

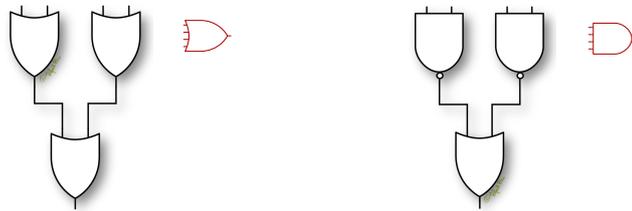
PROBLEMAS RESUELTOS: Introducción

1. Entendemos por *fan-in* el número de entradas que pueden atacar o estimular una puerta lógica sin distorsionar su funcionamiento. Asimismo, entendemos por *fan-out* el número de puertas a las que puede excitar una salida directamente sin menoscabo de la señal.

En muchas ocasiones, el resultado de una ecuación lógica impone el uso de puertas lógicas con un número de entradas n que excede su *fan-in* (f_{in}). La solución a este problema de implementación consiste en disponer las puertas lógicas en cascada. Evalúe la complejidad *hardware* tanto en coste como en retardo en función de n y f_{in} .

SOLUCIÓN:

La figura siguiente ilustra lo que queremos hacer. Si no disponemos de una puerta OR o NAND de 4 entradas podemos hacer uso del álgebra de Boole para construir un circuito lógico que nos de la función buscada con un arreglo en cascada.



Como vemos, el número de puertas es el correspondiente a una serie geométrica basada en el parámetro *fan-in* (f_{in}). Las n entradas se reparten en un primer nivel de n/f_{in} puertas lógicas. Las salidas de dichas puertas se reparten en un segundo nivel de $\frac{n/f_{in}}{f_{in}}$ puertas lógicas. Y así en sucesivos pasos hasta confluir en una única puerta. En definitiva, el número total de puertas será:

$$A(n) = \frac{n}{f_{in}} + \frac{n/f_{in}}{f_{in}} + \frac{n/f_{in}^2}{f_{in}} \dots = \frac{n}{f_{in}} + \frac{n}{f_{in}^2} + \frac{n}{f_{in}^3} + \dots$$

Por tanto, la serie geométrica es:

$$A(n) = \sum_{i=1}^{\infty} n \cdot \left(\frac{1}{f_{in}}\right)^i = \frac{n}{f_{in} - 1}$$

Recordamos:

La serie geométrica de razón $r < 1$ y término inicial a converge:

$$\sum_{i=0}^{\infty} ar^i = \frac{a}{1-r} \quad \rightarrow \quad \sum_{i=1}^{\infty} a \cdot r^i = \frac{a}{1-r} - a = \frac{ar}{1-r}$$

En definitiva, la complejidad *hardware* en coste es $A(n) = \mathcal{O}\left(\frac{n}{f_{in} - 1}\right)$

En cuanto al retardo, es proporcional al número de niveles que hayamos de pasar de la entrada a la salida:

$$L(n) = \mathcal{O}(\log_{f_{in}} n)$$

2. Disponemos de un programa de prueba del que conocemos el recuento r de instrucciones que se ejecutan cuando se procesa en una determinada arquitectura. También disponemos de dos máquinas de dicha arquitectura: una viene caracterizada por un MIPS de 80 millones instrucciones/s y una frecuencia de reloj de 50MHz y otra, más moderna, tiene un MIPS de 100 millones instrucciones/s a una frecuencia de 60MHz. Determine el *speed-up* obtenido al usar la máquina más moderna respecto a la antigua. ¿Hay mejora?

SOLUCIÓN:

Sabemos que el *speed-up* relaciona tiempos:

$$S = \frac{t_{\text{sin}}}{t_{\text{con}}} = \frac{t_{\text{antes}}}{t_{\text{después}}}$$

También sabemos que:

$$\left. \begin{aligned} t_p &= r_p \times CPI_p \times \tau \\ CPI_p &= \frac{f}{MIPS_p} \\ \tau &= \frac{1}{f} \end{aligned} \right\} \Rightarrow t_p = \frac{r_p}{MIPS_p}$$

De donde:

$$S = \frac{t_{\text{sin}}}{t_{\text{con}}} = \frac{MIPS_{\text{después}}}{MIPS_{\text{antes}}} = \frac{100}{80} = \frac{5}{4} = 1.25$$

Sí hay mejora puesto que el *speed-up* S sale mayor que 1.

3. Sea un procesador que trabaja a una velocidad de reloj de 500Mhz. El CPI medio para las aplicaciones que habitualmente corre es de 1.45 ciclos/instrucción. Sabemos que los tiempos consumidos en la ejecución de un determinado proceso son los siguientes: tiempo de pared 0.5s, tiempo de CPU 0.3s y tiempo de sistema 0.18s. Expresa cada uno de esos tiempos en ciclos y determine el recuento del proceso. Suponga que conseguimos reducir el recuento del proceso en un 8%. Bajo dicho supuesto, calcule el nuevo tiempo de ejecución y el *speed-up* obtenido.

SOLUCIÓN:

Los tiempos en ciclos son:

$$\begin{aligned} c_w &= t_w \times f = 0.5 \times 500 \cdot 10^6 = 250 \cdot 10^6 \text{ ciclos} \\ c_{\text{CPU}} &= t_{\text{CPU}} \times f = 0.3 \times 500 \cdot 10^6 = 150 \cdot 10^6 \text{ ciclos} \\ c_{\text{sys}} &= t_{\text{sys}} \times f = 0.18 \times 500 \cdot 10^6 = 90 \cdot 10^6 \text{ ciclos} \end{aligned}$$

El recuento r_p se refiere al tiempo de CPU:

$$r_p = \frac{c_{\text{CPU}}}{CPI_p} = \frac{150 \cdot 10^6}{1.45} = 103.44 \cdot 10^6 \text{ instrucciones}$$

Si el recuento r_p disminuye un 8% tendremos:

$$t_p \text{ nuevo} = 0.92 \cdot r_p \times CPI_p \times \tau$$

El *speed-up* S será:

$$S = \frac{t_{\text{sin}}}{t_{\text{con}}} = \frac{r_p \times CPI_p \times \tau}{0.92 \cdot r_p \times CPI_p \times \tau} = \frac{1}{0.92} = 1.086$$