EECS151/251A Introduction to Digital Design and ICs

Lecture 16: Logical Effort

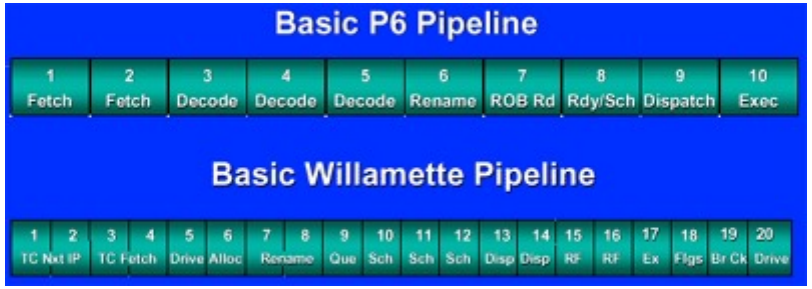
Sophia Shao

10GHz by 2005 running at < 1 volt (published 2000)

The P6 micro-architecture was introduced with the Pentium Pro at 150MHz in 1995 and is still with us today with the Pentium III at 1GHz. The P6 architecture will be with us for a little while longer, in the end offering clock speeds close to 1.3GHz if not higher which is close to a 9x increase in clock speed since the architecture's introduction. Thus it isn't too far fetched to assume a similar scalability from the Pentium 4's NetBurst architecture.

Realistically speaking, we should be able to see NetBurst based processors reach somewhere between 8 – 10GHz in the next five years before the architecture is replaced yet again. Reaching 2GHz isn't much of a milestone, however reaching 8 – 10GHz begins to make things much more exciting than they are today...These processors will run at less than 1 volt, 0.85v being the current estimate.

https://www.anandtech.com/show/680/6



Basic P6 Pipeline

1	2	3	4	5	6	7	8	9	10
Fetch	Fetch	Decode	Decode	Decode	Rename	ROB Rd	Rdy/Sch	Dispatch	Exec

Basic Willamette Pipeline

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
TC Nxt IP		TC Fetch		Drive	Alloc	Rename		Que	Sch	Sch	Sch	Disp	Disp	RF	RF	Ex	Flgs	Br Ck	Drive

Review

- Inverter Delay
  - $t_p = \ln 2 * R_{eq}C_{in}(1+C_L/C_{in}) = \tau_{INV}(1+f)$
    - $\tau_{INV} = \ln 2 * R_{eq}C_{in}$
    - Fanout = $f = C_L/C_{in}$
  - Normalized Delay to τ_{INV}
    - $D\,(\text{inv}) = 1 + f$

Parasitic delay Effort delay

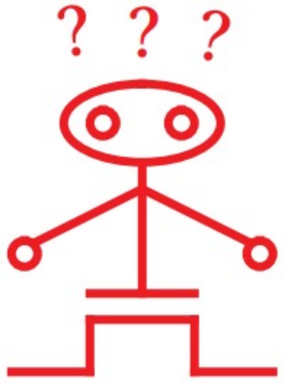
- Inverter Chain
  - Path Delay $Delay = \sum(1 + f_i) = N + \sum fi$
  - Path Fanout Effort $F = \prod f_i = C_L/C_{in,1}$
  - Size the inverters to minimize the delay of an inverter chain
    - Every inverter stage has the same effort delay, i.e., $f_i = \sqrt[N]{F}$
    - The size of each inverter stage can be determined by working backward
      - $C_{in,i} = \dfrac{C_{L,i}}{f_i}$

This Lecture: Generalize to other gates

- Logical Effort
  - Overview
  - Definition
  - Calculating LE for Gates
- Multi-Stage Network
  - Overview
  - Gate Sizing Problem
  - Branching Effort

Introduction

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

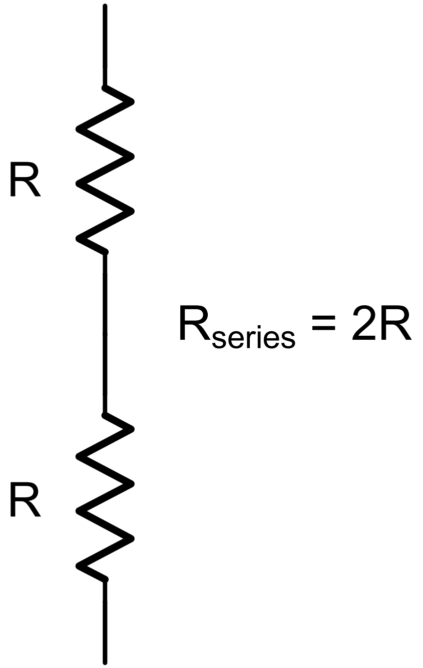


- **Logical Effort (LE) is a method to answer these questions**
  - Uses simple delay model
  - Back-of-the-envelop calculations
- **Who cares about LE?**
  - Circuit designers who waste time in simulate/tweak loop
  - High-speed logic designers need to know where time is going in their logic
  - CAD designers need to understand circuits to build better tools

