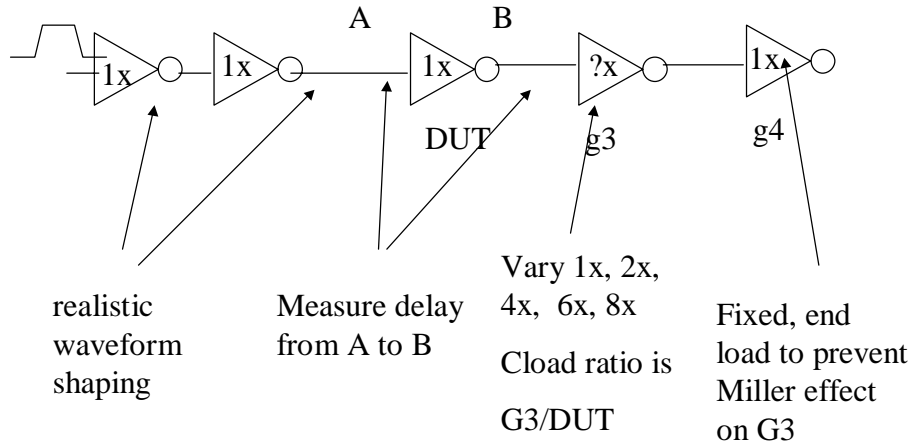


Logic Effort Revisited

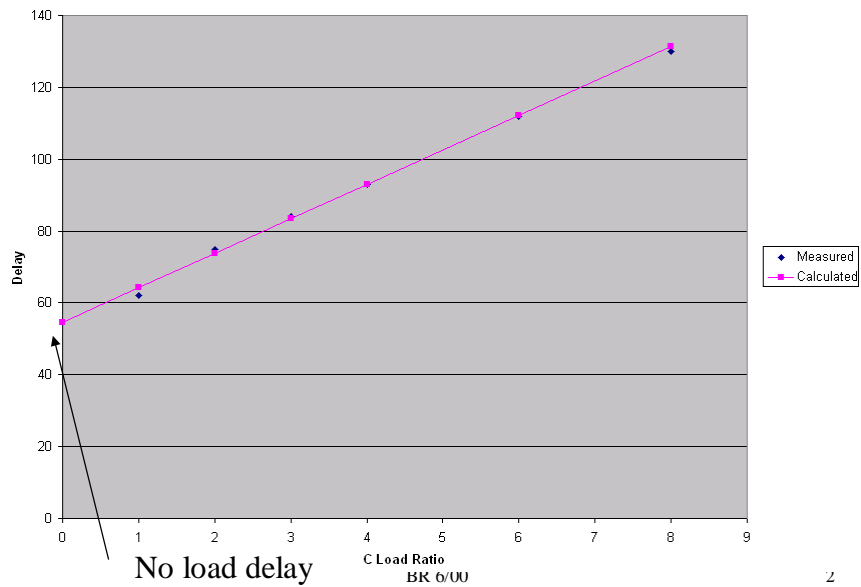
A better way to measure P, Tau



BR 6/00

1

Plot Delay, Fit to Straight line (delay = $mX + b$)



2

Tau, Pinv

By definition, $g_{inv} = 1.0$

From fitted line of $mx + b$, Tau can be calculated at any point as:

$$\begin{aligned} \text{delay} &= \text{tau} (g * h + \text{Pinv}) \\ &= \text{tau} * g * h + \text{tau} * \text{Pinv} \end{aligned}$$

When $X=0$, $\text{delay} = \text{tau} * \text{Pinv} = b$ (y-intercept).

So:

$$\text{Tau} = (\text{delay_measured} - b) / \text{Cload}$$

When Tau is known, can compute Pinv

BR 6/00

3

Old vs New

Using $V_{dd} = 2.5$, $L_{eda} = 0.25\mu$

| | Tau | Pinv | Pnand2 | Pnand4 |
|-----|-----|------|--------|--------|
| old | 8.4 | 6.6 | 11 | 23 |
| new | 9.6 | 5.7 | 12 | 30 |

Differences mainly due to realistic waveshaping of inputs.

Sizing values in Decoder value not changed much.

BR 6/00

4

Measuring Actual Logical Effort

When replace all gates in test circuit with Nand2, and plot:

$$\text{delay} = M_{\text{nand2}} * X + B$$

versus

$$\text{delay} = M_{\text{inv}} * X + B$$

the ratio of $M_{\text{nand2}}/M_{\text{inv}}$ is the logical effort of the Nand2 since the Cload ratios are the same, and Tau is the same.

