Design and Implementation of VLSI Systems
Lecture05

Delay

Delay

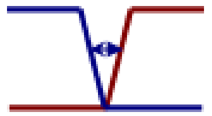
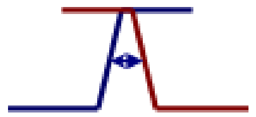
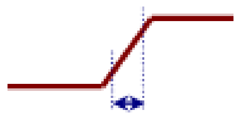
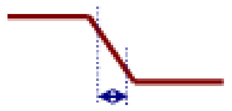
- Introduction
- Delay Estimation
- Logical Effort for Delay Estimation

Introduction

- Critical paths are those which require attention to timing details
- Timing analyzer is a design tool that automatically finds the slowest path in a logic design
  - Altera: Classic Timing Analyzer, TimeQuest Timing Analyzer
  - Synopsys: PrimeTime
- The critical paths can be affected at four main levels
  - The architecture/ micro-architecture level
  - The logic level
  - The circuit level
  - The layout level

Delay definitions

- tpdr: rising propagation delay
  - Max time: From input to rising output crossing VDD/2
- tpdf: falling propagation delay
  - Max time: From input to falling output crossing VDD/2
- tpd: average propagation delay. tpd = (tpdr + tpdf)/2
- tcdr: rising contamination (best-case) delay
  - Min time: From input to rising output crossing VDD/2
- tcdf: falling contamination (best-case) delay
  - Min time: From input to falling output crossing VDD/2
- tcd: average contamination delay. tcd = (tcdr + tcdf)/2
- tr: rise time - from output crossing 0.2 VDD to 0.8 VDD
- tf: fall time - from output crossing 0.8 VDD to 0.2 VDD

Delay definitions

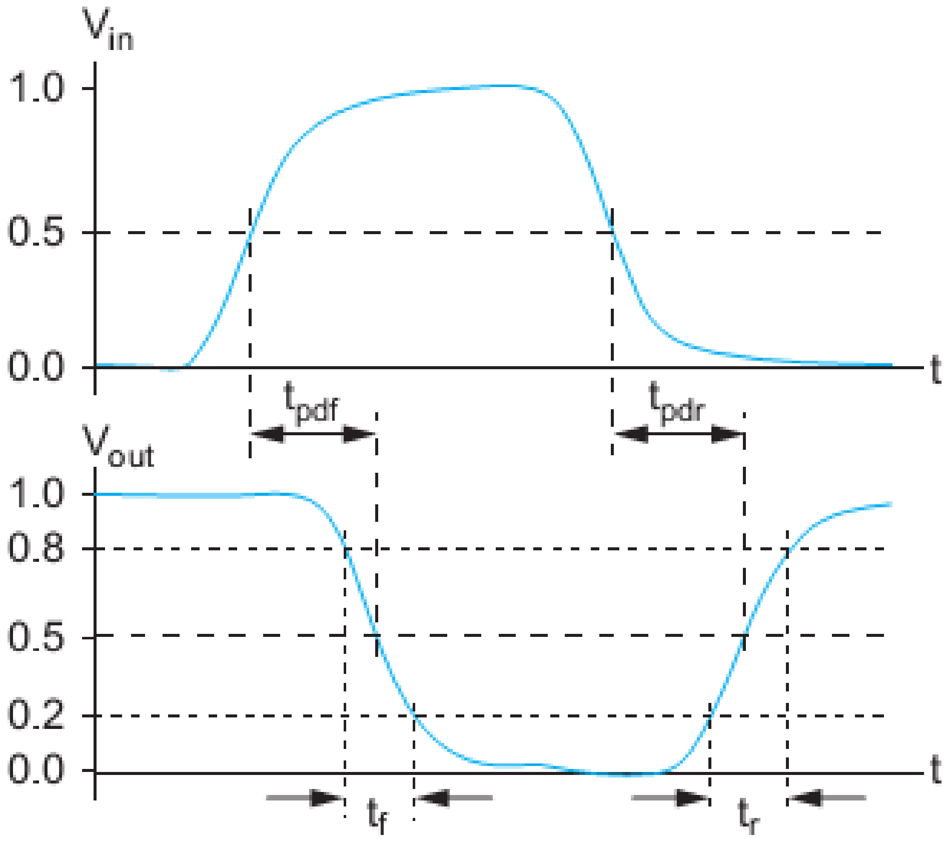


FIGURE 4.1 Propagation delay and rise/fall times

Delay definitions

- Driver: gate that charges or discharges a node
- Load: gates and wire being driven
- Delay: propagation delay
- Timing analyzer: computes the arrival times, i.e., the latest time at which each node in a block of logic will switch.
- The nodes: inputs, outputs, and internal nodes.
- The arrival time ai at internal node i depends on the propagation delay of the gate driving i and the arrival times of the inputs to the gate:

$$a_i = \max_{j \in fanin(i)} \left\{ a_j \right\} + t_{pd_i}$$

Delay definitions

- The timing analyzer computes the arrival times at each node and checks that the outputs arrive by their required time.
- The slack is the difference between the required and arrival times.
- Positive slack means that the circuit meets timing.
- Negative slack means that the circuit is not fast enough.
- Figure 4.2 shows nodes annotated with arrival times. If the outputs are all required at 200 ps, the circuit has 60 ps of slack.

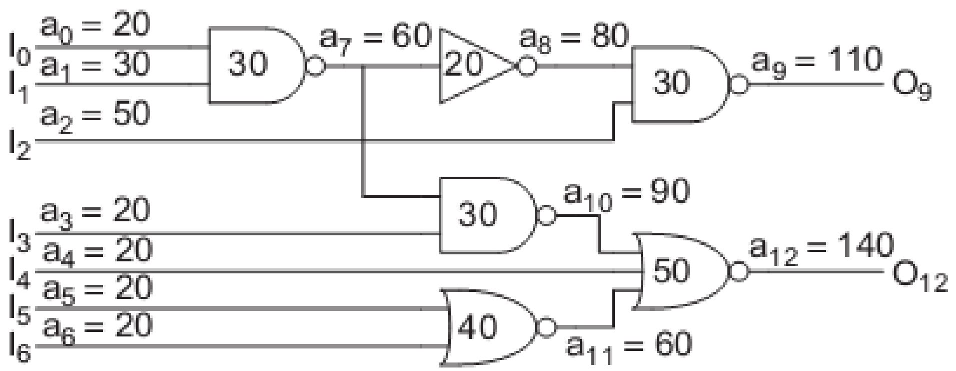


FIGURE 4.2 Arrival time example

Delay definitions

- There are a number of critical paths that limit the operating speed of the system and require attention to timing details.
- The critical paths can be affected at four main levels:
  - The architectural/micro architectural level
  - The logic level
  - The circuit level
  - The layout level

Delay definitions

- The most important: architecture or microarchitecture level
- Requires a broad knowledge on:
  - Algorithms that implement the function
  - Target technology: gate delays, how fast memories are accessed, how long signals take to propagate along a wire…
- Trade-offs at the micro architectural level include the number of pipeline stages, the number of execution units (parallelism), and the size of memories.

Delay definitions

- Logic level: the transformation from function to gates and registers can be done by experience, by experimentation, or, most often, by logic synthesis.
- Trade-offs include:
  - Types of functional blocks (e.g., ripple carry vs. look ahead adders)
  - Number of stages of gates in the clock cycle
  - Fan-in and fan-out of the gates.

Delay definitions

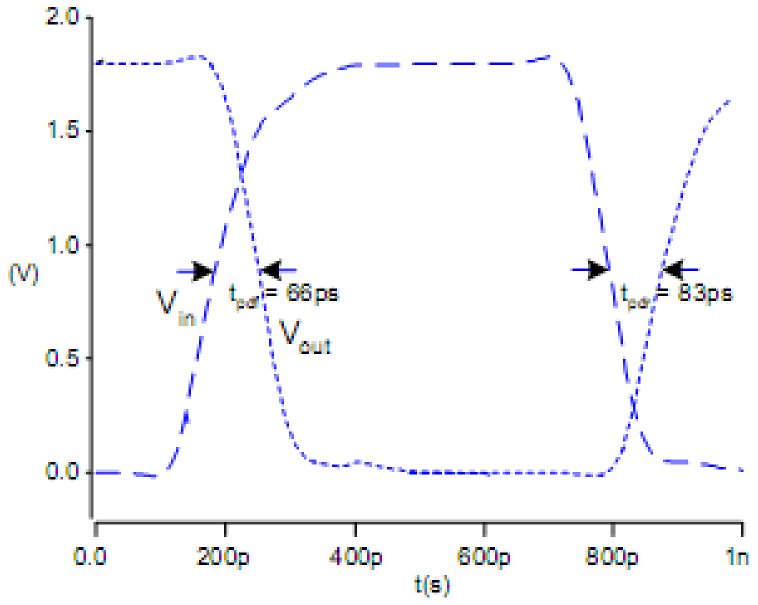
- Circuit level: by choosing transistor sizes or using other styles of CMOS logic.
- Layout level:
  - Floor plan (either manually or automatically generated) determines the wire lengths that can dominate delay.
  - Good cell layouts can also reduce parasitic capacitance

Delay definitions

- This chapter: logic and circuit optimizations
- Select number of stages of logic, the types of gates, and the transistor sizes.
- Examining the transient response of an inverter: use simple models that offer the designer more intuition – RC
- Method of Logical Effort simplifies the model even further and is a powerful way to evaluate delay in circuits.
- Other delay models used for timing analysis.

How to calculate delay

- Time consuming
- Not very useful for designers in evaluating different options and optimizing different parameters
- We need a simple way to estimate delay for “what if” scenarios.
- Fidelity vs. accuracy



Transistor resistance

$$R = \frac{\partial I_{ds}}{\partial V_{ds}}^{-1}$$

- In the linear region

$$I_{ds} \approx \beta(V_{gs} - V_t)V_{ds}$$
$$\rightarrow R \approx \frac{1}{\beta(V_{gs}-V_t)} = \frac{1}{\mu C_{ox}} \frac{L}{W} \frac{1}{(V_{gs}-Vt)}$$

- Not accurate, but at least shows that the resistance is proportional to L/W and decreases with Vgs

Switch level RC models

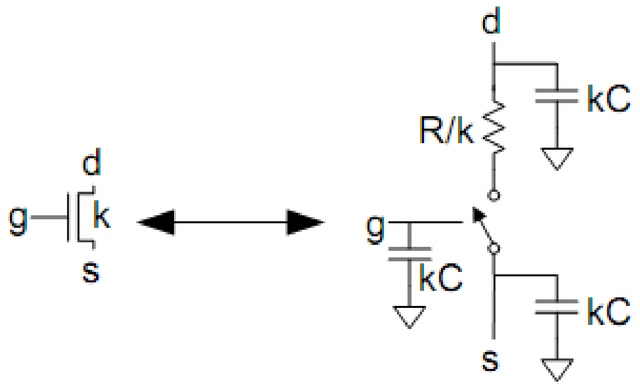
- An nMOS transistor with width of one unit is defined to have effective resistance R.
- The resistance of a pMOS transistor = 2-3× resistance of nMOS transistor of the same size due to the pMOS mobility.
- Wider transistors have lower resistance => a pMOS transistor of double-unit width has effective resistance R.
- R typically around 10K in this course
- Long-channel model: current decreases linearly with channel length
- However, if a transistor is fully velocity-saturated, current and resistance become independent of channel length.
- Real transistors operate somewhere between these two extremes.
- => neglect velocity saturation