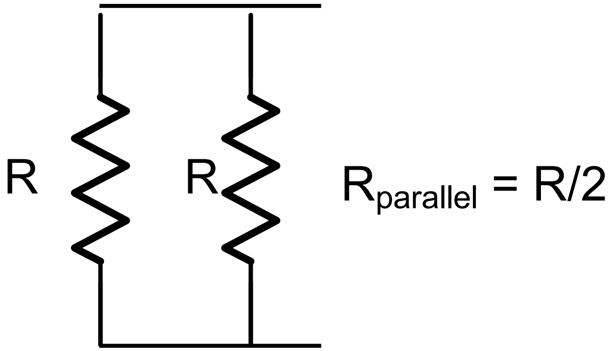
Courtesy: D. Harris

Series and Parallel

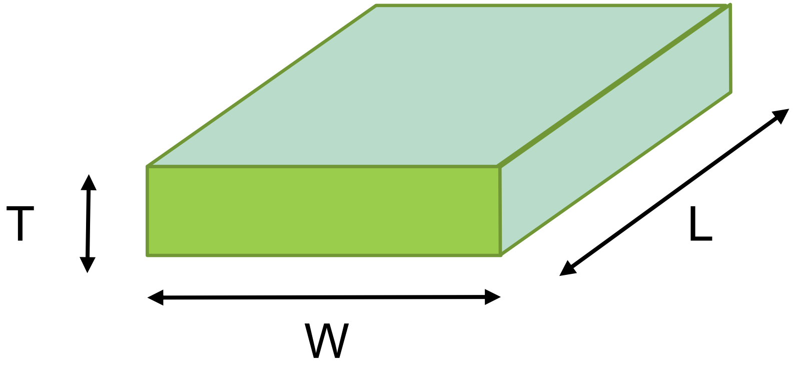
- With two identical resistors, R



Equivalent to doubling length



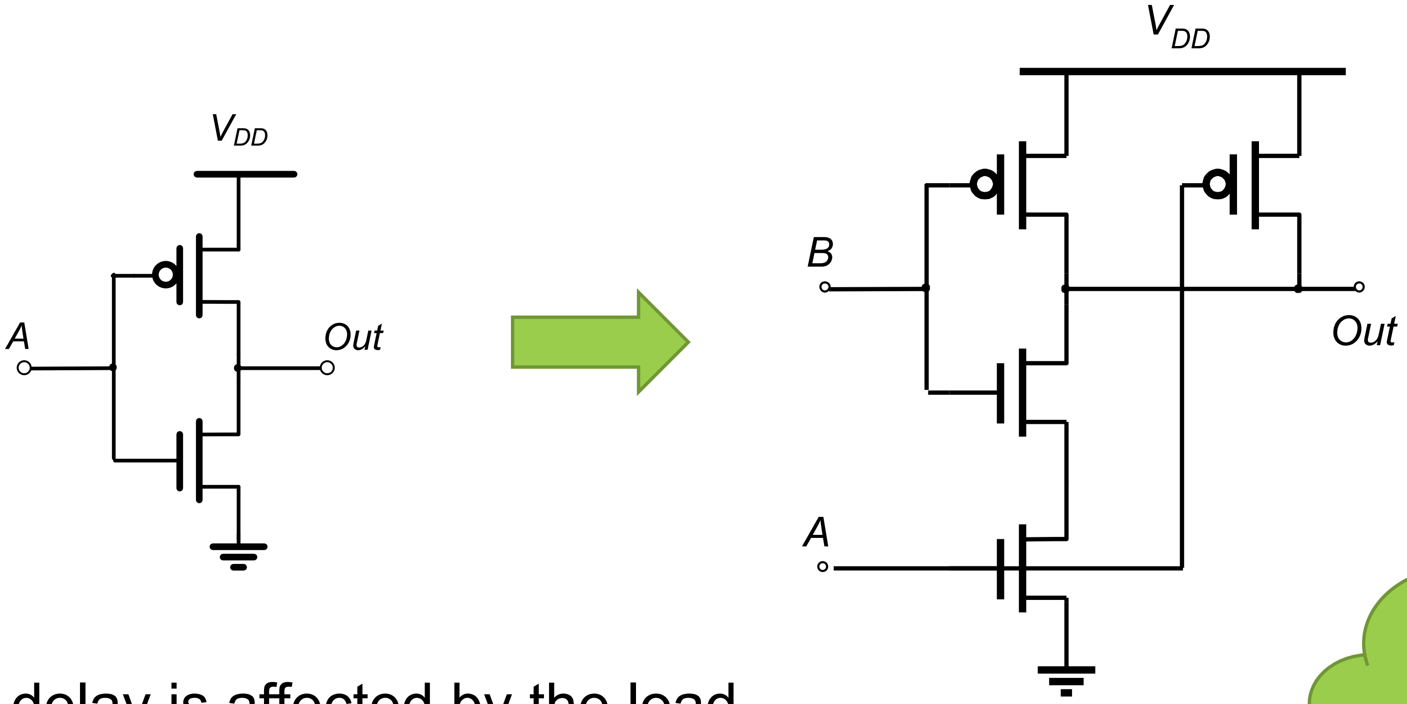
Equivalent to doubling width



$$R = \rho \frac{L}{TW}$$

- In a planar process, designer controls W and L

Single-Input Gate -> Multi-Input Gates



Logic Effort quantifies these effects.

- So far: how delay is affected by the load.
- Delay also depends on the logic function of the gate.
- Inverters, the simplest logic gates, drive loads best.
- Logic gates that compute other functions require more transistors, *some of which are connected in series*, making them poorer than inverters are driving current.

