Lecture

3

ECE M216A

Logic Design

Prof. Dejan Marković

ee216a@gmail.com

Agenda

- **Logic design concepts**
  - Fast vs. slow inputs
  - Transmission gates
- **More delay models**
  - Simulation-based
  - Library models
- **Optimal gate sizing**
  - Min delay
  - Delay > min delay

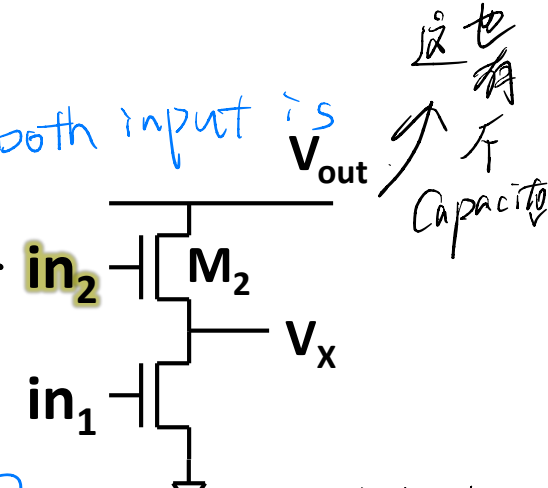
Faster Inputs

越靠近 output 的越晚 (faster)

- **Higher input of a stack is faster (in_2 is faster than in_1)**
  - V_X pre-discharged
- **Also body effect on M_2**
  - $V_X > 0$ increases V_{T2} and takes longer to discharge
- **Use higher inputs for the LATE arrivals**
  - Synthesis tools do this

in1 first, then in2

Faster input



both input is

这也有不 Capacitor

假如 in2 先到，

charge Vx，从而使 Vx 大，因此要更长时间去 discharge

① 先 in1，可以先对 Vx discharge to ground

So Vx discharge to 0

② in2，只需要考虑 output capacitor

假如先 in2，他会先给 Vx 充电，从而导致要更多次 discharge Vx

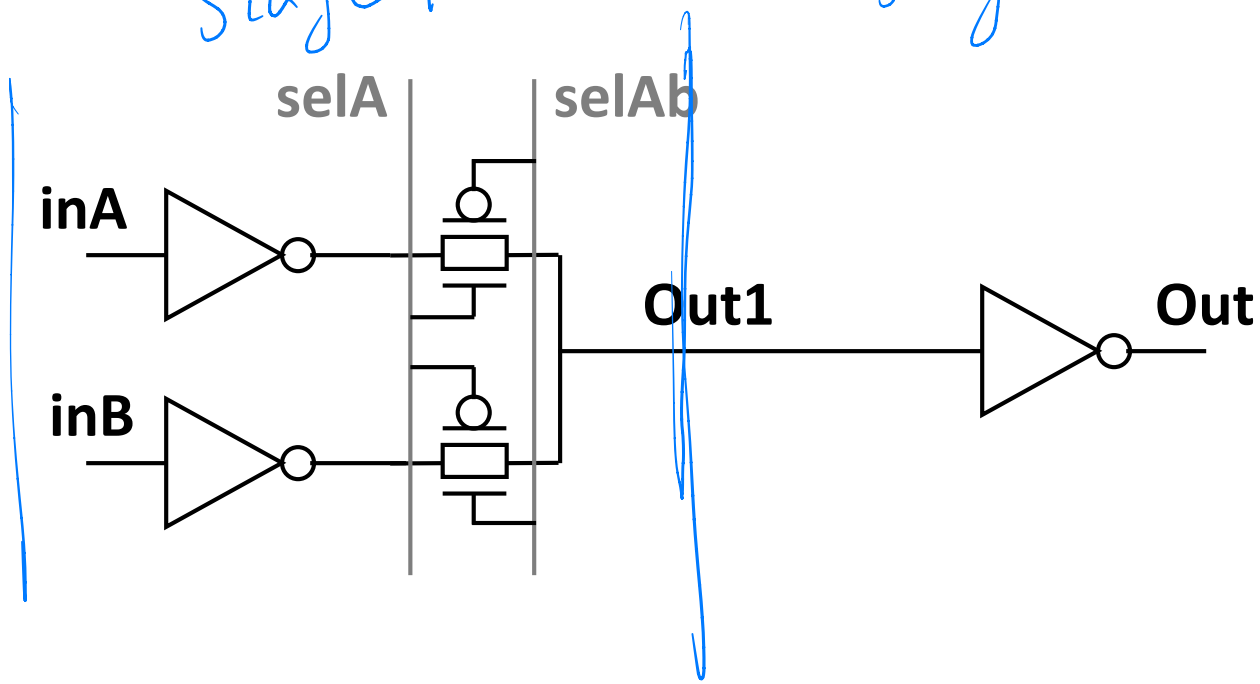
Mixing Static CMOS & Transmission Gates

- Static CMOS gates drive a TG switch network
- Output of network drives a static CMOS gate
- **Example:** a 2:1 Mux

$t_p = t_{p1} + t_{p2}$

Stage 1

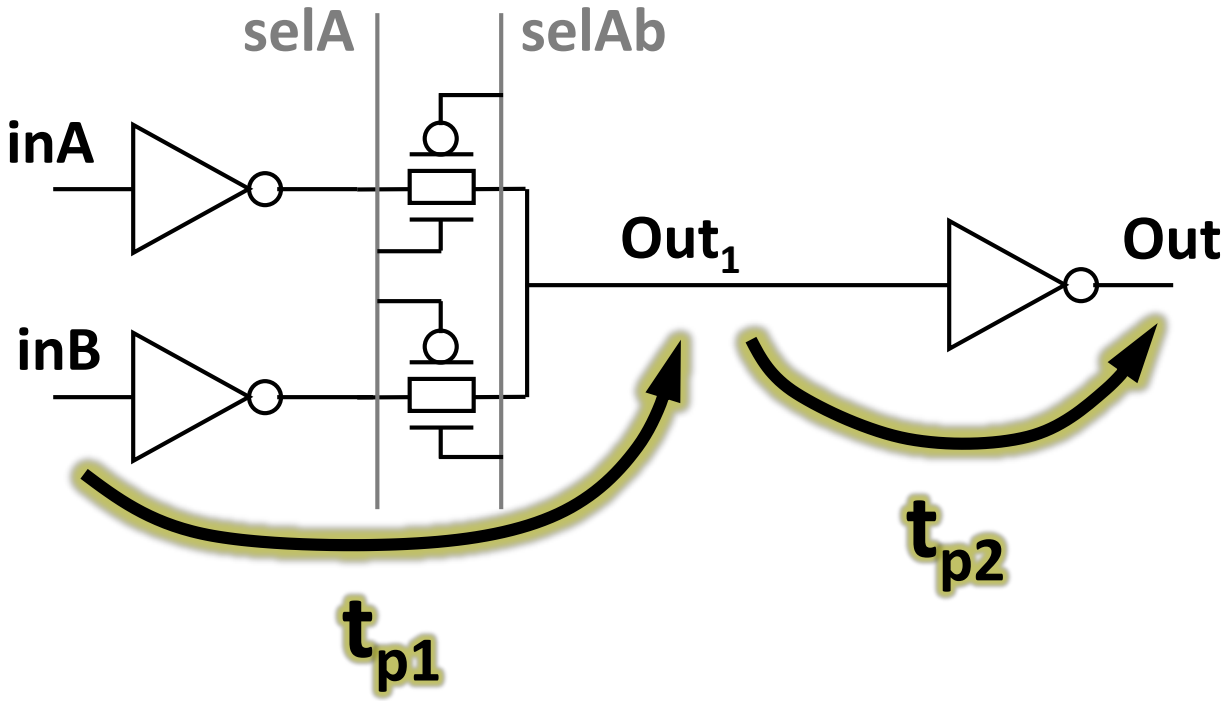
Stage 2



Example 3.1: Transmission Gate Delay

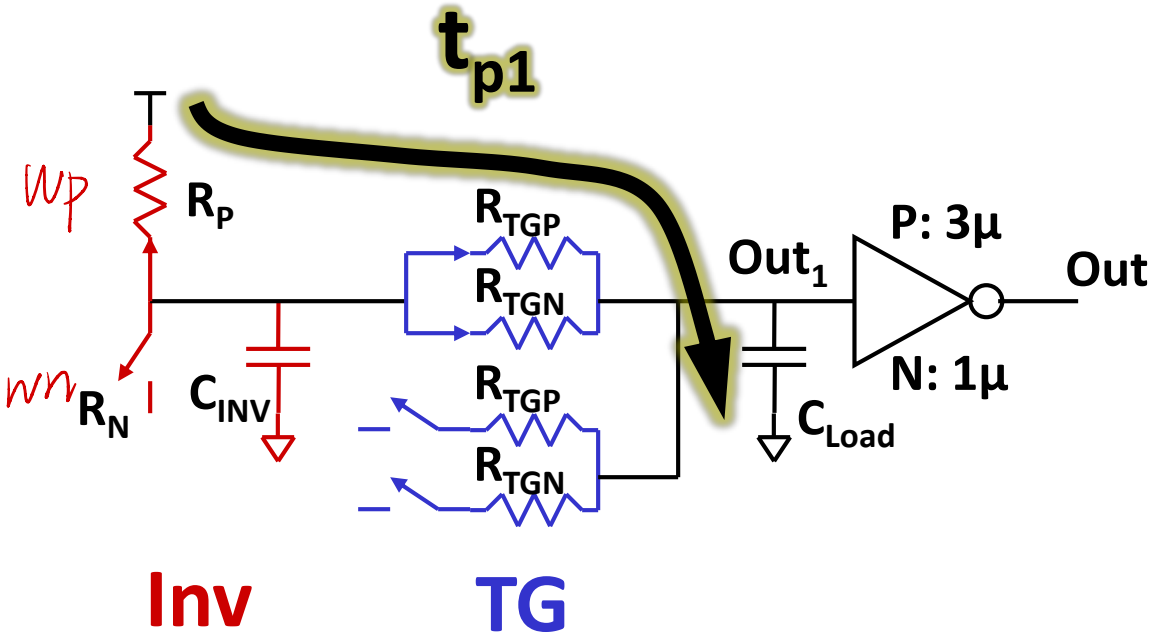
- **Delay depends on the gate that drives the switch**
- **Example:** path B selected, Out_1 pulling up

$$t_p = t_{p1} + t_{p2}$$



Modeling the Delay

- **Focus on the 1st (compound) stage**
  (2nd stage is a simple inverter)



Assume:

