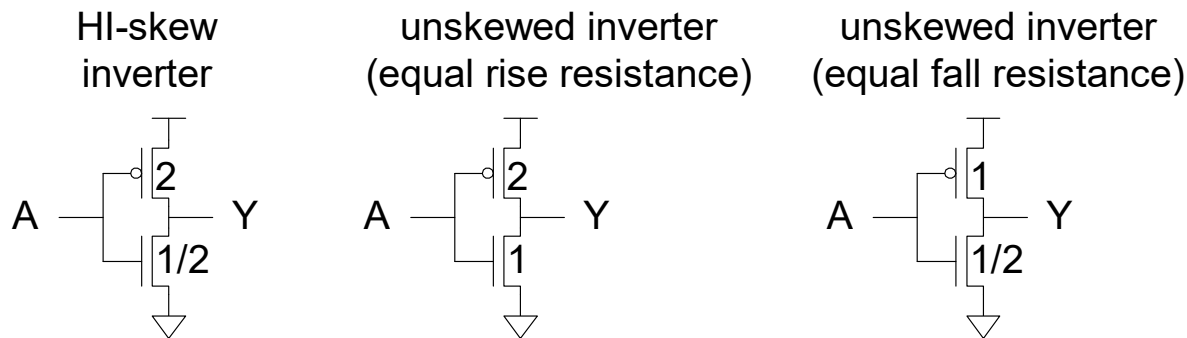
Symmetric Gates

- Inputs can be made perfectly symmetric



Skewed Gates

- ❑ Skewed gates favor one edge over another
- ❑ Ex: suppose rising output of inverter is most critical
 - Downsize noncritical nMOS transistor



- ❑ Calculate logical effort by comparing to unskewed inverter with same effective resistance on that edge.
 - $g_u =$
 - $g_d =$

HI- and LO-Skew

when $\frac{\beta_p}{\beta_n} > 1$ HI skew

Favors rising transition

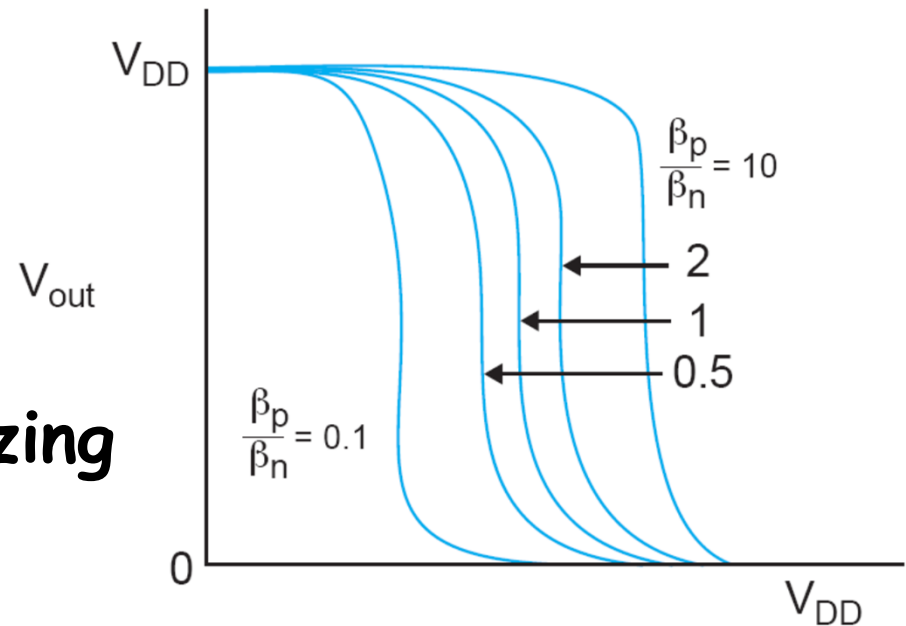
Done by downsizing nMOS

Skewing is done by downsizing MOSs by a factor of 2

when $\frac{\beta_p}{\beta_n} < 1$ LO skew

Favors falling transition

Done by downsizing pMOS



$$\beta_n = \left(\frac{W}{L}\right)_n$$
$$\beta_p = \left(\frac{W}{L}\right)_p$$

HI- and LO-Skew

- ❑ Def: Logical effort of a skewed gate for a particular transition is the ratio of the input capacitance of that gate to the input capacitance of an unskewed inverter delivering the same output current for the same transition.

- ❑ Skewed gates reduce size of noncritical transistors
 - HI-skew gates favor rising output (small nMOS)
 - LO-skew gates favor falling output (small pMOS)
- ❑ Logical effort is smaller for favored direction
- ❑ But larger for the other direction

HI- and LO-Skew

In calculating g_u of a complex gate:

Draw the unskewed inverter (2:1) whose pull-up resistance is equal to the equivalent resistance of the pull-up network of the skewed gate.

$$\text{Then } g_u = \frac{\text{input capacitance of the skewed gate}}{\text{input capacitance of the unskewed inverter}}$$

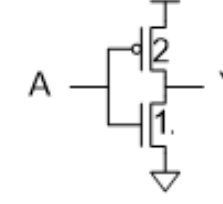
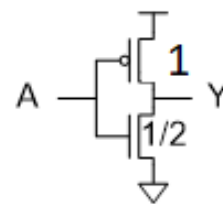
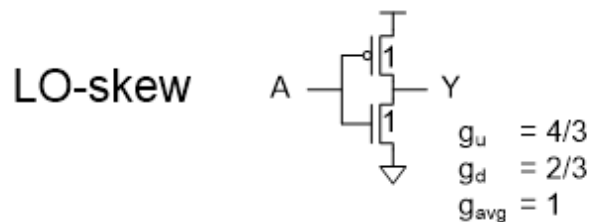
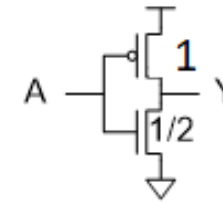
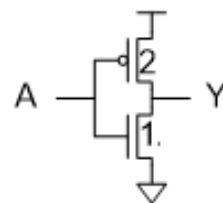
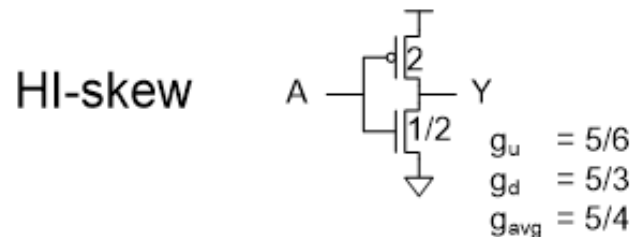
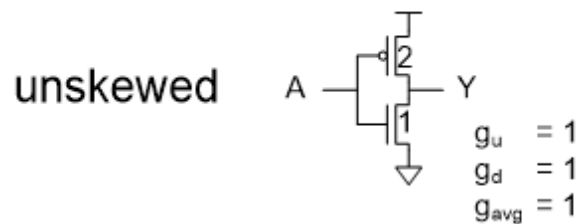
In calculating g_d of a complex gate:

