Lecture 5

Logical Effort

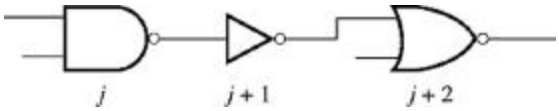
Boris Murmann
Center for Integrated Systems
Stanford University
murmann@stanford.edu

Overview

- **Reading**
  - Hodges 6.6
- **Additional References:**
  - Harris, Logical Effort talk slides (handout #7 on the web)
  - Sutherland, Sproull, Harris, *Logical Effort*, Kaufman Publishers, 1999.
- **Introduction**

  Having set up the basic optimization problems, we will next develop a formalism for doing sizing with real gates. This formalism is called logical effort. To get some practice using this method we will apply it later to the memory decoder we talked about in Lecture 3 and 4.

Re-Cap



- In this example, the total delay is:

$$Delay = t_{nand}\left(\frac{C_{j+1}}{C_j} + g_{nand}\right) + t_{inv}\left(\frac{C_{j+2}}{C_{j+1}} + g_{inv}\right) + t_{nor}\left(\frac{C_{j+3}}{C_{j+2}} + g_{nor}\right)$$

- Normalized to the intrinsic time constant of an inverter, we have:

$$\frac{Delay}{t_{inv}} = \frac{t_{nand}}{t_{inv}}\left(\frac{C_{j+1}}{C_j} + g_{nand}\right) + \frac{t_{inv}}{t_{inv}}\left(\frac{C_{j+2}}{C_{j+1}} + g_{inv}\right) + \frac{t_{nor}}{t_{inv}}\left(\frac{C_{j+3}}{C_{j+2}} + g_{nor}\right)$$

Logical Effort Formalism (1)

- ... since this is hard to fit on the back of an envelope, we define new symbols:

$$\frac{Delay}{t_{inv}} = \frac{t_{nand}}{t_{inv}}\left(\frac{C_{j+1}}{C_j} + g_{nand}\right) + \frac{t_{inv}}{t_{inv}}\left(\frac{C_{j+2}}{C_{j+1}} + g_{inv}\right) + \frac{t_{nor}}{t_{inv}}\left(\frac{C_{j+3}}{C_{j+2}} + g_{nor}\right)$$

$$D = (LE_{nand}FO_1 + P_{nand}) + (LE_{inv}FO_2 + P_{inv}) + (LE_{nor}FO_3 + P_{nor})$$

Logical Effort — **Fan-Out, "Electrical Effort"** — **Parasitic Delay**

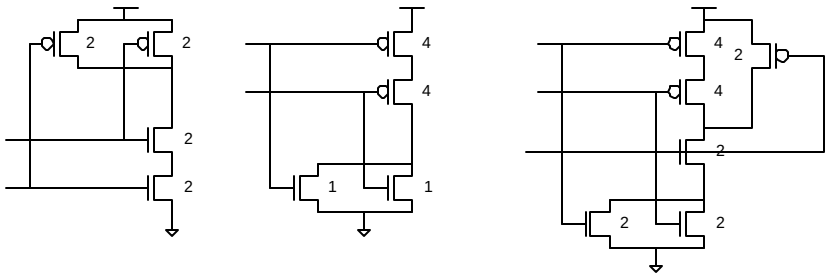
Logical Effort Formalism (2)

- More nomenclature:
  - D_{gate} = LE*FO+P = *EffortDelay* + *ParasiticDelay*
- Some options to find LE of a logic gate:
  - Set R_{drive} equal, then compare C_{in}
  - Set C_{in} equal, then compare R_{drive}
  - Or simply compare R and C ratio form first principles…

$$LE_{gate} = \frac{t_{gate}}{t_{inv}} = \frac{(R_{drive} \cdot C_{in})_{gate}}{(R_{drive} \cdot C_{in})_{inv}}$$

Calculating Logical Effort for a Gate (1)

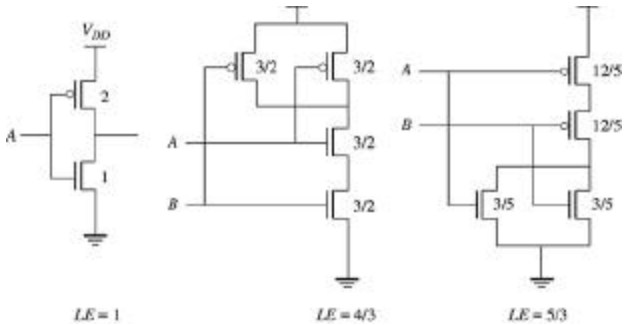
- Build the gates to have the same drive strength as a 2x pMOS, 1x nMOS inverter. The numbers on each transistor is relative to the 1x nMOS transistor in the inverter. The Cin of inverter is 3x.