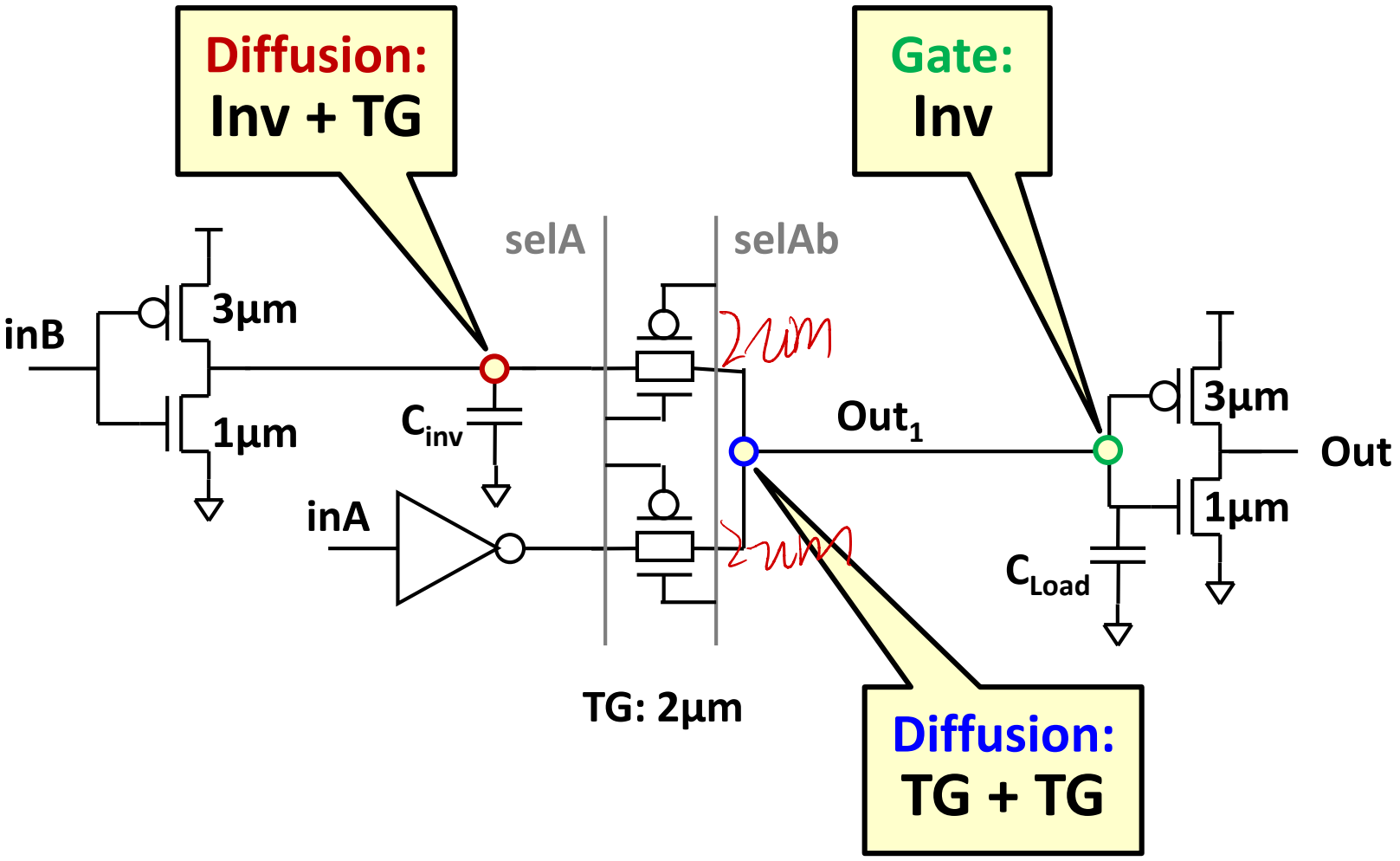
Inverters

- W_P = 3 µm
- W_N = 1 µm

Trans. Gates

- W_P = 2 µm
- W_N = 2 µm

Capacitance Components



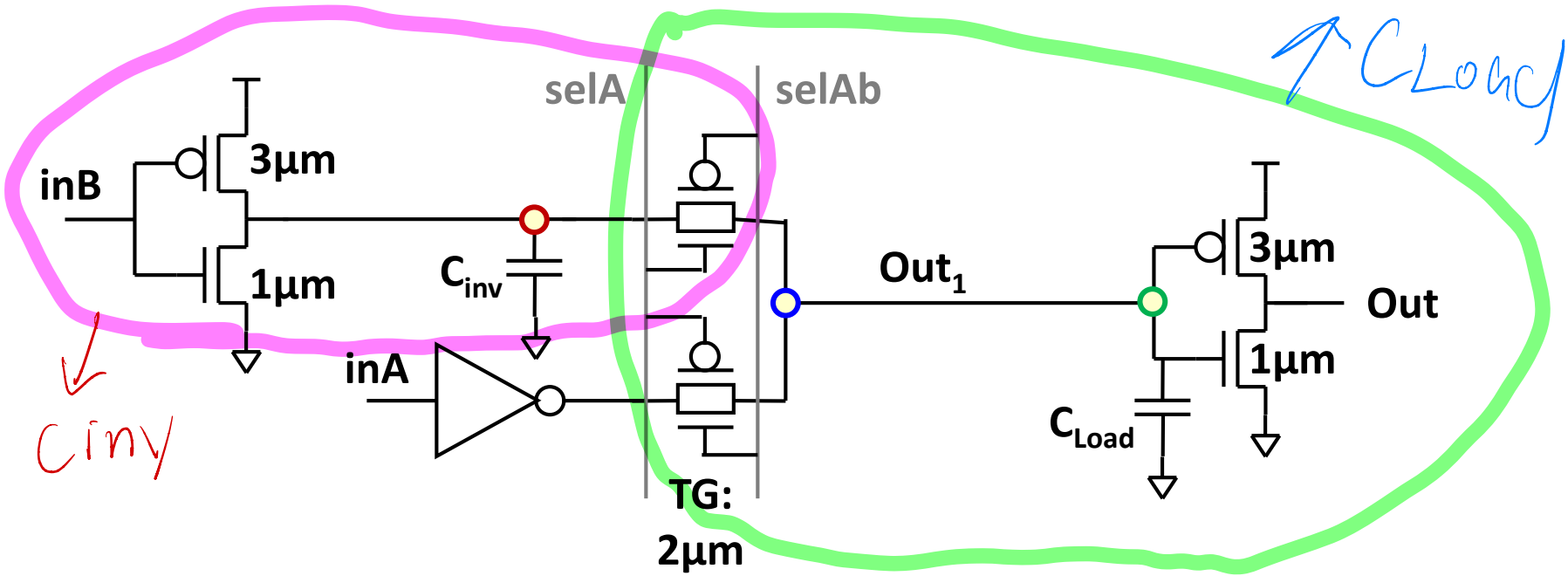
Calculate the Capacitances

Assumptions:
- C_D = 1.5 fF/µm
- C_G = 2 fF/µm

diffusion

$$C_{inv} = \overbrace{(3\mu+1\mu)\cdot 1.5f}^{\text{Inv}} + \overbrace{(2\mu+2\mu)\cdot 1.5f}^{\text{TG}} = 12\ fF$$

$$C_{Load} = \underbrace{(2\cdot 2\mu)\cdot 1.5f}_{\text{TG}} + \underbrace{(2\cdot 2\mu)\cdot 1.5f}_{\text{TG}} + \underbrace{(3\mu+1\mu)\cdot 2f}_{\text{Inv}} = 20\ fF$$



Calculate RC Time Constant τ_1

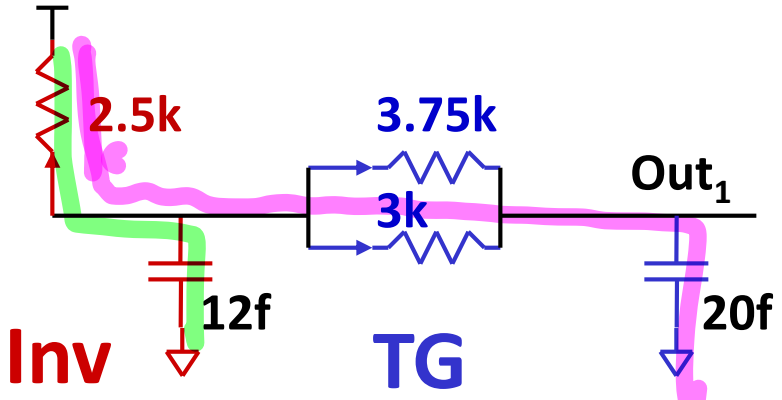
- R_P = 7.5 kΩ-μ / 3 μ = 2.5 kΩ
- R_{TGP} = 7.5 kΩ-μ / 2 μ = 3.75 kΩ
- R_{TGN} = 6 kΩ-μ / 2 μ = 3 kΩ

Assumptions:

- R_{ND} = 3 kΩ-μm
- R_{NU} = 6 kΩ-μm
- R_{PU} = 7.5 kΩ-μm
- R_{PD} = 15 kΩ-μm

up

down



select this "smart" worst case

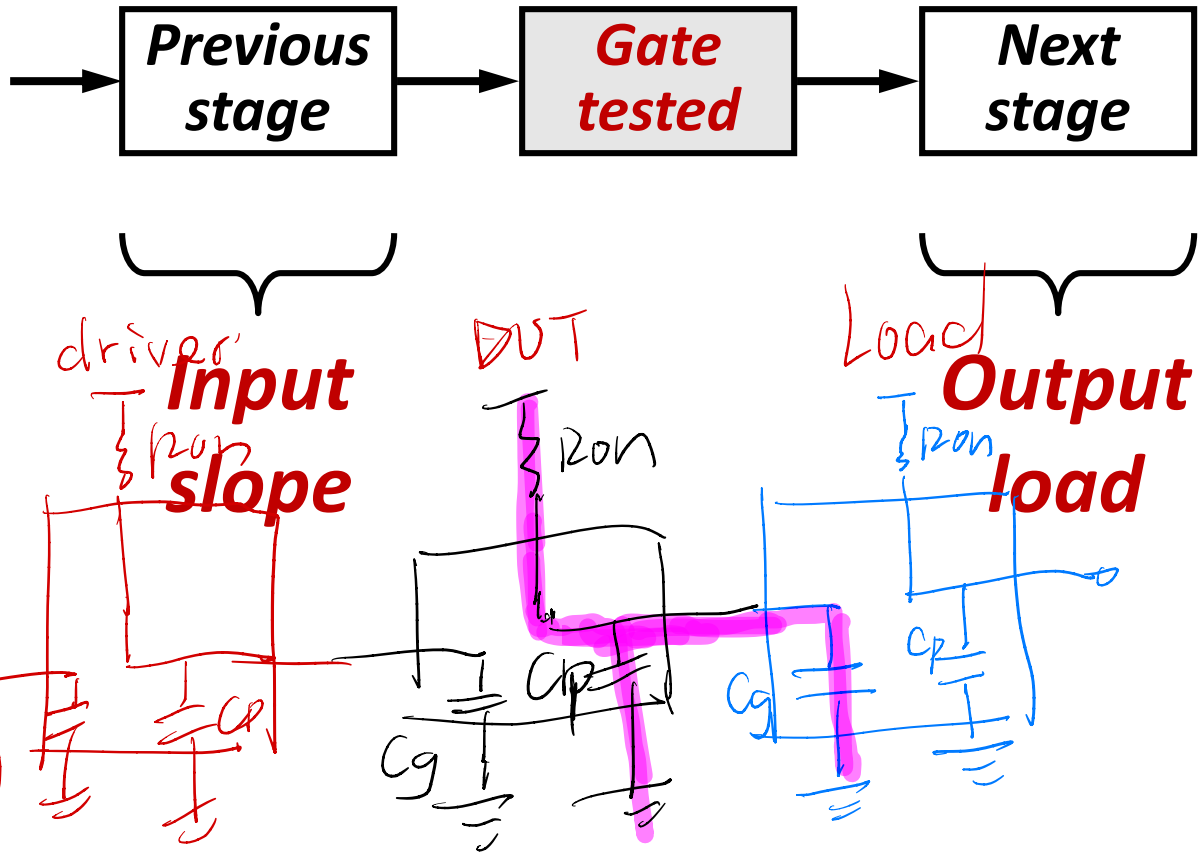
τ_1 = 2.5k·12f + (2.5k+3.75k||3k)·20f = 113ps

absolute worst case.

Simulation-Based Delay Model

Simulation-Based Parameter Extraction

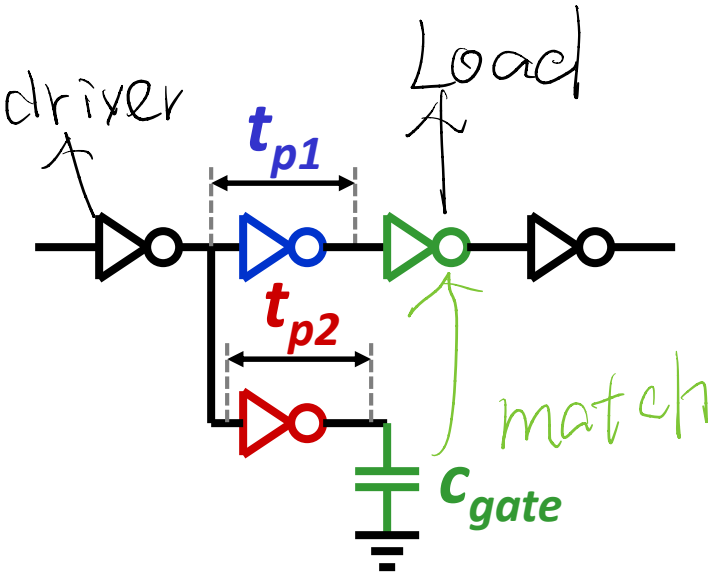
- **Typical simulation model**
  - Provides realistic input + load



Extracting C_{gate} by Simulation

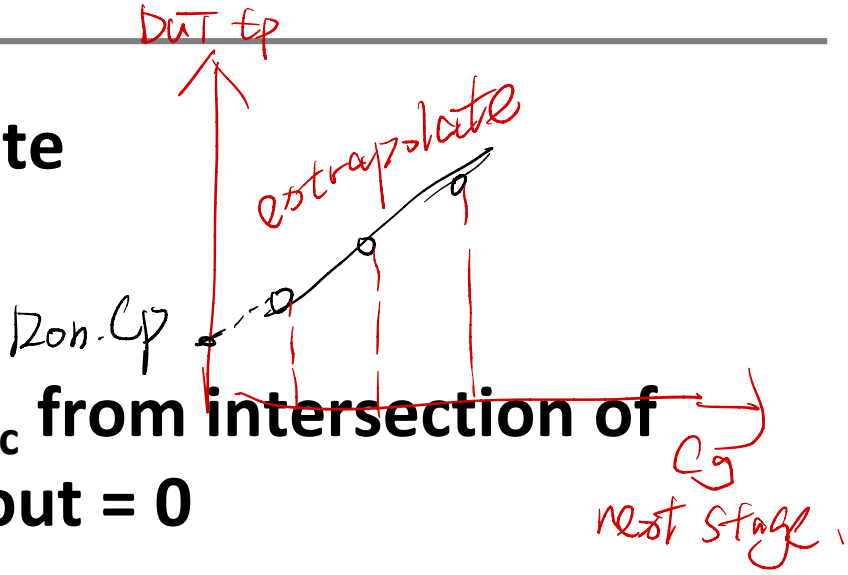
Curve-fitting:

