Laboratorio de Arquitectura de computadores GITT Entrega 3

Fecha de entrega: Viernes 09/11/2012 Grupo: Viernes

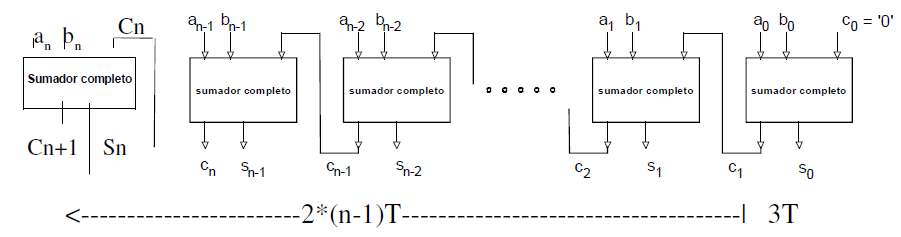
Autores: Manuel Montoya Catalá

**Cuestión 1**

Tal y como dijimos en la practica anterior:

- El primer cambio en el acarreo de salida tarda "3T".

- Teniendo un sumador de "N" bits, el cambio de acarreo se propaga "N-1" veces en el peor de los casos, por lo que el tiempo necesario para que el acarreo de salida del último sumador sea estable es "2T\*(N-1)"



Asi pues, el tiempo que tarda en estabilizarse el acarreo de salida del ultimo sumador es:

Retardo Acarreo = 2T\*(N-1) + 3T = 2T\*N + T

- Para el caso del retardo máximo de la salida de la suma, tenemos en cuenta que también tenemos que esperar a que se propaguen el cambio en los acarreos en el peor de los casos

hasta el ultimo sumador.

El acarreo de entrada del último sumador tarda en estabilizarse "2T\*(N-2) + T"

Un cambio en la salida debido a un cambio en el acarreo tarda "T" segundos en propagarse:

Retardo Suma = = 2T\*(N-2) + 3T + T = 2T\*N

# Ahora tenemos un sumador propagados de 32 bits (RCA), siendo el retardo de puerta T = 2ns, en el peor de los casos, los retardos son:

**Retardo Acarreo = 2T\*N + T = 130 ns**

**Retardo Suma = 2T\*N = 128 ns**

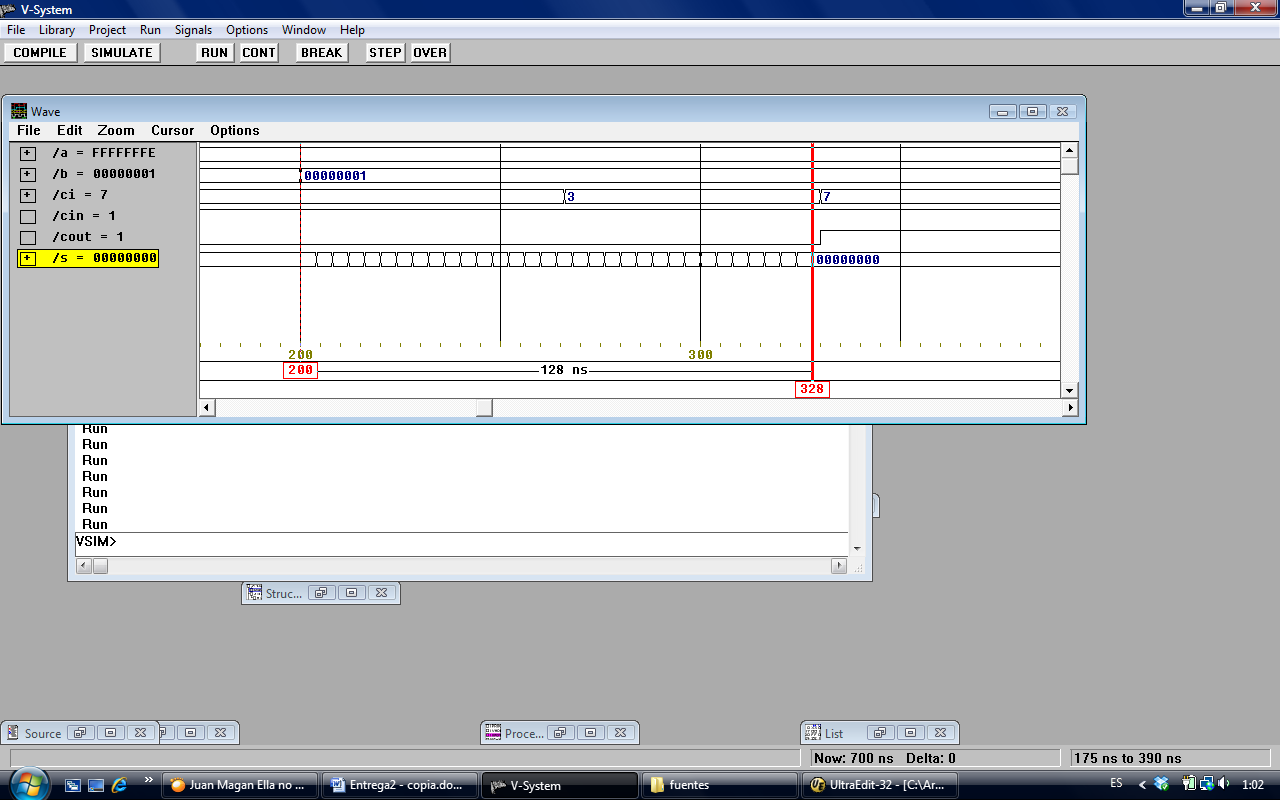
# Para ver el tiempo que tarda en producirse la suma y acarreo de nuestro sumador de 8 bits, establecemos el peor de los casos, aquel en el que un cambio en la entrada del sumador (a o b)