Generalize Inverter Delay to other Gates

- So far, inverter delay $t_{p,inv}$ = ln2 * $R_{eq,inv}(C_{p,inv} + C_L)$
  - $t_{p,inv}$ =ln2 * $R_{eq,inv}C_{in,inv}(1+C_L/C_{in,inv}) = \tau_{INV}(1+f)$
    - τ_{INV}= ln2 * $R_{eq,inv}C_{in,inv}$

Generalize Inverter Delay to other Gates

- So far, inverter delay $t_{p,inv}$ = ln2 * $R_{eq,inv}(C_{p,inv} + C_L)$
  - $t_{p,inv}$ =ln2 * $R_{eq,inv}C_{in,inv}(1+C_L/C_{in,inv}) = \tau_{INV}(1+f)$
    - τ_{INV}= ln2 * $R_{eq,inv}C_{in,inv}$
- For other gates $t_{p,gate}$ = ln2 * $R_{eq,gate}(C_{p,gate} + C_L)$
  - Normalized delay D = $t_{p,gate} / \tau_{INV}$
  - $D(gate) = \dfrac{ln2 * R_{eq,gate}(C_{p,gate} + C_L)}{ln2 * R_{eq,inv} * C_{in,inv}} = \dfrac{R_{eq,gate}C_{p,gate} + R_{eq,gate}C_L}{R_{eq,inv} * C_{in,inv}}$

$$= \frac{R_{eq,gate}C_{p,gate}}{R_{eq,inv}C_{in,inv}} + \frac{R_{eq,gate}C_L}{R_{eq,inv}C_{in,inv}}$$

$$= \frac{R_{eq,gate}C_{p,gate}}{\underset{\uparrow}{R_{eq,inv}C_{p,inv}}} + \frac{R_{eq,gate}C_{in,gate}}{\underset{\uparrow}{R_{eq,inv}C_{in,inv}}} * \frac{C_L}{\underset{\nwarrow}{C_{in,gate}}} = \boldsymbol{P + LE * FO}$$

Parasitic delay (P)

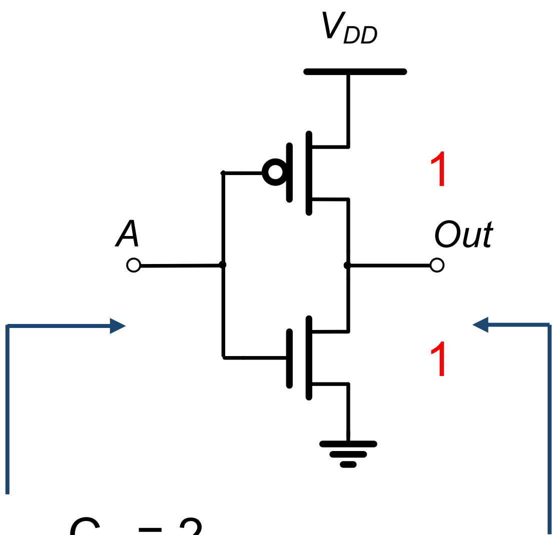
Logical Effort (LE)

Fanout (Electrical Effort) (FO)

Logical Effort

- $D(gate) = LE * FO + P =$ Effort Delay + Parasitic Delay
  - $LE = \frac{R_{eq,gate} C_{in,gate}}{R_{eq,inv} C_{in,inv}}$
- Recall D(inv) = f + 1
  - LE(inv) = 1
- Definition:
  - Logical effort is the ratio of the input capacitance to the input capacitance of a unit inverter delivering the same output current.
  - Only dependent on gate topology
- To calculate LE and P, size transistors so that R is the same as Rinv, then
  - LE = Cin,gate / Cin,inv
  - P = Cp,gate / Cp,inv