Sweep C_{gate} until $t_{p1} = t_{p2}$



- $t_{p1} = t_{p2}$ → C_{gate} **is equivalent cap of the green gate**
- **Limitations:** the model depends on signal rise times, voltage, temperature, process parameter variations

Extracting $C_{parasitic}$ by Simulation



1. • **Delay of the unloaded gate**
   - Beware of signal slopes

2. • **Another way: find $C_{parasitic}$ from intersection of delay(fanout) line at fanout = 0**

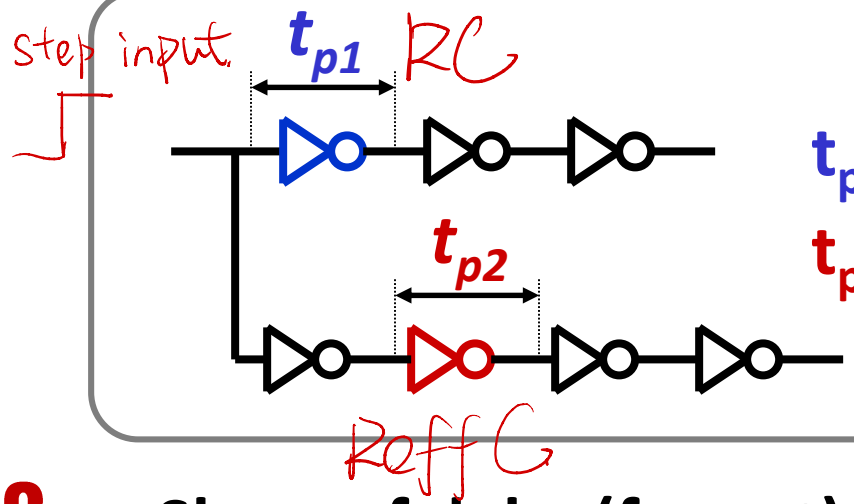
3. • **Play with AS, PS, AD, PD model parameters**
   - AS = AD = PS = PD = 0 [overlap + junction caps]

- **C_{gate} & C_{par} for t_{pLH} and t_{pHL} are different** [WHY?]
  - For delay analysis, we use the average value

Extracting R_{eff} by Simulation

1 • **Use the delay formula to find R_{eff}**

- $R_{eff} = t_{p,gate} / (0.69\ C_{gate})$
- Reminder: set AS, AD, PS, PD to 0!



t_{p1}: delay w/ step input

t_{p2}: delay w/ realistic slope

2 • **Slope of delay(fanout) line $\propto$ fanout$\cdot R_{eff} \cdot C_{gate}$**

- **R_{eff} for transistor stacks could be similarly found**
  - Make sure you factor in the parasitic cap

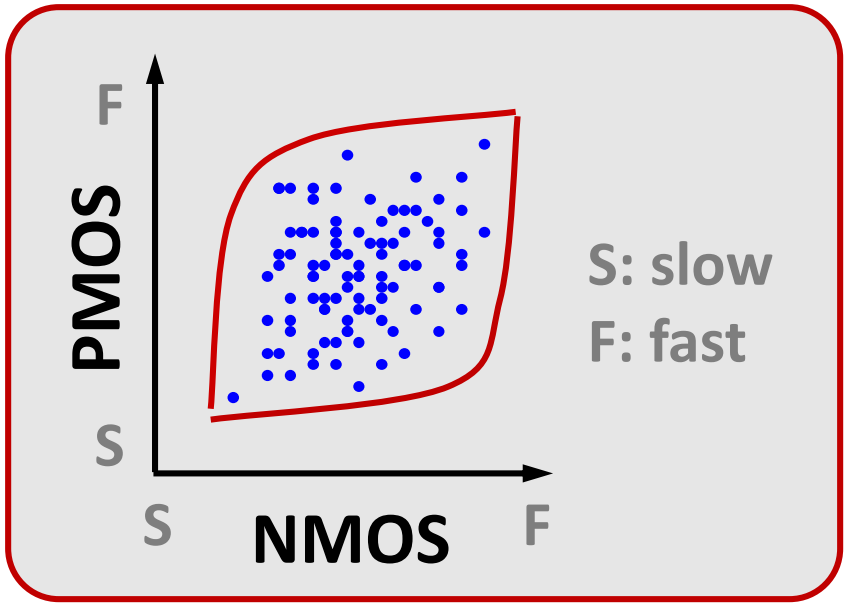
The “Flow”

- **Hand design**
  - Simple RC models provide **intuition** about circuits
  - Tradeoff analysis
  - Dominant effects
  - Reasonable starting point in the design process
  - Run simulations to refine the model

- **Simulation** (to confirm your intuition)
  - If the simulation results are way off, there is a bug (check schematics, simulation files, your models)
  - Don’t forget the corners

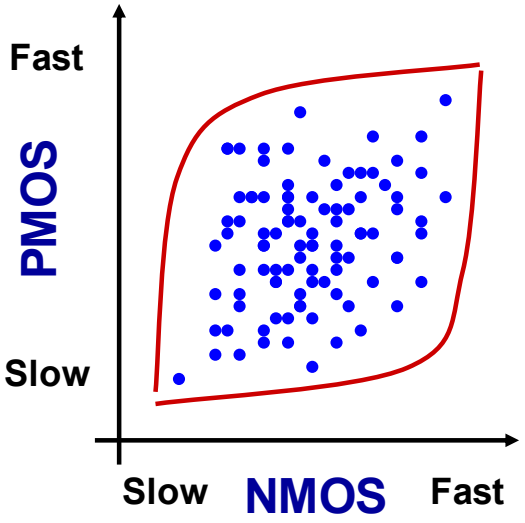
Process Variations

- **Not all devices (primarily transistors) are created equal**
  - Even if they nominally have the same exact drawing
  - Two on the same die (wafer, lot) can differ
- **Variations cause transistors to have different V_T and μ, speeding up (F) or slowing down (S) the devices**
  - Doping conc. (μ)
  - W/L
  - t_{ox}

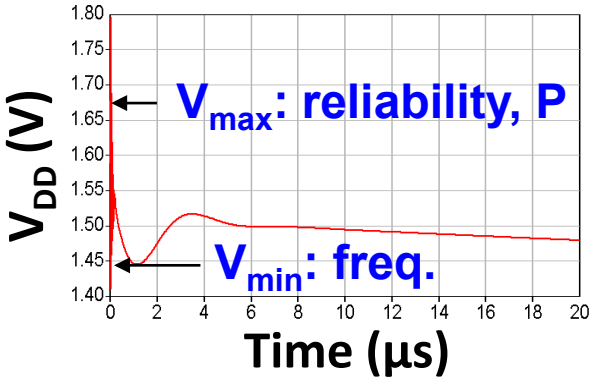


More Corners (PVT Variation)

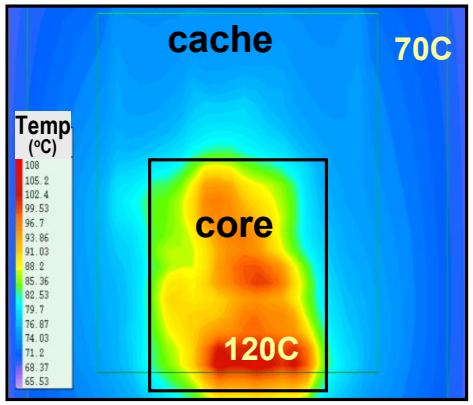
Process

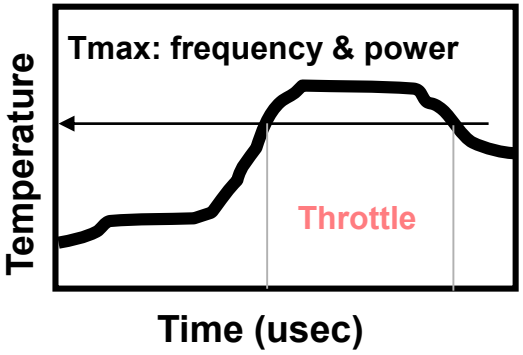


Voltage



Temperature

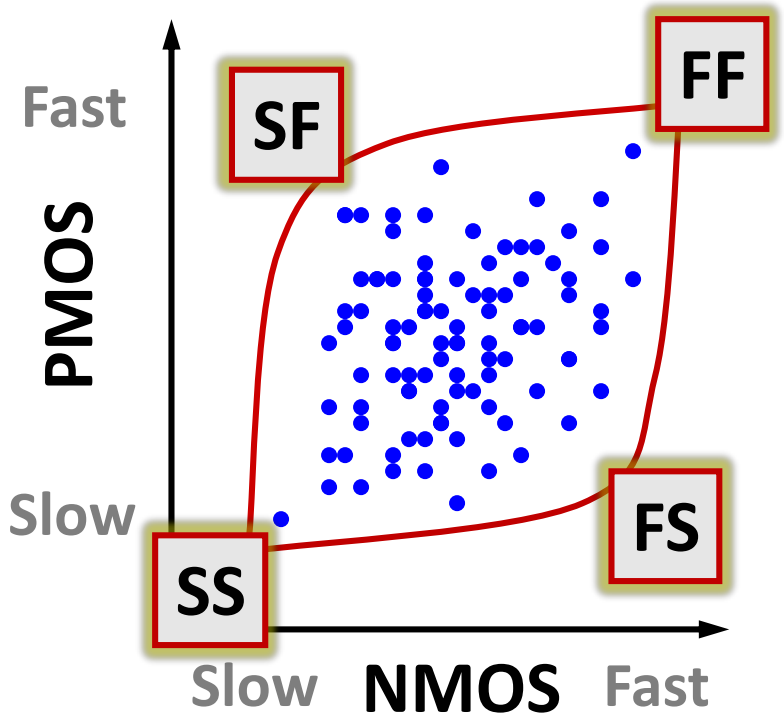




- **Typically, given corner parameters for devices**
  - Characterize effective parameters across corners

$$\mu = \mu_0 \cdot \left(\frac{T_0}{T}\right)^{1.5}$$

Process Corners Combined



Includes:

- **Process**
- **Voltage**
- **Temp**

variations

S: low V_{DD}, high T (t_{setup})
F: high V_{DD}, low T (t_{hold})

Delay and Transition Time

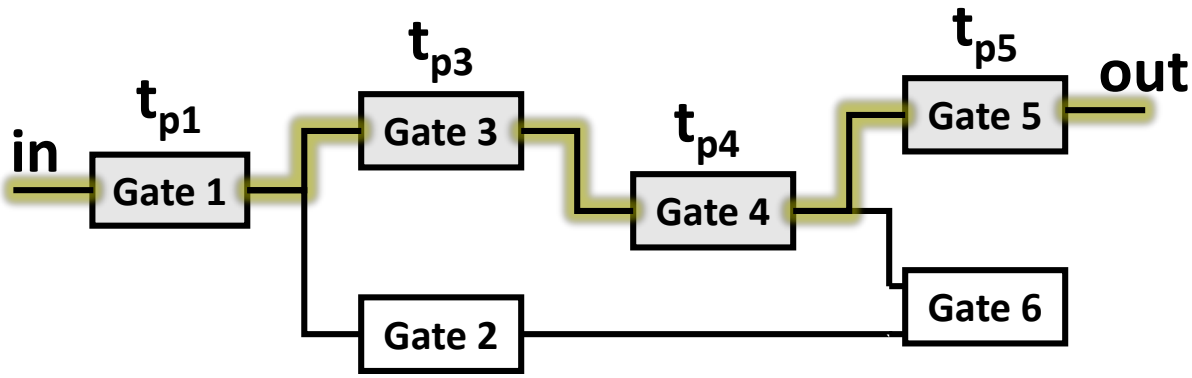
- **Gate delay is approximated as RC elements**
- **The transition time is proportional to the same RC**
  - Using ideal RC, the 10-90% is roughly 2.2RC
- **For a gate (inverter) driving another gate,**
  - Transition time is roughly $2t_{DELAY}$
  - Valid only for RC network
    - [What would happen in an inductive network?]
- **Useful for modeling noise coupling**
  - Aggressor transition time determines injected noise

Multi-Stage Gate Delay

- **Static CMOS:** total delay = sum of stage delays

$$t_{total} = \sum_{n} t_n$$

- **The R of the gate** (current stage)
- **The C of the subsequent gate(s)** (next stage)