produce un cambio en el acarreo de salida, dicho cambio se propaga por todos los sumadores hasta el último, donde además tiene que cambiar también el valor de la salida. Para ello:

- Ponemos un Byte de entrada con todo 1´s menos el más bajo b = x"FFFFFFFE"

y el otro a 0´s a= x"00000000" con el acarro de entrada a 1.

- Si ahora cambiamos a : a= x"00000001" tenemos el peor de los casos



- Como se puede ver, la suma tarda 128 ns en estabilizarse y el acarreo tarda 130 ns.

**# Código sumador\_rca\_Nbits**

entity sumador\_rca\_Nbits is

generic (n: integer := 16; t\_1puerta : time := 2 ns);

port (a: IN bit\_vector (N-1 downto 0);

b: IN bit\_vector (N-1 downto 0);

cin: In bit; --Acarreo

s: OUT bit\_vector (N-1 downto 0);

cout: OUT bit); --Acarreo salida

end;

architecture sumador\_rca of sumador\_rca\_Nbits is

signal ci: bit\_vector(N downto 0);

begin

ci(0) <= cin;

gen: for i in 0 to N-1 generate

add1: entity work.sum\_elemental generic map (t\_1puerta)

port map (a(i),b(i),ci(i),s(i),ci(i+1));

end generate;

cout <= ci(N);

end;

**# Código sumador\_rca\_32\_tb**

entity sumador\_rca\_32\_tb is

end;

architecture tst of sumador\_rca\_32\_tb is

signal a,b,s: bit\_vector(31 downto 0):= x"00000000";

signal cin,cout : bit := '0';

signal ci: bit\_vector(2 downto 0);

begin

ci(0)<=cin;

sumador1: entity work.sumador\_rca\_Nbits generic map (16, 2 ns)

port map (a(15 downto 0 ), b(15 downto 0 ),ci(0),s(15 downto 0 ),ci(1));

sumador2: entity work.sumador\_rca\_Nbits generic map (16, 2 ns)

port map (a(31 downto 16 ),b(31 downto 16 ),ci(1),s(31 downto 16 ),ci(2));

cout<=ci(2);

process

begin

a<=x"FFFFFFFE"; b<=x"00000000"; cin <='1';

wait for 200 ns;

b<=x"00000001";

wait;

end process;

end;

**Cuestión 2**

**1)**

# Funcionamiento del sumador CSK:

- El sumador CSK está formado por bloques de "N" bits que implementan el RCA.

- Para implementar CSK dividimos nuestro sumador de M bits en etapas de sumadores

mas pequeños de "N" bits cada uno.

- La primera y última etapa son propagados RCA normales y el resto, las etapas intermedias, son propagadores que implementan la función "P" del grupo por lo que son capaces de anticipar el carry de salida de la etapa de "N" bits intermedia:

- Tenemos qu el acarreo de salida del bloque CSK es:

**Cout = + P ∙ Cin**

Siendo la función P el AND de todos los :

**P =**  siendo  **=**

- De una forma cualitativa, los bloque que implementan la función "P" miran por casos donde el carry de entrada a un conjunto de bits "N" es el mismo que el de salida.

