

Logical Effort

12 de septiembre de 2024

1. El método de Logical Effort

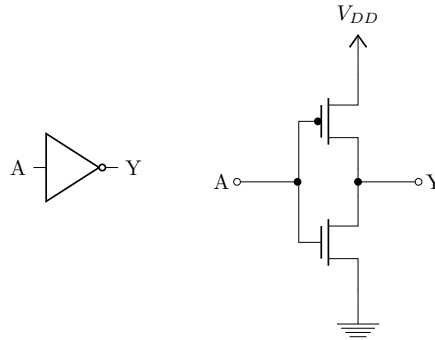


Figura 1: Inversor, símbolo y circuito.

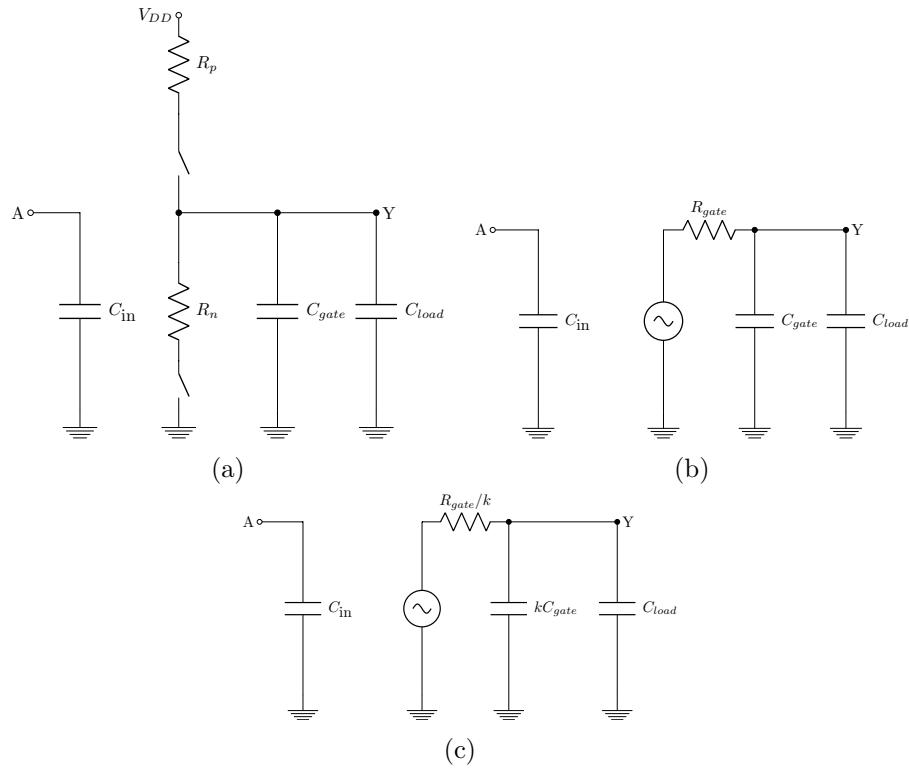


Figura 2: Inversor, modelo de primer orden

Definimos delay absoluto como:

$$d_{abs} = R_{gate}(C_{load} + C_{gate}) = \underbrace{R_{gate}C_{load}}_{\text{depende de la carga}} + \underbrace{R_{gate}C_{gate}}_{\text{constante}}$$

Definimos parasitic delay como:

$$d_p = R_{gate}C_{gate} \Rightarrow d_{abs} = R_{gate}C_{load} + d_p$$

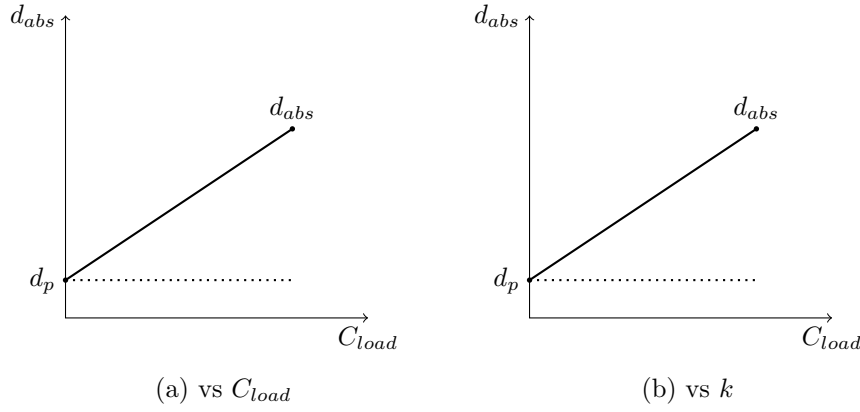


Figura 3: Parasitic Delay

Si se aumenta k veces el tamaño de la compuerta podemos suponer que tenemos el circuito representado en la Figura 2c.

Luego:

$$d_{abs} = \frac{R_{gate}}{k}(kC_{gate} + C_{load}) = \frac{R_{gate}}{k}kC_{gate} + \frac{R_{gate}}{k}C_{load} = R_{gate}C_{gate} + \frac{R_{gate}}{k}C_{load}$$

$$d_{abs} = d_p + \frac{R_{gate}}{k}C_{load}$$

Definición Esfuerzo Eléctrico (h)

$$h = \frac{C_{load}}{C_{in}}$$

Supongamos que podemos escribir $C_{in_{gate}} = \alpha C_{in_0}$ donde C_{in_0} es la capacidad de entrada del inversor de tamaño mínimo. Entonces:

$$\alpha = \frac{C_{in_{gate}}}{C_{in_0}} \Rightarrow \frac{1}{\alpha} = \frac{C_{in_0}}{C_{in}}.$$

Lo mismo sucede con la capacidad de salida $C_{gate} = \alpha C_{gate_0}$

Esto indica que si los transistores de la compuerta aumentan α veces respecto a la misma compuerta de tamaño mínimo entonces su capacidad aumenta α veces y su resistencia disminuye α veces.