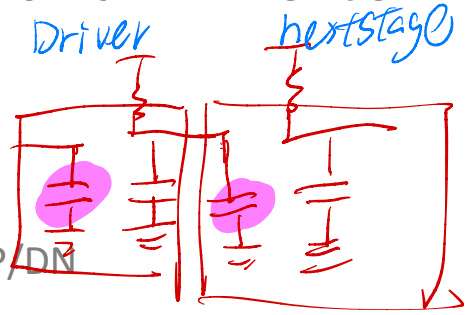


$$t_{total} = t_{p1}(R_1, C_3, C_2) + t_{p3}(R_3, C_4) + t_{p4}(R_4, C_5, C_6) + t_{p5}(R_5, C_L)$$

(C_L: Load)

Fan-In and Fan-Out

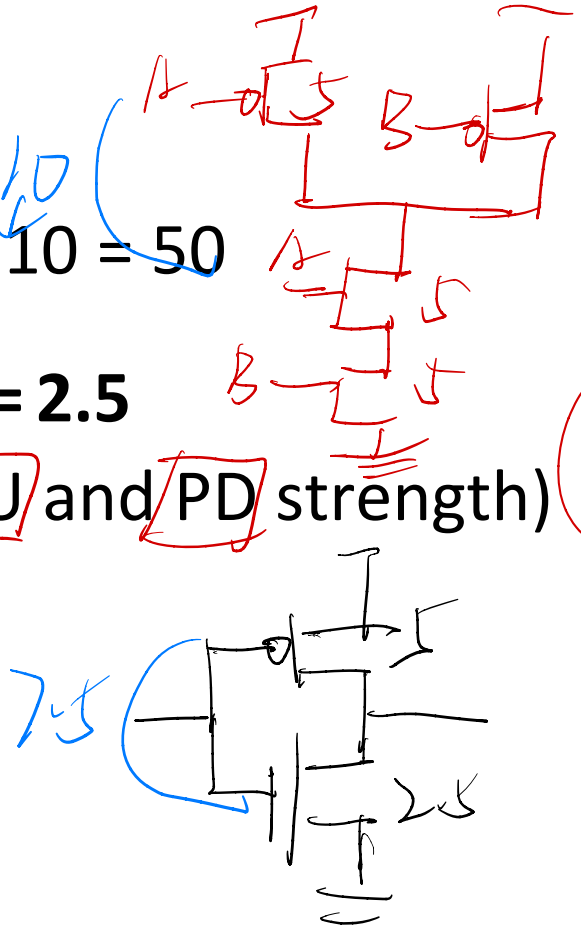
- **Fan-In:** the number of inputs (logic gate)
  - An indication of the input load that the gate presents to a predecessor gate
    - Because the series stack is roughly the number of inputs
    - Later we will use **Logical Effort** to embed this concept
- **Fan-Out:** effective output load (relative to Inv) ratio of Cgate of next stage and Cgate of Driver
  - An indication of the loading (gate type dependent)
  - Useful to normalize the loading to C_{gate} of an inverter with equal drive strength as the gate
    - $FO = C_{LOAD}/C_{INV}$
      where $C_{INV} = C_0'(W_P+W_N)$ and $R_{INV} = R_{PULL\_UP/DN}$



Example 3.2: Fanout Calculation

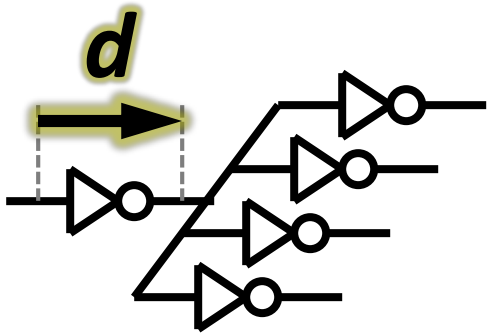
NAND gate driving 5 equal NAND gates

- **NAND gate: $W_p = 5$, $W_n = 5$**
  - Total input width = 10
  - Total load gate width = 5·10 = 50
- **Equivalent Inv: $W_p = 5$, $W_n = 2.5$**
  - Equivalent (in terms of PU and PD strength)
  - Total gate width = 7.5
- **Fanout = 50/7.5 = 6.6**



Another Metric: FO4 Inverter Delay

- **Measures quality of design independent of technology**



Cadence 90nm technology:

FO4 = 33ps

Ring Osc Stage = 13ps

Mini assignment:
Simulate FO4 delay in 32nm (slide 2.34)

Model Used in Cell Libraries

Propagate two quantities:

- **Delay**
- **Signal slope**

Modeled as linear or table lookups

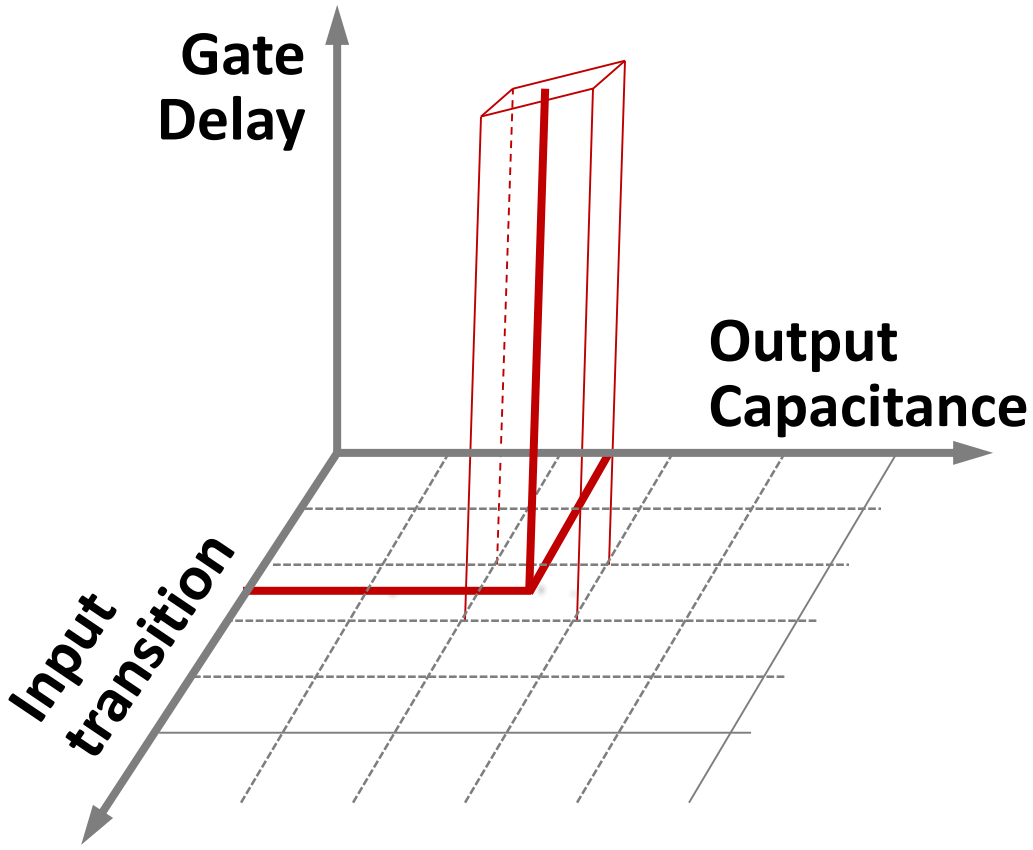
Gate Delay Estimation

- **Depends on transistor sizes, input signal slew, output capacitive load and PVT conditions**

- **For each PVT, timing library is provided by vendor**
  - Delay, output slew, dynamic and static power are characterized for different input slew and output loads

- **Characterization values are stored as 2-D look-up tables**
  - Output delay = f (input slew, output load capacitance)

Gate Delay Estimation



Interpolation with 4 points

Courtesy: Synopsys

Design Problem: Interconnect Delay

- Timing assumption during pre-layout synthesis widely differs from the post-layout reality
- This happens because the interconnect delay dominates the overall delay in scaled technologies
- As a result, timing closure becomes a challenge

Courtesy: Synopsys

CAD Problem: Interconnect Delay

- **New industry trends = new IC design flows**
- **The major contributing factors:**
  - Interconnect delay and area
  - The number of metal layers
  - Planning requirements
- **Need to consider interconnect delay early in design**

- **A few techniques for reducing interconnect delay:**
  - Chip-level signal planning
  - Over-the-Cell routing

Courtesy: Synopsys

Wire Load Models (WLMs)

- **In the absence of physical design information DC uses statistically generated WLMs to estimate wire lengths**
  - WLMs provide a fanout vs. length relationship

- **Interconnect delay estimation:**
  - By knowing fanout, estimate the wire lengths
  - Next, given unit-length R and C and the estimated length, estimate R and C to give an estimated delays

- **Note:** WLMs are area dependent
  - Unit-length R and C increase with area

Courtesy: Synopsys

Example WLM

```
wire_load('30x30') {
capacitance : 3.0 ; /* C per unit length */
resistance : 30.0 ; /* R per unit length */
area : 1.5 ; /* area per unit length */
slope : 1.5 ; /* extrapolation slope */
fanout_length(1,1) ; /* fanout/length pairs */
fanout_length(2,2.2);
fanout_length(3,3.3);
fanout_length(4,4.4); }
```

Model from a library compiler (LC) source file:

依次确定

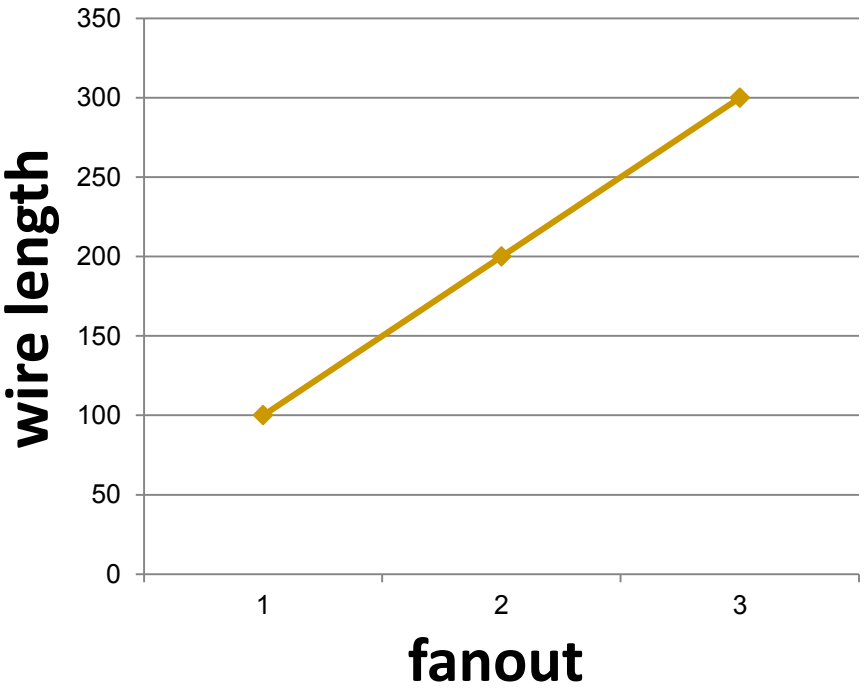
Fanout

⇩

Wire length

⇩

RC delay

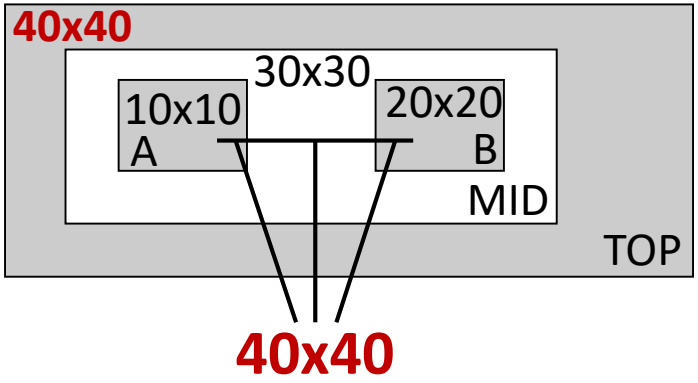


Courtesy: Synopsys

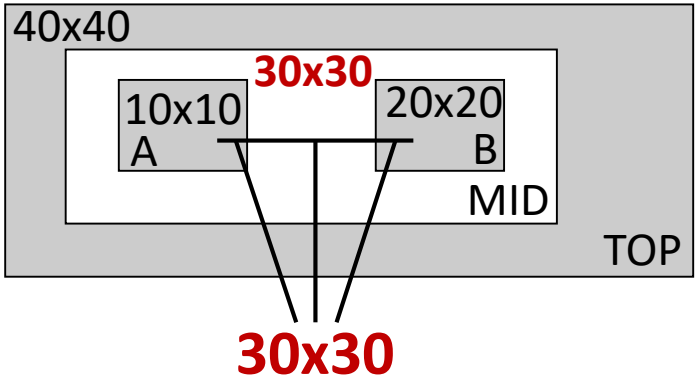
Hierarchical Wire Load Models

DC supports 3 modes for nets that cross hier. boundaries

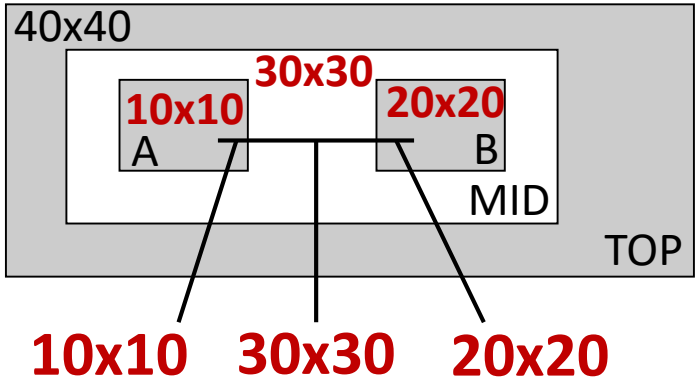
Mode = **top**



Mode = **enclosed**



Mode = **segmented**



Courtesy: Synopsys

Speed Optimization via Gate Sizing

Speed Optimization via Gate Sizing

- **Gate sizing basics**
  - P:N ratio
  - Complex gates
  - Velocity saturation
  - Tapering

- **Developing intuition**
  - Number of stages vs. fanout
  - Popular inverter chain example

Basic Gate Sizing Relationships

- **Rise and fall delays are determined by the pull-up and pull-down "strength"**
  - Besides the W/L, strength depends on μ, V_T
  - PMOS is weaker because of lower μ_p
    - Larger P network than N network
- **Increasing size of gate can reduce delay**
  - $R_{on} \propto 1/W$
  - BUT it can slow down the gate driving it
    - $C \propto W$. So be careful!