BR 6/00

5

Logical Effort of Nand2, Nand4

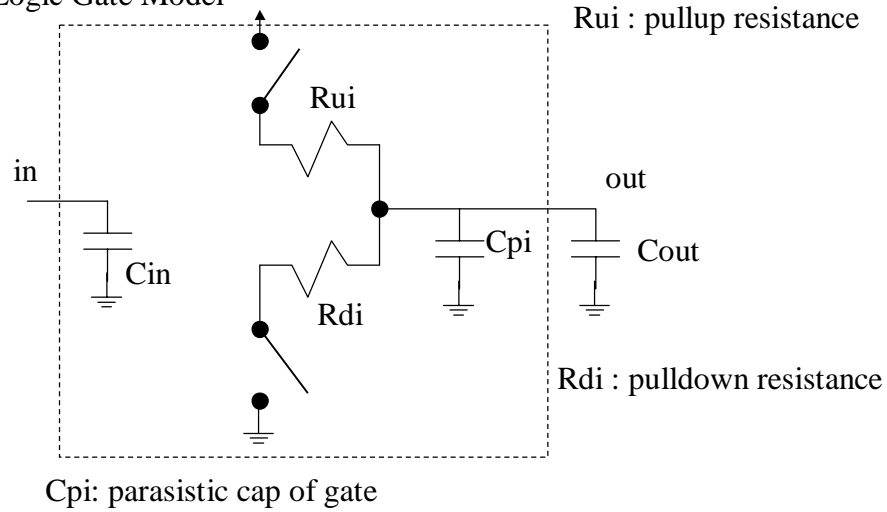
| | Nand2 g | Nand4 g |
|----------|---------|---------|
| Book | 1.33 | 2 |
| Measured | 1.6 | 2.2 |

BR 6/00

6

Derivation of Logical Effort Equations

Logic Gate Model



BR 6/00

7

Tau

Tau (τ) is the absolute delay of a 1x inverter driving 1x inverter *with no parasitics*. We assume equal pullup/pulldown R_{inv} , and $C_{in} = C_{inv}$, so:

$$\text{Tau} = \kappa * R_{inv} * C_{inv}$$

where κ is a constant characteristic of the fabrication process that relates RC time constants to delay.

Note: Tau is NOT the no-load delay of an inverter. Also, it is not the delay of a 1x inverter driving a 1x inverter since this includes the parasitic delay!

BR 6/00

8

Template Circuit

A template circuit is chosen as the basis upon which other gates are scaled. The scaling factor is α .

C_t is the input cap of the template.

R_t is the pullup or pulldown resistance of the template.

C_{pt} is the parasitic capacitance of the template.

$$C_{in} = \alpha * C_t$$

$$R_i = R_{ui} = R_{di} = R_t / \alpha$$

$$C_{pi} = \alpha * C_{pt}$$

BR 6/00

9

RC Delay

$$D_{abs} = \kappa R_i (C_{out} + C_{pi})$$

$$= \kappa (R_t / \alpha) C_{in} (C_{out} / C_{in}) + \kappa (R_t / \alpha) (\alpha C_{pt})$$

$$= (\kappa R_t C_t) (C_{out} / C_{in}) + \kappa R_t C_{pt}$$

Written in this form, can see relation to logical effort model:

$$D_{abs} = \tau (gh + p)$$

$$\tau = \kappa R_{inv} C_{inv} \quad (\text{previous definition})$$

$$g = (R_t C_t) / (R_{inv} C_{inv}) \quad \text{Note: if template = 1X inverter, then } g = 1 \text{ !!!!}$$

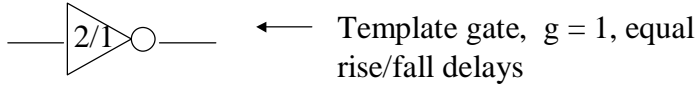
$$h = C_{out} / C_{in}$$

$$p = (R_t C_{pt}) / (R_{inv} C_{inv}) \quad \text{Note: book value of } P_{inv} = 1 \text{ only true if } C_{pt} = C_{inv}!!$$

BR 6/00

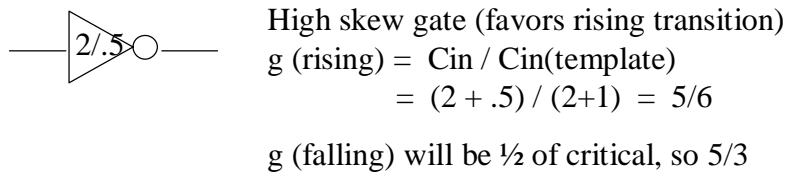
10

Logical Effort of Skewed Gates (unequal rise/fall delays)



The Leffort definition of a skewed gate is a gate that will produce the same output current for the critical transition as the template gate, and produce less current for the non-critical transition.