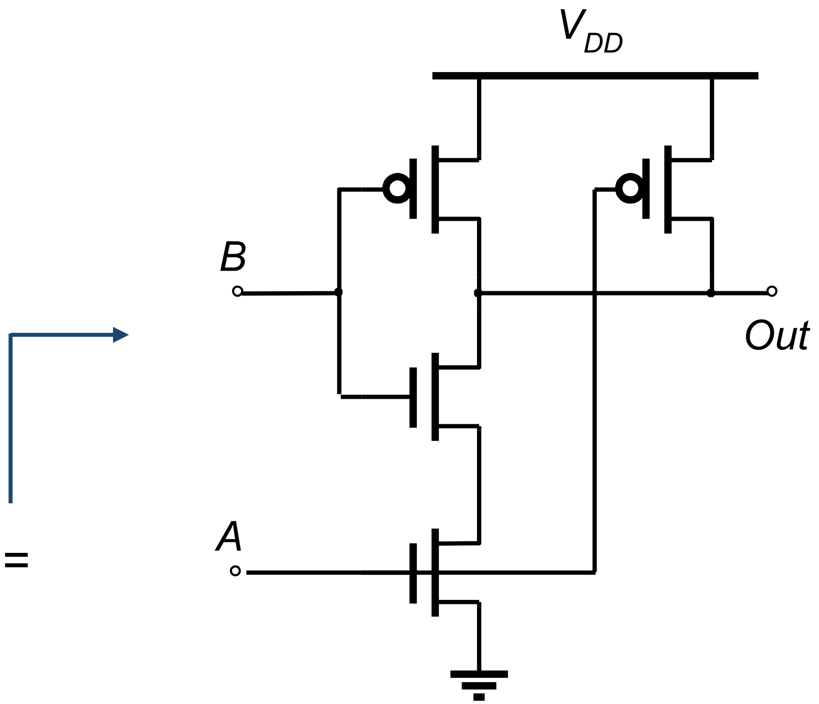
NAND2 Gate



$R_{eq} = 1$

$C_{in} = 2$

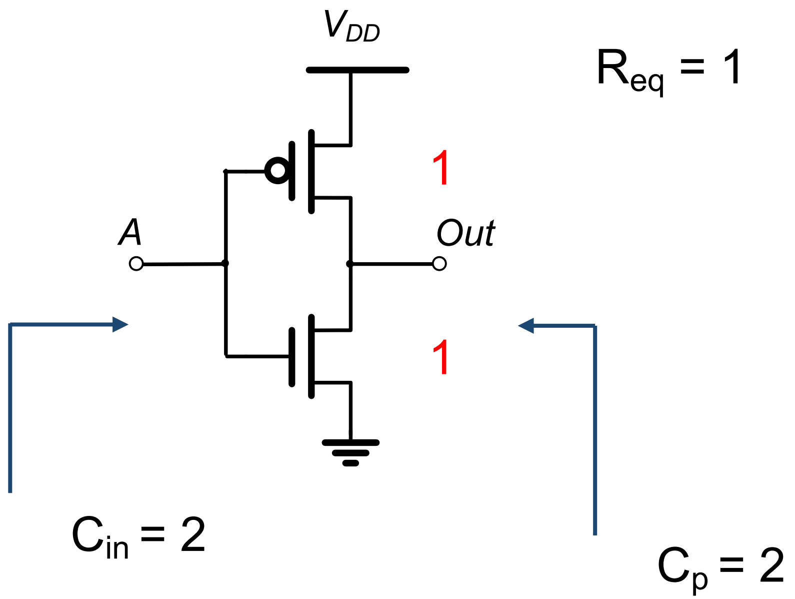
$C_p = 2$



$R_{eq} = 1$

$C_{in} =$

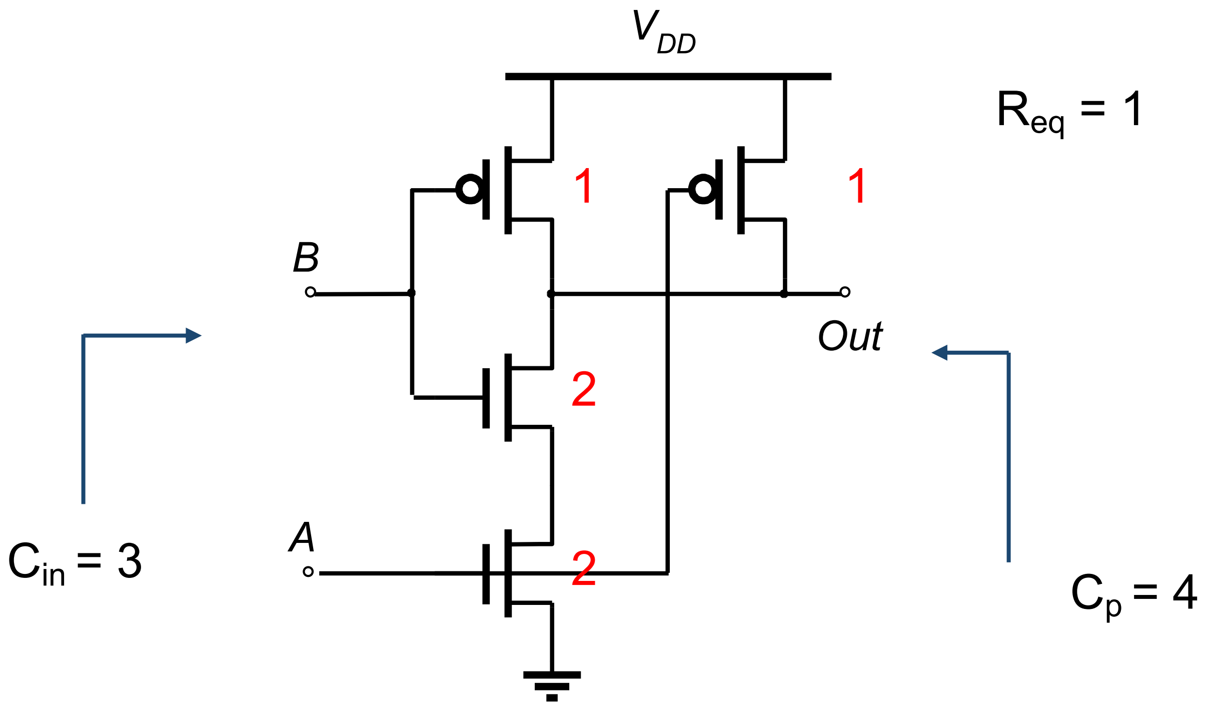
NAND2 Gate



$R_{eq} = 1$

$C_{in} = 2$

$C_p = 2$



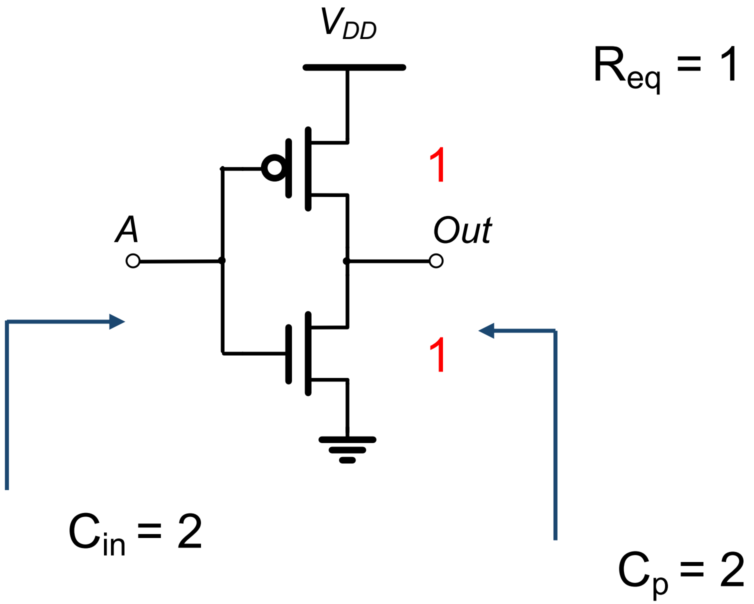
$R_{eq} = 1$

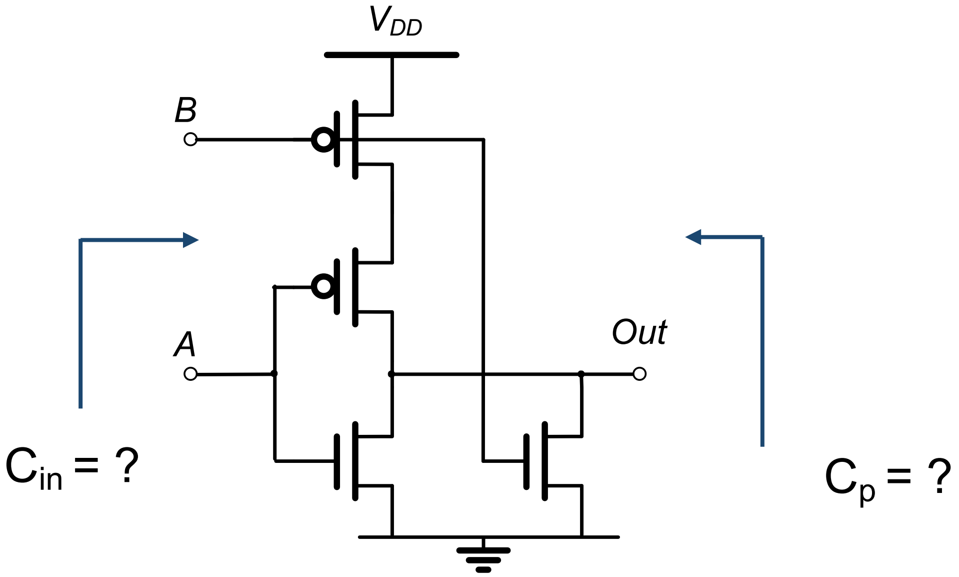
$C_{in} = 3$

$C_p = 4$

- LE(NAND2) = 3/2
- P(NAND2) = 4/2

NOR2 Gate





- Calculate the LE (logical Effort) and P (normalized parasitic delay) for NOR2 gate.
- A: LE (NOR2) = 3/2, P (NOR2) = 4/2
- B: LE (NOR2) = 4/2, P (NOR2) = 3/2
- C: LE (NOR2) = 1, P (NOR2) = 1

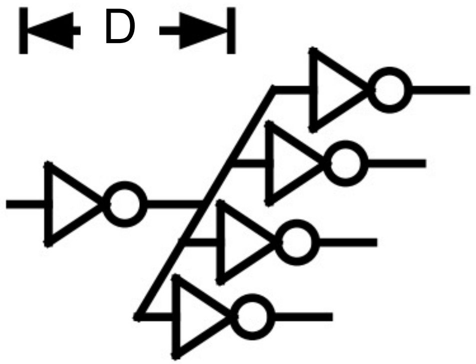
Administrivia

- Have fun during Spring break!
- Project released this week.
  - Checkpoint 1 after spring break.
- Homework 6 released this week.
  - Due after spring break.

- Logical Effort
  - Overview