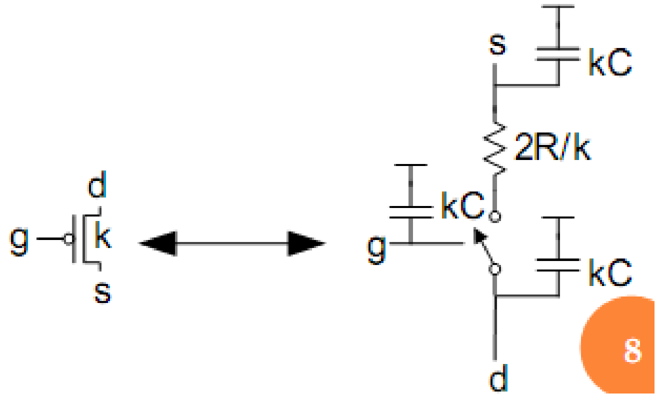
Switch level RC models

- Each transistor has gate and diffusion capacitance.
- Define C to be the gate capacitance of a unit transistor. A transistor of k times unit width has capacitance kC.
- Assume the contacted source or drain of a unit transistor to also have capacitance of about C.
- Wider transistors have proportionally greater diffusion capacitance.
- Increasing channel length increases gate capacitance proportionally but does not affect diffusion capacitance.
- Use a single average value for capacitances.
- Roughly estimate C for a minimum length transistor to be 1 fF/um of width.
- In a 65 nm process with a unit transistor being 0.1 um wide, C is thus about 0.1 fF

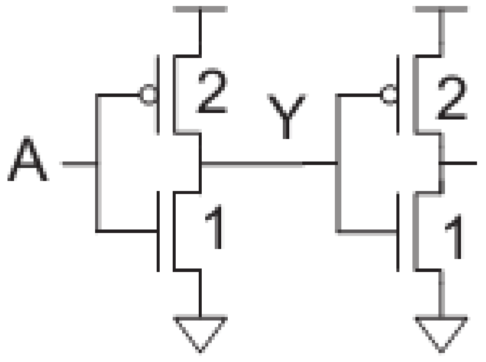
Switch level RC models

- Figure: equivalent RC circuit models for nMOS and pMOS transistors of width k with contacted diffusion on both source and drain
- A transistor of k unit width has kC capacitance and R/k resistance





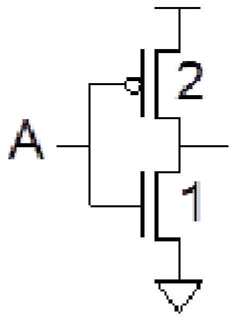
Switch level RC models

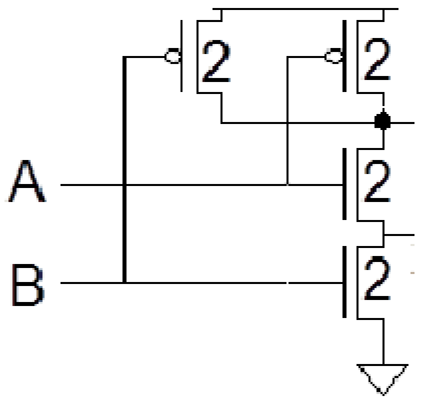


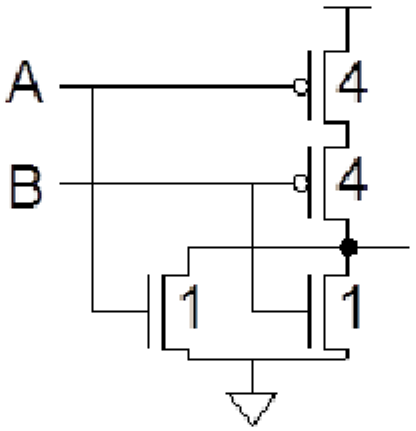
(a)

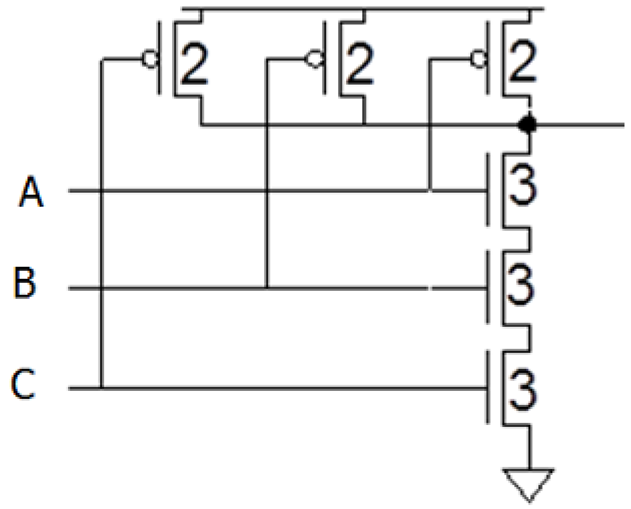
Draw equivalent circuit for the inverter

Calculate K









Example: 3-input NAND gate

- Sketch a 3-input NAND with transistor widths chosen to achieve effective rise and fall resistances equal to a unit inverter (R).

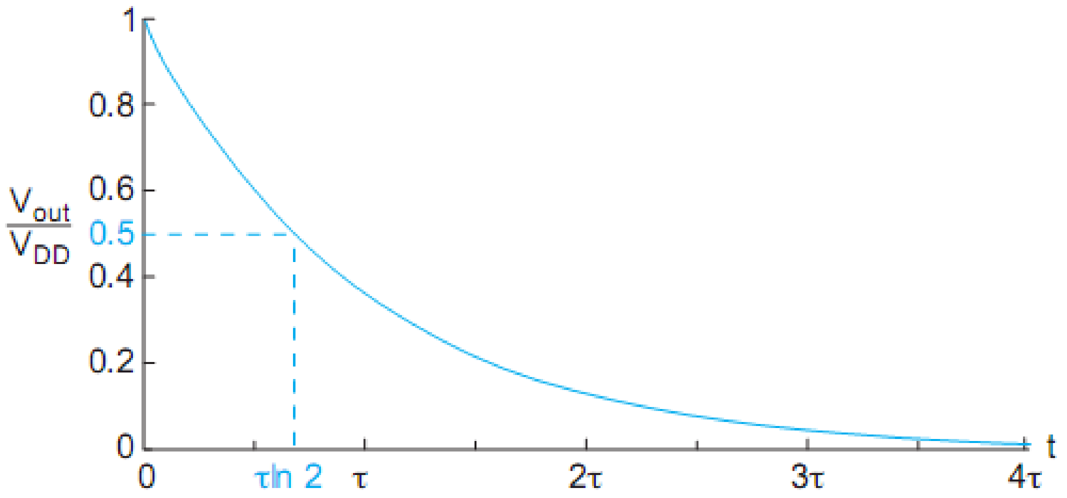
Example: 3-input NAND gate

- Worse cases

Falling Raising

Transient response

- Transfer function $H(s) = \frac{1}{1 + sRC}$
- Step response $V_{out}(t) = V_{DD} e^{-t/\tau}$
- The propagation delay: V_{out} reaches $V_{DD}/2$ $t_{pd} = RC \ln 2$
- Choose t_{pd} = RC, R effective
- First-order step response

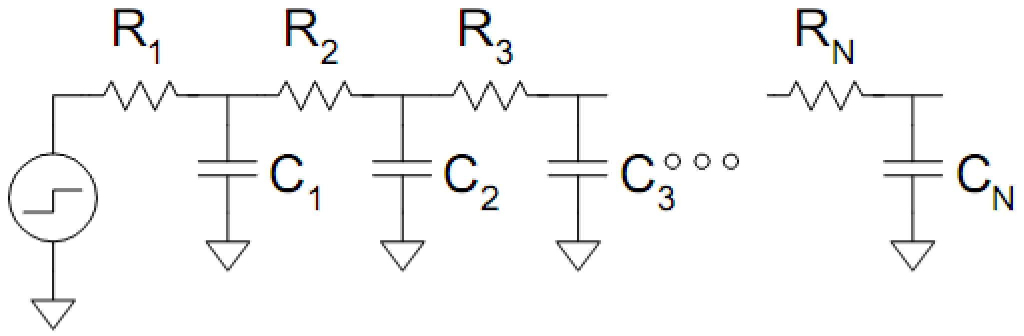


Elmore delay model

- ON transistors look like resistors
- Pullup or pulldown network modeled as RC ladder
- Elmore delay of RC ladder

$$t_{pd} \approx \sum_{\text{nodes } i} R_{i-to-source} C_i$$

$$= R_1 C_1 + \left(R_1 + R_2\right) C_2 + \ldots + \left(R_1 + R_2 + \ldots + R_N\right) C_N$$

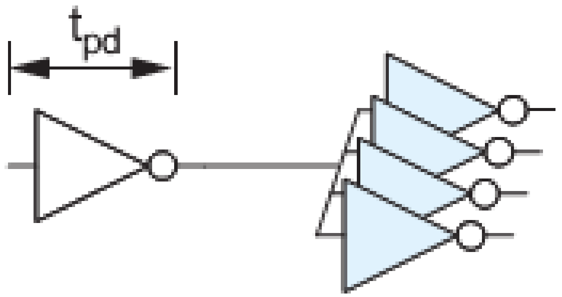


Compute raise and fall delay

- Estimate t_{pd} for a unit inverter driving m identical unit inverters.

Compute raise and fall delay

- If a unit transistor has R=10 kΩ and C= 0.1 fF in a 65 nm process, compute the delay, in picoseconds, of the inverter in Figure with a fanout of h=4.

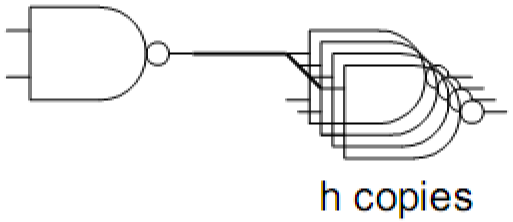


Compute raise and fall delay

- Estimate rising and falling propagation delays of a 2-input NAND driving h identical gates.

Compute raise and fall delay

- Estimate rising and falling propagation delays of a 2-input NAND driving h identical gates.

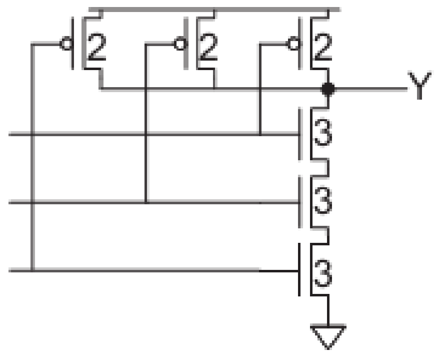


Compute raise and fall delay

- Estimate rising and falling propagation delays of a 3-input NAND driving h identical gates.

Compute raise and fall delay

- Estimate rising and falling propagation delays of a 3-input NAND driving h identical gates.

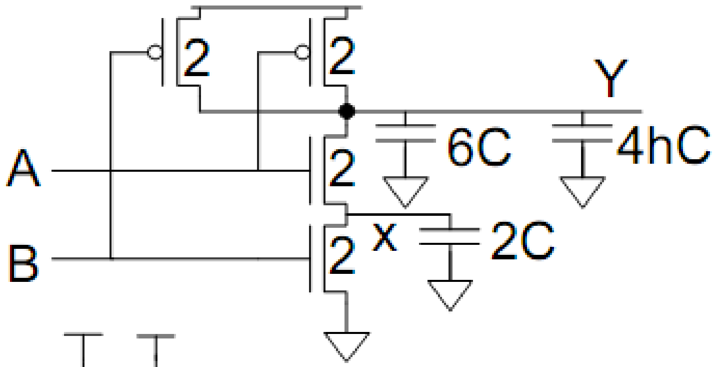


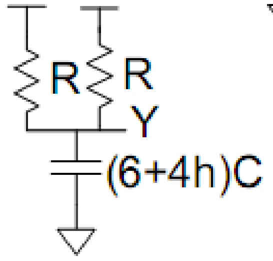
Compute raise and fall delay

- Estimate tpdf and tpdr for the 3-input NAND gate with transistor widths chosen to achieve effective rise and fall resistance equal to that of a unit inverter (R) if the output is loaded with h identical NAND gates.

Contamination delay

- Best-case (contamination) delay can be substantially less than propagation delay.
- Ex: If both inputs fall simultaneously
- Order of inputs also impact propagation delay. Which is better AB = 10 => 11 or AB = 01=>11?