Luego:

$$R_{gate} = \frac{R_{gate_0}}{\alpha} = R_{gate_0} \frac{C_{in_0}}{C_{in}}$$

Sabemos que:

$$\begin{aligned} d_{abs} &= R_{gate} C_{load} + R_{gate} C_{gate} \\ &= \frac{R_{gate_0}}{\alpha} C_{load} + \frac{R_{gate_0}}{\alpha} \alpha C_{gate_0} \\ &= \frac{R_{gate_0}}{\alpha} C_{load} + R_{gate_0} C_{gate_0} \\ &= R_{gate_0} \frac{C_{in_0}}{C_{in}} C_{load} + R_{gate_0} C_{gate_0} \\ &= \underbrace{R_{gate_0} C_{in_0}}_g \underbrace{C_{load}/C_{in}}_h + \underbrace{R_{gate_0} C_{gate_0}}_p \\ &= gh + p \end{aligned}$$

Donde:

- g : Logical effort (depende de la compuerta)
- h : Electrical effort
- p : Parasitic delay

Notar que:

$$h_0 = \frac{C_{load}}{C_{in}} = \text{cte} \Rightarrow d_{abs} = \text{cte}$$

En general se usa una constante τ que depende del proceso y se expresa todo en unidades $d_{abs} = d'_{abs} \tau = \tau(gh + d_p)$.

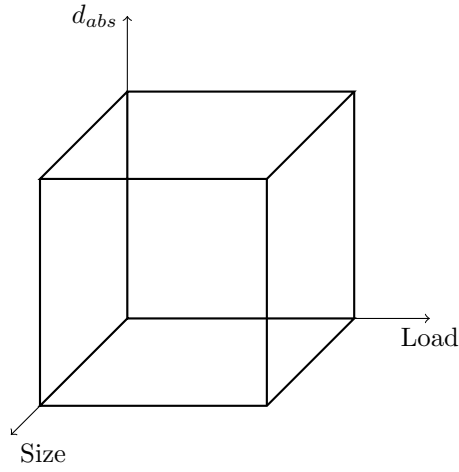


Figura 4

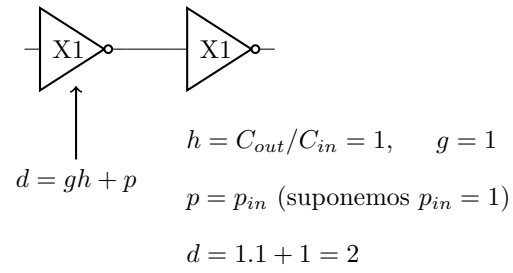
Cantidad de entradas						
g	1	2	3	4	5	n
Inverter	1	-	-	-	-	-
NAND	-	$4/3$	$5/3$	$6/3$	$7/3$	$(n+2)/3$
NOR	-	$5/3$	$7/3$	$9/3$	$11/3$	$(2n+1)/3$
XOR	-	4	12	32	-	-

(a) Algunos valores de g

Gate	p
Inversor	p_{inv}
n -NAND	np_{inv}
n -NOR	np_{inv}
2-XOR	$4p_{inv}$

(b) Algunos valores de p

2. Ejemplo



3. Ejemplo

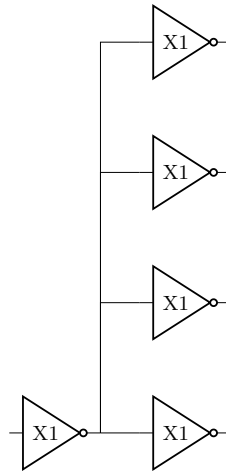
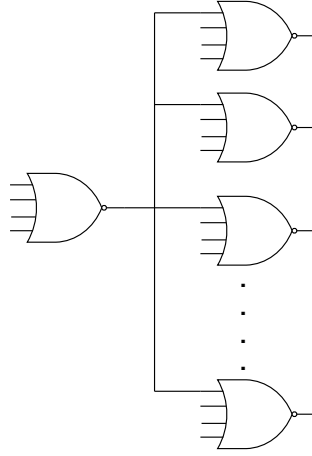


Figura 5: $h = C_{out}/C_{in} = 4, g = 1, p_{inv} = 1, d = gh + p = 4 \times 1 + 1 = 5$

4. Ejemplo 4-Input NOR

A continuación vemos un ejemplo de una NOR de 4 entrada.



$$h = C_{out}/C_{in} = 10$$

$$g = 9/3 = 3$$

$$p = Np_{inv} = N1 = 4$$

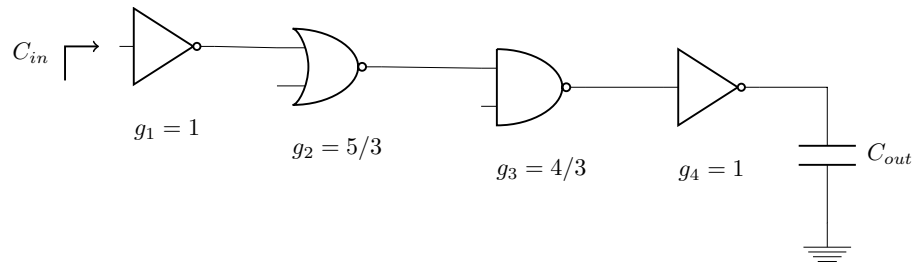
$$d = gh + p = 10 \times 3 + 4 = 34$$

Figura 6: Ejemplo 4-Input NOR

5. Circuitos multinivel

A lo largo de un path el logical effort es:

$$G = \prod g_i, H = \frac{C_{out}}{C_{in}}$$



$$G = 1 \times 5/3 \times 4/3 \times 1 = 20/9$$

Figura 7: Circuitos multinivel

En general $F \neq GH$, ya que existen circuitos con ramificaciones

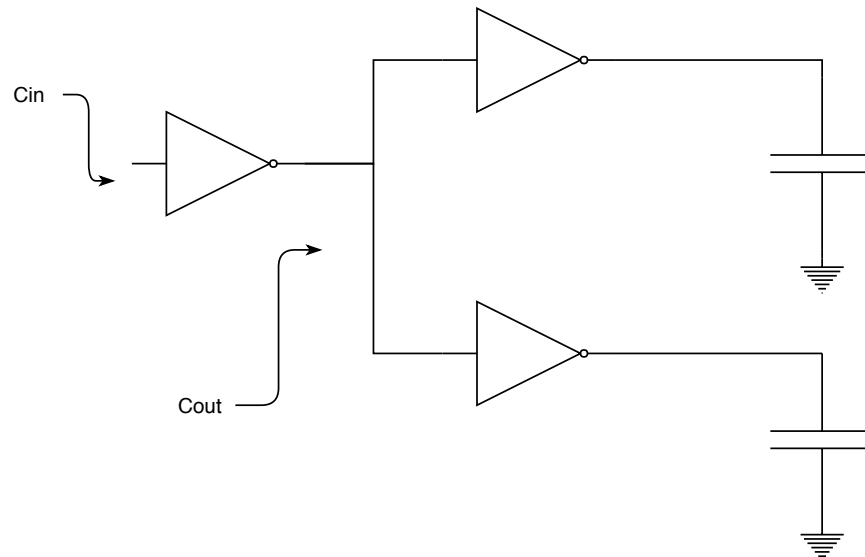


Figura 8: Branching effort

5.1. Branching Effort

$$h = \frac{C_{in}}{C_{out}} = \frac{C_{on\ path} + C_{off\ path}}{C_{in\ path}} = \frac{C_{on\ path}}{C_{in\ path}} b$$

$$b = \frac{C_{on\ path} + C_{off\ path}}{C_{in\ path}} \leq 1$$

A lo largo de un path,

$$B = \prod b_i$$

luego,

$$F = GBH \text{ (Path effort)}$$

$$\begin{aligned} BH &= \prod b_i \left(\frac{C_{out}}{C_{in}} \right) \\ &= \prod b_i \left(\frac{C_{out}}{C_{in_N}} \times \frac{C_{in_N}}{C_{in_{N-1}}} \times \dots \times \frac{C_{in_2}}{C_{in_1}} \right) \\ &= \prod \left(b_i \frac{C_{in_{i+1}}}{C_{in_i}} \right) \\ &= \prod h_i \end{aligned}$$

Observación: El *Path Effort* depende sólo de la topología del circuito y la carga y no del tamaño de las compuertas utilizadas.

Más aún, si se agregan inversores o se quitan inversores del *path* F permanece constante ya que el esfuerzo lógico del inversor es 1.

- Path Delay:

$$D = \sum d_i, \text{ donde } d_i = \text{es el delay de cada etapa a lo largo del path.}$$

- Path effort delay:

$$D_f = \sum g_i h_i$$

- Path parasitic delay

$$P = \sum p_i$$

Entonces Path Delay queda definido como:

$$D = \sum d_i = \sum (g_i h_i + p_i) = \sum g_i h_i = D_f + P$$

Teorema 1

El path delay es mínimo cuando cada etapa a lo largo del path tiene el mismo stage effort f_i . Es decir $f_i = f_j$ para todo $i, j \leq N$. Por lo tanto $f = \sqrt[N]{F}$. demostramos el valor optimo de f por el simbolo \hat{f} . Es decir: $\hat{f} = \sqrt[N]{F}$.

Colorario:

El valor mínimo del path delay es $\hat{D} = N\hat{f} + P$, donde $N\hat{f} = \hat{D}f$ para ecualizar f en cada etapa de forma que $f_i \cdot \hat{f}$, para todo i , se debe elegir los tamaños de compuertas adecuados.

Sabemos que para una compuerta $f = g.h$, entonces $\hat{h}_1 = \frac{\hat{f}_i}{g_i} = \frac{\sqrt[N]{F}}{g_i}$.

$$h_i = \frac{C_{out_i}}{C_{in_i}} \Rightarrow C_{in_i} \frac{C_{out_i}}{h_i} \Rightarrow \widehat{C_{in_i}} = \frac{C_{out_i} g_i}{\sqrt[N]{F}}$$

Es decir se empieza por la última etapa y se computan hacia atrás los valores óptimos de C_{in_i} .

Ejemplo:

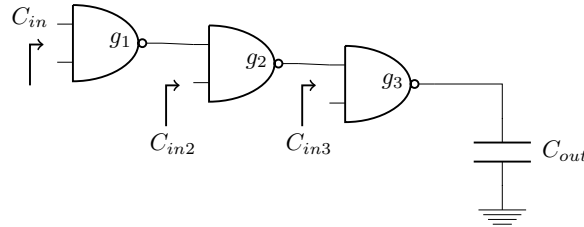


Figura 9: Path delay

Sea $C_{out} = C_{in}$:

- Cual es el mínimo delay posible ?
- Cual debe ser el tamaño de los transistores para lograr mínimo delay?

$$\left. \begin{aligned} G &= g_1 \cdot g_2 \cdot g_3 = \frac{4}{3} \frac{4}{3} \frac{4}{3} = \frac{4^3}{3^3} \\ B &= b_1 \cdot b_2 \cdot b_3 = (1) \cdot (1) \cdot (1) = 1 \\ H &= C_{out}/C_{in} = 1 \end{aligned} \right\} F = GBH = \frac{4^3}{3^3}$$

Luego:

$$\hat{D} = N \sqrt[N]{F} + P = 3 \sqrt[3]{\left(\frac{4}{3}\right)^3} + 6 = 10$$

Donde:

$$P = \sum p_i = 3.(2p_{inv}) = 6p_{inv} = 6$$

$$p_{inv} = 1$$

Por lo que el stage effort optimo seria:

$$\hat{f} = \sqrt[N]{F} = \sqrt[3]{\frac{4^3}{3}} = \frac{4}{3}$$

$$\left. \begin{array}{l} Cin_3 = \frac{C_{out.g_3}}{4/3} = C_{out} \\ Cin_2 = \frac{C_{in_3.g_2}}{4/3} = C_{out} \\ Cin_1 = C_{out} \end{array} \right\} \text{Las tres compuertas deben ser iguales}$$

y su tamaño de forma tal que $Cin = C$.