  - Definition
  - Calculating LE for Gates
- Multi-Stage Network
  - Overview
  - Gate Sizing Problem
  - Branching Effort

Warm-up Example

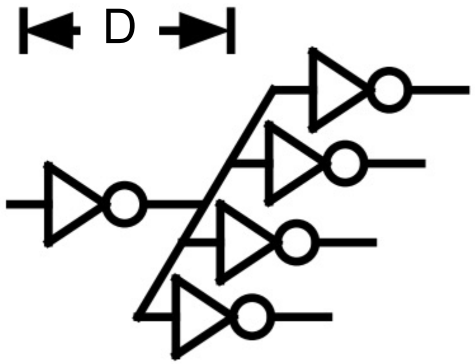
- Estimate the delay of a fanout-of-4 (FO4) inverter:



- Logical Effort: LE =
- Fanout Effort: FO =
- Parasitic Delay: P =
- State Delay: D = LE*FO + P =

Warm-up Example

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



- Logical Effort: LE = 1
- Fanout Effort: FO = C_out / C_in = 4
- Parasitic Delay: P = p_inv = 1
- State Delay: D = LE*FO + P = 5

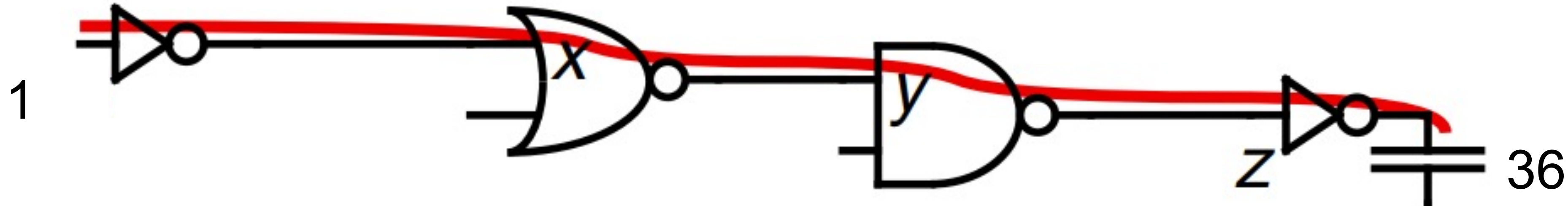
Multi-Stage Networks

- $Delay = \sum_{i=1}^{N}(P_i + LE_i * FO_i)$
  - Stage Effort: $SE_i = LE_i * FO_i$
  - Path Fanout Effort: $FO_{path} = \frac{C_{load}}{C_{in}}$
  - Path Logical Effort: $LE_{path} = LE_1 * LE_2 \ldots * LE_N$
  - Path Effort: $PE = LE_{path} * FO_{path}$