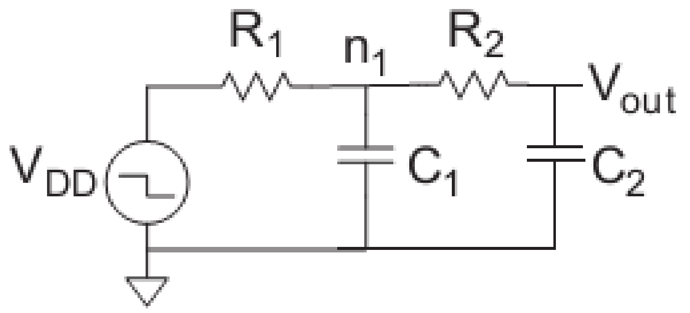
$$t_{cdr} = (3 + 2h)RC$$

Elmore delay

- Most circuits can be represented as an RC tree
- The root of the tree is the voltage source and the leaves are the capacitors at the ends of the branches.
- The Elmore delay model estimates the delay from a source switching to one of the leaf nodes changing as the sum over each node i of the capacitance C_i on the node, multiplied by the effective resistance R_{is} on the shared path from the source to the node and the leaf.

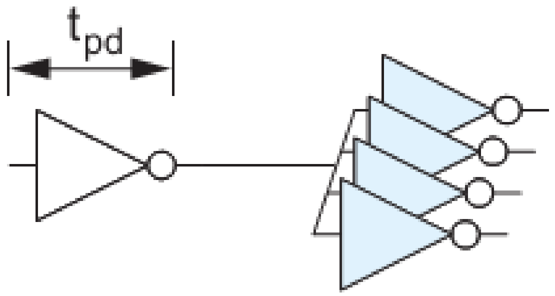
$$t_{pd} = \sum_i R_{is} C_i$$

Elmore delay



- Example:
- Compute the Elmore delay for Vout in the 2nd order RC system from Figure above

Elmore delay



Example:

- If a unit transistor has R=10 kΩ and C= 0.1 fF in a 65 nm process, compute the delay, in picoseconds, of the inverter with a fanout of h=4.

Elmore delay

Example:

- Estimate t_{pdf} and t_{pdr} for 3-input NAND gate from if the output is loaded with h identical NAND gates.

Elmore delay

Example:

- Estimate contamination delays t_{cdf} and t_{cdr} for the 3-input NAND gate if the output is loaded with h identical NAND gates

Diffusion capacitance

- In a good layout, diffusion nodes are shared wherever possible to reduce the diffusion capacitance.
- Uncontacted diffusion nodes between series transistors are usually smaller than those that must be contacted. => have less capacitance
- Estimate capacitances before layout: assume uncontacted diffusion between series transistors and contacted diffusion on all other nodes.
- Example: NAND3 layout shares one diffusion contact

Diffusion capacitance

- Example: NAND3 layout shares one diffusion contact

Diffusion capacitance

- The diffusion capacitance can be decreased by folding wide transistors.
- Figure 4.18(a): conventional layout of a 24/12 λ inverter.
- A unit (4 λ) transistor has diffusion capacitance C => inverter has a total diffusion capacitance of 9C.
- The folded layout in Figure 4.18(b) constructs each transistor from two parallel devices of half the width.
- The diffusion area has shrunk by a factor of two, reducing the diffusion capacitance to 4.5C.

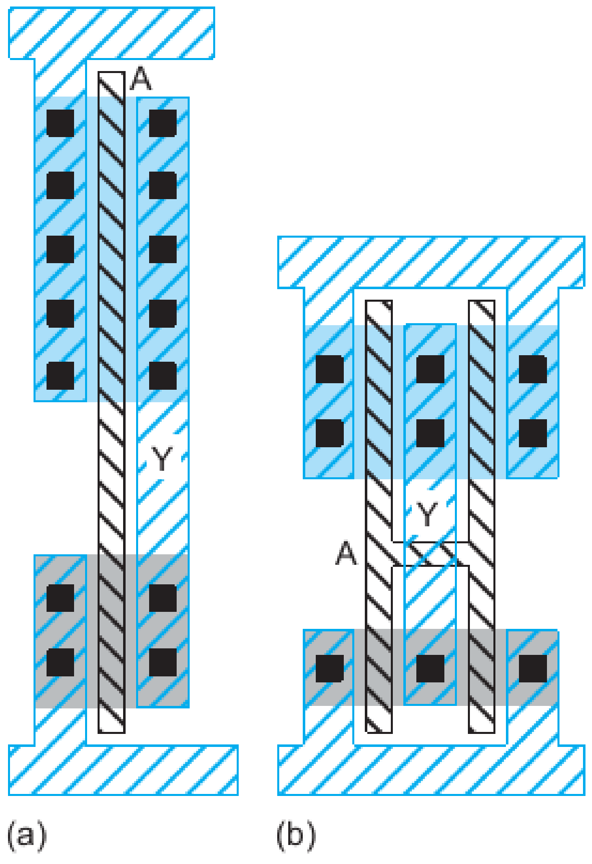


FIGURE 4.18 Layout styles: (a) conventional, (b) folded

Diffusion capacitance

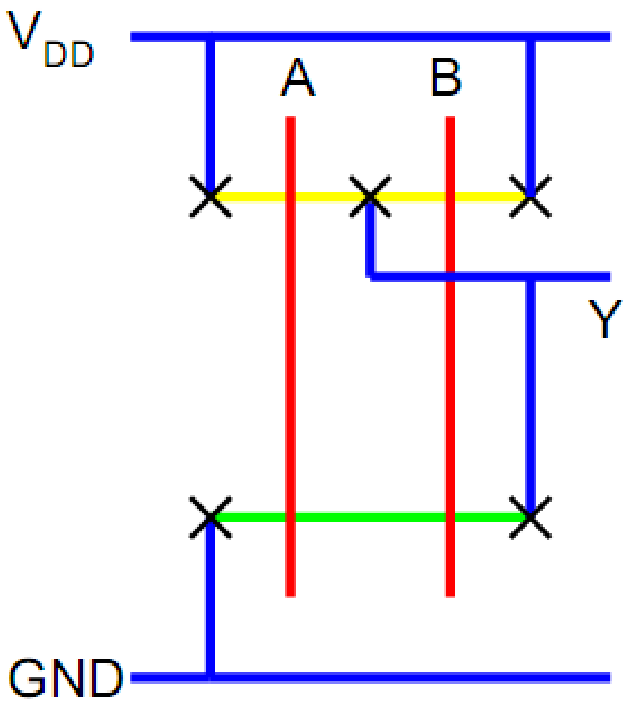
- In general, folded layouts:
  - Offer lower parasitic delay than unfolded layouts.
  - Fit better in a standard cell of limited height
  - Have lower resistance since have shorter polysilicon lines .

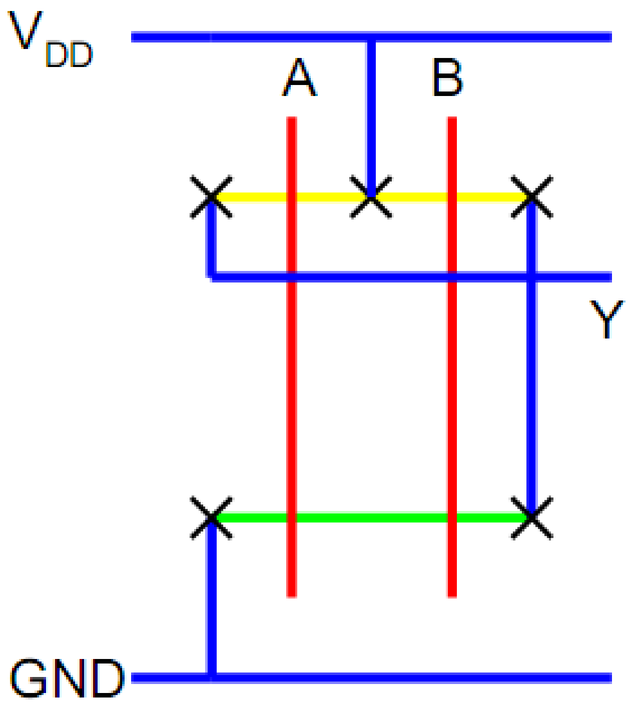
=> wide transistors are folded whenever possible.

- In some nanometer processes (generally 45 nm and below), transistor gates are restricted to a limited choice of pitches to improve manufacturability and reduce variability.
- Moreover, using a single standard transistor width may reduce variability.

Layout comparison

- Which layout is better?





Circuit characterization and performance estimation

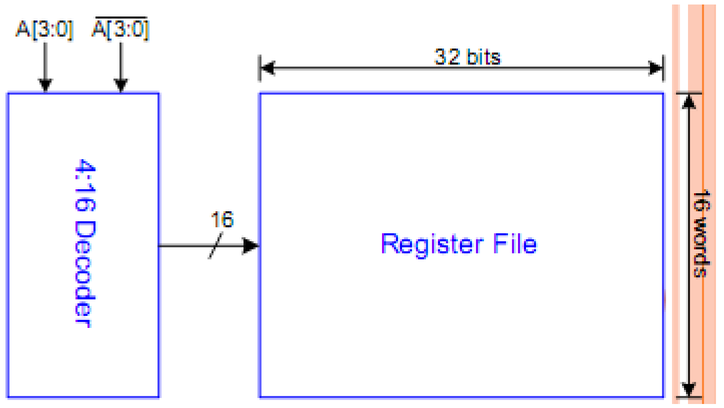
- Delay Estimation
- Logical Effort for Delay Estimation

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

Introduction

- 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?



Delay in a logic gate

- Normalize delay has two components: $d = f + p$
- f: effort delay or stage effort, depend on complexity and fanout of gate
- $f = gh$
- g: logical effort
  - Represent complexity
  - More complex gates have greater logical efforts => take longer to drive a given fanout.
  - For example, the logical effort of the 3-input NAND gate from the previous example is 5/3.

Delay in a logic gate

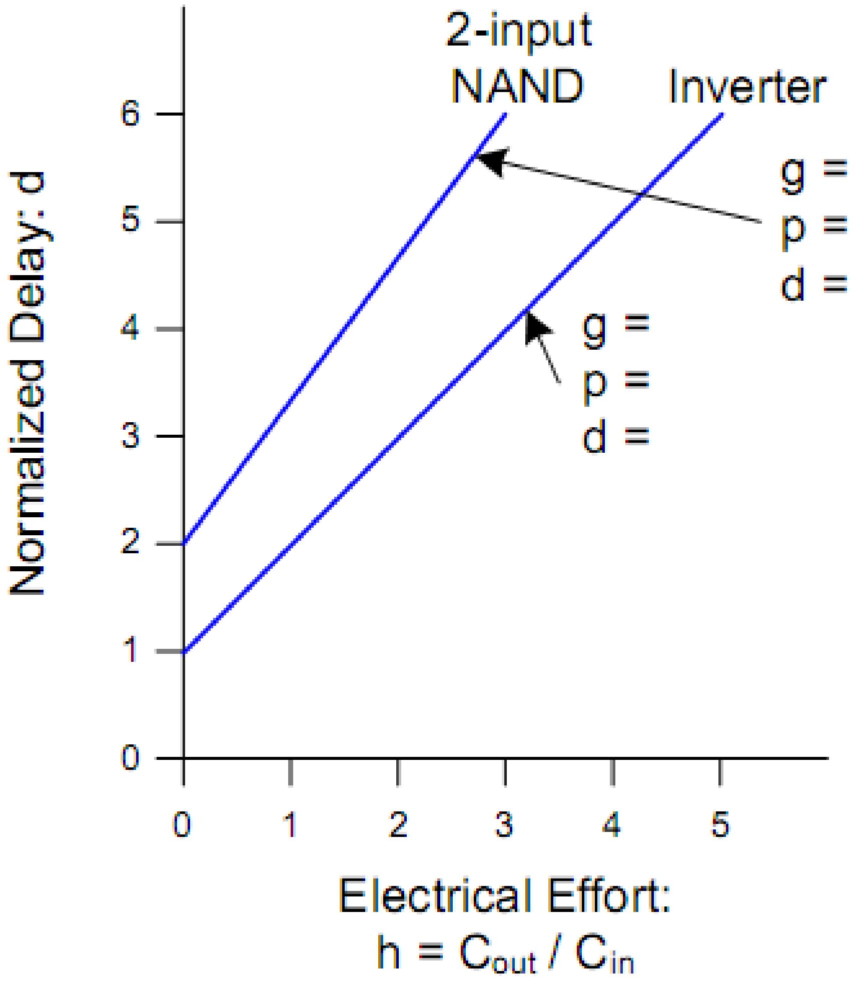
- A gate driving h identical copies of itself is said to have a fanout or electrical effort of h.
- If the load does not contain identical copies of the gate, the electrical effort can be computed as:

$$H = C_{out}/C_{in}$$

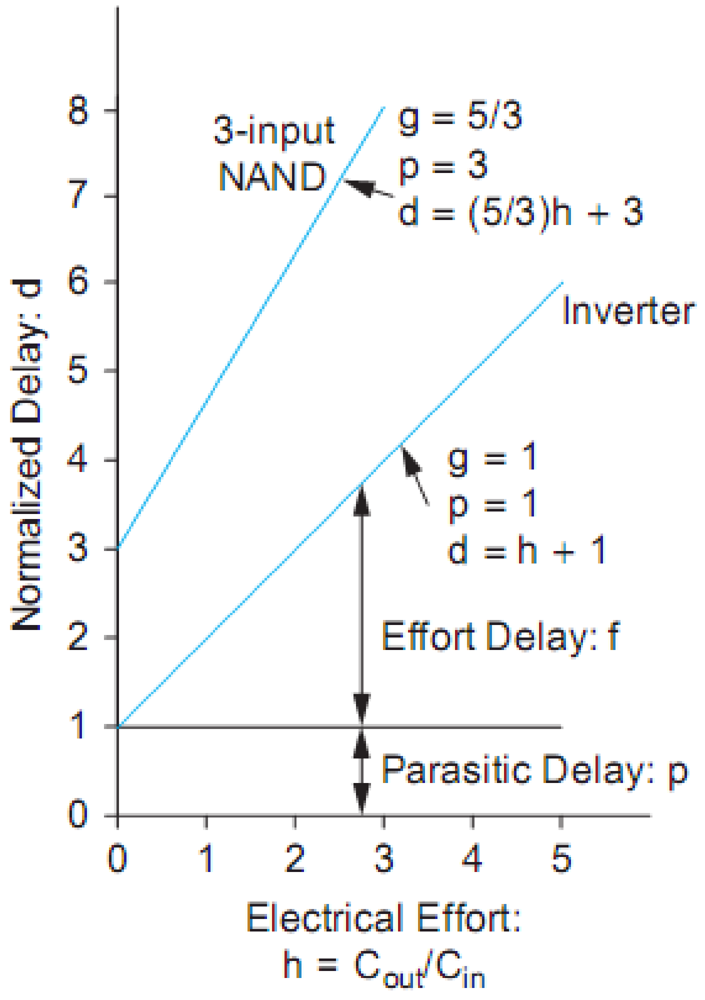
  - Cout: capacitance of external load being driven
  - Cin: input capacitance of the gate
  - g Ξ1 for inverter
- p: parasitic delay
  - Represents delay of gate driving no load
  - Set by internal parasitic capacitance

Delay plot

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

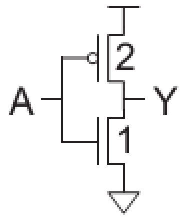


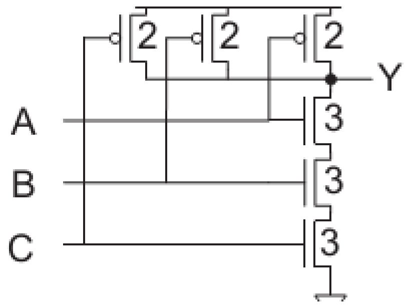
Delay plot

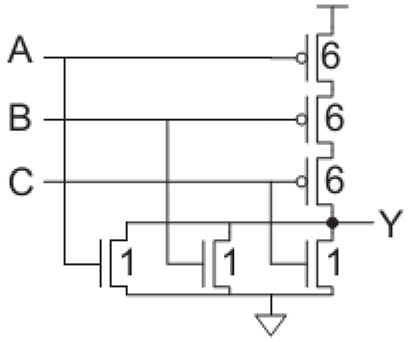

Normalized Delay: d
0
1
2
3
4
5
6
7
8
3-input
NAND
g = 5/3
p = 3
d = (5/3)h + 3
Inverter
g = 1
p = 1
d = h + 1
Effort Delay: f
Parasitic Delay: p
0
1
2
3
4
5
Electrical Effort:
$h = C_{out}/C_{in}$

Compute logical effort

- Inverter presents three units of input capacitance.
- NAND presents 5C on each input => logical effort = 5/3
- NOR presents 7C => logical effort = 7/3.
- This matches our expectation that NANDs are better than NORs because NORs have slow pMOS transistors in series

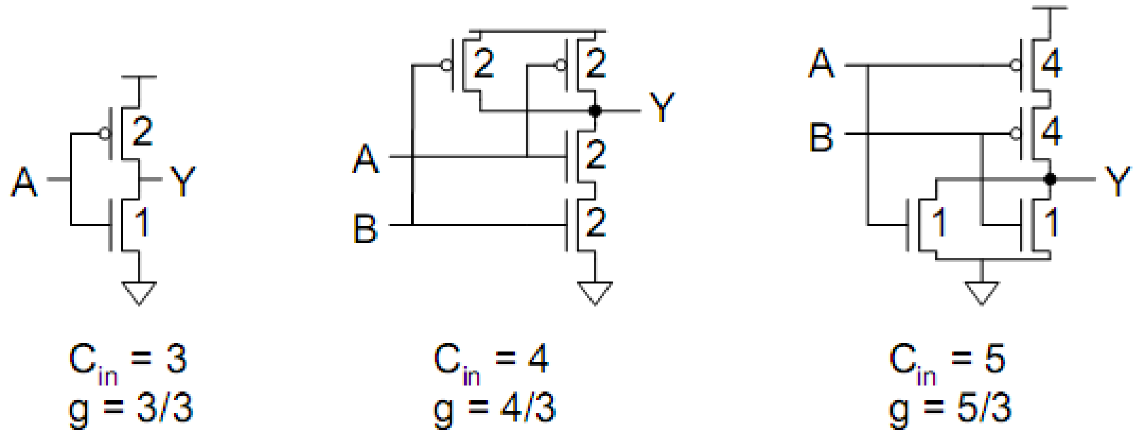




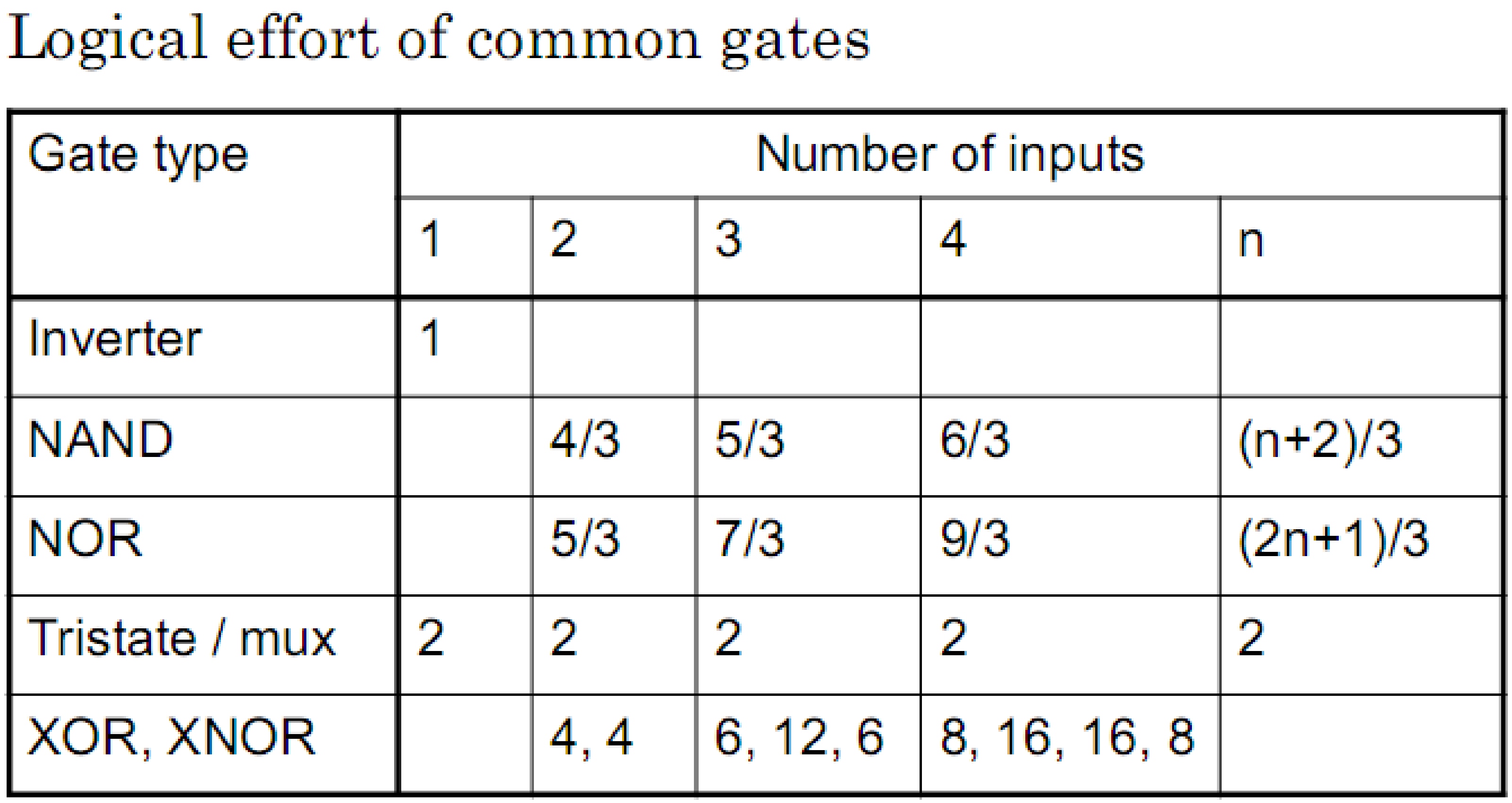


Compute logical effort

- Inverter presents three units of input capacitance.
- NAND presents 5C on each input => logical effort = 5/3
- NOR presents 7C => logical effort = 7/3.
- This matches our expectation that NANDs are better than NORs because NORs have slow pMOS transistors in series