Draw the unskewed inverter (2:1) whose pull-down resistance is equal to the equivalent resistance of the pull-down network of the skewed gate.

$$\text{Then } g_d = \frac{\text{input capacitance of the skewed gate}}{\text{input capacitance of the unskewed inverter}}$$

Calculations of g'_u s and g'_d s

Inverters



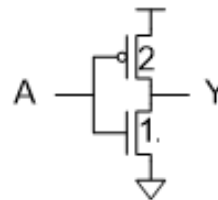
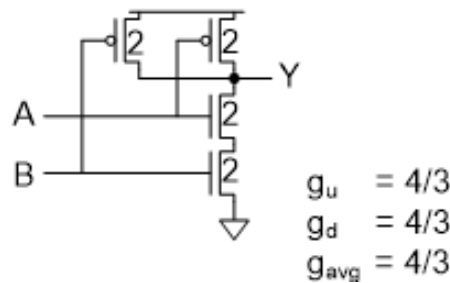
Equal rise time

Equal fall time

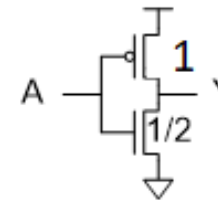
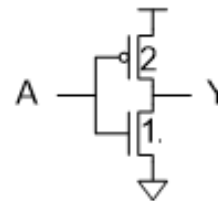
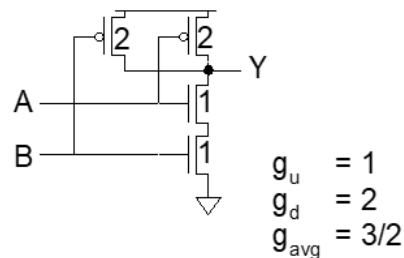
Calculations of g'_u s and g'_d s

NAND gates

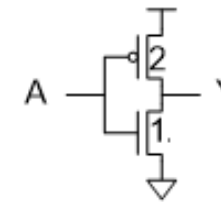
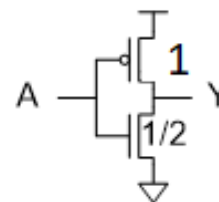
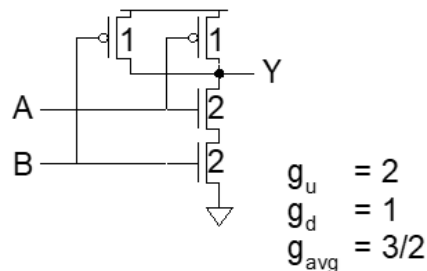
Unskewed



HI-skewed



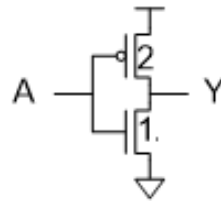
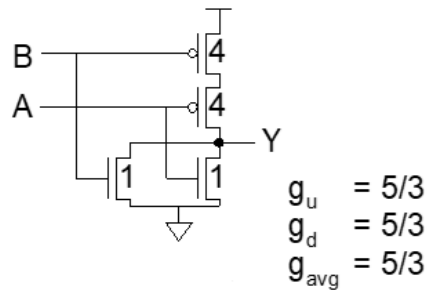
LO-skewed



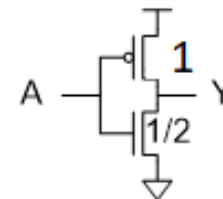
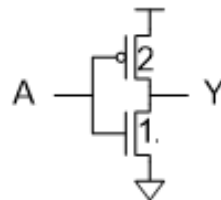
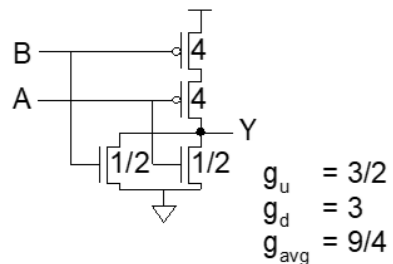
Calculations of g'_u s and g'_d s

NOR gates

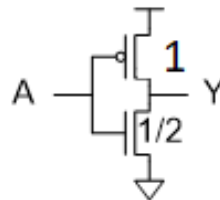
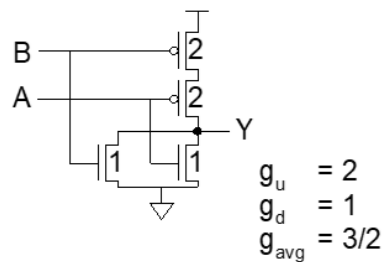
Unskewed



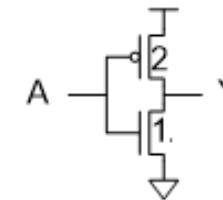
HI-skewed



LO-skewed



Equal rise time



Equal fall time

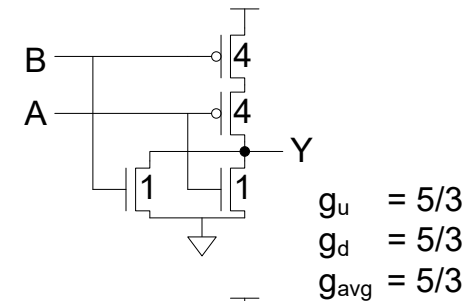
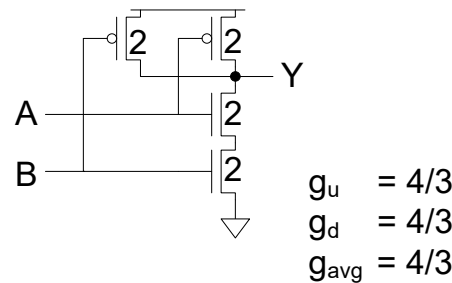
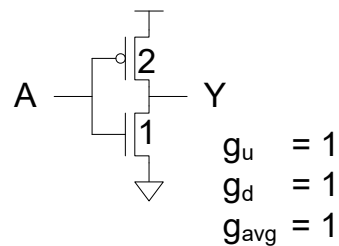
Catalog of Skewed Gates