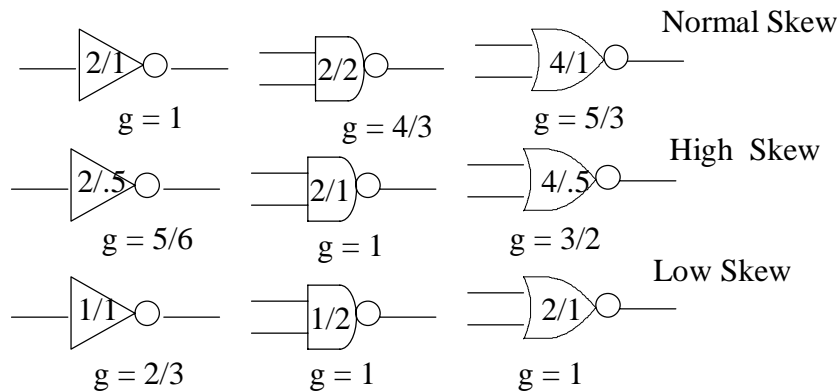
Assume that for a high skew gate, we reduce the non-critical current by $\frac{1}{2}$ (.5). Then:



BR 6/00

11

Skewed Gates

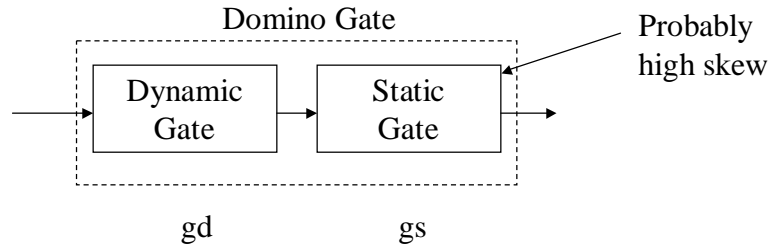


Logical efforts shown for critical transitions. Note that skewed gates always have lower logical efforts for the critical transition than the corresponding non-skewed version.

BR 6/00

12

Logical Effort of Domino Gates



$$g(\text{domino}) = g(\text{dynamic}) * g(\text{static})$$

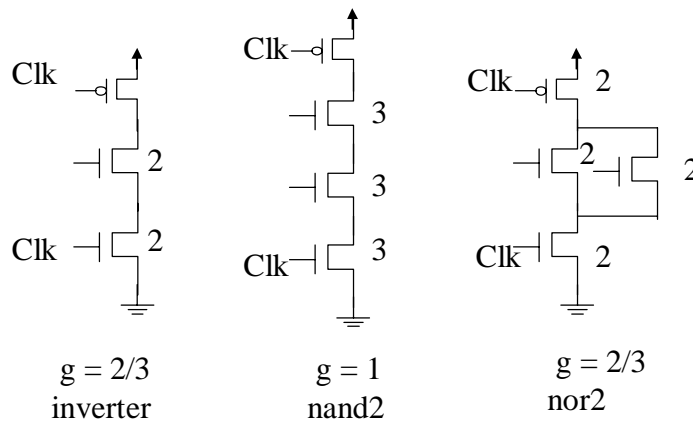
Both dynamic and static portions enter into the stage count for sizing purposes (so a minimum 2-stage circuit)

BR 6/00

13

Logical Effort Calculation for Dynamic Gates

Only compute logical effort based on N-tree, size N-tree relative to Template gate, which is 2/1 inverter