Ejemplo:

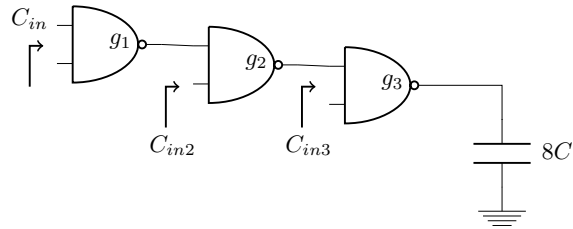


Figura 10: Path delay

$$B = (1).(1).(1)$$

$$G = (4/3)^3$$

$$H = 8C/C = 8$$

$$F = GBH = (4/3)^3.(8)$$

$$\hat{D} = N \sqrt[N]{F} + P = 3 \sqrt[3]{\left(\frac{4}{3}\right)^3.(8)} + 3p_{inv} = 3.\frac{4}{3}.(2) + 6 = 14$$

$$\hat{f} = \sqrt[N]{F} = \sqrt[3]{\frac{4^3}{3}} \cdot (8) = \frac{8}{3}$$

Se observa que cada compuerta duplica el ancho de los transistores de la anterior (lo cual era de esperarse).

$$\begin{cases} Cin_3 = \frac{Cout.g_3}{\hat{f}} = \frac{8.C.(4/3)}{8/3} = 4C \\ Cin_2 = \frac{Cin_3.g_2}{\hat{f}} = \frac{4.C.(4/3)}{8/3} = 2C \\ Cin_1 = \frac{Cin_2.g_3}{\hat{f}} = \frac{2.C.(4/3)}{8/3} = C \end{cases}$$

Debe elegirse la primer componente de forma tal que. La primera tenga una capacidad de entrada igual a C.

Ejemplo:

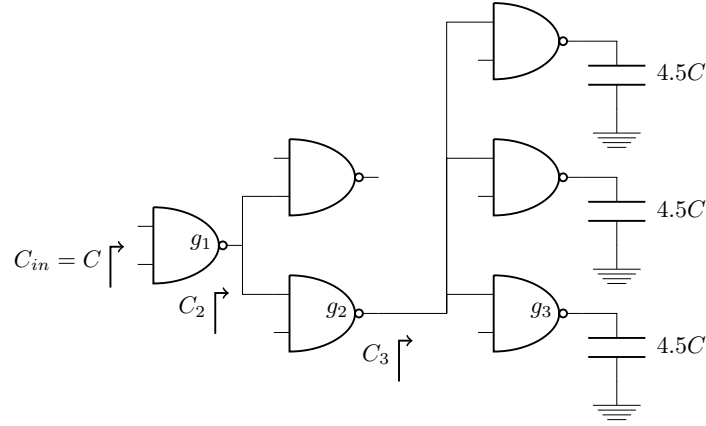


Figura 11: Path delay

$$G = g_1 \cdot g_2 \cdot g_3$$

$$B = b_1 \cdot b_2 \cdot b_3 = (2) \cdot (3) \cdot (1) = 6$$

$$H = 4,5C/C = 4,5 = 9/2$$

$$F = GBH = (4/3)^3 \cdot (6) \cdot (9/2) = 64$$

$$\hat{D} = N \sqrt[N]{F} + P = 3 \cdot \sqrt[3]{64} + 3 \cdot (2p_{inv}) = (3) \cdot (4) + (3) \cdot (2) = 18$$

$$\hat{f} = \sqrt[3]{F} = 4$$

$$Cin_3 = \frac{C_{out.g_3}}{\hat{f}} = \frac{4,5C \frac{4}{3}}{4} = 1,5C$$

$$Cin_2 = 3 \frac{Cin_3.g_2}{\hat{f}} = \frac{(3), 1,5C \frac{4}{3}}{4} = 1,5C$$

$$Cin_1 = 2 \frac{Cin_2.g_3}{\hat{f}} = \frac{(2), 1,5C \frac{4}{3}}{4} = C$$

Ejemplo:

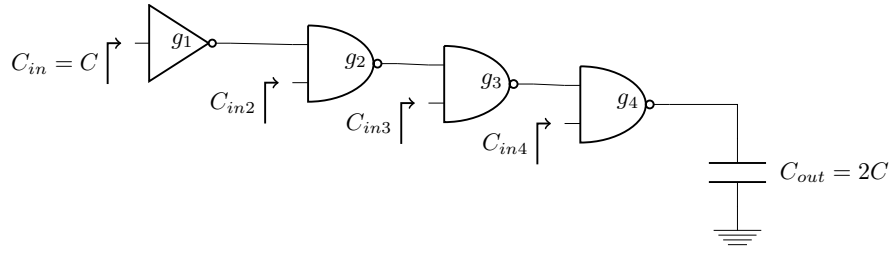


Figura 12: Path delay

$$B = b_1 \cdot b_2 \cdot b_3 \cdot b_4 = (1) \cdot (1) \cdot (1) \cdot (1) = 1$$

$$H = C_{out} = C_{in} = 2$$

$$G = g_1 \cdot g_2 \cdot g_3 \cdot g_4 = 1 \cdot (5/3) \cdot (4/3) \cdot (1) = 20/9$$

$$F = GBH = 40/9, \text{ donde } P = p_{inv} + 2p_{inv} + 2p_{inv} + p_{inv} = 6p_{inv} = 6$$

$$\hat{D} = N \sqrt[N]{F} + P = 4 \cdot \sqrt[4]{40/9} + 6 = (4) \cdot 1,45 + 6 = 11,8$$

$$\hat{F} = \sqrt[4]{F} = \sqrt[4]{40/9} = 1,45$$

$$C_{in4} = \frac{C_{out} \cdot g_4}{\hat{f}} = \frac{2C \cdot (1)}{1,45} = 1,38C$$

$$C_{in3} = \frac{C_{in4} \cdot g_3}{\hat{f}} = \frac{1,38C \cdot (4/3)}{1,45} = 1,27C$$

$$C_{in2} = \frac{C_{in3} \cdot g_2}{\hat{f}} = \frac{1,27C \cdot (5/3)}{1,45} = 1,45C$$

$$C_{in1} = \frac{C_{in2} \cdot g_1}{\hat{f}} = \frac{1,45C}{1} \cdot (1) = 1,45C$$