Inverter

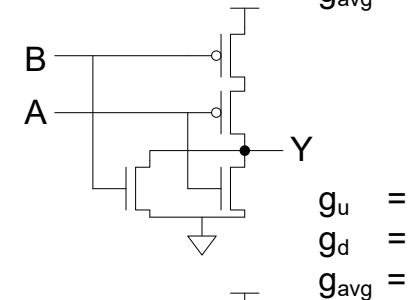
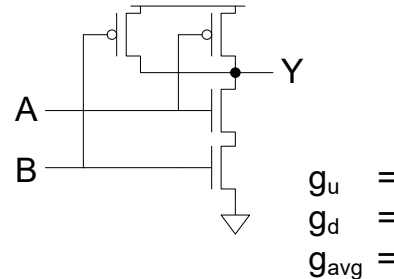
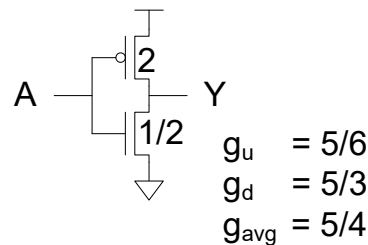
NAND2

NOR2

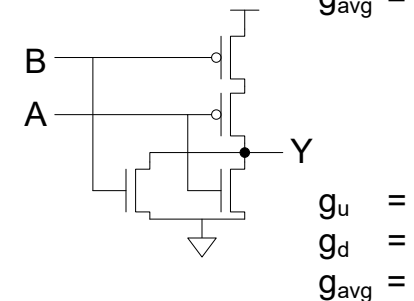
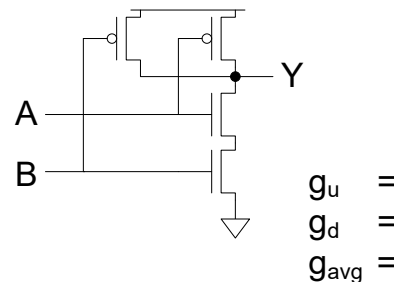
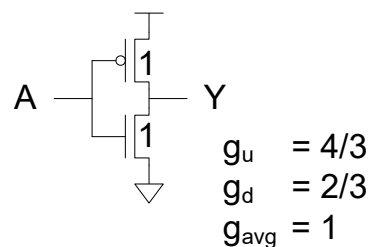
unskewed



HI-skew

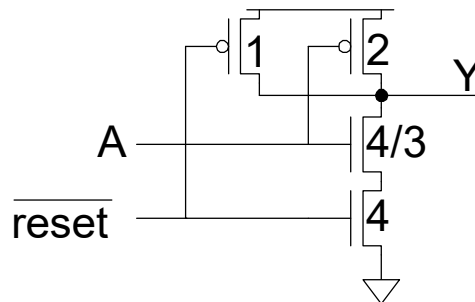
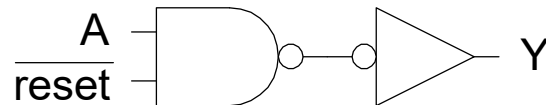


LO-skew



Asymmetric Skew

- Combine asymmetric and skewed gates
 - Downsize noncritical transistor on unimportant input
 - Reduces parasitic delay for critical input

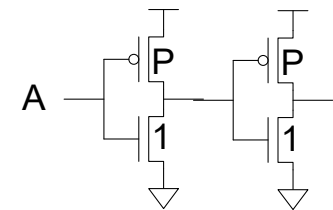


Best P/N Ratio

- We have selected P/N ratio for unit rise and fall resistance ($\mu = 2-3$ for an inverter). $\mu = \frac{\mu_n}{\mu_p} = 2$
- Alternative: choose ratio for least average delay

□ Ex: inverter

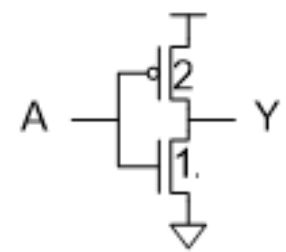
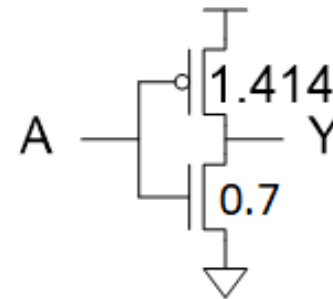
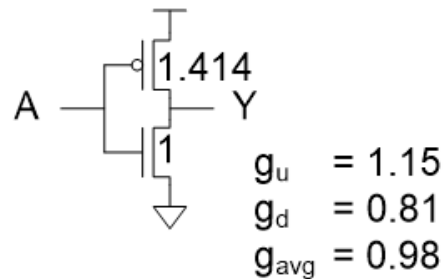
- Delay driving identical inverter



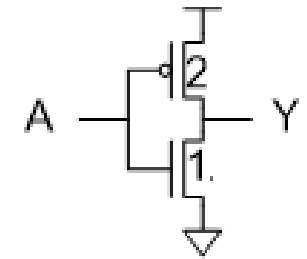
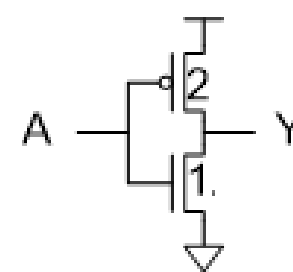
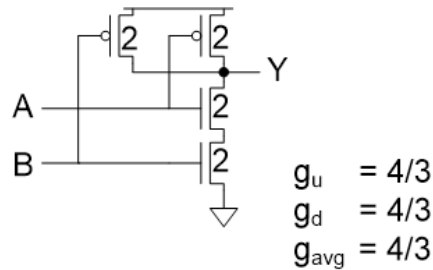
- $t_{pdf} = 2C(P+1). R$
- $t_{pdr} = 2C(P+1). R(\mu / P)$
- $t_{pd} = 1/2(t_{pdf} + t_{pdr}) = 1/2[2CR(P+1)(1 + \mu/P)] = (P + 1 + \mu + \mu/P)CR$
- $dt_{pd} / dP = (1 - \mu/P^2) = 0$
- Least delay for $P = \sqrt{\mu}$