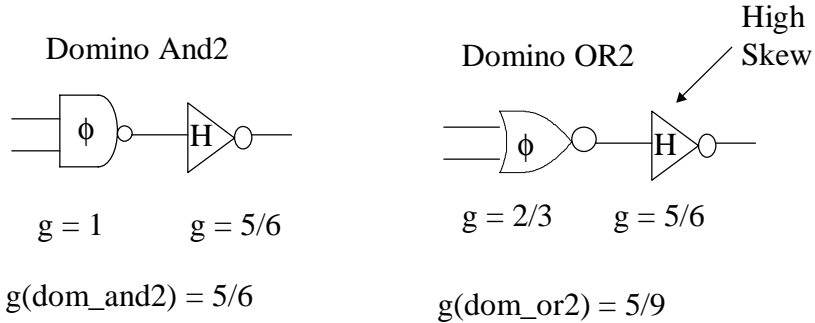


Clock capacitance load does not count in logical effort

BR 6/00

14

Domino Logic Effort



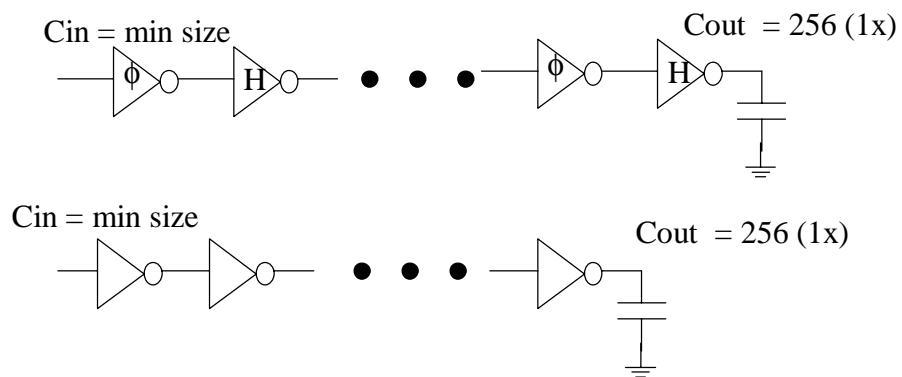
Note that logical efforts of dynamic gates less than static counterparts. This is because of less load on inputs. Also dynamic gates start switching at $V_{in} = V_t$ since no P-tree.

BR 6/00

15

A Sizing Problem

Determine Number of Stages to drive a load

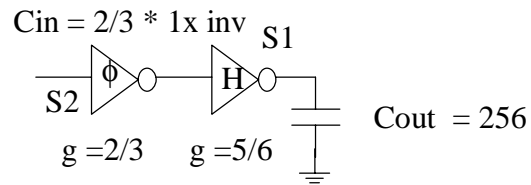


Will not worry about inversion/non-inversion of signal

BR 6/00

16

Dynamic Logic Sizing



$$G = 2/3 * 5/6 = 0.56, \quad B = 1, \quad H = 256 / (2/3) = 384$$

$$F = 0.56 * 1 * 384 = 215$$

$$F_{min} = (215)^{1/2} = 14.6$$

$$S1 = g * 256 / F_{min} = (5/6) * 256 / 14.6 = 14.6$$

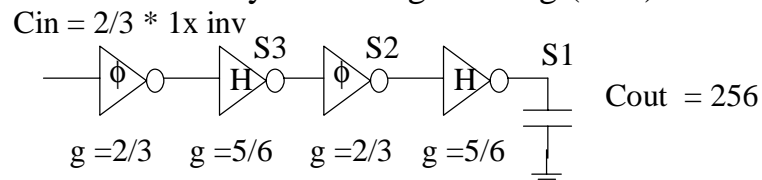
$$S2 = g * 14.6 / F_{min} = (2/3) * (14.6) / (14.6) = 2/3 \text{ (consistent)}$$

$$\begin{aligned} \text{Delay} &= 2(14.6) + P_{invH} + P_{dynamic} \\ &= 29.2 + P_{invH} + P_{dynamic} \end{aligned}$$

BR 6/00

17

Dynamic Logic Sizing (cont).



$$H = 256 / (2/3) = 384$$

$$G = (2/3 * 5/6)^2 = (0.56)^2 = .31$$

$$F = 0.31 * 1 * 384 = 119$$

$$F_{min} = (119)^{1/4} = 3.3$$

$$S1 = g * 256 / 3.3 = (5/6) * 256 / 3.3 = 64.6$$

$$S2 = g * 64.6 / 3.3 = (2/3) * 64.6 / 3.3 = 13$$

$$S3 = g * 13 / 3.3 = (5/6) * 13 / 3.3 = 3.3$$

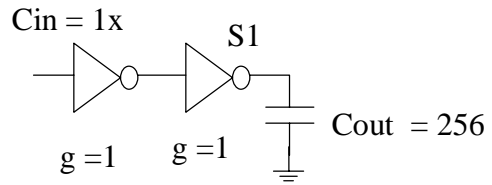
$$S4 = g * 3.3 / 3.3 = (2/3) * 1 = 2/3 \text{ (consistent)}$$

$$\begin{aligned} \text{Delay} &= 4 * (3.3) + 2 * P_{invH} + 2 * P_{dynamic} \\ &= 13.3 + 2 * P_{invH} + 2 * P_{dynamic} \end{aligned}$$