P:N Ratio for "Equal" Rise and Fall Delay

- **Good to have $t_{pHL} \approx t_{pLH}$**
  - Don't need to worry about a worst-case sequence
  - Size P's to compensate for mobility
    - C_{OX}, V_T, L are roughly the same

$$R_{on} \propto \frac{1}{I} \propto \frac{1}{\mu W}$$

- **Make $R_p = R_n$**
  - $R_n/R_p = 1$
    $(\mu_p W_p/\mu_n W_n = \beta/k)$

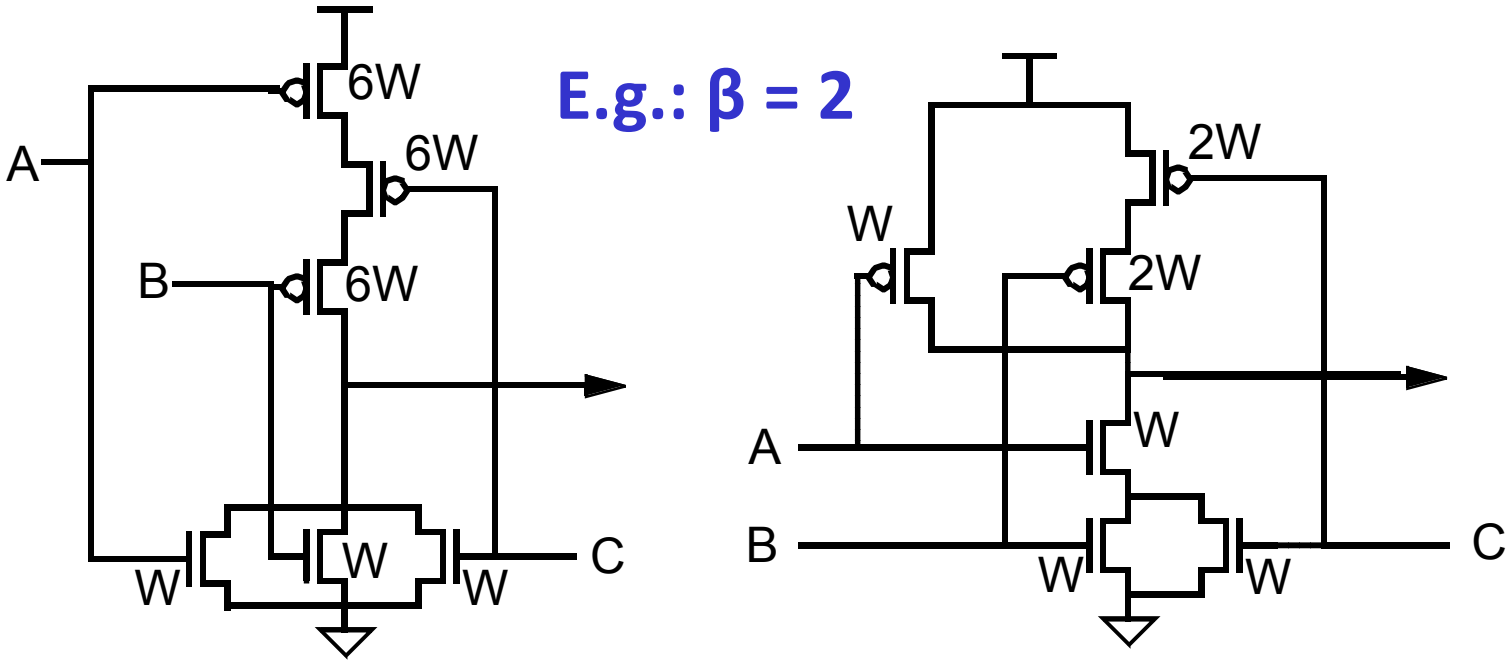
$t_{pLH} = t_{pHL}$

$$\frac{W_p}{W_n} = \beta = k = \frac{\mu_n}{\mu_p}$$

Complex Gate Sizing

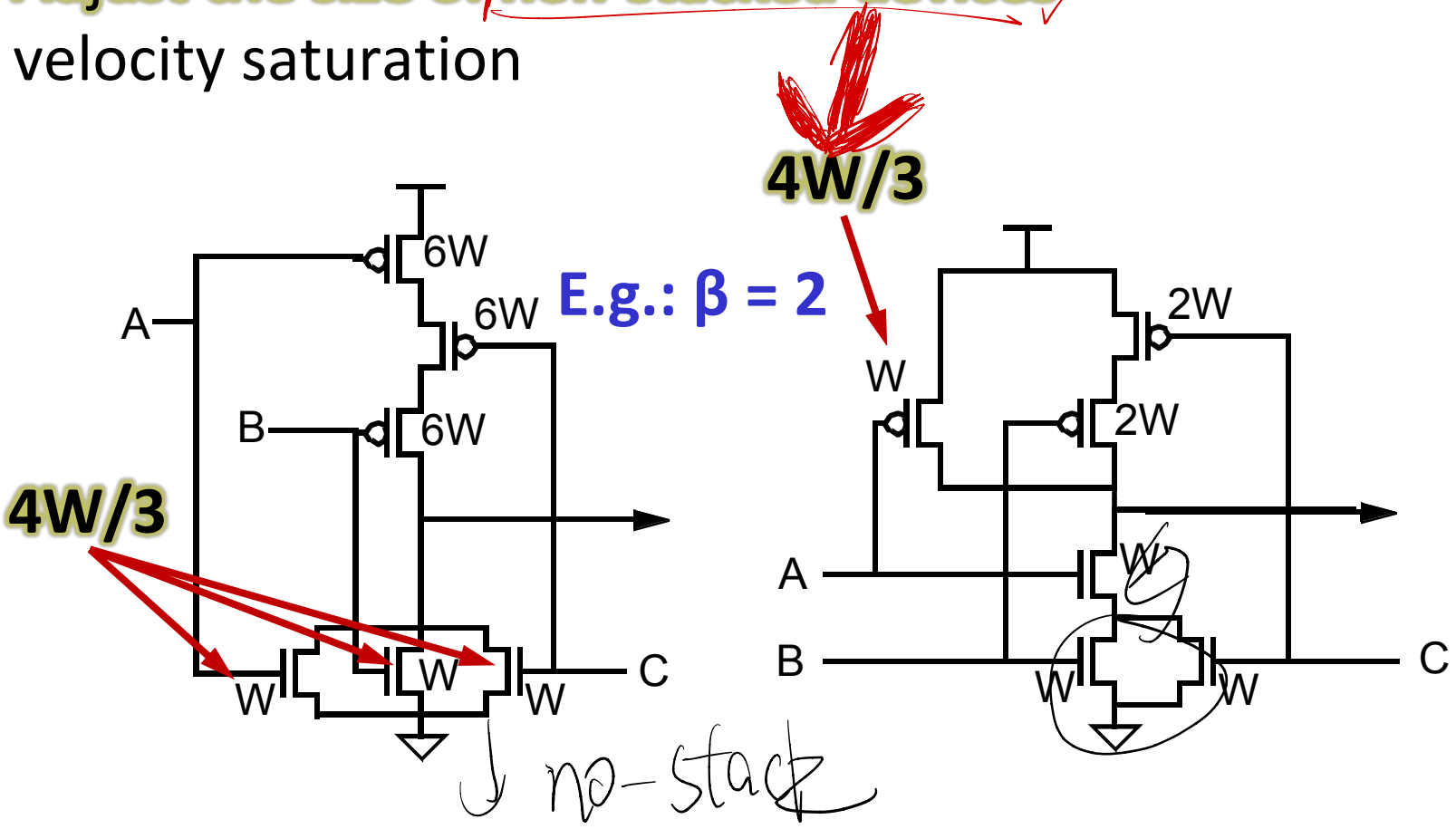
Stacking

- **N-stack series devices need N times lower resistance**
  - N×Width
- **Make worst-case strength of each path equal**
  - Multi-input transition can result in stronger network
- **Long series stacking is VERY bad**



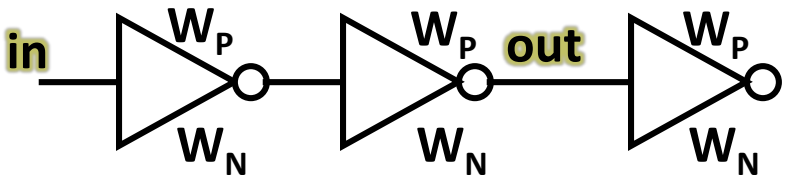
Accounting for Velocity Saturation

- **Series stacking is actually less velocity saturated**
  - If we use $R_{no\_stack} = (4/3)R_{stack}$
  - Adjust the size of non-stacked devices to account for velocity saturation



P:N Ratio for Minimum Delay?

- **Delay of 2 inverters:**



Assumptions:

- $R_{PDRV} \sim R_0'/W_P\mu_P$
- $R_{NDRV} \sim R_0'/W_N\mu_N$
- $C_G \sim C_0(1+W_P/W_N)$

- $t_p = t_{p1} + t_{p2} = R_0'(1/W_P\mu_P + 1/W_N\mu_N)\, C_0(1+W_P/W_N)$
- $t_p = t_N(1+1/k\beta)(1+\beta)$
- $dt_p/d\beta = 0$
  $= t_N(1-k/\beta^2)$

