Departamento de Cs. e Ingeniería de la Computación
Universidad Nacional del Sur

Arquitectura de Computadoras para Ingeniería Trabajo

Práctico Nº 2

Implementación de las Operaciones Básicas: Suma y Resta

Primer Cuatrimestre de 2016

Ejercicios

1. A partir de la tabla de verdad de un *semi-sumador* o *half-adder*,

   a) Deducir las funciones lógicas para la suma S_i y el *carry* de salida C_{i+1}, dadas dos entradas X_i e Y_i.

   b) Implementar un *subcircuito* utilizando *LogiSim*. Dicho subcircuito debe servir como bloque de construcción para futuras implementaciones.

   c) Testear la funcionalidad de la implementación realizada utilizando un circuito que haga uso de la misma.

2. Deducir las funciones lógicas para la suma S_i y para el *carry* de salida C_{i+1} de un *full adder*. Luego,

   a) Implementar el sumador en *LogiSim* utilizando como base las funciones encontradas.

   b) Implementar el sumador en *LogiSim* utilizando como base el subcircuito desarrollado en el ejercicio $1b$.

   c) ¿Cómo modela el *carry propagado* en ambas implementaciones?

3. Utilizando *LogiSim* implemente y simule un ***sumador serie*** a partir del *full adder* desarrollado en el ejercicio $2a$, tres registros de desplazamiento, un reloj y un flip-flop D.

4. Dada la siguiente expresión lógica para la resta:

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

   a) Verifique su validez analizando todas las combinaciones posibles.

   b) Implemente un subcircuito a partir de las mismas.

   c) Implemente un circuito que permita computar la resta entre dos operandos de 4 bits. Compruebe el funcionamento del mismo tomando como entradas $X = 0101$ e $Y = 0010$. Indique claramente b_{in} y b_{out}.

5. 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.

6. 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) Verifique el comportamiento del mismo utilizando el subcircuito desarrollado en el ejercicio $2a$. ¿Presenta alguna ventaja con respecto al desarrollo del ejercicio 3?

   c) ¿Cuál es el tiempo de retardo máximo que se debe contemplar para computar la suma al operar de forma **sincrónica**?

   d) 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) $X = 1\,1\,0\,1\,0\,0\,1\,0$ e $Y = 0\,0\,1\,0\,1\,1\,0\,1$.
      2) $X = 1\,1\,0\,1\,0\,0\,1\,0$ e $Y = 0\,0\,1\,1\,1\,1\,0\,1$.
      3) $X = 1\,1\,0\,1\,0\,0\,1\,0$ e $Y = 0\,0\,1\,0\,0\,1\,0\,1$.
      4) $X = 1\,1\,0\,0\,1\,1\,1\,0$ e $Y = 1\,1\,0\,1\,0\,1\,1\,0$.
      5) $X = 0\,1\,1\,0\,0\,1\,1\,0$ e $Y = 1\,0\,1\,1\,1\,0\,0\,1$.

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

7. 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.

   d) Compute para el inciso anterior el valor del carry por adelantado al sumador CLAA de la segunda posición i=2, si los operandos al primer CLAA son $X = 0\,1\,1\,1$ e $Y = 1\,0\,1\,1$ con C_{in}=0.

   e) Muestre cómo esquematizaría de forma eficiente el sumador en configuración *Carry Look-Ahead* de dos niveles, si en lugar de sumar operandos de 32 bits se trabajara con operandos de 24 bits.

   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 y g.
   - Para los **CLAG**:
     - $1t$ para obtener p y g.
     - $2t$ para obtener los c_{n+x}.

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

   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 carry de salida de 13 ns, esquematizar y calcular el tiempo de ejecución de la suma trabajando con dos niveles de Look Ahead.

   c) Dados los siguientes operandos:

   $$X = 0\,0\,0\,1\,1\,0\,1\,0\,1\,1\,1\,0\,0\,0\,1\,0$$
   $$Y = 1\,1\,1\,0\,1\,0\,0\,1\,1\,1\,0\,0\,1\,1\,0\,0$$

   Obtener el valor lógico de cada uno de los **P** y los **G** de los CLAA, y, a modo de verificación, indique cuál será 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 una función independiente de la suma, tomando en cuenta los **P** y los **G** precedentes junto con el carry inicial C_0 para definir el C_{in} del correspondiente bloque CLAA y los **p** y **g** (internos) de las posiciones que le preceden dentro del CLAA.

9. Diseñar un circuito *Carry-Skip Adder* para 32 bits, considerando las siguientes particiones:

   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.

   c) En caso de poder utilizar una cantidad mayor de bloques, ¿podría obtenerse alguna mejora? Muestre los tiempos involucrados y justifique claramente su decisión.

   Ponderar los retardos en cada caso, asumiendo un retardo t por cada dos niveles de compuertas.

10. Suponga que se cuenta con un sumador *Carry-Skip Adder* de n bits construido con bloques de k bits. Asumiendo que el tiempo de suma es proporcional a la cantidad de niveles lógicos involucrados, ¿cuál es el tiempo de suma del sumador?.

11. 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.