BR 6/00

18

Static Logic Sizing



$$G = 1, H = 256, B = 1$$

$$F_{min} = (256)^{1/2} = 16$$

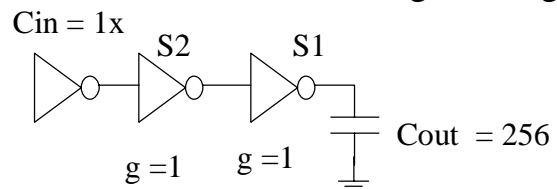
$$S1 = 256/16 = 16$$

$$S2 = 16/16 = 1 \text{ (consistent)}$$

BR 6/00

19

Static Logic Sizing (cont)



$$G = 1, H = 256, B = 1$$

$$F_{min} = (256)^{1/3} = 6.3$$

$$S1 = 256/6.3 = 40.6$$

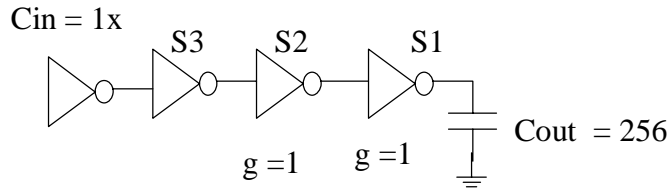
$$S2 = 40.6/6.3 = 6.3$$

$$S3 = 6.3/6.3 = 1 \text{ (consistent)}$$

BR 6/00

20

Static Logic Sizing (cont)



$$G = 1, H = 256, B = 1$$

$$F_{min} = (256)^{1/4} = 4$$

$$S1 = 256/4 = 64$$

$$S2 = 64/4 = 16$$

$$S3 = 16/4 = 4$$

$$S4 = 4/4 = 1 \text{ (consistent)}$$

BR 6/00

21

Results

| Dynamic | Meas (T _{plh}) | Predicted |
|-------------|--------------------------|-----------|
| Two Stage | 518.0 | 392.8 |
| Four Stage | 488.0 | 351.7 |
| Static | Meas (T _{avg}) | Predicted |
| two stage | 634.0 | 416.6 |
| Three Stage | 547.0 | 345.6 |
| Four Stage | 562.0 | 372.5 |

Leffort correctly predicted relative magnitudes. Leffort does not capture all of the speedup that will occur in a dynamic implementation. Dynamic implementation benefits from more stages than static.

BR 6/00

22

Optimum Number of Stages?

- What is the optimum number of stages? (N_{opt})
- Depends on the relative magnitude of the parasitic gate delays to the delay due to the stage effort
 - Why? Because as you add more stages, you add a fixed parasitic delay, and the delay via each stage effort gets smaller. Larger parasitic delays means N_{opt} is smaller!
- Authors of logical effort model derive equations that relate parasitic delay to N optimum
 - We will not cover this, since when talking about *absolute delay*, also need to account for slope dependence of delay (which Leffort does not do).
- Practical approach is simply to try a few different N values (will usually not be that many choices), and see what is best. Use Leffort to size gates for a particular N

BR 6/00

23

Optimum Number of Stages? (cont).

- Rule of Thumb from Leffort authors which probably is true:

The optimum stage effort found for a string of inverters driving a load will be close to the optimum stage effort for general logic driving a load. (assuming no significant off-path load).
- For our static inverter design, we found that a stage effort of about 6 is best. So, for a general logic problem, pick number of stages such that stage effort is about 6 and this will probably be the best number of stages.
- For static logic, the book recommends 4 as the optimum stage effort, but they assume P_{inv} is between 0.7 and 2.5. Our P_{inv} is about 6, so higher parasitics means higher stage effort, which will result in less stages.

BR 6/00

24

Optimum Number of Stages? (cont.)

- For our example, the optimum stage effort for static logic was about 6, for dynamic logic was about 3.5.
- This means that to drive the same load, dynamic logic will benefit from more stages than static logic
- If the number of stages chosen is non-optimum by +/- 1, will not significantly affect the delay.
- Other considerations such as Power, Area will definitely affect the choice of the number of stages.