Cuando se cumple que todos los pi del grupo de bits "N" son 1, el grupo propaga el carry de entrada directamente a la salida con un retardo de 2 tiempos de puerta:

# Sabiendo los retardos de un RCA en el peor de los casos son:

- Para la primera etapa de todas => Retardo Acarreo = 2T\*N + T

- Para la última etapa => Retardo Acarreo = 2T\*N

- Como se tardan 2T en propagar el acarreo de un bloque a otro:

Un tiempo por la puerta "and" de la función P y otro por la puerta "or" de P y

# Para un sumador CSK de N bits en bloques de "m" bits, por lo que tenemos que implementar**: k = M/N** bloques, en el peor de los casos tenemos un retardo de:

- Primer bloque RCA: **2T\*N + T**

- Bloques intermedios CSK: (**k-2)\*2T**

- Bloque final RCA: **2T\*N**

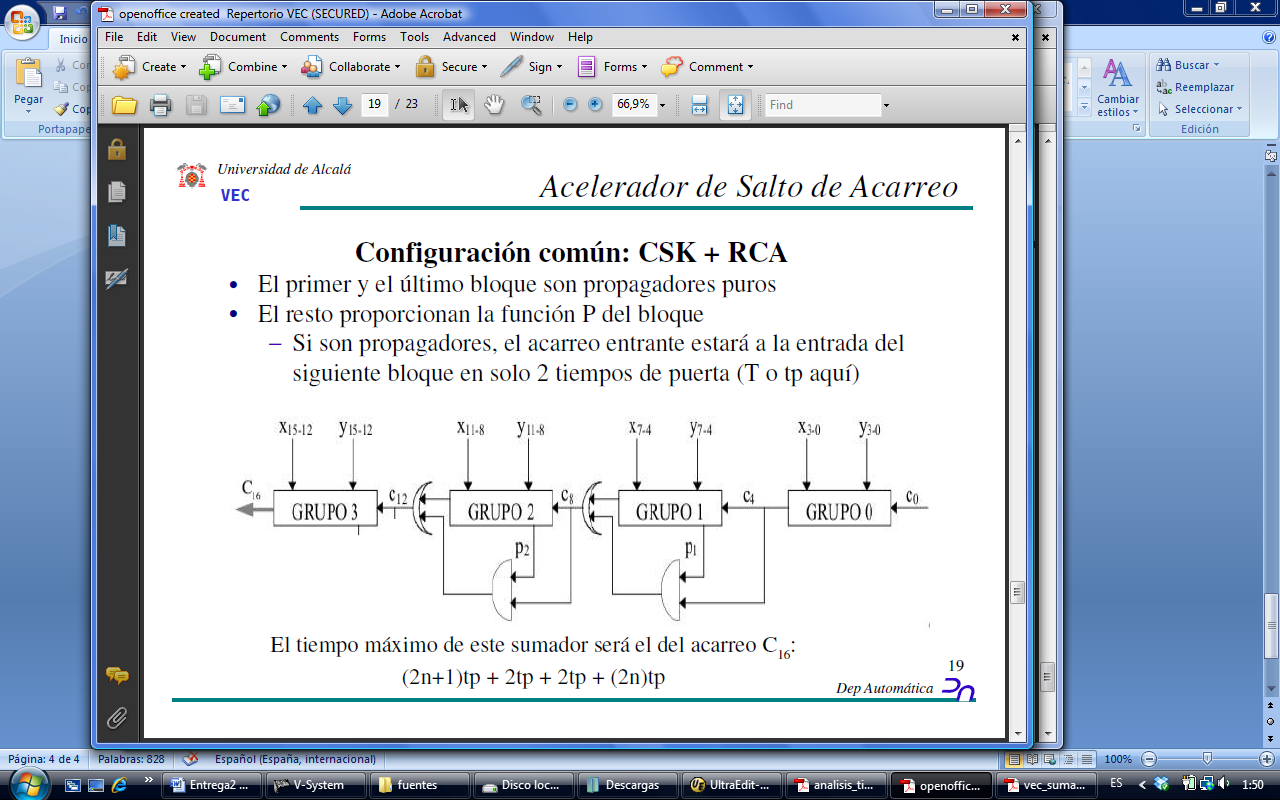
# Retardo total CSK = **2T\*(2N + k - 2 ) + T**

- El retardo para la suma será como en los RCA un T menos:

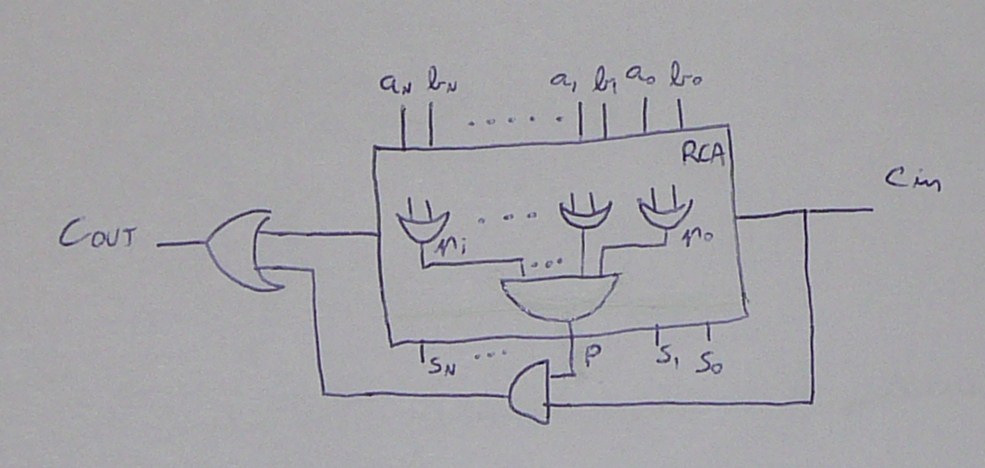
Retardo\_suma\_CSK = **2T\*(2N + k - 2 )**

- Internamente, las sumas y acarreos internos de cada etapa CSK tardan lo mismo en generarse que en un sumador RCA

Visión gráfica:

****

**# Caso único sumador CSK:**

-# Caso de primer sumador y Cin a 1 con un cambio en la entrada

- En cuanto a la suma: lo mismo que un RCA normal ya que el CSK no lo modifica:

- En cuanto al acarreo de salida **Cout** del bloque tardará **4\*t\_puerta:**

- 1t\_puerta por el cálculo del nuevo p0

- 1t\_perta por la función P (and de los pi)

-1t\_puerta por multiplicar P y Cin

-1t\_puerta para la "or" entre PCin y Crcaout

Darse cuenta que  se generará en el mismo tiempo que un RCA normal

**(2N\* + 1)\*t\_puerta** sin embargo, como **P ∙ Cin** será 1, la puerta or dará 1 cuando este lo sea.

# He aquí un ejemplo de dos sumadores CSK de 8 bits seguidos, con t\_puerta = 2ns:

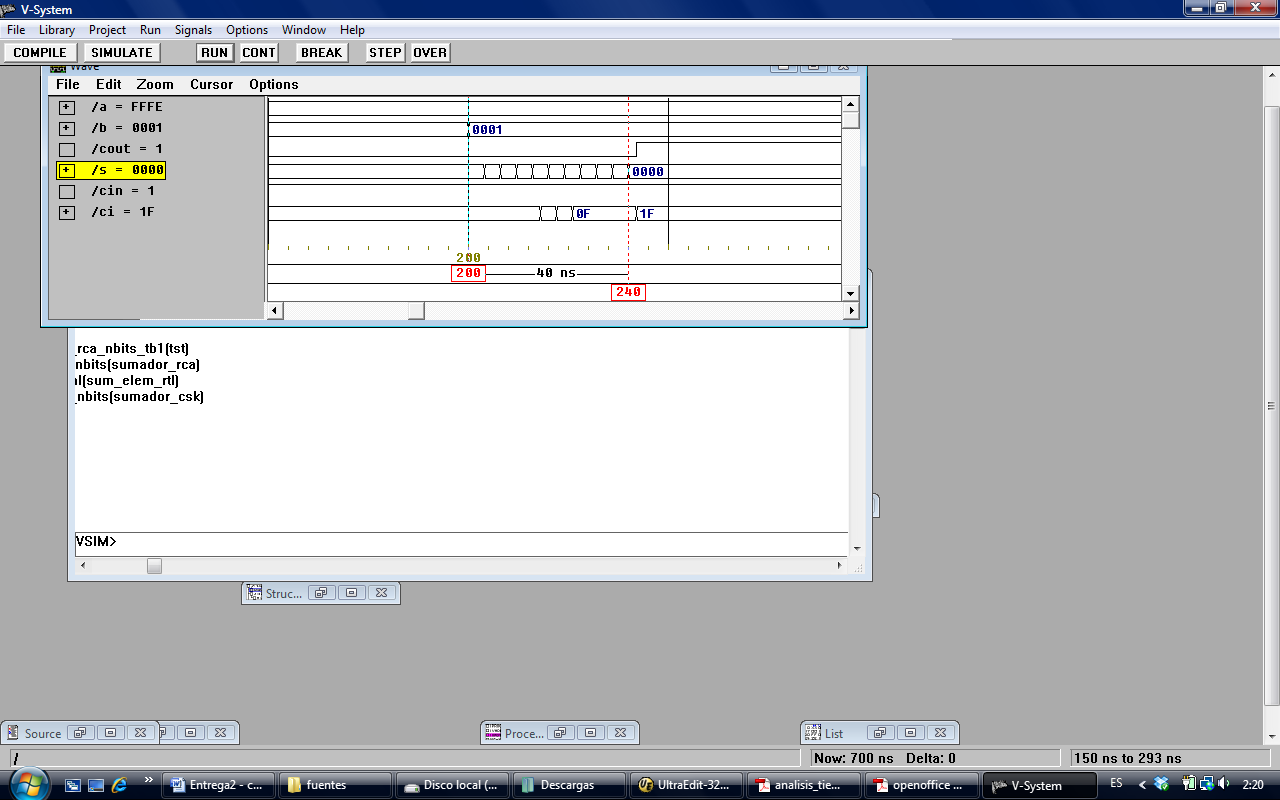
# En nuestro caso, queremos hacer un sumador CSK de **M = 16 bits**, en bloques de **N = 4 bits**