Ejemplo:

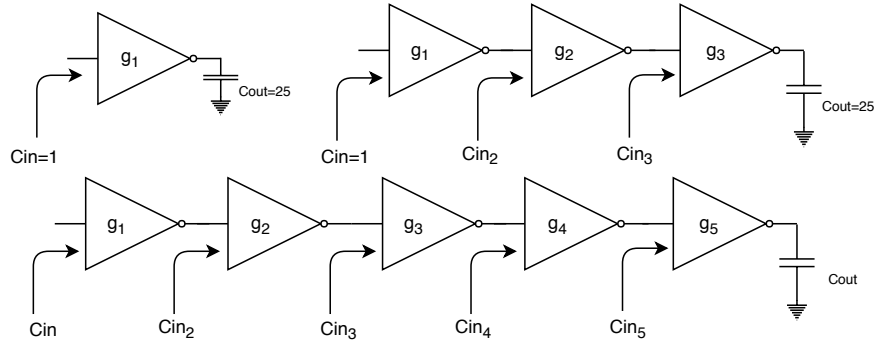


Figura 13: Ejemplos calculos Delay mínimo

$$H = 25$$

$$P = Np_{inv} = N$$

$$B = 1$$

$$G = \prod g_i = 1$$

$$F = GBH = 25$$

- 1 INV, entonces $\hat{D} = N \sqrt[N]{F} + P = 25 + 1 = 26$
- 2 INV, entonces $\hat{D} = 3 \sqrt[3]{25} + 3 = 11,17$
Este es la mejor elección.
- 5 INV entonces $\hat{D} = 5 \sqrt[5]{25} + 5 = 14,51$

5.2. Demostración del teorema 1

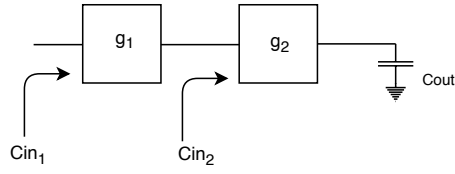


Figura 14: Esquema Calculo Delay path

$$D = \sum (g_1 k_i + p_i) = g_1 \frac{C_{in2}}{C_{in1}} + p_i + g_2 \frac{C_{out}}{C_{in2}} + p_2$$

$$b_1 = b_2 = 1$$

$$H = h_1 h_2 = \frac{C_{out}}{C_{in1}}$$

Entonces:

$$h_2 = H/h_1$$

Luego:

$$D = g_1 h_1 + p_1 + g_2 \frac{H}{h_1} + p_2$$

Para buscar el mínimo derivamos e igualamos a cero.

$$\frac{dD}{dh_1} = g_1 - g_2 h \frac{1}{h_1^2} = g_1 - g_2 \frac{h_2}{h_1} = 0$$

Entonces

$$g_1 h_1 = g_2 h_2$$

$$f_1 = f_2$$

.

Este resultado se puede extender a cualquier número arbitrario de etapas y branching efforts.

Sabemos que:

$$BH = h_1 h_2 \dots h_n$$

$$G = g_1 g_2 \dots g_n$$

$$(g_1 h_1)(g_2 h_2) \dots (g_n h_n) = GBH = F$$

Todos los factores, entre paréntesis, deben ser iguales para obtener mínimo delay, entonces:

$$\hat{f} = \sqrt[N]{F}$$

.

Veamos ahora cuántas etapas deben tener el path para obtener el mínimo delay.

Sea un path que contiene n_1 compuertas al que se le agregan n_2 inversores obteniéndose $N = n_1 + n_2$. Se asume que las n_1 compuertas originales sólo pueden cambiar de tamaño ya que generan la función lógica requerida mientras que la cantidad n_2 de inversores puede ser alterada para reducir el retardo del path. Aunque n_2 debe ser par para no alterar la función lógica, se asume que si n_2 es impar se puede cambiar la función lógica a su valor negado. Se supone conocido el path effort $F = GBH$, el lógico effort G y el branching effort B dependen de las n_1 compuertas originales y no pueden ser alterados agregando inversores. H es constante ya que depende de las cargas de entrada y salida que consideramos inalterables. El retardo mínimo de las N etapas es la suma del retardo de las n_1 compuertas originales y los n_2 inversores.

$$\hat{D} = N \sqrt[N]{F} + \sum_{i=1}^{N_1} f_i + (N - n_1) p_{inv}$$

donde $N - n_1 = n_2$.

$$\frac{d\hat{D}}{dN} = F^{1/N} - F^{1/N} \ln(F^{1/N}) + p_{inv}$$

Entonces: \hat{N} es el valor de N que hace que

$$\frac{d\hat{D}}{dN} = 0$$

Sea ahora $\rho = F^{1/\hat{N}}$ entonces $p_{inv} + \rho(1 - \ln \rho) = 0$

Cuando cada etapa tiene un esfuerzo lógico ρ , se alcanza el mínimo retardo. Se denomina ρ como *best stage effort*.

Observación 1:

- \hat{f} determina el f óptimo cuando N es conocido fijo.
- ρ determina el f óptimo que tiene cada etapa cuando el N es óptimo ($N = \hat{N}$).