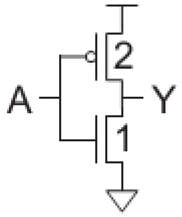
Logical effort



Logical effort of common gates

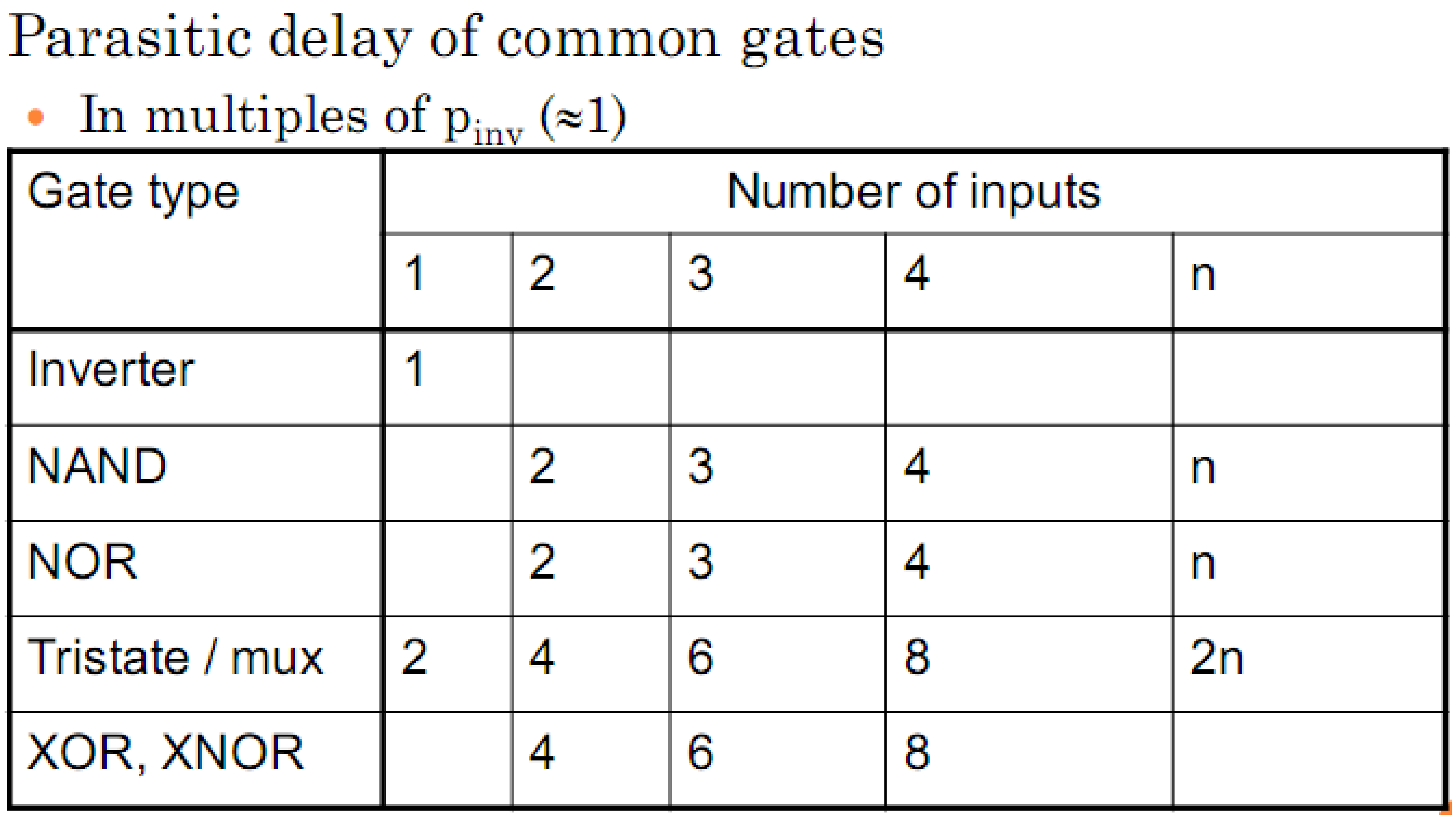
Gate type	Number of inputs				
	1	2	3	4	n
Inverter	1				
NAND		4/3	5/3	6/3	(n+2)/3
NOR		5/3	7/3	9/3	(2n+1)/3
Tristate / mux	2	2	2	2	2
XOR, XNOR		4, 4	6, 12, 6	8, 16, 16, 8	

Parasitic delay



- Parasitic delay of a gate is delay of the gate when it drives zero load.
- It can be estimated with RC delay models.
- Hand calculations: count diffusion capacitance on output node.
- For example:
  - Gates in Figure, each transistor on the output node has its own drain diffusion contact.
  - Transistor widths chosen to give a resistance of R in each gate.
  - Inverter has three units of diffusion capacitance on the output, so the parasitic delay is 3RC= τ.
- Call normalized parasitic delay p_{inv}
- p_{inv} is the ratio of diffusion capacitance to gate capacitance in a particular process.
- It is usually close to 1 and will be considered to be 1 in many examples for simplicity.
- 3-input NAND and NOR each have 9 units of diffusion capacitance on the output => parasitic delay is $3p_{inv}$ or simply 3.

Parasitic delay



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	

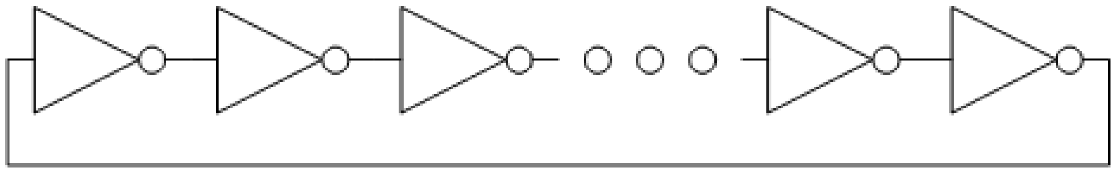
parasitic delay of common gates

Diffusion capacitance

Example:

- Use the linear delay model to estimate the delay of the fanout-of-4 (FO4) inverter. Assume the inverter is constructed in a 65 nm process with Y=3 ps.

Ring oscillator



Example:

- Estimate the frequency of an N-stage ring oscillator, as shown in Figure. N is odd.

Solution: (continue)

- An N-stage ring oscillator has a period of 2N stage delays because a value must propagate twice around the ring to regain the original polarity.

=> the period is T=2 ×2N.

- The frequency = 1/4N.
- A 31-stage ring oscillator in a 65 nm process has a frequency of 1/(4 ×31 ×3 ps) = 2.7 GHz.

F04 inverter

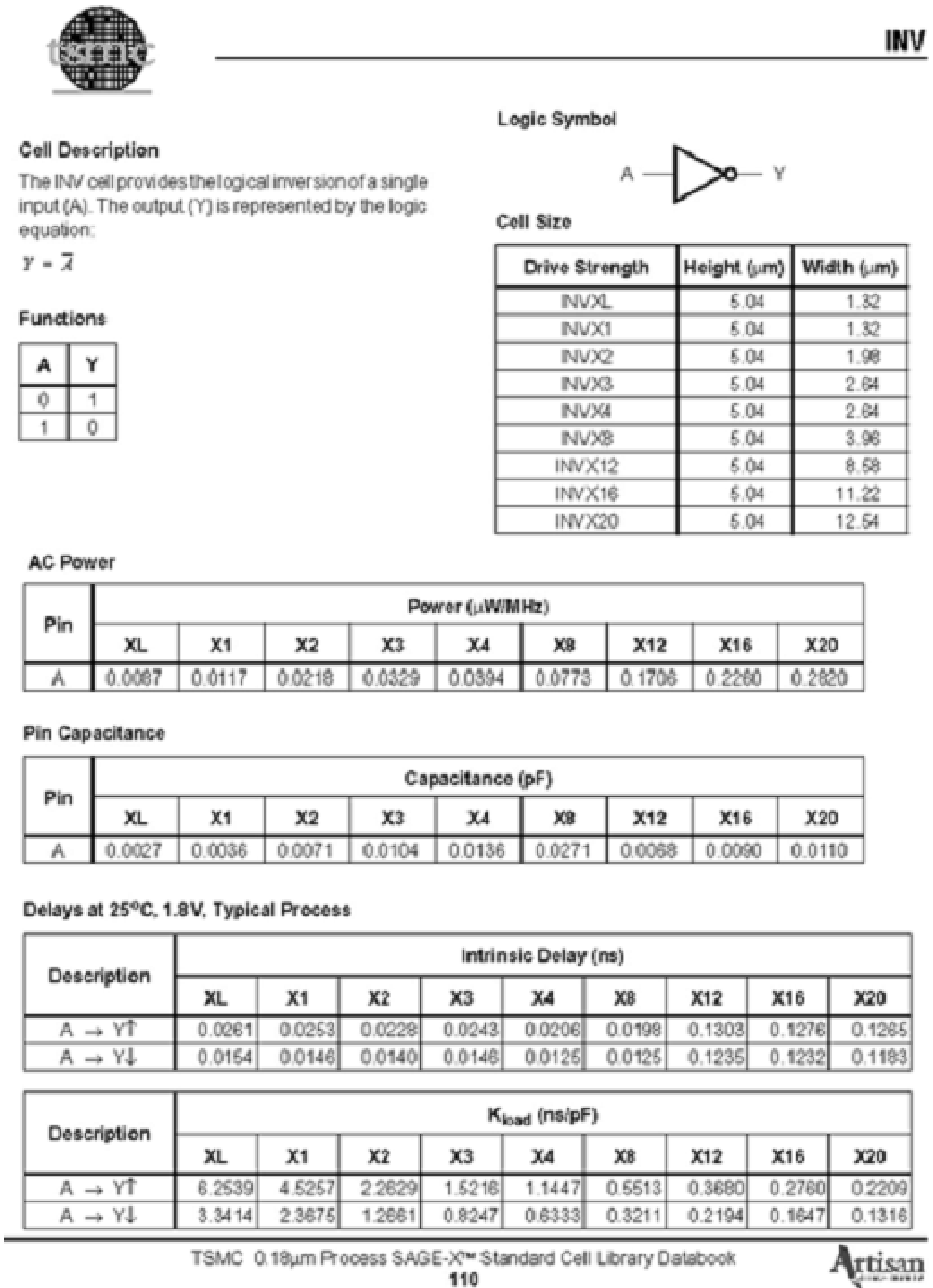
Example:

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

Extracting Logical Effort from Datasheets

- Using a standard cell library, you can often extract logical effort of gates directly from the datasheets.
- Next slides shows INV and NAND2 datasheets from Artisan Components library for the TSMC 180 nm process.
- The gates in the library come in various drive strengths.
- INVX1 is the unit inverter; INVX2 has twice the drive. INVXL has the same area as the unit inverter but uses smaller transistors to reduce power consumption on noncritical paths.
- The X12–X20 inverters are built from three stages of smaller inverters to give high drive strength and low input capacitance at the expense of greater parasitic delay.

Artisan Components cell library datasheets



INV

Cell Description

The INV cell provides the logical inversion of a single input (A). The output (Y) is represented by the logic equation:

$Y = \overline{A}$

Functions

A	Y
0	1
1	0

Logic Symbol

A — Y

Cell Size

Drive Strength	Height (µm)	Width (µm)
INVXL	5.04	1.32
INVX1	5.04	1.32
INVX2	5.04	1.98
INVX3	5.04	2.64
INVX4	5.04	2.64
INVX8	5.04	3.96
INVX12	5.04	6.58
INVX16	5.04	11.22
INVX20	5.04	12.54

AC Power

Pin	Power (µW/MHz)								
	XL	X1	X2	X3	X4	X8	X12	X16	X20
A	0.0067	0.0117	0.0218	0.0329	0.0394	0.0773	0.1706	0.2260	0.2620

Pin Capacitance

Pin	Capacitance (pF)								
	XL	X1	X2	X3	X4	X8	X12	X16	X20
A	0.0027	0.0036	0.0071	0.0104	0.0136	0.0271	0.0068	0.0090	0.0110