Optimal Effort per Stage

- When each stage bears the same effort:
  - Stage Effort: $SE_i = LE_i * FO_i$
  - Path Effort: $PE = LE_{path} * FO_{path}$
  - $SE^N = \prod LE * FO = PathEffort$
  - $SE_* = \sqrt[N]{PathEffort}$
- Where the minimum path delay
  - $Delay = \sum_{i=1}^{N}(P_i + LE_i * FO_i) = N * SE_* + P$
- In this case, the effective fanout of each stage:
  - $FO_i = SE_*/LE_i$
  - => solve gate sizing problems

Gate Sizing Example



LE_1 =
FO_1 =

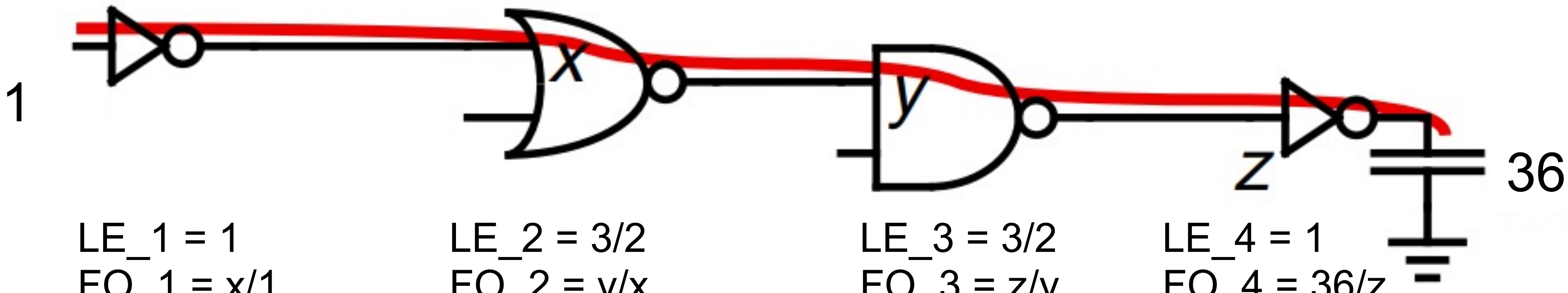
LE_2 =
FO_2 =

LE_3 =
FO_3 =

LE_4 =
FO_4 =

- First compute path effort:
- $PathEffort =$
- The optimal stage effort is:
- $SE_* =$

Gate Sizing Example



LE_1 = 1
FO_1 = x/1

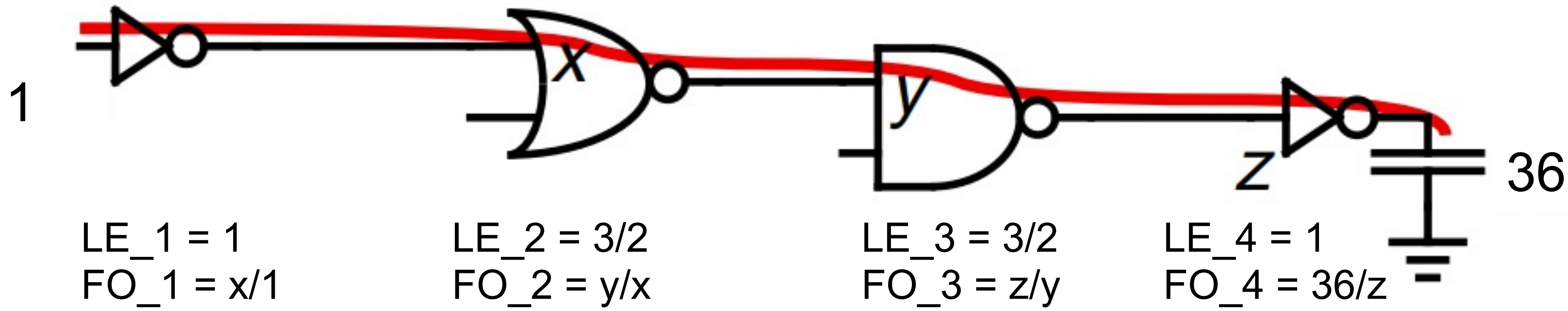
LE_2 = 3/2
FO_2 = y/x

LE_3 = 3/2
FO_3 = z/y

LE_4 = 1
FO_4 = 36/z

- First compute path effort:
- $PathEffort = \prod LE * FO = 1 * \left(\frac{x}{1}\right) * \frac{3}{2} * \left(\frac{y}{x}\right) * \frac{3}{2} * \left(\frac{z}{y}\right) * 1 * \left(\frac{36}{z}\right) = 81$
- The optimal stage effort is:
- $SE_* = \sqrt[N]{PathEffort} = \sqrt[4]{81} = 3$