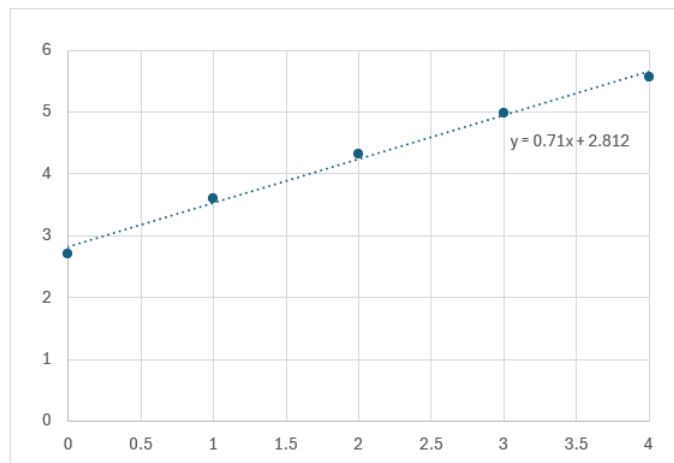
Observación 2:

Se puede tabular la solución de la ecuación $\rho = F^{1/\hat{N}}$ entonces $p_{inv} + \rho(1 - \ln \rho) = 0$ para los distintos valores de p_{inv} :

p_{inv}	ρ
0	2.71 (e)
1	3.59
2	4.32
3	4.97
4	5.57

Esta tabla suele aproximar por $\rho = 0,71 + 2,812p_{inv}$, y para el caso particular de $p_{inv} = 1$ se tiene que $\rho = 3,59$.

p_{inv}	ρ
0	2.71
1	3.59
2	4.32
3	4.97
4	5.57



5.3. Procedimiento

1. Hallar ρ .
2. Conociendo F hallar $\hat{N} = \log_\rho(F) = \log_{10}(F)/\log_{10}(\rho)$.
3. Teniendo ρ y b_i hallar el tamaño de las compuertas (capacidades de entradas).
4. El delay mínimo será $\hat{D} = \hat{N}\rho + \sum p_i$.

6. Qué sucede si no se usa el valor óptimo de \hat{N}

Sabemos que

$$\hat{D} = \hat{N} \sqrt[\hat{N}]{\hat{N}F} + \sum p_i \quad (1)$$

Luego

$$D(N) = N \left(\sqrt[N]{F} + \frac{\sum p_i}{N} \right) \quad (2)$$

Sea $p = \sum p_i/N$ y sea $N = s\hat{N}$, entonces

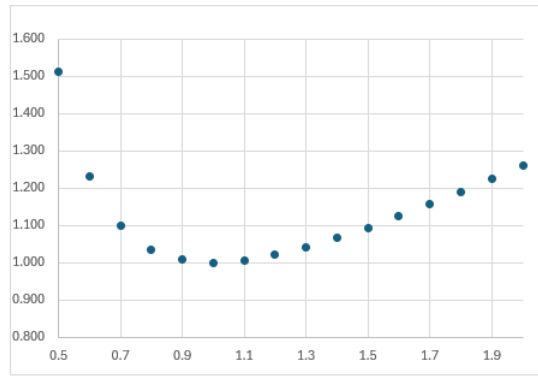
$$r = \frac{D(N)}{D(\hat{N})} = \frac{D(s\hat{N})}{D(\hat{N})} = \frac{s(\rho^{1/s} + p)}{\rho + p} \quad (3)$$

Si $p = 1$ entonces $\rho = 3,59$, entonces por ejemplo:

- Si $s = 2$, $N = 2\hat{N}$ entonces $D(N) = 1,26D(\hat{N})$
- Si $s = 0,5$, $N = \hat{N}/2$ entonces $D(N) = 1,51D(\hat{N})$

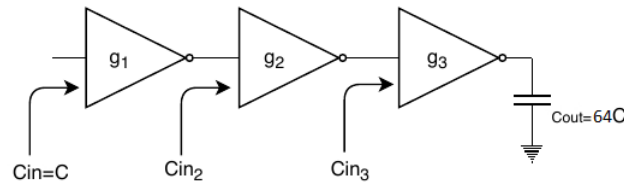
Se concluye entonces que equivocarse entre la mitad o el doble de la cantidad de etapas óptimas sube el retardo entre un +26 % y un +51 %.

s	D(N)
0.5	1.513
0.6	1.231
0.7	1.099
0.8	1.036
0.9	1.007
1	1.000
1.1	1.006
1.2	1.020
1.3	1.040
1.4	1.065
1.5	1.093
1.6	1.123
1.7	1.156
1.8	1.190
1.9	1.225
2	1.261



7. Qué sucede si no se usa el tamaño óptimo para cada compuerta

Considerar el siguiente ejemplo:



$$g_1 = g_2 = g_3 = 1, G = 1$$

$$b_1 = b_2 = b_3 = 1, B = 1$$

$$H = 64C/C = 64$$

$$F = GBH = 64$$

$$\hat{f} = \sqrt[3]{F} = 4$$

Luego

$$\begin{aligned} C_{IN3} &= \frac{C_{OUT} \cdot g_3}{\hat{f}} = \frac{64C \cdot 1}{4} = 16C \\ C_{IN2} &= \frac{C_3 \cdot g_2}{\hat{f}} = \frac{16C \cdot 1}{4} = 4C \\ C_{IN1} &= \frac{C_2 \cdot g_1}{\hat{f}} = \frac{4C \cdot 1}{4} = C \end{aligned}$$