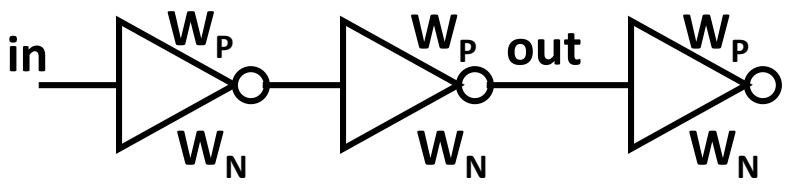
⇨

Min $(t_{pLH} + t_{pHL})/2$

$$\frac{W_p}{W_n} = \beta = \sqrt{k} = \sqrt{\frac{\mu_n}{\mu_p}}$$

P:N Ratio for Equal/Minimum Delay

- **Delay of 2 inverters:**



$$t_{pLH} = t_{pHL}$$

$$\frac{W_p}{W_n} = \beta = k = \frac{\mu_n}{\mu_p}$$

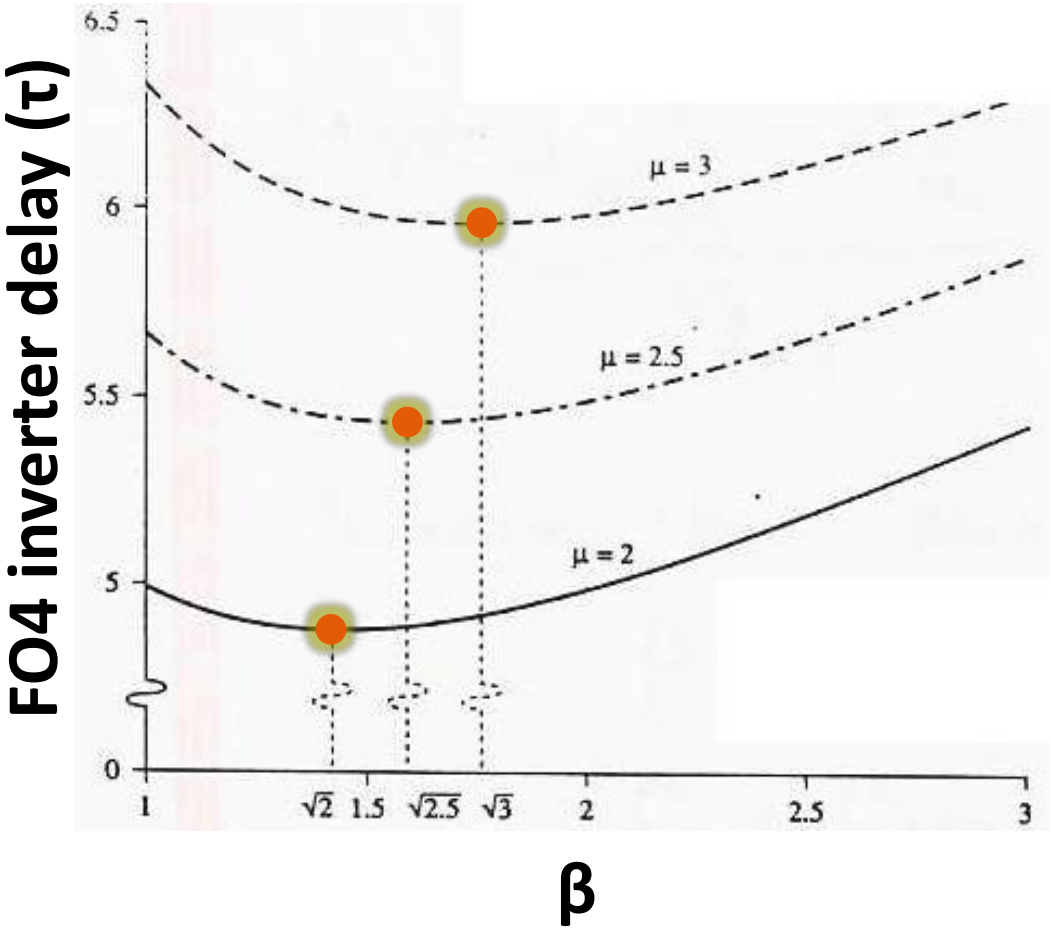
NMOS: more drive for a given size, so it is better to use more NMOS

$$\text{Min } (t_{pLH} + t_{pHL})/2$$

$$\frac{W_p}{W_n} = \beta = \sqrt{k} = \sqrt{\frac{\mu_n}{\mu_p}}$$

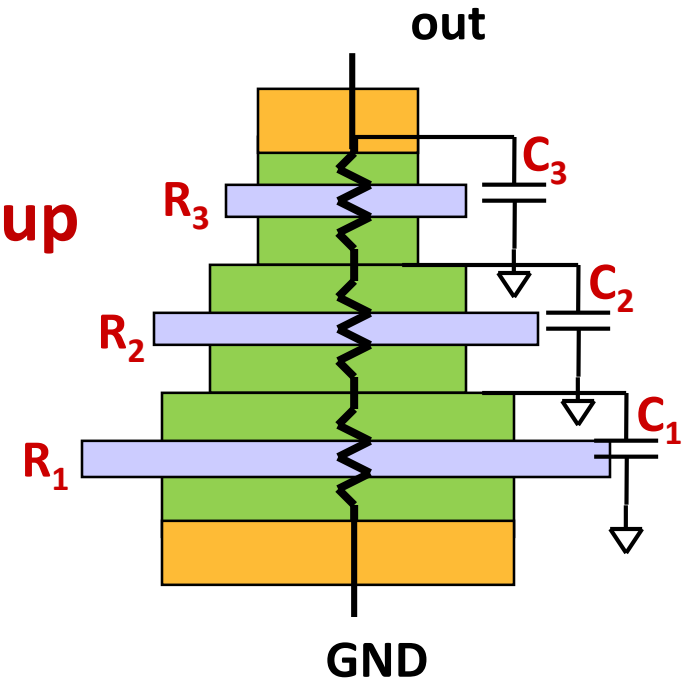
FO4 Inverter Delay vs. P:N Ratio β

- **Optimal β = sqrt(μ) for minimum delay**
  - Curve is relatively flat so not a strong delay tradeoff



An Idea: Tapering

- **C closer to the v-source: less effect on delay**
  - N = 2: $t_p = R_1(C_1)+(R_1+R_2)(C_2)$
    - C_1 has less effect on delay than C_2
- **So taper stacked devices for speedup**
  - Make the bottom ones bigger
    - R_1 (many occurrences) has less resistance
    - C_3 (multiplying larger R) has smaller capacitance

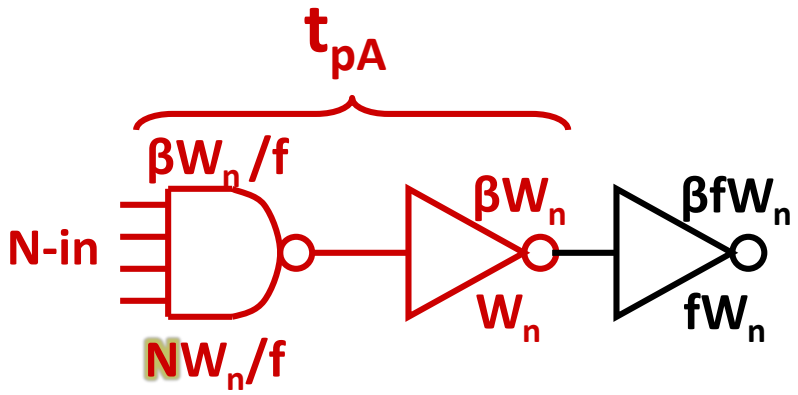


In reality, tapering doesn't win as much because layout is less compact when stacking unequal sized transistors (causing more C)

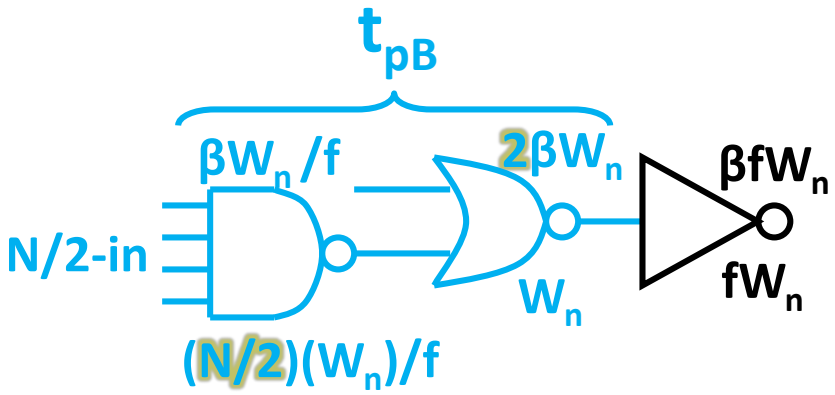
Example 3.3: Gate Design for Min Delay

- **Which scenario (A, B) has faster delay?**
  - Drive the same output load

A: N-in NAND + INV



B: N/2-in NAND + NOR



Let's analyze building blocks: NAND, NOR, INV

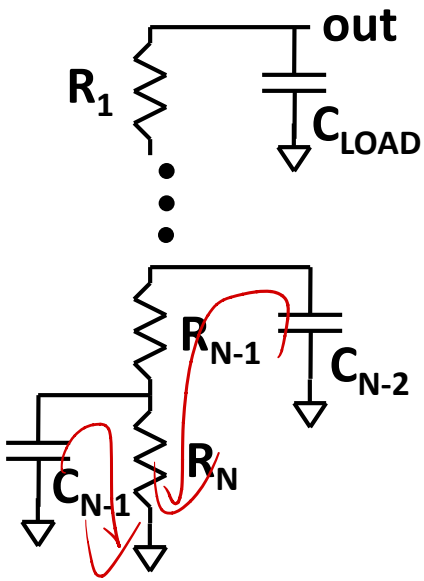
A1: Delay of N-input NAND

Assumptions:

- $C_G = C_D = C_0$ [fF/μm]
- $R_n = R_0$W-μm

N-stack NMOS:

$$N \cdot W_N / f$$



For N inputs:

- $R_1 = \ldots = R_N = R_0/(NW_n/f)$ $= R_i$
- $C_1 = \ldots = C_N = NC_0W_n/f$ $= C_i$

$R_iC_i = R_0C_0$

$$t_p = \sum_{i=1}^{N-1} iR_iC_i + R_{tot}\left(C_N + NC_0\frac{\beta W_n}{f} + C_{gate}\right)$$

A1: Delay of N-input NAND

$$t_p = \sum_{i=1}^{N-1} iR_iC_i + R_{tot}(C_N + NC_0\frac{\beta W_n}{f} + C_{gate})$$

$R_{tot} = \frac{R_0}{W_n f}$ $R_i = \frac{R_0}{i W_n f}$

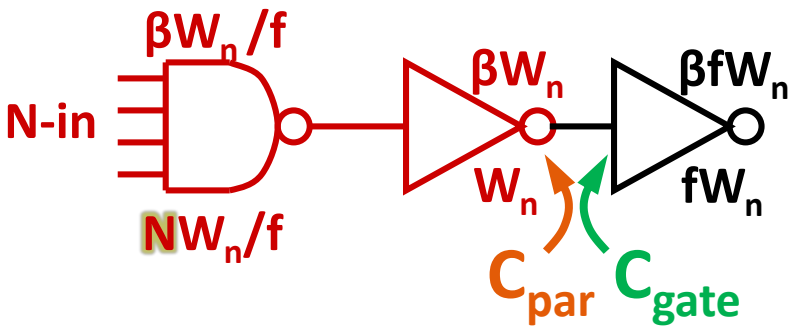
$$t_p = \sum_{i=1}^{N-1} iR_0C_0 + R_0(NC_0 + NC_0\beta + C_{gate}\frac{f}{W_n})$$

n transistor elements

NMOS PMOS Output

$$t_p = R_0C_0\left(\frac{N(N-1)}{2} + N\right) + R_0C_0\left(\beta N + \frac{C_{gate}}{C_0}\frac{f}{W_n}\right)$$

A2: Delay of Inverter

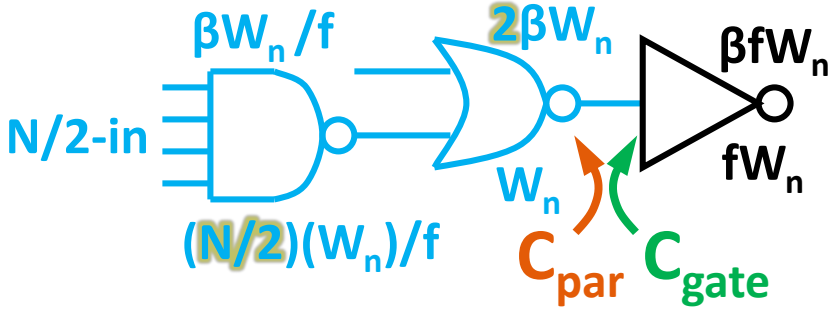


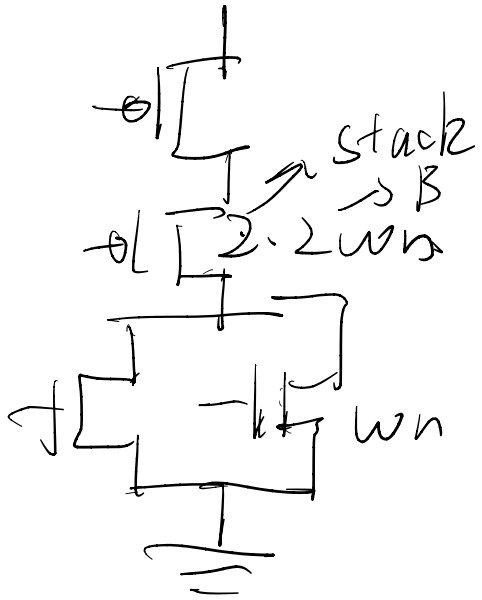
Inverter:

- $R_{INV} = R_0/W_n$
- $C_{L,INV} = C_{par} + C_{gate} = C_0(W_n(1+\beta) + f\,W_n(1+\beta))$
- $t_{INV} = R_0C_0(1+\beta)(1+f)$

$$C_{gate,INV} = C_0(W_n(1+\beta))$$

B2: Delay of NOR2





NOR2:

2 NMOS

- $R_{NOR} = R_0/W_n$
- $C_{L,NOR} = C_{par} + C_{gate} = C_0(W_n(2+2\beta) + f\,W_n(1+\beta))$
- $t_{NOR} = R_0C_0(1+\beta)(2+f)$

$$C_{gate,NOR} = C_0(W_n(1+2\beta))$$

Delay Comparison (β = 2)