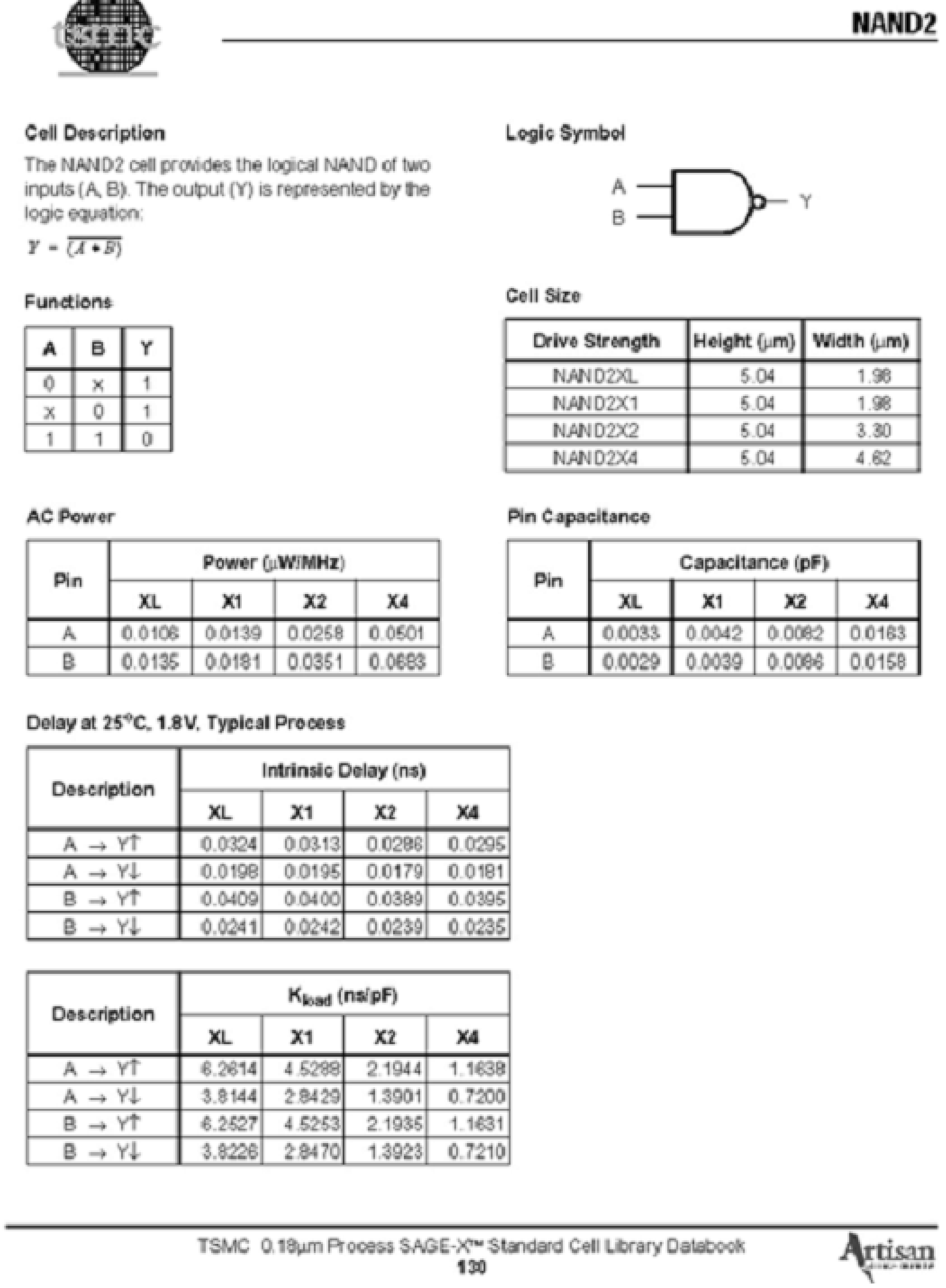
Delays at 25°C, 1.8V, Typical Process

Description	Intrinsic Delay (ns)								
	XL	X1	X2	X3	X4	X8	X12	X16	X20
A → Y↑	0.0261	0.0253	0.0228	0.0243	0.0206	0.0198	0.1303	0.1276	0.1265
A → Y↓	0.0164	0.0146	0.0140	0.0148	0.0126	0.0125	0.1235	0.1232	0.1183

Description	K_{load} (ns/pF)								
	XL	X1	X2	X3	X4	X8	X12	X16	X20
A → Y↑	6.2539	4.5257	2.2629	1.5216	1.1447	0.5513	0.3680	0.2760	0.2209
A → Y↓	3.3414	2.3675	1.2681	0.8247	0.6333	0.3211	0.2194	0.1647	0.1316

TSMC 0.18µm Process SAGE-X™ Standard Cell Library Databook
110

Artisan Components cell library datasheets



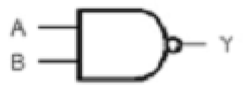
NAND2

Cell Description

The NAND2 cell provides the logical NAND of two inputs (A, B). The output (Y) is represented by the logic equation:

$Y = \overline{(A \bullet B)}$

Logic Symbol



Functions

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

Cell Size

Drive Strength	Height (μm)	Width (μm)
NAND2XL	5.04	1.98
NAND2X1	5.04	1.98
NAND2X2	5.04	3.30
NAND2X4	5.04	4.62

AC Power

Pin	Power (μW/MHz)			
	XL	X1	X2	X4
A	0.0106	0.0139	0.0258	0.0501
B	0.0135	0.0181	0.0351	0.0683

Pin Capacitance

Pin	Capacitance (pF)			
	XL	X1	X2	X4
A	0.0033	0.0042	0.0082	0.0163
B	0.0029	0.0039	0.0086	0.0158

Delay at 25°C, 1.8V, Typical Process

Description	Intrinsic Delay (ns)			
	XL	X1	X2	X4
A → Y↑	0.0324	0.0313	0.0286	0.0295
A → Y↓	0.0198	0.0195	0.0179	0.0181
B → Y↑	0.0409	0.0400	0.0389	0.0395
B → Y↓	0.0241	0.0242	0.0239	0.0235

Description	K_{load} (ns/pF)			
	XL	X1	X2	X4
A → Y↑	6.2614	4.5288	2.1944	1.1638
A → Y↓	3.8144	2.8429	1.3901	0.7200
B → Y↑	6.2527	4.5253	2.1935	1.1631
B → Y↓	3.8226	2.8470	1.3923	0.7210

Extracting Logical Effort from Datasheets

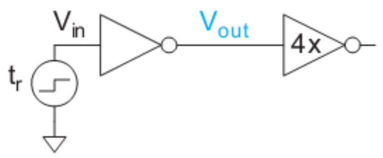
- From the datasheet, unit inverter C_{in}
- The rising and falling delays are specified separately.
- The average intrinsic or parasitic delay
- The slope of the delay vs. load capacitance curve is the average of the rising and falling K_{load} values. An inverter driving a fanout of h will thus have a delay of

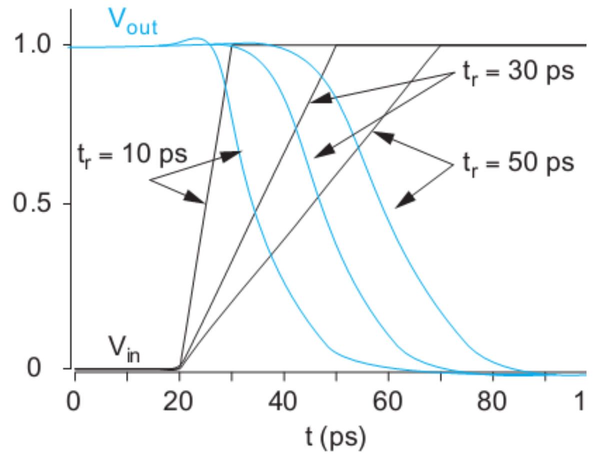
Extracting Logical Effort from Datasheets

- Slope of delay vs. fanout curve indicates Y
- Y-intercept indicates p_{inv} in normalized terms.
- This is larger than delay of 1 estimated earlier, probably because it includes capacitance of internal wires.
- By a similar calculation, we find the X1 2-input NAND gate has an average delay from the inner (A) input of
- Thus, the parasitic delay is and the logical effort is

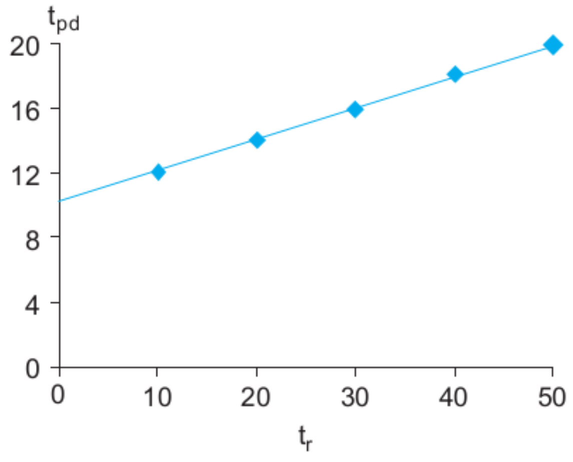
Limitations of linear delay model

- The largest source of error in the linear delay model is the input slope effect.
- Figure shows a fanout-of-4 inverter driven by ramps with different slopes.
- ON current increases with the gate voltage for an nMOS transistor.
- Transistor is OFF for Vgs<Vt, fully ON for Vgs=VDD, and partially ON for intermediate gate voltages.
- As the rise time of the input increases, the delay also increases because the active transistor is not turned fully ON at once.
- Figure (b) plots average inverter propagation delay vs. input rise time.





(a)



(b)

Limitations of linear delay model

- Input Arrival Times

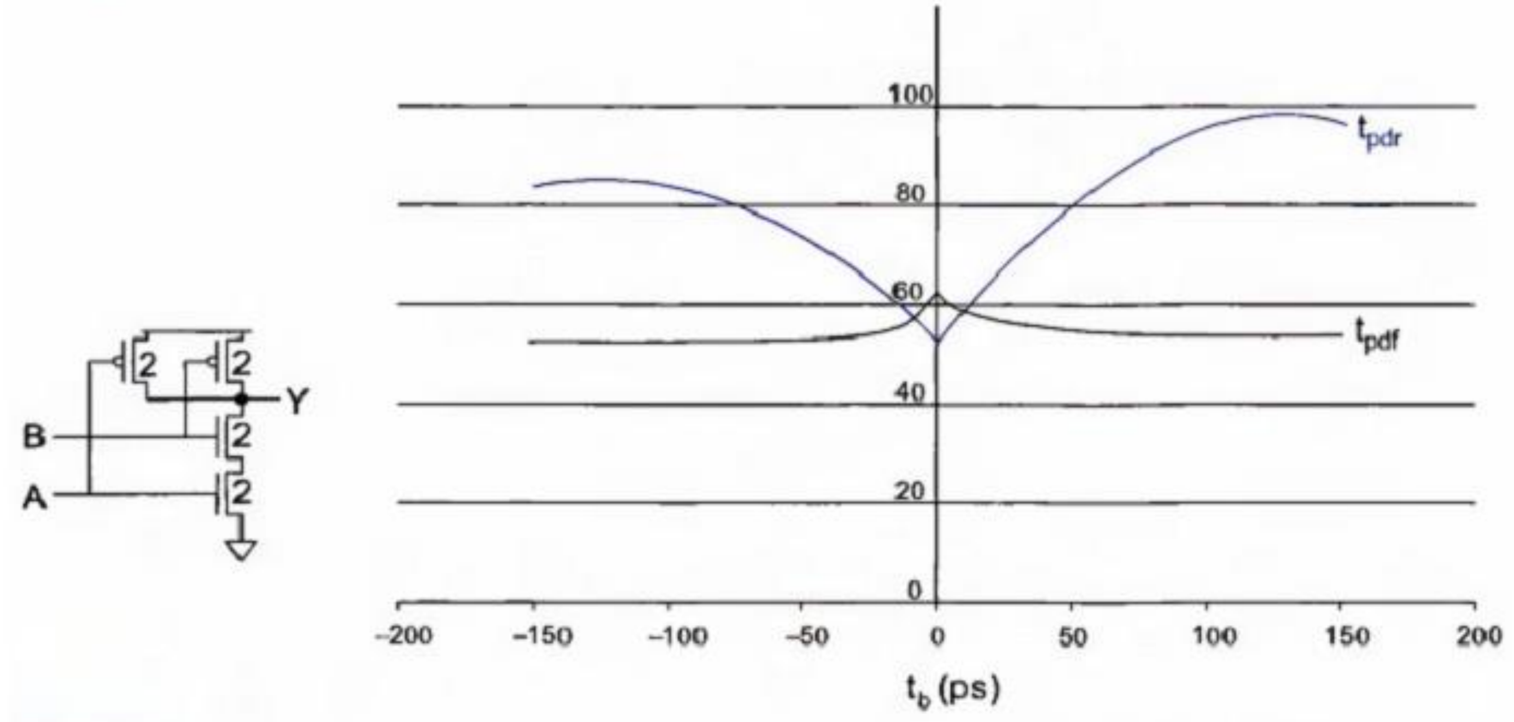


FIG 4.12 Delay sensitivity to input arrival time

- Source of error in the linear delay model: the assumption that one input of a multiple-input gate switches while others are completely stable.
- When two inputs to a series stack turn ON simultaneously, the delay will be slightly longer than predicted
- When two inputs to a parallel stack turn ON simultaneously, the delay will be shorter than predicted

Limitations of linear delay model

Velocity saturation:

- Estimated logical efforts assume that N transistors in series must be N times as wide to give equal current.
- However, series transistors see less velocity saturation and hence have lower resistance than we estimated
- N transistors in series = 1 transistor with N times channel length.
- Substituting L and NL into EQ (2.28) shows ratio of I_{dsat} for two series transistors to that of a single transistor:

$$\frac{I_{\text{dsat}-N-\text{series}}}{I_{\text{dsat}}} = \frac{\left(V_{DD} - V_t\right) + V_c}{\left(V_{DD} - V_t\right) + NV_c}$$

- In the limit that the transistors are not at all velocity saturated (Vc >>VDD-Vt), the current ratio reduces to 1/N as predicted.
- In the limit that the transistors are completely velocity saturated, the current is independent of the number of series transistors.