Supongamos ahora que $C_{IN2} = s \cdot 4C$ entonces:

$$h_1 = \frac{C_{IN_2}}{C_{IN_1}} = \frac{s \cdot 4C}{C} = 4s$$

$$h_2 = \frac{C_{IN_3}}{C_{IN_2}} = \frac{16C}{s \cdot 4C} = \frac{4}{s}$$

$$h_3 = \frac{C_{OUT}}{C_{IN_3}} = \frac{64C}{16C} = 4$$

Luego,

$$f_1 = g_1 h_1 = 1 \cdot 4s = 4s$$

$$f_2 = g_2 h_2 = 1 \cdot 4/s = 4/s$$

$$f_3 = g_3 h_3 = 1 \cdot 4 = 4$$

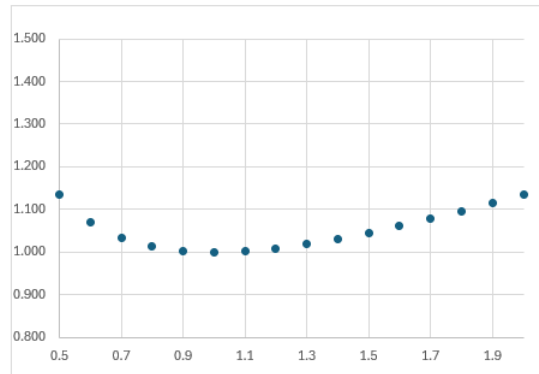
$$F = f_1 f_2 f_3 = 4s \cdot \frac{4}{s} \cdot 4 = 64$$

$$D = \sum f_i + \sum p_i = \left(4s + \frac{4}{s} + 4\right) + (1 + 1 + 1) = 4(s + s^{-1}) + 7$$

$$\frac{D(s)}{D(1)} = \frac{4}{15}(s + s^{-1}) + 0,467$$

$$\frac{dD}{ds} = 4 \left(1 - \frac{1}{s^2}\right) = 0 \quad \text{si y sólo si} \quad s = 1$$

s	D(s)/D(1)
0.3	1.436
0.4	1.240
0.5	1.133
0.6	1.071
0.7	1.034
0.8	1.013
0.9	1.003
1.0	1.000
1.1	1.002
1.2	1.009
1.3	1.018
1.4	1.030
1.5	1.044
1.6	1.060
1.7	1.077
1.8	1.095
1.9	1.114
2.0	1.133



Observación: Si se modifica el tamaño de la compuerta entre 0.5 y 2 veces respecto del valor óptimo, se tiene un aumento del retardo de un 13 %.

8. Definiciones equivalentes de Logical effort

- El logical effort de una compuerta se define como la cantidad de veces que la resistencia de salida de una compuerta es mayor a la de un inversor de idéntica capacidad de entrada.
- El logical effort de una compuerta se define como la relación entre su capacidad de entrada y la de un inversor con misma resistencia de salida.
- El logical effort de una compuerta se define como la pendiente de su retardo en función de la carga (C_L) dividida por la pendiente de la curva de retardo en función de la carga (C_L) de un inversor de la misma capacidad de entrada.

9. Multiple inputs gates

- Esfuerzo lógico por entrada: Efectividad de una entrada en controlar la corriente de salida respecto a un inversor de misma a_n .
- Esfuerzo lógico de un grupo de entradas: Efectividad de un grupo de entradas en controlar la corriente de salida respecto a un inversor de misma f_{in} .
- Esfuerzo lógico total: Efectividad de la totalidad de las entradas en controlar la corriente de salida respecto de un inversor de misma c_{in}

Sea b un grupo de entrada de una compuerta específica entonces:

$$g_b = \frac{C_b}{C_{inv}} = \frac{\sum_b C_i}{C_{inv}}$$

Donde c_{inv} es la capacidad de entrada de un inversor que entrega la misma cantidad de corriente.

Luego: $g_b = \sum_b g_i$, suma de los esfuerzos lógicos de cada entrada.

Supongamos que en un compuerta CMOS todos los transistores tienen el mismo L y todos los NMOS tiene un width W_N y los PMOS W_P .

Podemos decir que:

La capacidad de entrada de un transistor NMOS es:

$$C_{in} = k.W_N$$

Y la capacidad de entrada para un PMOS es:

$$C_{in} = k.W_P$$

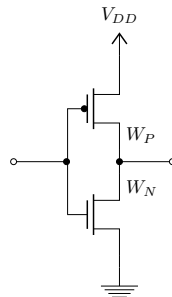


Figura 15: Esquema inversor CMOS

Supongamos $W_P = \alpha W_N$, entonces:

$$\begin{aligned} C_{in} &= k.W_P + k.W_N = k(W_P + W_N) \\ &= k.W_N \frac{W_P}{W_N} + \frac{W_N}{W_N} = kW_N \left(\frac{W_P}{W_N} + 1 \right) \end{aligned}$$

Sea: $k.W_N = W$, entonces $C_{in} = W(\alpha + 1)$.

Supongamos $\alpha = 2$, entonces $C_{in} = 3w$.

9.1. Ejemplo NAND

L igual para todos los T_r

AB	A.B	\overline{AB}
00	0	1
01	0	1
10	0	1
11	1	0

Cuadro 2: Tabla de verdad ejemplo NAND

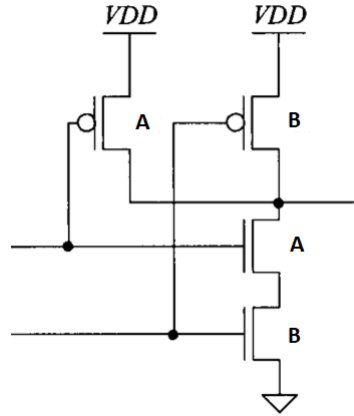


Figura 16: CMOS NAND

Supongamos que R_{on_N} y R_{on_P} para cierto $\alpha = \frac{W_P}{W_N}$, (L igual para todos los $T_{r's}$).