Best P/N Ratio

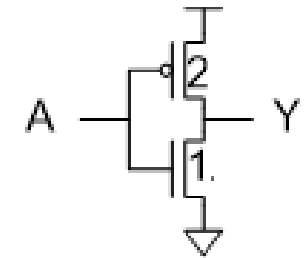
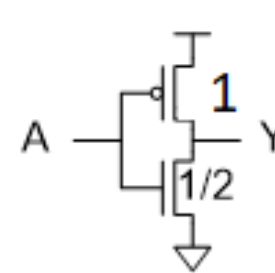
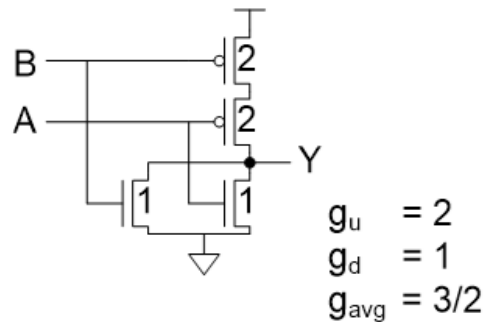
Inverters



NAND gate



NOR gate



Equal rise time

Equal fall time

P/N Ratios

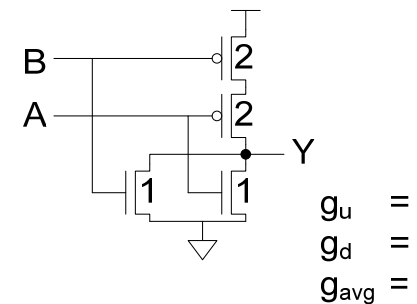
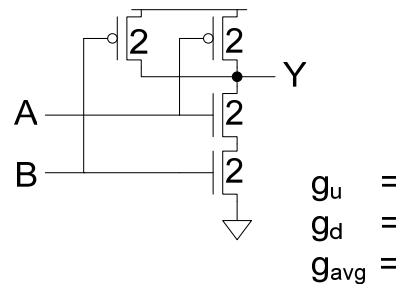
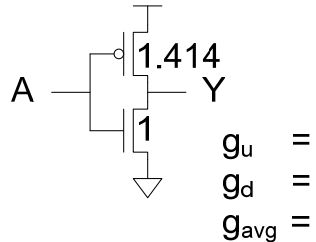
- In general, best P/N ratio is sqrt of equal delay ratio.
 - Only improves average delay slightly for inverters
 - But significantly decreases area and power

Inverter

NAND2

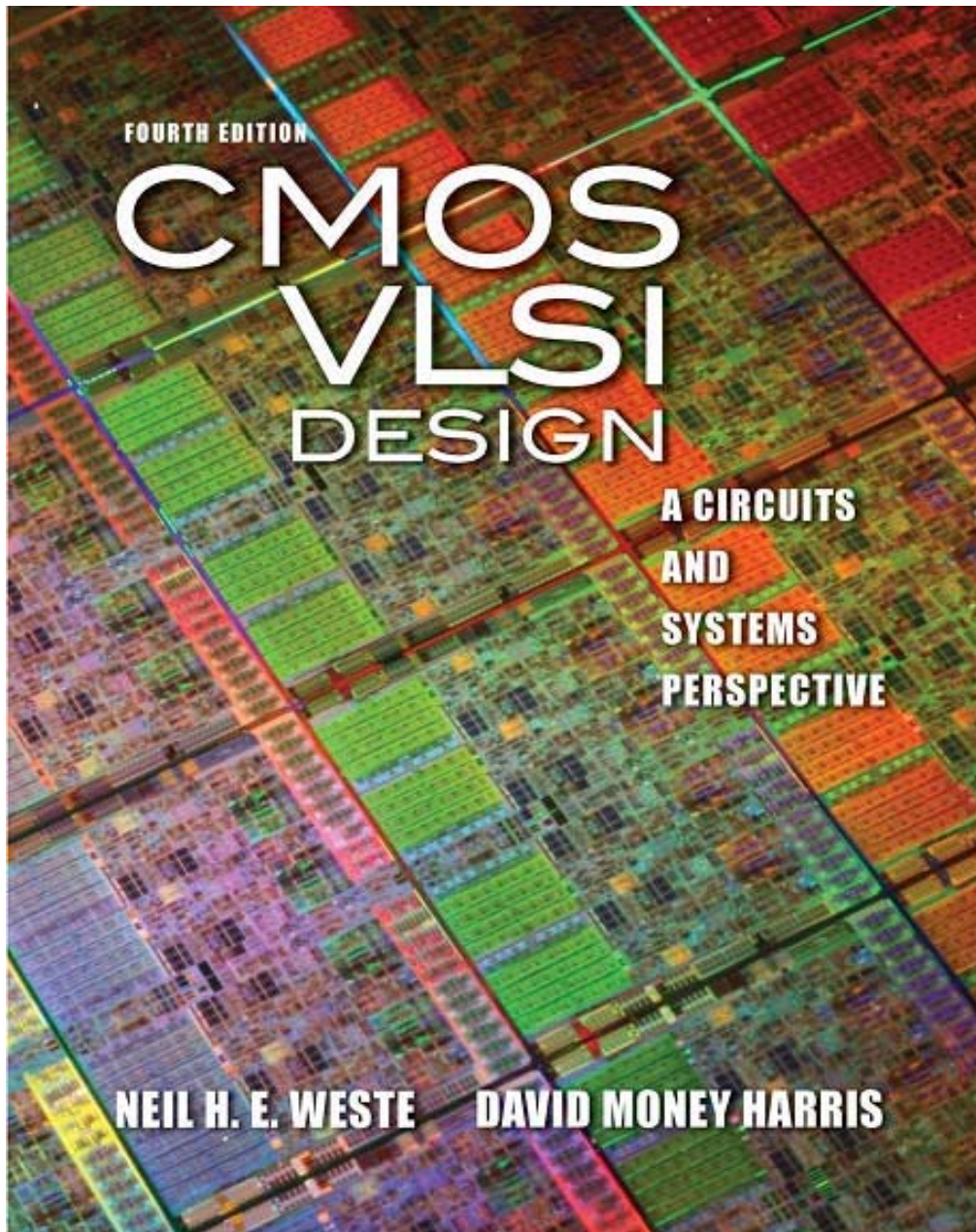
NOR2

fastest
P/N ratio



Observations

- For speed:
 - NAND vs. NOR
 - Many simple stages vs. fewer high fan-in stages
 - Latest-arriving input
- For area and power:
 - Many simple stages vs. fewer high fan-in stages