Limitations of linear delay model

Example:

- Determine the relative saturation current of 2- and 3-transistor nMOS and pMOS stacks in a 65 nm process. V_{DD}=1.0 V, V_t =0.3 V. Use V_c =E_cL=1.04 V for nMOS devices and 2.22 V for pMOS devices.

Limitations of linear delay model

Voltage dependence:

- Delay is proportional to CVDD/I and using the α-power law model, RC time constant:
  where k reflects process parameters.

$$\tau = k\frac{CV_{DD}}{\left(V_{DD} - V_t\right)^{\alpha}}$$

- Alternatively, using straight line saturation current model from EQ (2.32) for velocity-saturated transistors:

$$\tau = k\frac{CV_{DD}}{\left(V_{DD} - V_t^{*}\right)} = \frac{kC}{1 - \frac{V_t^{*}}{V_{DD}}}$$

=> supply voltage can be reduced without changing the delay of a velocity-saturated transistor so long as the threshold is reduced in proportion.

- When $V_{DD}<V_t$, delay instead depends on the subthreshold current:

$$\tau = k\frac{CV_{DD}}{I_{\text{off}}10^{\frac{V_{DD}}{S}}}$$

Limitations of linear delay model

- Gate-Source Capacitance

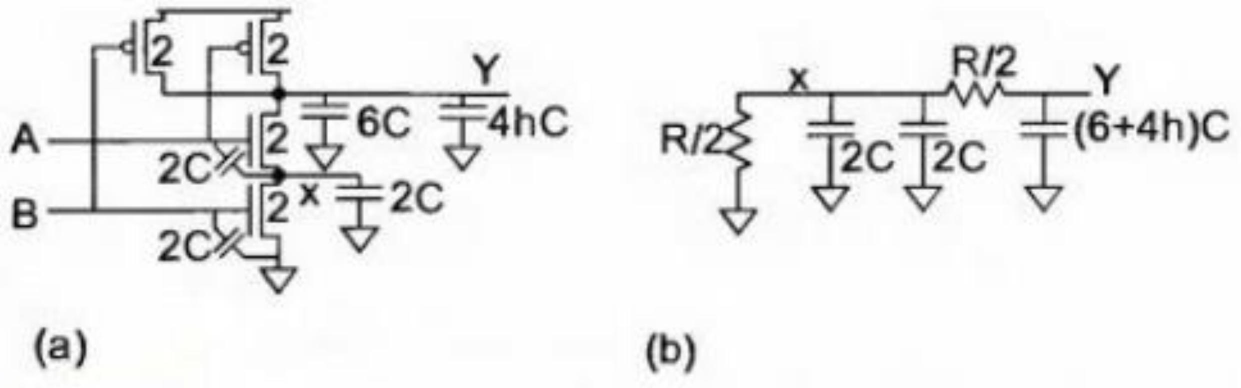
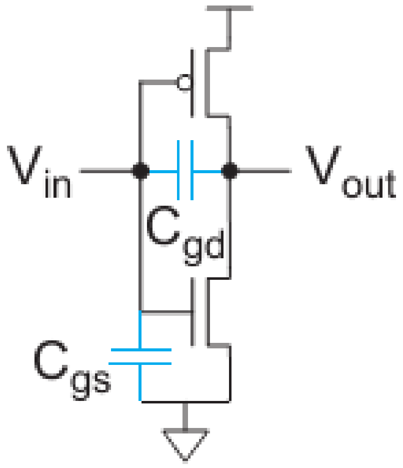


FIG 4.13 NAND gate delay estimation with gate-source capacitance modeled

- Bottom terminal of the gate oxide capacitor is the channel, which connected to the source when the transistor is ON.

=>as the source of a transistor changes value, charge is required to change the voltage on Cgs
=>more delay for series stacks.

Limitations of linear delay model



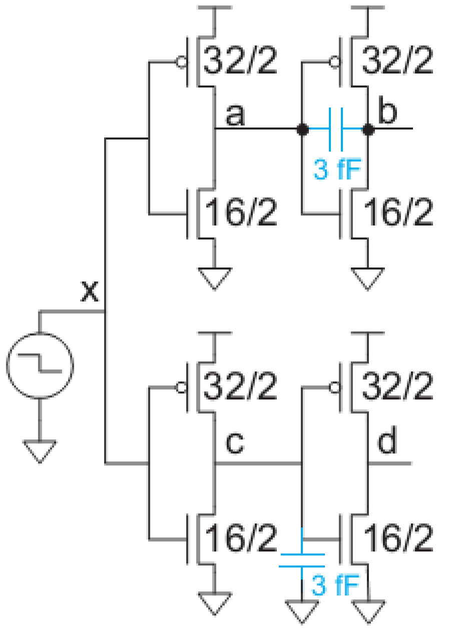
Bootstrapping:

- Transistors have capacitance from gate to drain.
- This capacitance couples the input and output in an effect known as bootstrapping
- Our models so far have only considered Cin (Cgs).
- This figure considers Cgd, the gate to drain capacitance.
- Input is rising (the output starts high), the effective input capacitance is Cgs+Cgd.
- When the output starts to fall, the voltage across Cgd changes, requiring the input to supply additional current to charge Cgd
- In other words, the impact of Cgd on gate capacitance is effectively doubled.

Limitations of linear delay model

Bootstrapping:

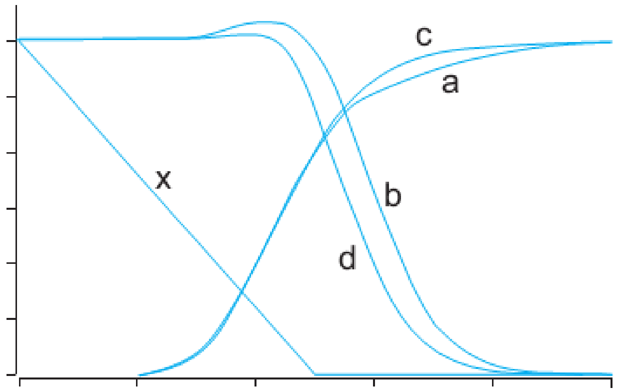
- Figure shows two inverter pairs.
- The top pair has an extra bit of capacitance between the input and output of the second inverter.
- The bottom pair has the same amount of extra capacitance from input to ground.
- When x falls, nodes a and c begin to rise
- At first, both nodes see the same capacitance, consisting of the two transistors and the extra 3 fF.
- As node arises, it initially bumps up b or “lifts b by its own bootstraps.”
- Eventually the nMOS transistors turn ON, pulling down band d.



Limitations of linear delay model

Bootstrapping:

- As b falls, it tugs on a through the capacitor, leading to the slow final transition visible on node a.
- Also observe that b falls later than d because of the extra charge that must be supplied to discharge the bootstrap capacitor.
- Extra capacitance has greater effect when connected between input and output as compared to connected between input and ground.
- C_{gd} is small, bootstrapping is only a little effect in digital circuits.
- However, if the inverter is biased in its linear region near $V_{DD}/2$, the C_{gd} is multiplied by the large gain of the inverter => Miller effect

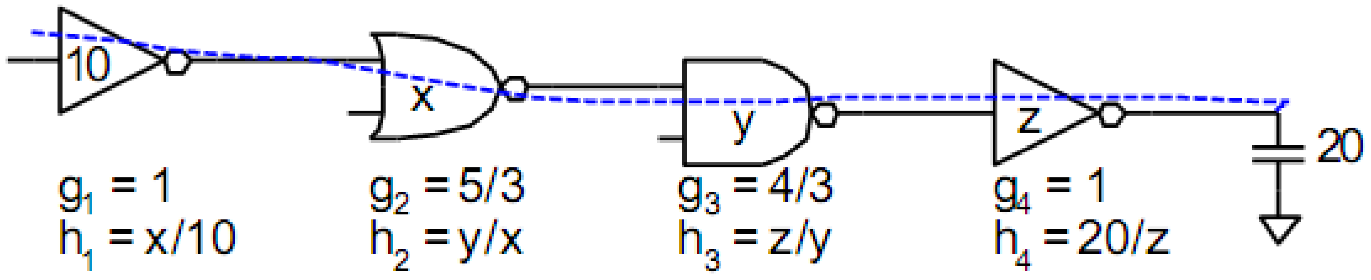


Multistage logic networks

- Designers need to choose the fastest circuit topology and gate sizes for a particular logic function and estimate delay of design
- Simulation or timing analysis only determine how fast a particular implementation will operate
- Logical Effort provides a simple method to choose best topology and number of stages of logic for a function.
- Based on the linear delay model, designer can estimate the best number of stages for a path, the minimum possible delay for the given topology, and the gate sizes that achieve this delay.
- The techniques of Logical Effort will be revisited throughout this text to understand the delay of many types of circuits.

Multistage logic networks

- Logical effort generalizes to multistage networks
- *Path Logical Effort* $G = \prod g_i$
- *Path Electrical Effort* $H = \dfrac{C_{\text{out-path}}}{C_{\text{in-path}}}$
- *Path Effort* $F = \prod f_i = \prod g_i h_i$

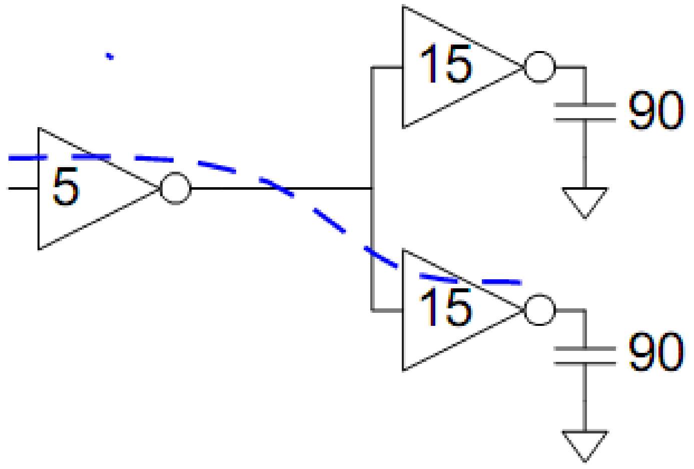


Multistage logic networks

- 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* $F = \prod f_i = \prod g_i h_i$
- 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
  - F = GBH

Multistage delay

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

Design fast circuits

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

- Delay is smallest when each stage bears same effort
- $$\hat{f} = g_i h_i = F^{\frac{1}{N}}$$
- 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?