Quiero:

$$(R_{PON} = R_{on_{NA}} + R_{on_{NB}}) = R_{PON} = R_{on_{PA}} = R_{on_{PB}}$$

Donde:

$$R_{on_{NA}} + R_{on_{NB}} = 2R_{on_{NA}} = 2R_{on_{NB}}$$

Si definimos:

$$Cin_A = 2 + 2 = 4$$

$$Cin_B = 2 + 2 = 4$$

$$g_i = \frac{Cin_i}{C_{inv}} = 4/3$$

$$g = g_A + g_B = 4/3 + 4/3 = 8/3$$

Entonces:

$$\begin{cases} \text{Luego } 2Ron_{NA} = Ron_{PA} \Rightarrow Ron_{NA} = \frac{1}{2}Ron_{PA} \\ \text{Como } Ron_N = \beta_N/W_N; Ron_P = \beta_P/W_P \\ \beta_N/W_N = \frac{1}{2}\beta_P/W_P = \beta_P(\alpha W_N) \Rightarrow \alpha = \frac{1}{2}\frac{\beta_P}{\beta_N} = 1 \end{cases}$$

9.2. Ejemplo NOR

L igual para todos los T_r

AB	OR	NOR
00	0	1
01	1	0
10	1	0
11	1	0

Cuadro 3: Tabla de verdad ejemplo NOR

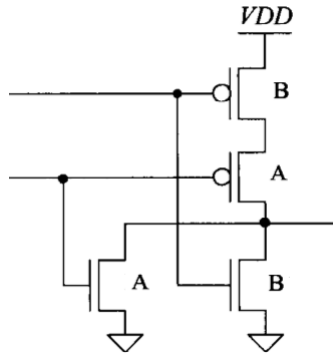


Figura 17: CMOS NOR

Suponemos que:

$$Cin_A = 5$$

$$Cin_B = 5$$

$$g_i = \frac{Cin_i}{Cin_o} = \frac{5}{3}$$

$$g = g_A + g_B = \frac{10}{3}$$

Entonces:

$$R_{PON} = R_{PDN}$$

$$Ron_{PA} + Ron_{PB} = Ron_{NA} = Ron_{NA}$$

$$2Ron_{PA} = Ron_{NA}$$

$$Ron_N = \beta_N / W_N$$

$$Ron_P = \beta_P / W_P$$

Donde $\alpha = \frac{W_P}{W_N}$

$$2\beta_P / W_P = \beta_N / W_N$$

$$2\beta_P / W_{NO} = \beta_N / W_N$$

$$\alpha = 2 \frac{\beta_P}{\beta_N} = 4$$

Spice simulation para determinar el logical effort

Sabemos que el logical effort de un inversor 1. Luego $d_{abs} = (g.h + p_{inv})\tau$ como $g = 1$, entonces $d_{abs} = \tau(h + p_{inv})$. Se levanta la siguiente curva por medio de simulación.

$$d_{abs} = (h + p_{inv})\tau$$

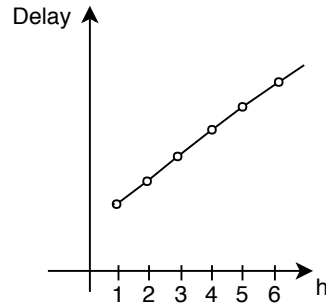


Figura 18: Curva de Delay

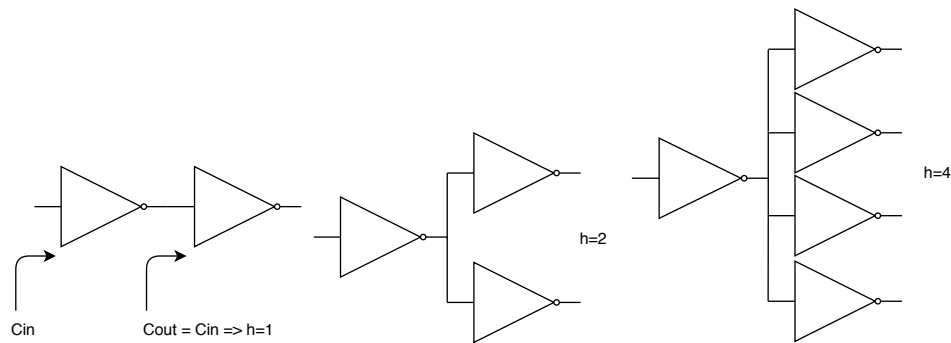


Figura 19: Modelos simulados

- Si $h = 0$, entonces $d_{abs} = p_{inv}\tau$. $p_{inv} = 1$, entonces obtengo el τ del proceso.
- Luego con este valor de τ se caracterizan las demás compuertas por medio de un procedimiento similar.

Observación

Pueden utilizarse compuertas asimétricas donde el lógico effort sea distinto para cada entrada. Esto da flexibilidad para la optimización de ciertos paths.

Por ejemplo:

Si en el siguiente circuito quisieramos optimizar el path 2 sin alterar el path 1 podemos utilizar una NAND asimétrica.

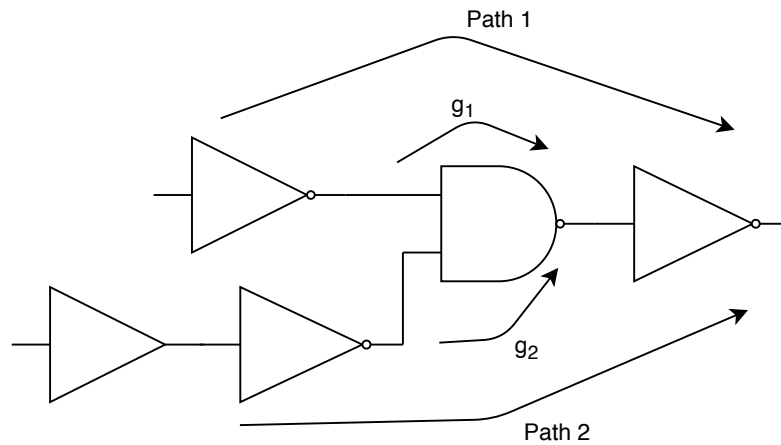


Figura 20: NAND Asimétrica.

10. Preguntas para pensar

1. Qué sucede si para un determinado período de operación requerido de clock, no se cumple en un path R2R con la restricción de setup time? Como puede hacer que el dicho path propague más rápido?
2. Discutir la necesidad de compuertas de misma función lógica pero distinto tamaño.
3. Discutir la necesidad de compuertas multi entrada asimétricas de forma de optimizar un path pero no alterar el otro.
4. Discutir como influyen los parásitos R-C en las interconexiones.