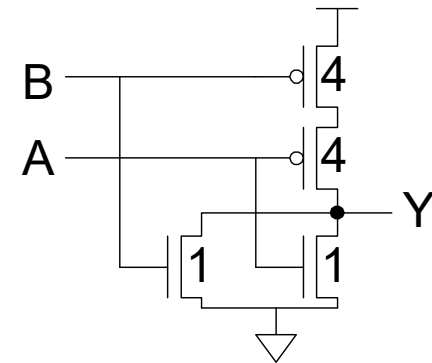
Lecture_10: Circuit Families

Outline

- Pseudo-nMOS Logic
- Dynamic Logic
- Pass Transistor Logic

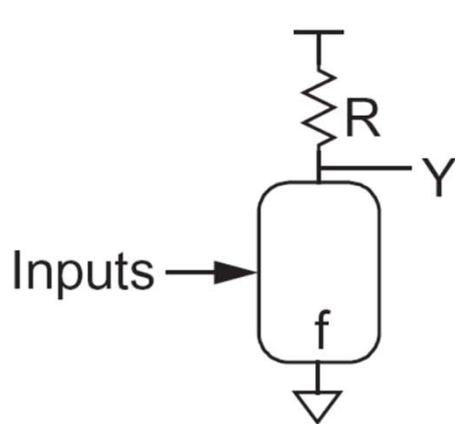
Introduction

- ❑ What makes a circuit fast?
 - $I = C \, dV/dt \quad \rightarrow \quad t_{pd} \propto (C/I) \, \Delta V$
 - low capacitance
 - high current
 - small swing
- ❑ Logical effort is proportional to C/I
- ❑ pMOS are the enemy!
 - High capacitance for a given current
- ❑ Can we take the pMOS capacitance off the input?
- ❑ Various circuit families try to do this...

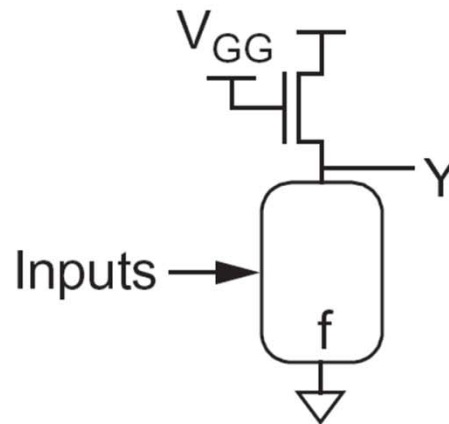


Ratioed circuits: nMOS Technology

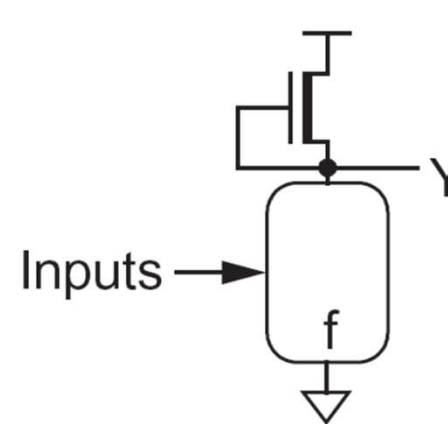
- ❑ nMOS only Technology.
- ❑ Popular 1970 – to -1980 before CMOS.
- ❑ Pulldown network off, static load (R or T) pulls output high.
- ❑ Pulldown network on, PDN fights the always on static load.
- ❑ Enhancement nMOS requires additional Supply V_{GG} for strong V_{OH} , use instead depletion mode MOS



(a)



(b)



(c)

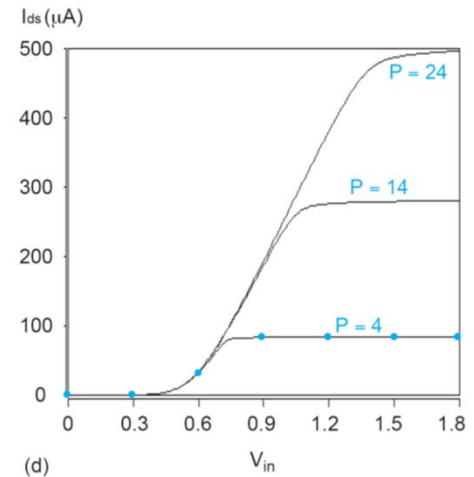
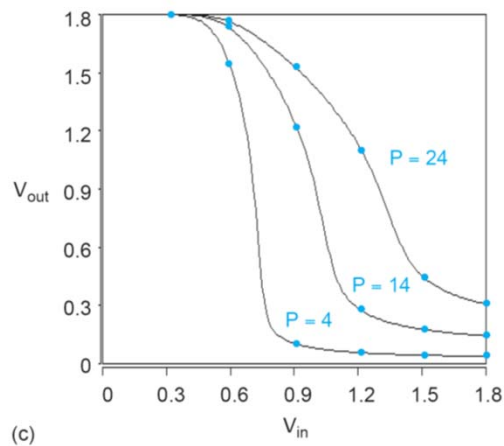
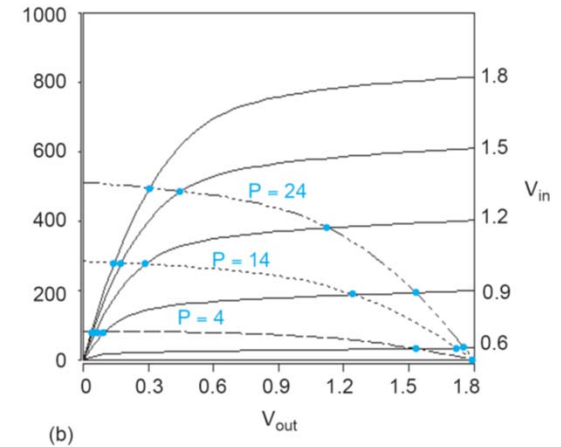
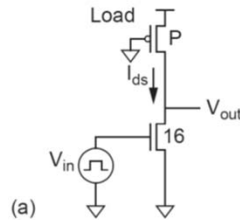
Pseudo-nMOS