- LE = 4/3 LE=5/3 LE=2; 4/3
- Note that the logical effort of all inputs does not always match

Calculating Logical Effort for a Gate (2)

- Make input capacitance equal to that of inverter.
- LE now follows from ratio of R_{drive}



- Note that uniform scaling of a gate does not change its LE, this is why this approach works...

LE Catalog of Gates

Gate type	Number of inputs					
	1	2	3	4	5	*n*
inverter	1					
NAND		4/3	5/3	6/3	7/3	(*n*+2)/3
NOR		5/3	7/3	9/3	11/3	(2*n*+1)/3
multiplexer		2	2	2	2	2
XOR, XNOR		4	12	32		

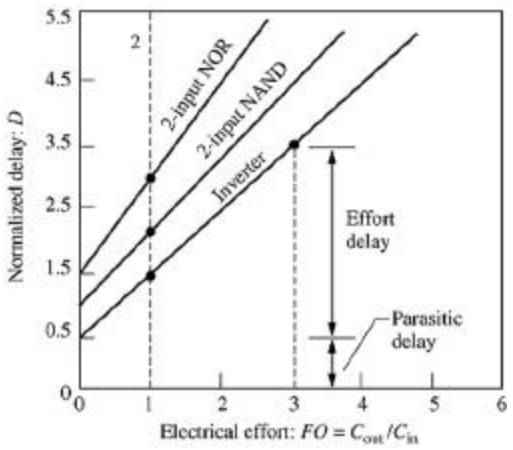
- Ref: Harris' slides

Parasitic Term

- Technology and gate dependent
- Typically, we have P_{inv}=0.5...1
- Once P_{inv} is known, we can estimate P of many other gates: [Hodges, p. 291]

	1 input	2 inputs	3 inputs	4 inputs
INV	P_{inv}			
NAND		$2P_{inv}$	$3P_{inv}$	$4P_{inv}$
NOR		$3P_{inv}$	$4.5P_{inv}$	$6P_{inv}$

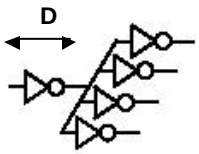
LE and P from Simulation/Graphical Data



- From plot:
  - Slope is LE
  - Y intercept is parasitic delay
- More complex gates
  - Have larger LE
  - Have larger parasitics

Warm-ups with Logical Effort

- Frequency of an N stage ring oscillator -> see HW#2
- Delay of a inverter with a fan-out of 4
  - LE=
  - FO=
  - P=
  - D=



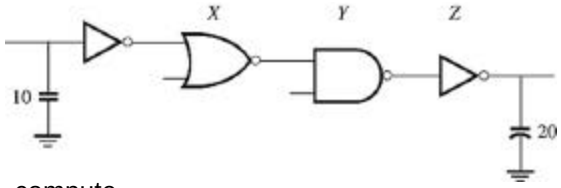
- Normally we will report delays in terms of FO4 inverter delays
- Delay in FO4 units is therefore roughly equal to normalized delay divided by:

Gate Sizing Problem

- We will use the LE of the gate to help find the correct sizes
  - We know that the LE*Fanout for each gate should be the same
- Before we do more math, we need to set a convention:
  - What does a gate of size '2' mean?
  - For an inverter, it is simple:
    - It has twice the capacitance and ½ the resistance of an inverter of size '1'
  - For a gate, you have two options:
    - Can define it to mean it has twice the capacitance of an inverter

      OR
    - Can define it to mean it has ½ the resistance
- **In EE313, the size is a measure of the input capacitance**
  - **A size 2 gate has twice the C_{in} of an inverter**

Gate Sizing Example (1)



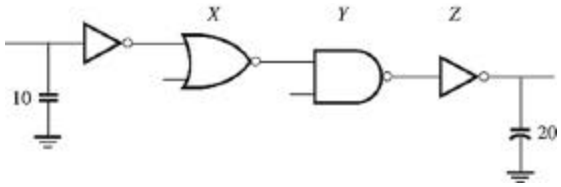
- First, compute

$$PathEffort = \prod LE \cdot FO$$
$$= 1\left(\frac{X}{10}\right)\times\frac{5}{3}\left(\frac{Y}{X}\right)\times\frac{4}{3}\left(\frac{Z}{Y}\right)\times 1\left(\frac{20}{Z}\right) = \frac{400}{90}$$