Gate Sizing Example



LE_1 = 1
FO_1 = x/1

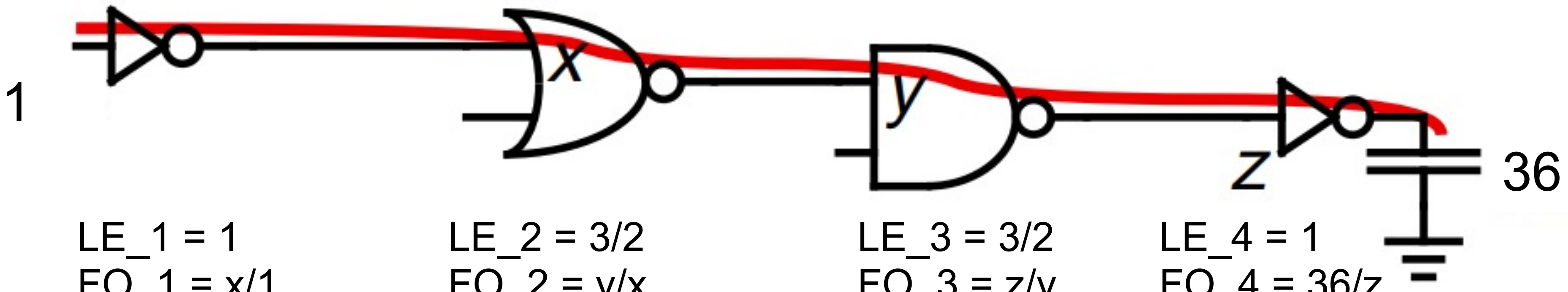
LE_2 = 3/2
FO_2 = y/x

LE_3 = 3/2
FO_3 = z/y

LE_4 = 1
FO_4 = 36/z

- The total normalized path delay is:
- $D = 4 * SE_* + \sum P =$

Gate Sizing Example



LE_1 = 1
FO_1 = x/1

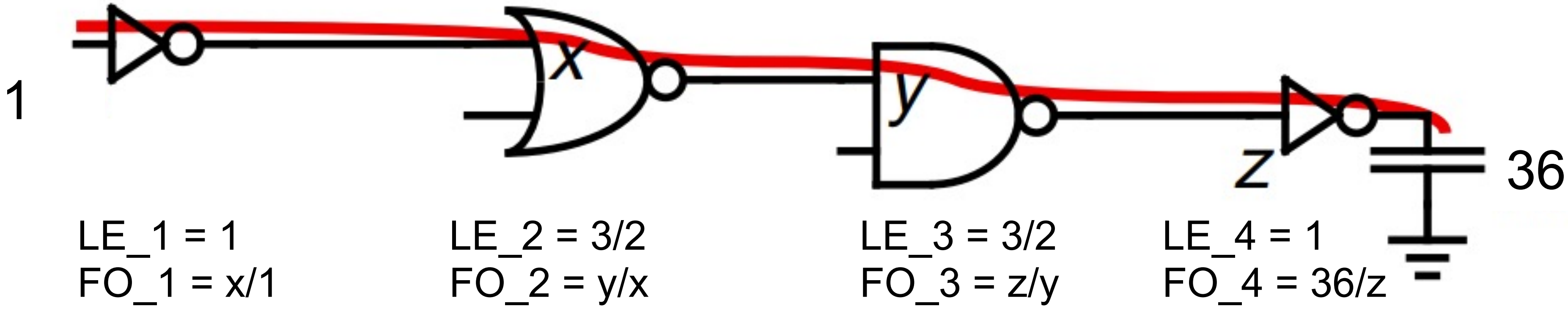
LE_2 = 3/2
FO_2 = y/x

LE_3 = 3/2
FO_3 = z/y

LE_4 = 1
FO_4 = 36/z

- The total normalized path delay is:
- $D = 4 * SE_* + \sum P = 4 * 3 + \left(1 + \frac{4}{2} + \frac{4}{2} + 1\right) = 18$

Gate Sizing Example



LE_1 = 1
FO_1 = x/1

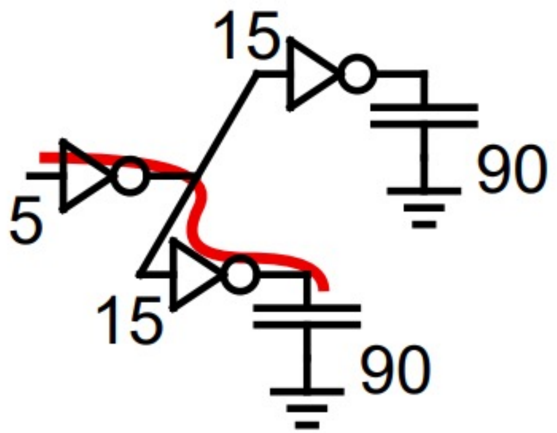
LE_2 = 3/2
FO_2 = y/x

LE_3 = 3/2
FO_3 = z/y

LE_4 = 1
FO_4 = 36/z

- Now we can size the gates, since for all the them:
- $SE_* = 3,\ C_{in} = LE * \frac{C_{out}}{SE_*}$
- We have:
  - Z =1*36/3=12
  - Y=3/2*z/3=6
  - X=3/2*y/3=3

Add Branching Effort



LE =1

FO_path = C_load/C_in=90/5=18

PathEffort = 18 (wrong!)

==================================

SE_1=(15+15)/5=6

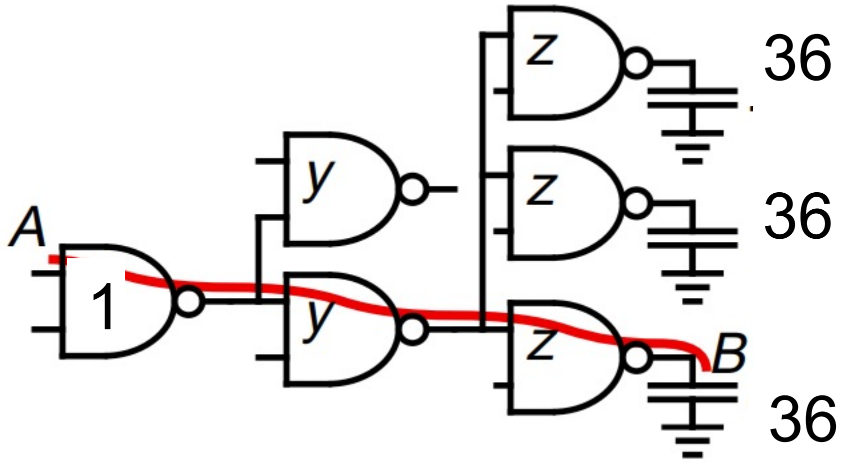
SE_2=90/15=6

PathEffort = 6*6 = 36 (not 18!)