□ In CMOS, use a pMOS that is always ON

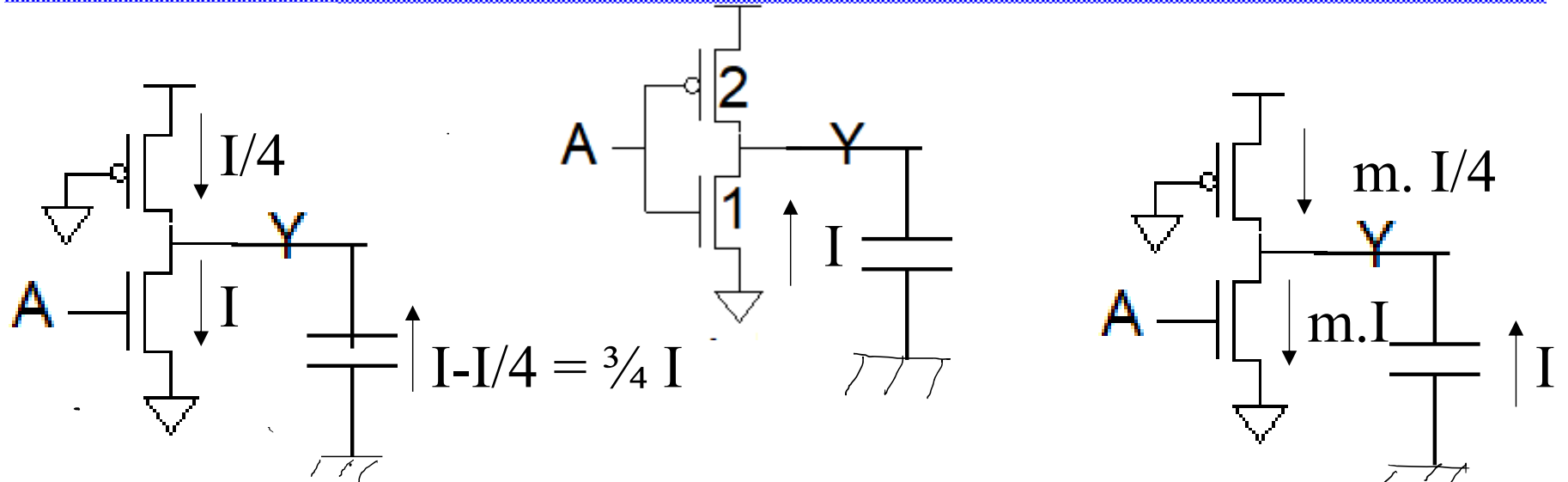
□ **Ratio issue**

Make pMOS about $\frac{1}{4}$ effective strength of pulldown network.

$$P = (2 \times 16) / 4 = 8$$



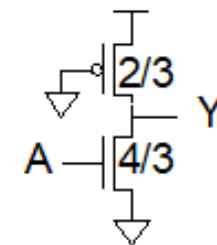
Pseudo-nMOS



Need the discharging current of the capacitor to I as a unit-sized inverter I . Required transistor size m to do so, keeping the pMOS transistor of $1/4$ the strength of the nMOS.

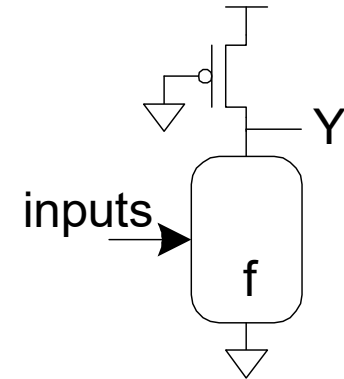
$$m \cdot I - m \cdot I/4 = I \text{ which gives } m = 4/3$$

$$\text{Which gives } \mu(4/3) * \frac{1}{4} = \frac{2}{3}$$

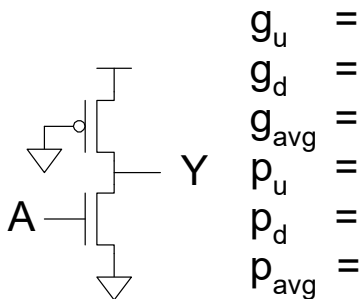


Pseudo-nMOS Gates

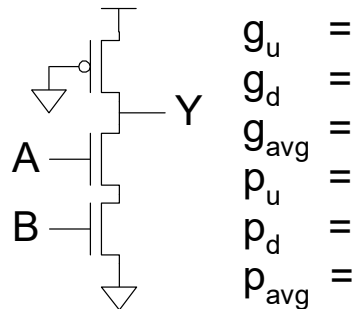
- Design for unit current on output to compare with unit inverter.
- pMOS fights nMOS



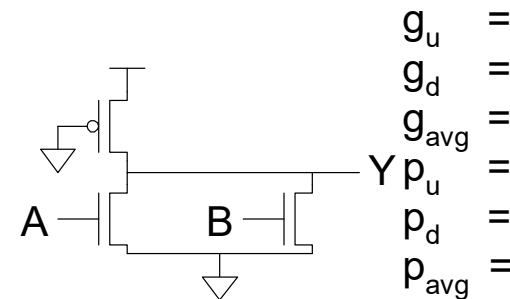
Inverter



NAND2

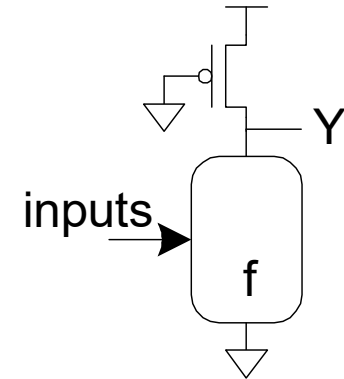


NOR2

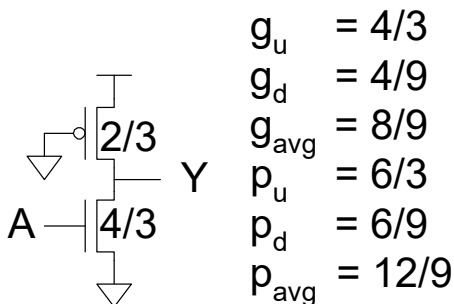


Pseudo-nMOS Gates

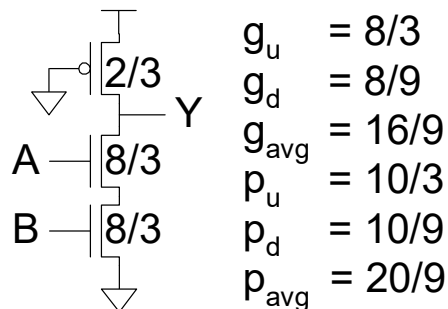
- Design for unit current on output to compare with unit inverter.
- pMOS fights nMOS



