



ARQUITECTURA DE COMPUTADORAS

Trabajo Práctico N° 4

Implementación de Operaciones Básicas: Suma y Resta ¹

Primer Cuatrimestre de 2018

Ejercicios

1. Verificar la validez de las siguientes expresiones analizando todas las combinaciones posibles:

a) Expresión lógica para la suma:

$$\begin{aligned}s_i &= x_i \oplus y_i \oplus c_i \\ c_{i+1} &= x_i \cdot y_i + (x_i \oplus y_i) \cdot c_i\end{aligned}$$

b) Expresión lógica para la resta:

$$\begin{aligned}s_i &= x_i \oplus y_i \oplus b_i \\ b_{i+1} &= \bar{x}_i \cdot y_i + (\bar{x}_i \oplus y_i) \cdot b_i\end{aligned}$$

2. Reescribir las expresiones para la suma y la resta introducidas en el ejercicio anterior de forma tal que para una posición genérica i , el valor de s_i , c_{i+1} o de b_{i+1} se calcule como una suma de productos entre x_i , y_i , c_i o b_i según corresponda.
3. Suponiendo un retardo de $2t$ en cada uno de los *full adders* de un *ripple adder* de 8 (ocho) bits, llevar adelante las siguientes tareas:

- a) Esquematizar el diagrama asociado a este sumador.
- b) ¿Cuál es el tiempo de retardo máximo que se debe contemplar para el tiempo de suma al operar de forma **sincrónica**?
- c) En el caso de operar de forma **asincrónica**, asumido un carry inicial $c_0 = 1$, determinar el tiempo de suma requerido en cada uno de los siguientes casos:
 - 1) $X = 11010010$ e $Y = 00101101$.
 - 2) $X = 11010010$ e $Y = 00111101$.
 - 3) $X = 11010010$ e $Y = 00100101$.

OBS: Tener en cuenta que el carry de entrada a una posición cualquiera $i + 1$ se expresa como $c_{i+1} = x_i y_i + (x_i + y_i) \times c_i = g_i + p_i c_i$, donde g_i denota la generación de carry y p_i la propagación de carry.

¹Fecha sugerida de finalización de este trabajo práctico: jueves 19 de abril de 2018.

4. Esquematizar un sumador paralelo de 32 bits empleando ocho sumadores CLAA de cuatro bits.

- a) En configuración *ripple*.
- b) En configuración *Carry Look-Ahead* de dos niveles.
- c) En configuración *Carry Look-Ahead* de tres niveles.

Calcular en cada caso el tiempo que insume la suma asumiendo que los retardos de los distintos componentes son los siguientes:

- Para los CLAA:
 - $6t$ para obtener la suma.
 - $4t$ para obtener el c_{out} .
 - $2t$ para obtener $P_{i,j}$ y $G_{i,j}$.
- Para los CLAG:
 - $1t$ para obtener $P_{i,j}$ y $G_{i,j}$.
 - $2t$ para obtener los c_{n+x} .

5. Trabajando con sumadores CLAA de cuatro bits con tiempos para la suma de 24ns y para generar el carry de salida de 13ns.

- a) Construir un sumador en configuración ripple de 16 bits, esquematizarlo y obtener el tiempo de suma para operación sincrónica.
- b) El CLAA suministra salidas P y G, correspondientes a la propagación y generación de carry en función de los sumandos de entrada, que se resuelven en 10ns. Empleando circuitos generadores de carry por adelantado, CLAG, con tiempos de respuesta para sus carries de salida de 13ns, esquematizar y calcular el tiempo de ejecución de la suma trabajando con dos niveles de Look Ahead.
- c) Dados los siguientes operandos:
 - 1) $X = 0001101011100010$
 - 2) $X = 1110100111001100$

Obtener el valor lógico de cada uno de los P y los G de los CLAA, y, a modo de verificación, indique cual sería el valor del carry que ingrese al FA de la posición 14 (tercer full adder del último de los cuatro CLAA) suponiendo que el carry inicial C_0 tiene valor 1.

Obs: Deberá expresarlo y calcularlo como función independiente de la suma. La función tomará en cuenta los P y los G precedentes junto con el carry C_0 inicial para definir el $carry_{in}$ del correspondiente CLAA y los p y g (*internos*) de las posiciones que le preceden dentro del CLAA para así evaluar el $carry_{in}$ del FA de la posición 14.

6. Diseñar un circuito *Carry-Skip Adder* para 32 bits realizando la partición de la siguiente manera:

- a) Usando ocho sumadores ripple de 4 bits cada uno.
- b) Usando ocho sumadores ripple, pero con las siguientes capacidades: 2, 3, 4, 5, 6, 5, 4 y 3 bits.

Ponderar los retardos en cada caso, asumiendo un retardo t por cada dos niveles de compuertas. Verificar si es el caso que la suma ha sido realizada *en cada bloque* en el tiempo propuesto.

7. En el contexto de un circuito *Carry-Skip Adder* para 32 bits, ¿cuál sería la partición óptima si los distintos bloques tuvieran que ser del mismo tamaño? Ponderar una vez más los retardos, asumiendo un retardo t por cada dos niveles de compuertas, verificando que la suma haya sido realizada *en cada bloque* en el tiempo propuesto.
8. Diseñar un *Carry-Select Adder* de 32 bits con seis bloques en configuración *ripple*, agrupando los *full-adders* de forma tal de optimizar los niveles de compuerta que deben atravesarse para completar la suma. Luego, adaptar el diseño propuesto para que admita 64 bits, usando en esta ocasión siete bloques. Finalmente, comparar ambos resultados en cuanto a cómo creció la complejidad en niveles de compuertas.

OBS: Tener en cuenta para el diseño que a medida que se avanza hacia las posiciones más significativas se deben incorporar módulos con *full-adders* adicionales.

Referencias

- [Bae80] BAER, J. L. *Computer Systems Architecture*. Computer Science Press, 1980.
- [HP96] HENNESSY, J., AND PATTERSON, D. *Computer Architecture*, second ed. Morgan Kaufmann, 1996.
- [Kor02] KOREN, I. *Computer Arithmetic Algorithms*, 2da ed. A. K. Peters, Natick, 2002.