A: N-in NAND + INV

$$t_{pA} = R_0 C_0 \left(\frac{N(N-1)}{2} + 3N + 3f \right) + R_0 C_0 (3f + 3)$$

B: N/2-in NAND + NOR

$$t_{pB} = R_0 C_0 \left(\frac{N\left(\frac{N}{2} - 1\right)}{4} + \frac{3N}{2} + 5f \right) + R_0 C_0 (3f + 6)$$

Mini assignment:

Assume f = k

N = ? @ crossover

Sweep N: Delay Comparison (β = 2)

N	t_{pA}	t_{pB}	f = 3	f = 4	f = 5
4	21 + 6f	13 + 8f	39 | 37	45 | 45	51 | 53
6	36 + 6f	21 + 8f	54 | 42	60 | 53	66 | 61
8	55 + 6f	30 + 8f	73 | 54	79 | 62	85 | 70
			A | B	**A | B**	**A | B**

Min t_p: don't build gates with fan-in > 4

Transmission Gate Sizing

Try to make a TG with $R_{PD} = R_{PU}$

P:N ratio of k is not good for delay

- **NMOS still has some pull-up strength (even if not all the way to V_{DD})**
  - No need to have wide PMOS
- **PMOS has some pull-down (but very weak)**
  - PD slightly faster (NMOS is stronger)

TG Sizing: Some Common Numbers

2x penalty, weak transition

$R_{N,DN} = R_0$ kΩ-μm | $R_{N,UP} = 2R_0$ kΩ-μm

$R_{P,UP} = 2.5R_0$ kΩ-μm | $R_{P,DN} = 5R_0$ kΩ-μm

- Let's try $W_p = W_n$
  - Parallel Up, $R_{TGUP} = R_{N,UP} || R_{P,UP} = 1.1R_0$
  - Parallel Down, $R_{TGDN} = R_{N,DN} || R_{P,DN} = 0.83R_0$

Slightly faster

↑ for transmission gate

- So, using $W_p/W_n = 1$ is fairly reasonable
- Actual size may depend on the process technology

Delay Analysis (So Far): Summary

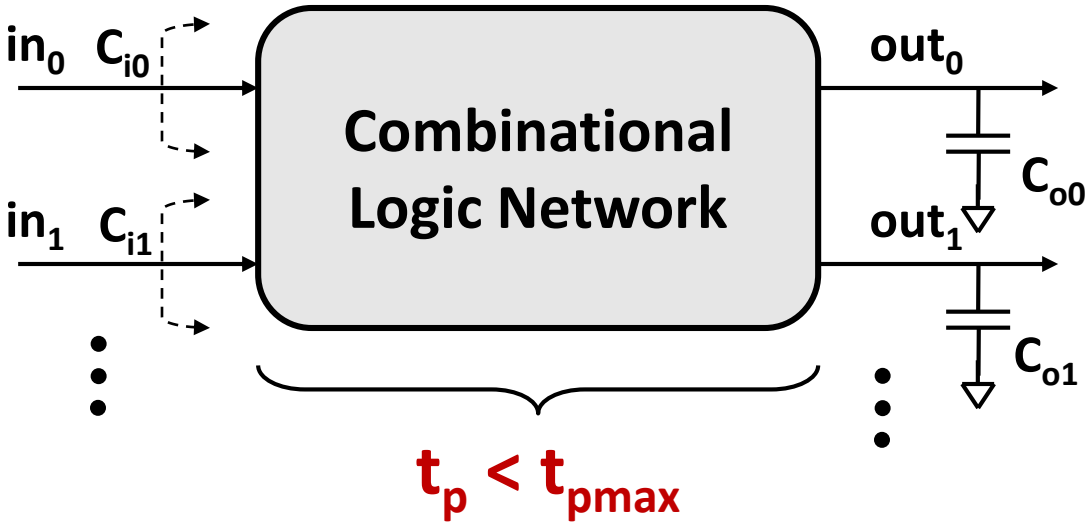
- **Device R and C determine circuit performance**
- **Elmore Delay approximation: initial insight into design**
  - Step response, does not account for signal slopes
  - Need slope correction [see slide 2.63]
- **The sizing of the transistors (a first glimpse)**
  - Determines V_M
  - Determines R_{DRV} as well C_{Load} it presents to the preceding gate which effects the delay
- **Large fan-in gates:**
  large self-loading and loading to the preceding gate
  - Split into 2 gates when fan-in > 4

Speed Optimization via Gate Sizing

- **Gate sizing basics**
  - P:N ratio
  - Complex gates
  - Velocity saturation
  - Tapering

- **Developing intuition**
  - Number of stages vs. fanout
  - Popular inverter chain example

Problem Statement

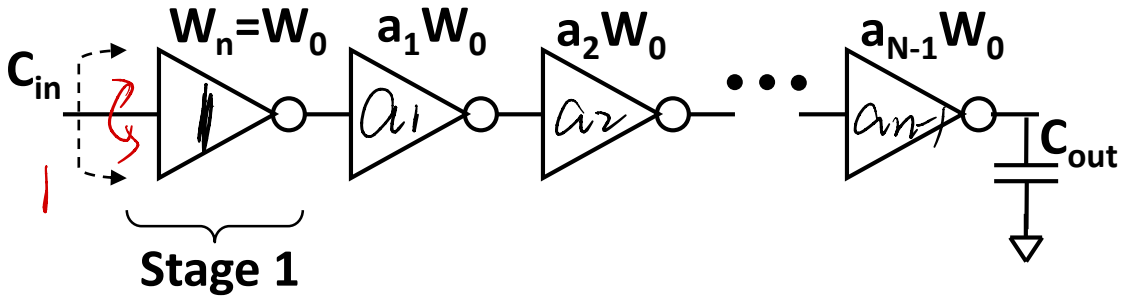


- **Constraints:**
  - C_{out} (load)
  - C_{in} (driver load)
  - Max delay

- **Given:**
  - Arbitrary logic function
  - Gate-level implementation

- **How do we decide the relative size of each gate?**

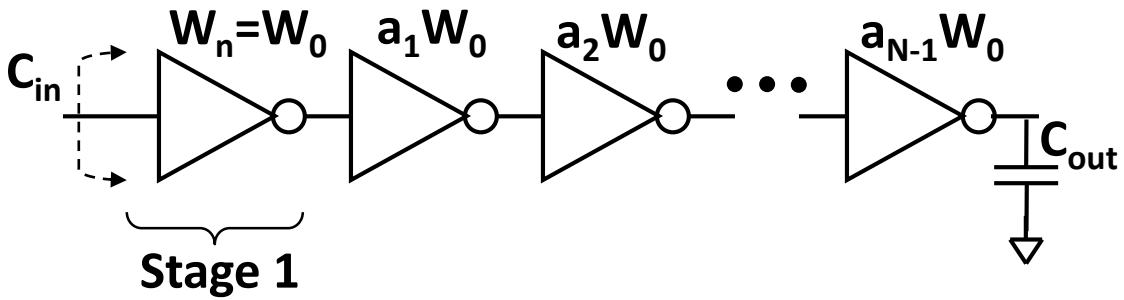
Simplified Problem: Buffering



- **Assume: β = k**
  - R_0 = Pull down for NMOS with size W_0 or PMOS with size βW_0
  - C_0 = Gate capacitance of N+PMOS of size W_0, βW_0
    - Ignore Source/Drain & Wire Capacitance for now
  - $\tau_0 = R_0C_0$

- **Goal: find $a_1, a_2, \ldots, a_{N-1}$ to minimize delay**

Delay Calculation



ignore C_{par}: $a_1 + \frac{a_2}{a_1} + \cdots + \frac{a_k}{a_{k-1}} + \cdots + \frac{C_{out}}{a_{out-1}}$

- **Delay of stage 1: $a_1C_0R_0 = a_1\tau_0$**
- **Delay of stage 2: $a_2C_0R_0/a_1 = a_2/a_1\tau_0$**

$a_k = \sqrt{a_{k+1} a_{k-1}}$

R_1, C_2

$\frac{\partial}{\partial a_k} = \frac{1}{a_{k-1}} - \frac{a_{k+1}}{a_k^2} \Rightarrow$

Optimal Fanout for Given N

- **Fanout calculation:**
  - Stage 1 = a_1, Stage 2 = a_2 / a_1
- **Assuming that the fanout of each stage is equal, a_0**
  - $a_1 = a_0$, $a_2 = a_0^2$, $a_3 = a_0^3$
  - $C_{out} = C_0 a_0^N$
- **Total Delay = Sum (Delay of stage 1:N)**
  - Delay = $\tau_0 N a_0$
- **Since $C_{in} = C_0$**
  - $C_{out} / C_{in} = a_0^N$

$$a_0 = \left(\frac{C_{out}}{C_{in}}\right)^{\frac{1}{N}}$$

Optimum Number of Stages

For an arbitrary N:

$$N = \frac{ln\left(\frac{C_{out}}{C_{in}}\right)}{ln(a_0)}$$

To a0 N

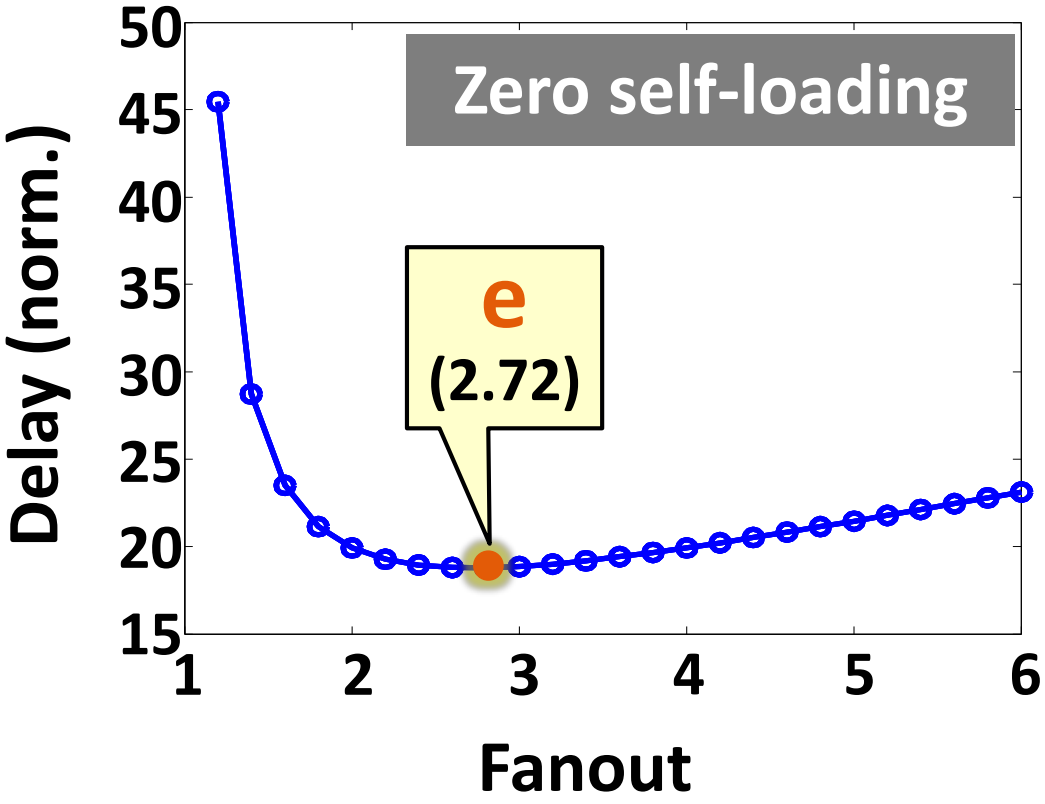
Min delay:

$$\frac{\partial Delay}{\partial a_0} = \frac{\partial\left[\tau_0 a_0 \frac{ln\left(\frac{C_{out}}{C_{in}}\right)}{ln(a_0)}\right]}{\partial a_0} = 0$$

$$\frac{\partial Delay}{\partial N}\frac{1}{\tau_0} = \left(\frac{C_{out}}{C_{in}}\right)^{\frac{1}{N}} - \frac{\left(\frac{C_{out}}{C_{in}}\right)^{\frac{1}{N}} ln\left(\frac{C_{out}}{C_{in}}\right)}{N} = 0$$

$$\Rightarrow \boxed{a_0 = e}$$

Optimum Number of Stages



Set number of stages to reach optimal fanout

Constant Fanout Per Stage?

Intuition:

What if we increase the size of a stage by (1+Δ)?