- From lecture 4, we know that the optimal stage effort is

$$SE^* = LE \cdot FO = \left(\frac{400}{90}\right)^{1/4} = 1.45$$

Gate Sizing Example (2)



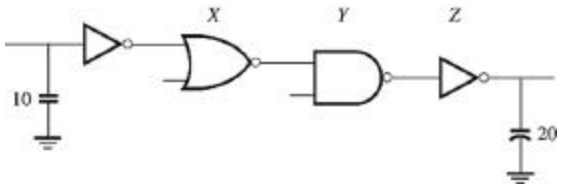
- With this information, we can now size the gates, since for all of them

$$C_{in} = LE \cdot \frac{C_{out}}{SE^*}$$

- We have

$$Z = 1 \cdot \frac{20}{1.45} = 13.8 \qquad X = \frac{5}{3} \cdot \frac{Y}{1.45} = 14.5$$
$$Y = \frac{4}{3} \cdot \frac{Z}{1.45} = 12.7 \qquad C_{in} = 1 \cdot \frac{X}{1.45} = 10$$

Gate Sizing Example (3)



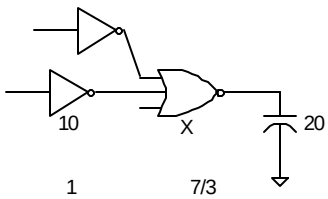
- The normalized delay is (assuming P_{inv}=0.5)

$$D = 4SE^* + \sum P = 4 \cdot 1.45 + 0.5 + 1.5 + 1 + 0.5 = 9.3$$

- Can we do better than this?
  - SE*=1.45 is small. Speed may improve if we get closer to 4, and use fewer stages.
  - Can try to reduce the number of stages (not always possible)

Reducing Number of Logic Stages

- If you have too low EF, use more complex gates, with fewer stages
- Often need to use AND-OR-Invert gates or OR-AND-Invert gates

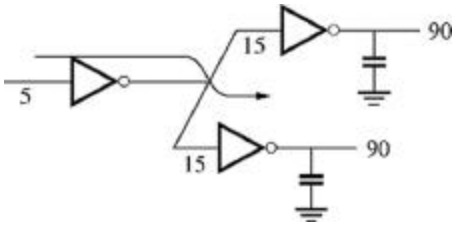


LE	1	7/3
Fanout	x/10	20/x

$$SE^* = \left(\frac{14}{3}\right)^{1/2} = 2.16 \qquad D = 2SE^* + \sum P = 4.32 + 0.5 + \frac{9}{4} = 7.1$$

(Note: this was the wrong optimization if the "top" input was critical - since we have slowed down this path. Make sure you are sizing the critical path)

Branching Effort

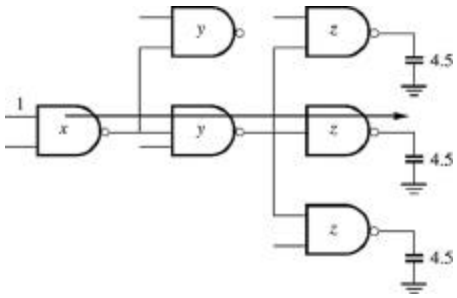


- If there is branching in the logic path, the equation for the total path effort becomes

$$PathEffort = \prod LE \cdot FO \cdot BE$$

where BE is the "branching effort" (2 in the above example).

Branching Effort Example



$$PathEffort = \prod LE \cdot FO \cdot BE = \left(\frac{4}{3}\right)^3 \cdot 4.5 \cdot 2 \cdot 3 = 64$$

$$SE^* = (64)^{1/3} = 4$$

Logical Effort "Design Flow"

- Estimate the path effort $PathEffort = \prod LE \cdot FO \cdot BE$
- Estimate the optimal number of stages $N^* = \log_4 (PathEffort)$
- Estimate the minimum delay $D^* = 4N^* + \sum P$
- Determine the actual number and type of gates
  - Fit the required logic in N stages, where N is close to N*
  - This may slightly change the path effort
- Determine the new stage effort $SE^* = (PathEffort)^{1/N}$
- Working from either end, determine gate sizes $C_{in} = LE \cdot B \cdot \frac{C_{out}}{SE^*}$