- Introduce branching effort to account for branching
  - Branching Effort: $b = \frac{C_{on-path}+C_{off-path}}{C_{on-path}}$
  - Path Branching Effort: $B = \prod b_i$
- Now we can compute the path effort
  - Path Effort $PE = LE_{path} * FO_{path} * B_{path}$

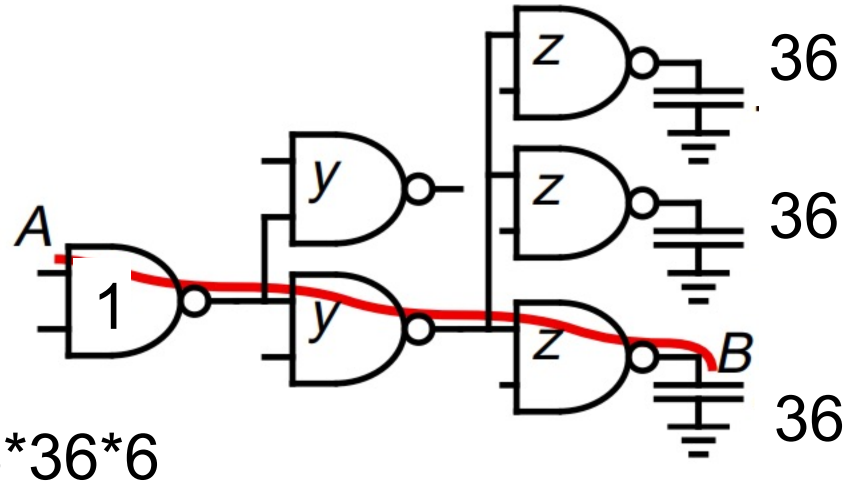
Branching Example

- Size gate sizes y and Z to minimize delay of the highlighted path.
- Logical Effort: LE_{path}
- Fanout Effort: FO_{path}=Cout/Cin
- Branching Effort: B_{path} =
- Path Effort: PE = $LE_{path} * FO_{path} * B_{path}$
- ==
- Best Stage Effort $SE_* = \sqrt[N]{PathEffort}$ =
- Path Delay D =
- Work backwards for sizes: z = , y=



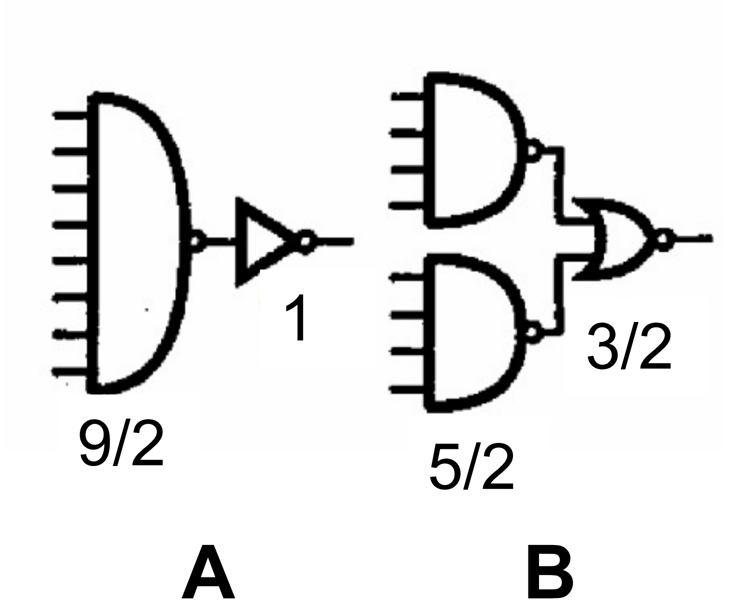
Branching Example

- Size gate sizes y and Z to minimize delay of the highlighted path.
- Logical Effort: LE_{path}=(3/2)^3
- Fanout Effort: FO_{path}=Cout/Cin = 36
- Branching Effort: B_{path} = 2*3=6
- Path Effort: PE = $LE_{path} * FO_{path} * B_{path}$=(3/2)^3*36*6
- ==



- Best Stage Effort $SE_{*} = \sqrt[N]{PathEffort}$ = ((3/2)^3*36*6)^1/3=9
- Path Delay D = 3*9+3*4/2=33
- Work backwards for sizes: z = 3/2*36/9=6, y=3/2*3*z/9=3

Quiz: Multi-Level Logic, which is faster?



- To build an 8-input AND logic, ignore parasitic delays, which option gives us the shortest delay?
  - The LE for each gate is labelled explicitly.
  - Hint: what's LE_{path} for each choice?

Review

- Logical Effort:
  - Logical effort is the ratio of the input capacitance to the input capacitance of a unit inverter delivering the same output current.
  - Only dependent on gate topology
  - To calculate LE and P, size transistors so that R is the same as Rinv, then
    - LE = Cin,gate / Cin,inv
    - P = Cp,gate / Cp,inv
- Multi-Stage Network:
  - $Delay = \sum_{i=1}^{N}(P_i + LE_i * FO_i)$
    - Stage Effort: $SE_i = LE_i * FO_i$
    - Path Fanout Effort: $FO_{path} = \frac{C_{load}}{C_{in}}$
    - Path Logical Effort: $LE_{path} = LE_1 * LE_2 \ldots * LE_N$
    - Path Effort: $PE = LE_{path} * FO_{path}$