- $R_{DRV} \propto 1/(1+\Delta)$
- C_{Load} (previous stage) $\propto (1+\Delta)$

R reduces less quickly than C increases

Delay is summed and would increase

(Constant FO) Mathematically

- **Delay = $\tau_0(a_1 + a_2/a_1 + a_3/a_2 + a_4/a_3 + a_5/a_4 + \ldots)$**

- **$dDelay/da_1 = 0$**
  - $dDelay/da_1 = \tau_0(1 - a_2/a_1^2)$
  - So $a_1^2 = a_2$

- **$dDelay/da_2 = 0$**
  - $dDelay/da_2 = \tau_0(1/a_1 - a_3/a_2^2)$
  - So $a_2^2 = a_1 a_3$, thus $a_1^3 = a_3$

⇨ **Min delay: equal fanout**

Optimal Buffering with Self-Loading

- **Intuition: without self-loading**
  - Delay decreases proportional to the decrease in N
  - But increasing fanout increases delay proportionally
  - The two are **equal @ the optimal N and fanout**
- **Intuition: with self-loading**
  - Increasing FO no longer increases delay proportionally
    - Delay = $R_0(FO \cdot C_0 + C_{par})$
  - **New N would be less and FO is bigger**

(Buffering with Self-Loading) Mathematically

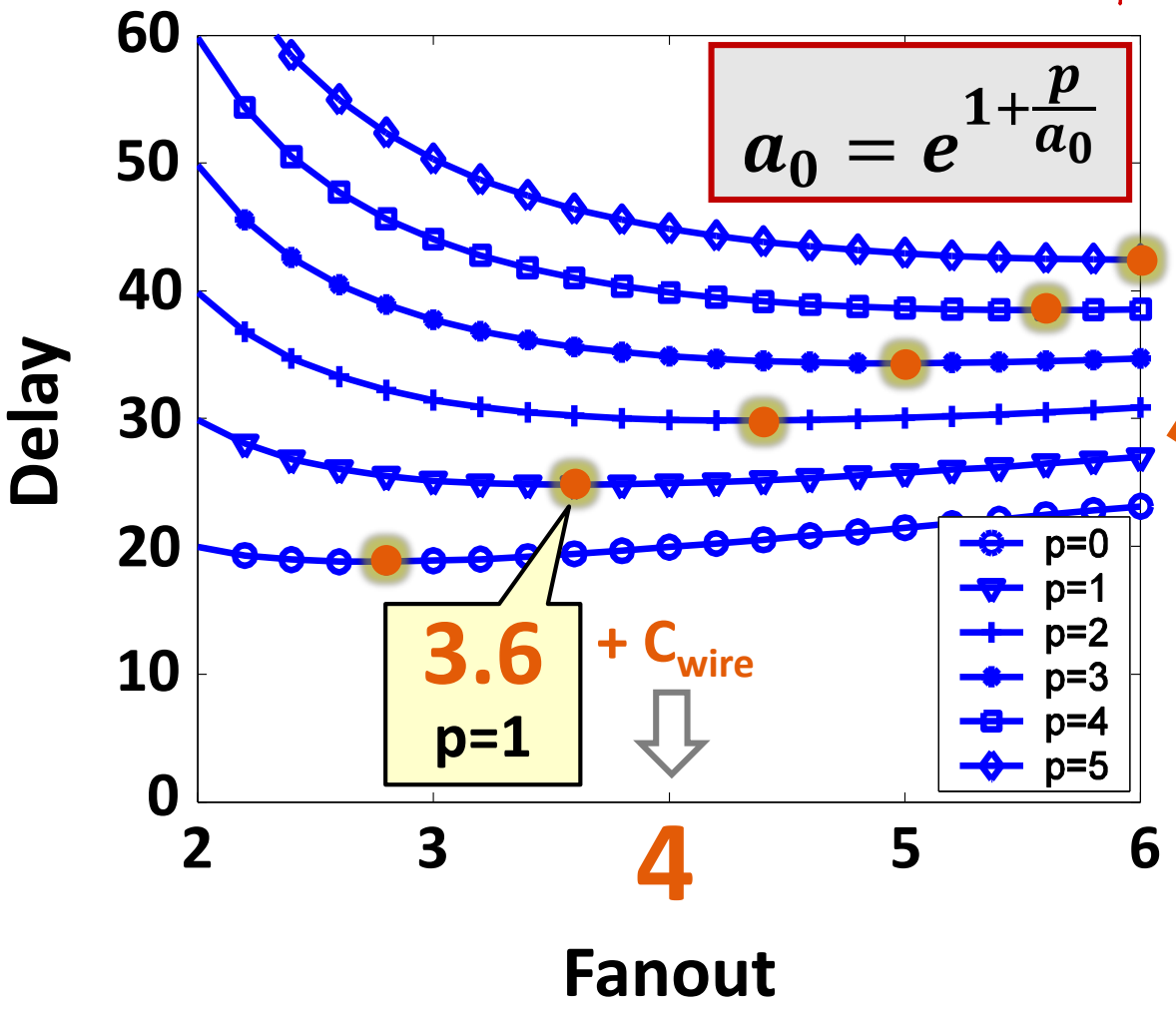
All equations remain the same except Delay

$$\frac{\partial Delay}{\partial a_0} = \frac{\partial\left[\tau_0\left(a_0 + \frac{C_{par}}{C_0}\right)\frac{ln\left(\frac{C_{out}}{C_{in}}\right)}{ln(a_0)}\right]}{\partial a_0} = 0$$

$$\Rightarrow \boxed{a_0 = e^{1+\frac{p}{a_0}}}$$

Optimal Buffering as *fn* (Self-Loading)

- A reasonable choice for optimal delay is **fanout of 4**



drive big load

Higher self-loading

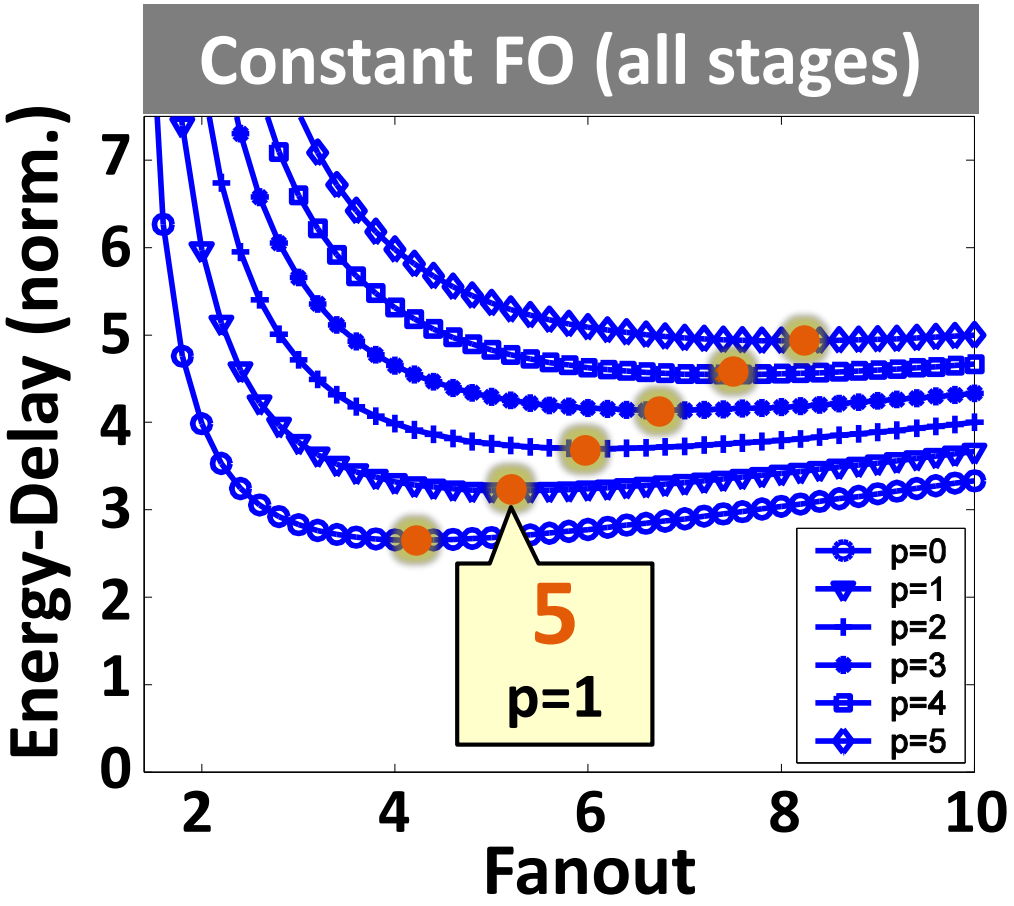
$p = C_{par} / C_{gate}$

Optimal FO for

$t_p > t_{p,min}$

Buffer Optimization for Energy-Delay

- **Minimizing for Energy is trivial:** min gate size
- **Instead minimize Energy-Delay product**



Result: larger FO

FO = 5 is pretty reasonable

Issue with Optimal Energy-Delay

- **Constant fanout is not a good assumption**
- **Intuition:**
  - Size of the final stage maters the most (use max fanout)
  - Reduce fanout of prior stages to compensate…
- **Example: C_{in} = 1, C_{out} = 1000, 4 gate stages**

Design	Gate 1	Gate 2	Gate 3	Gate 4	EDP
C_{in} | FO	1 | 5.6	5.6 | 5.6	31.6 | 5.6	177.8 | 5.6	32200
C_{in} | FO	1 | 4.8	4.8 | 4.9	23.1 | 5.4	124.5 | 8.0	31100

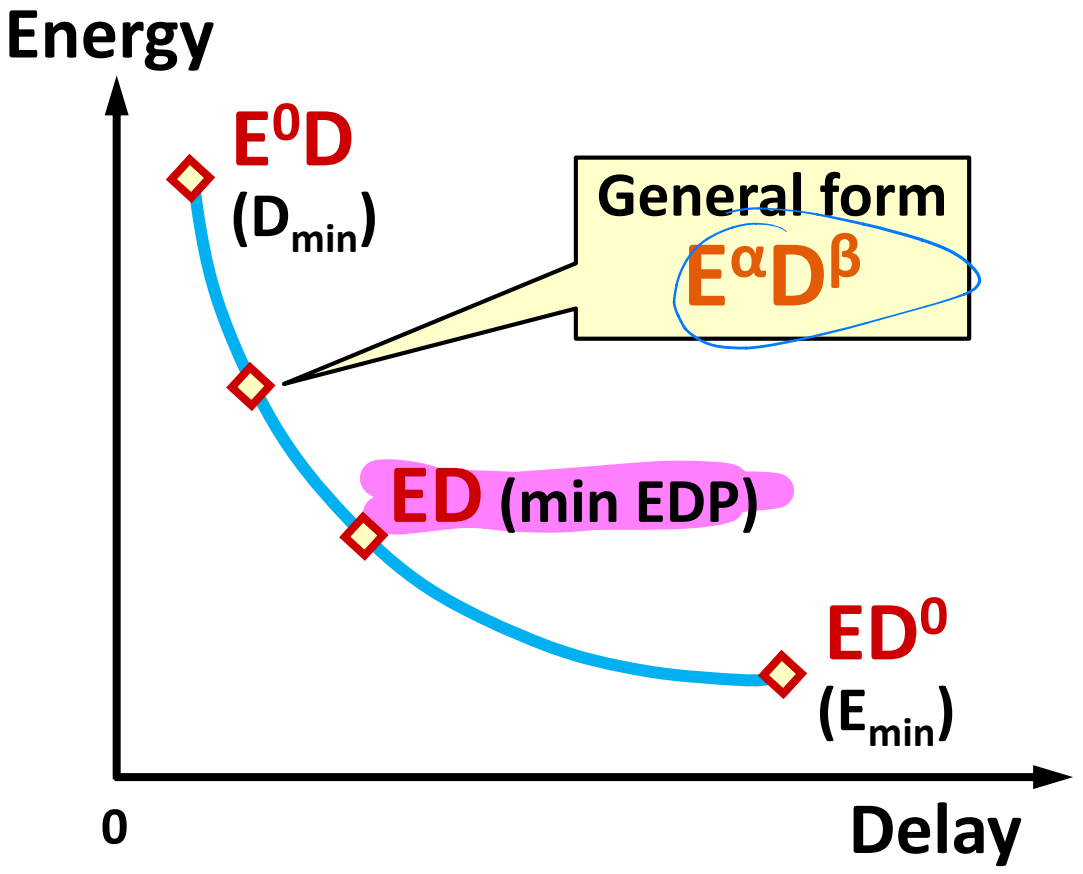
Tapered fanout reduces EDP

What if Fanout is Low?

- **Example:** large N (each stage drives small fanout)
  - Delay is logic limited, so reduce N
  - Balance Fanout so that they are equal
- **Try to adjust N such that each stage has FO ~ 4**
  - More complex logic for smaller N
- **OK, but not very systematic…**
  - Logical effort (next lecture) to formalize sizing

Energy-Delay Optimization (Lecture 14)

Variables: Gate size, Supply Voltage, Threshold Voltage



Summary

- **Delay of a logic network depends significantly on the relative size of logic gates**
  - Not transistors within a gate

- **Inverter buffering is a simple example of the analysis**
  - The analysis leads to FO ~ 4 as being optimal fanout for driving larger capacitive loads
  - The number of stages is optimized when FO ~ 4
  - For delays longer than minimum, tapered FO works best for minimizing power (more on this in Lec 14)