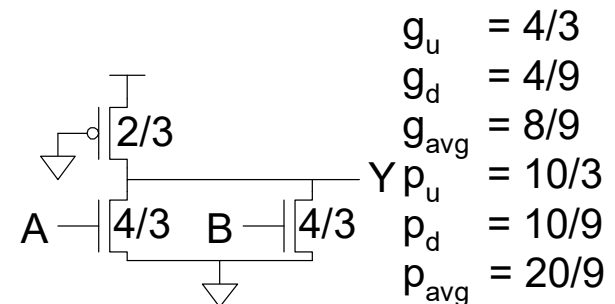
Inverter



NAND2

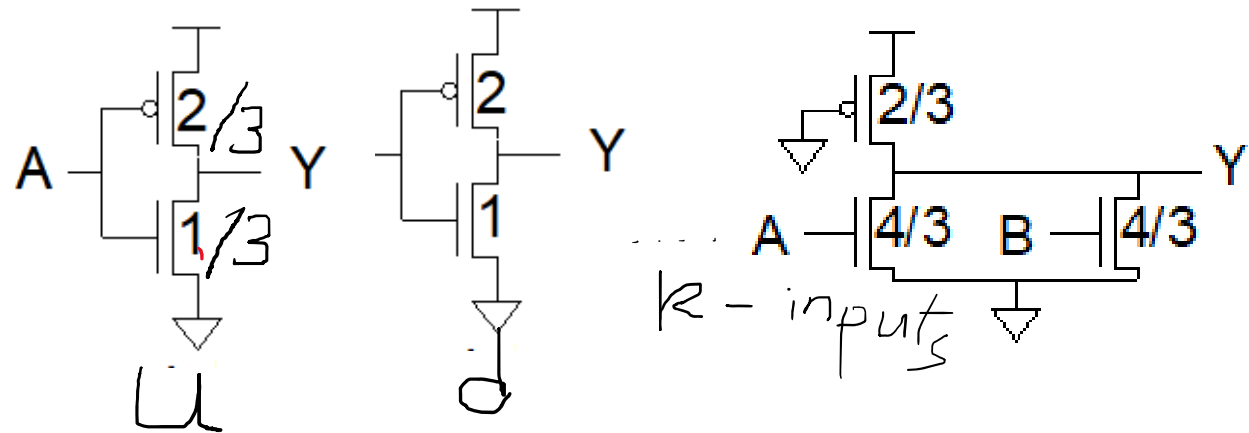


NOR2



Pseudo-nMOS Gates

Calculate g_{ave} and P_{ave} for k-input pseudo-nMOS NOR gate



$$g_u = (4/3)/1 = 4/3$$

$$g_d = (4/3)/3 = 4/9$$

$$g_{ave} = \frac{1}{2}(4/3 + 4/9) = 8/9 \text{ independent of } k$$

$$P_u = (2/3 + k \times 4/3)/1$$

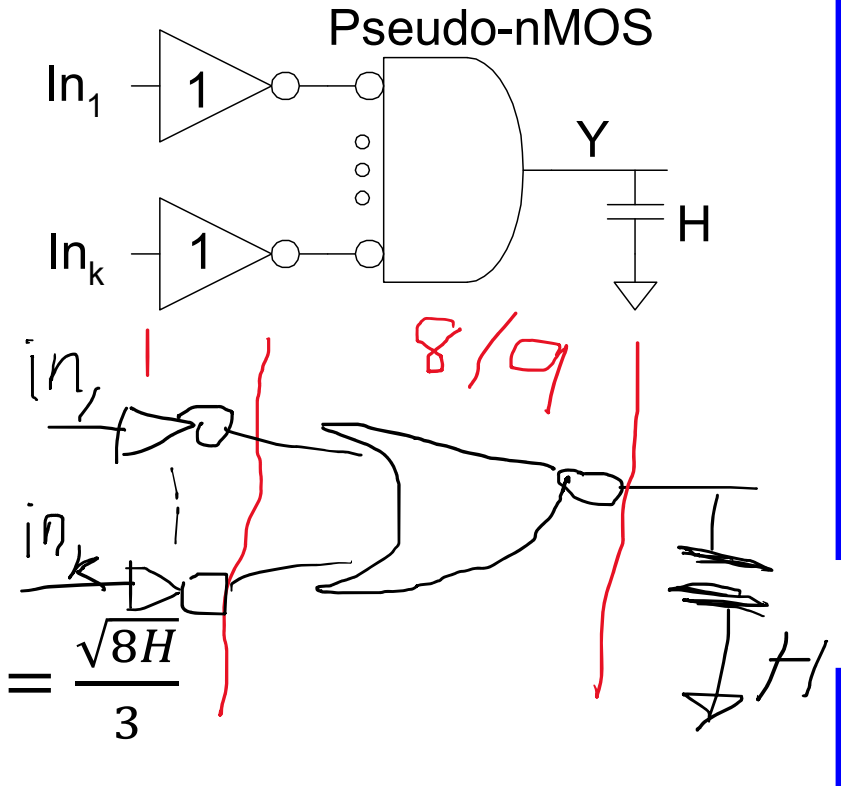
$$P_d = (2/3 + k \times 4/3)/3$$

$$P_{ave} = \frac{1}{2} [2/3 + 4/3 \times k + 2/9 + 4/9 \times k] = 4/9 + 8k/9$$

Pseudo-nMOS Design

- Ex: Design a k-input AND gate using pseudo-nMOS. Estimate the delay driving a fanout of H