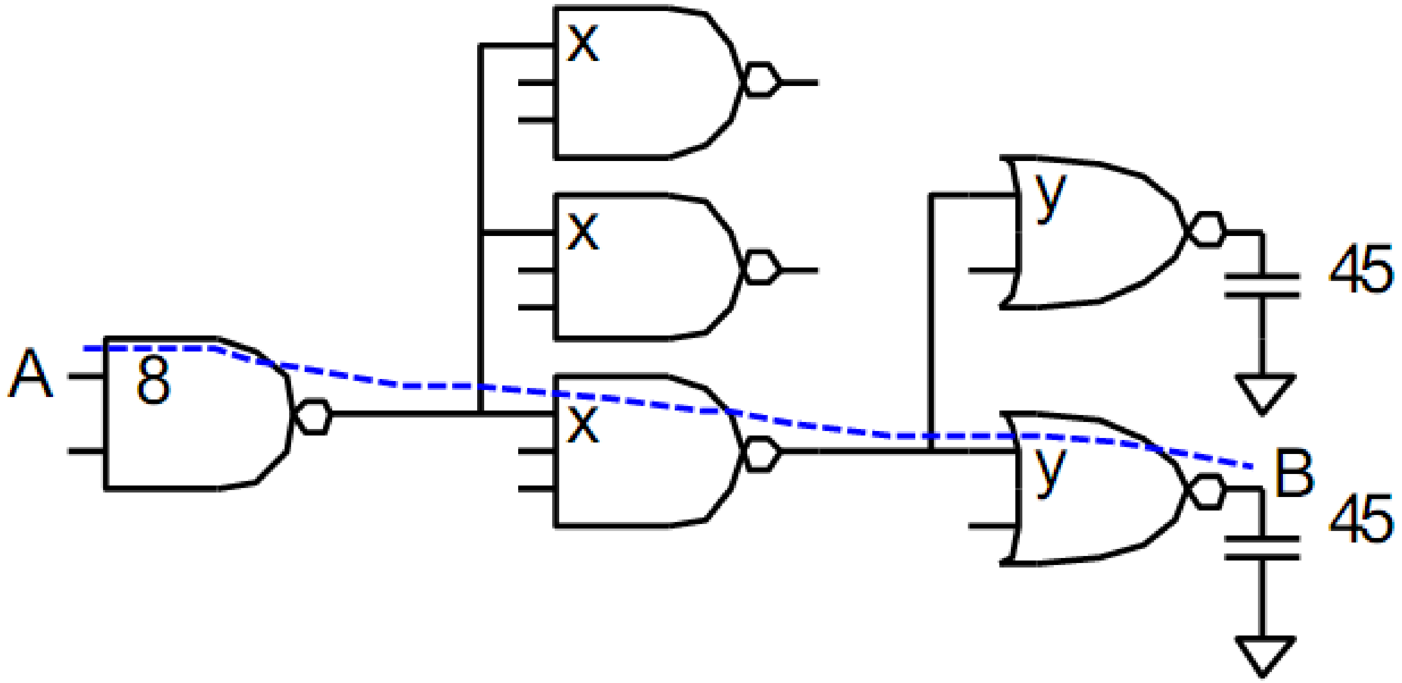
$$\hat{f} = gh = g\frac{C_{out}}{C_{in}}$$

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

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

Example: 3 stages path

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

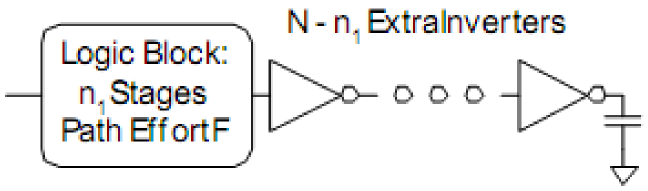


Best number of stages

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

Derivation

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


N - n_1 Extra Inverters
Logic Block: n_1 Stages Path Effort F

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

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

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

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

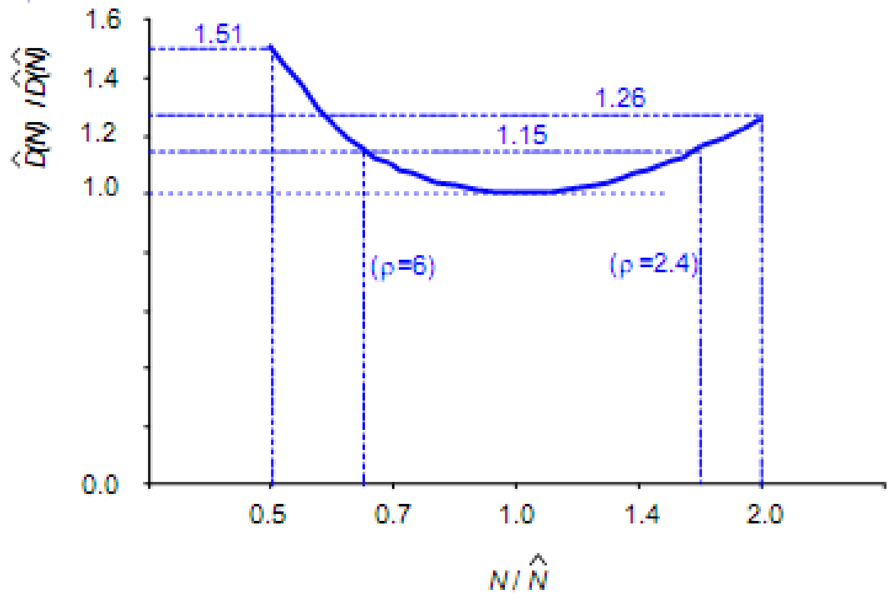
Best stage effort

- $p_{inv} + \rho(1 - \ln \rho) = 0$ has no closed-form solution

- Neglecting parasitics ($p_{inv} = 0$), we find $\rho = 2.718$ (e)
- For $p_{inv} = 1$, solve numerically for $\rho = 3.59$

Sensitivity analysis

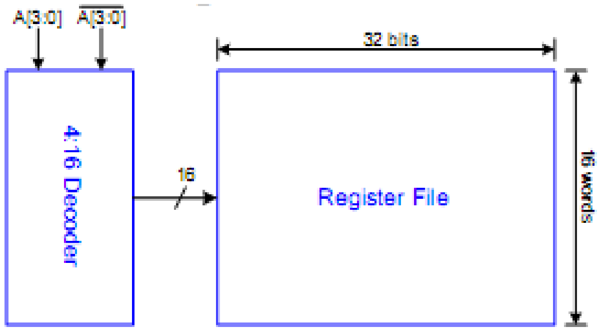
- How sensitive is delay to using exactly the best number of stages?



- $2.4 < \rho < 6$ gives delay within 15% of optimal
  - We can be sloppy!
  - I like $\rho = 4$

Introduction

- 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?



Number of stages

- Decoder effort is mainly electrical and branching
  Electrical Effort: H = (32*3) / 10 = 9.6
  Branching Effort: B = 8
-
- If we neglect logical effort (assume G = 1)
  Path Effort: F = GBH = 76.8

  Number of Stages: N = log4F = 3.1

- Try a 3-stage design

Gate size and delay

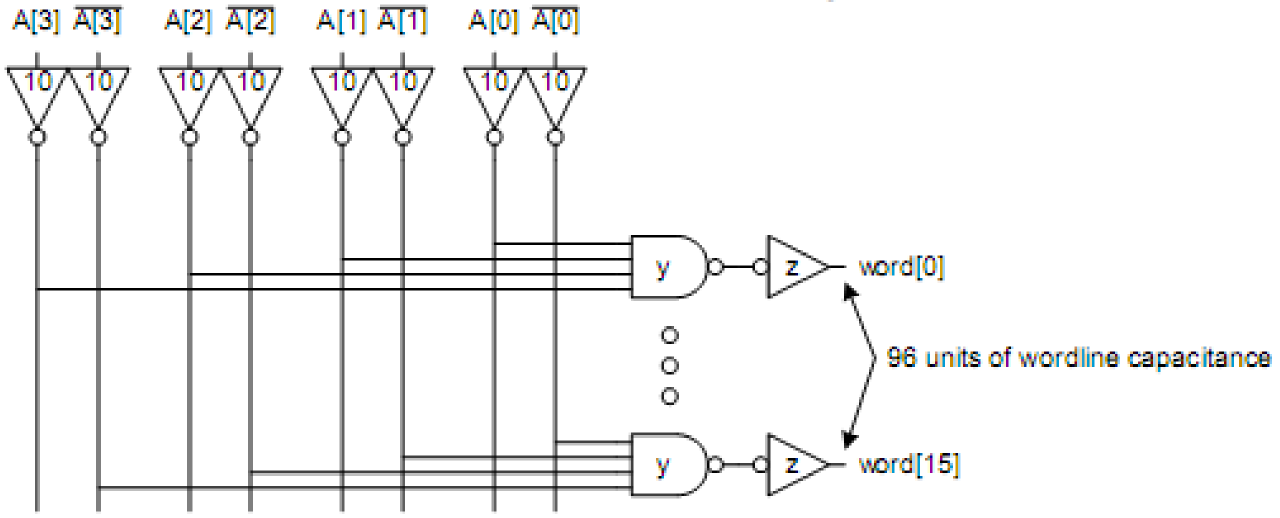
Logical Effort: $G = 1$

Path Effort: $F = G$

Stage Effort: $\hat{f} =$

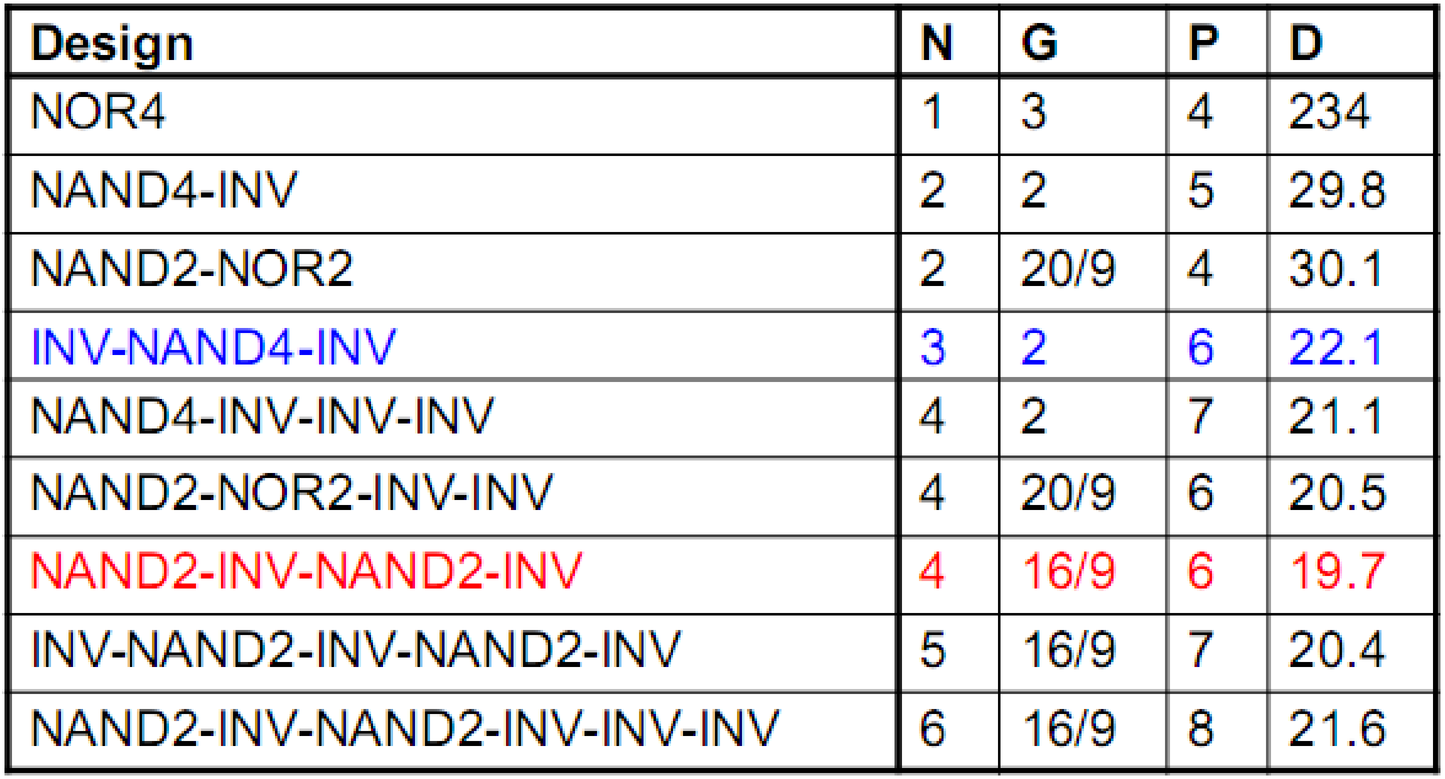
Path Delay: $D =$

Gate sizes: $z = 96*1/5.36$ 2/5.36



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 definition

Term	Stage	Path
number of stages	1	N
logical effort	g	$G = \prod g_i$
electrical effort	$h = \frac{C_{\text{out}}}{C_{\text{in}}}$	$H = \frac{C_{\text{out-path}}}{C_{\text{in-path}}}$
branching effort	$b = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}}$	$B = \prod b_i$
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 $C_{in_i} = \frac{g_i C_{out_i}}{\hat{f}}$

Limit 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 log4F FO4 inverter delays
  - Inverters and NAND2 best for driving large caps
- Provides language for discussing fast circuits
  - But requires practice to master

Homework

- Chapter 4: 1, 2, 3, 5, 7, 9, 10, 11, 12, 14, 15, 16, 17, 19, 21, 24