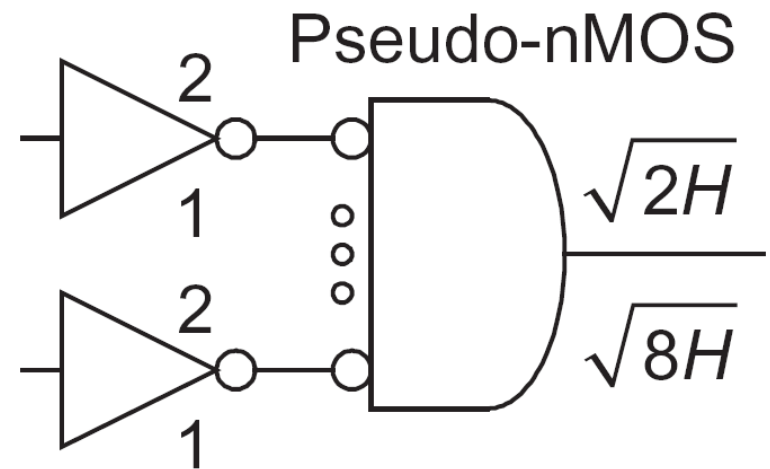
- $G =$
- $F =$
- $P = 1 + (4+8k)/9 = (8k+13)/9$
- $N =$
- $D =$



Which gives :
$$C_{in} = \frac{g C_{out}}{\hat{f}} = \frac{\frac{8}{9}H}{\frac{2\sqrt{2H}}{3}} = \frac{\sqrt{8H}}{3}$$

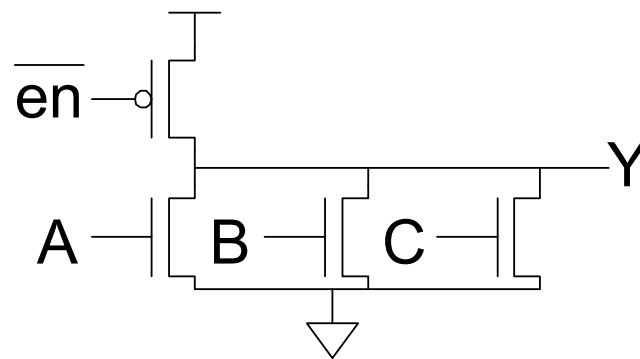
Pseudo-nMOS Design

Since the unit-sized inverter has an input capacitance of 3 units, the sizing of the nMOS NOR gate transistors should be $\sqrt{8H}$ and the size of the pMOS NOR gate would be $2 \cdot (\sqrt{8H})/4$ which makes it one fourth the nMOS strength.

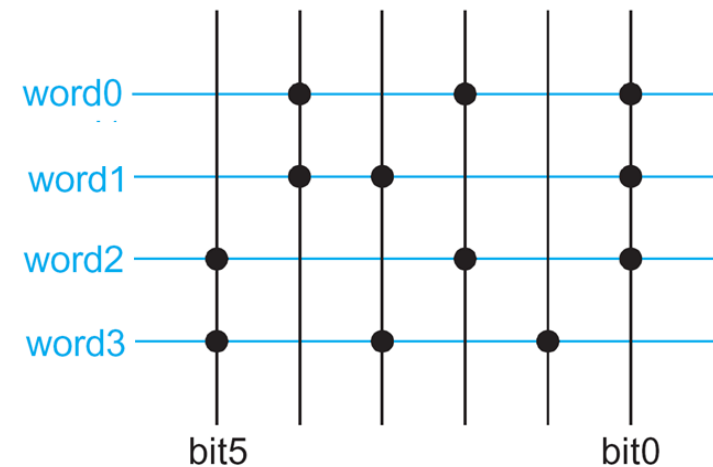
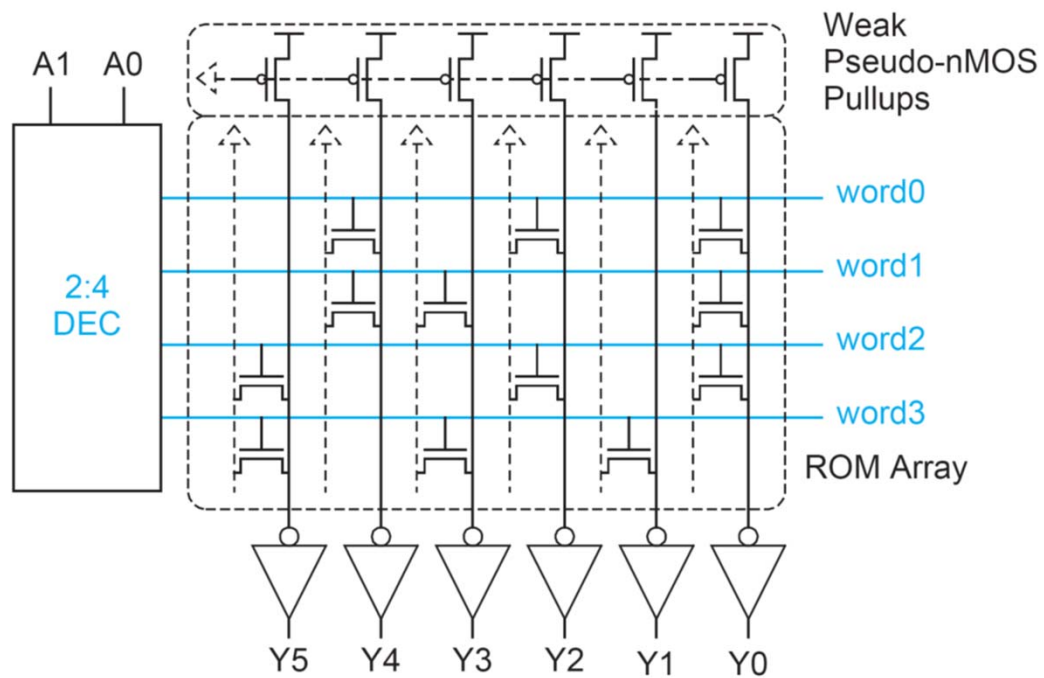


Pseudo-nMOS Power

- ❑ Pseudo-nMOS draws power whenever $Y = 0$
 - Called static power $P = I_{DD}V_{DD}$
 - A few mA / gate * 1M gates would be a problem
 - Explains why nMOS went extinct
- ❑ Use pseudo-nMOS sparingly for wide NORs
- ❑ Turn off pMOS when not in use



Pseudo nMOS ROM



Ratio Example

- The chip contains a 32 word x 48 bit ROM
 - Uses pseudo-nMOS decoder and bitline pullups
 - On average, one wordline and 24 bitlines are high
- Find static power drawn by the ROM
 - $I_{\text{on-p}} = 36 \mu\text{A}$, $V_{\text{DD}} = 1.0 \text{ V}$

□ Solution:

$$P_{\text{pull-up}} =$$

$$P_{\text{static}} =$$