por lo que tenemos **k = 4 etapas**, el retardo teórico entonce será, para T = 2ns:

- Retardo\_acarreo\_CSK = **2T\*(2N + k - 2 ) + T = 42 ns**

- Retardo\_suma\_CSK = **2T\*(2N + k - 2 ) = 40 ns**

# Para la parte experimental, hemos utilizado el mismo peor caso que en el sumador RCA ya que al fin y al cabo, el CSK está hecho con RCA's:



- Como se puede ver, la suma tarda 40 ns en estabilizarse y el acarreo tarda 42 ns.

**# Código sumador\_csk\_Nbits**

entity sumador\_csk\_Nbits is

generic (n: integer := 16; t\_1puerta : time := 2 ns);

port (a: IN bit\_vector (N-1 downto 0);

b: IN bit\_vector (N-1 downto 0);

cin: In bit; --Acarreo

s: OUT bit\_vector (N-1 downto 0);

cout: OUT bit); --Acarreo salida

end;

architecture sumador\_csk of sumador\_csk\_Nbits is

signal ci: bit\_vector(N downto 0);

signal aux : bit\_vector(N downto 0);

signal p, and\_skip : bit;

begin

ci(0) <= cin;

aux(0)<= '1';

gen: for i in 0 to N-1 generate

add1: entity work.sum\_elemental generic map (t\_1puerta)

port map (a(i),b(i),ci(i),s(i),ci(i+1));

aux(i+1) <= (a(i) or b(i)) and aux(i); --Acumulador pi

end generate;

p<=aux(n) after 2 \* t\_1puerta ; -- or y and fan-in ilimitado

and\_skip<= p and cin after t\_1puerta;

cout <= ci(N) or and\_skip after t\_1puerta;

end;

**# Código sumador\_csk\_rca\_Nbits\_tb1**

entity sumador\_csk\_rca\_Nbits\_tb1 is

end;

architecture tst of sumador\_csk\_rca\_Nbits\_tb1 is

signal a,b,s: bit\_vector(15 downto 0);

signal cin,cout : bit := '0';

signal ci: bit\_vector(4 downto 0);

begin

ci(0)<=cin;

sumador1: entity work.sumador\_rca\_Nbits generic map (4, 2 ns)

port map (a(3 downto 0 ), b(3 downto 0 ),ci(0),s(3 downto 0 ),ci(1));

sumador2: entity work.sumador\_csk\_Nbits generic map (4, 2 ns)

port map (a(7 downto 4 ),b(7 downto 4 ),ci(1),s(7 downto 4 ),ci(2));

sumador3: entity work.sumador\_csk\_Nbits generic map (4, 2 ns)

port map (a(11 downto 8 ),b(11 downto 8 ),ci(2),s(11 downto 8 ),ci(3));

sumador4: entity work.sumador\_rca\_Nbits generic map (4, 2 ns)

port map (a(15 downto 12 ),b(15 downto 12 ),ci(3),s(15 downto 12 ),ci(4));

cout<=ci(4);

process

begin

a<=x"FFFE"; b<=x"0000"; cin <='1';

wait for 200 ns;

b<=x"0001";

wait;

end process;

end;

**2)**

# Como hemos explicado reiteradas veces, el retardo del sumador RCA de 16 bits es:

**Retardo Acarreo RCA = 2T\*N + T = 66 ns**

- El retardo del sumador CSK de 16 bits en etapas de 4 bits es:

**Retardo\_acarreo\_CSK = 2T\*(2N + k - 2 ) + T = 42 ns**

# Como ya hemos explicado,esta diferencia es debida a que las etapas intermedias del sumador CSK son capaces de propagar el acarreo de salida de su etapa directamente gracias al uso de la función "P".

La aceleración del circuito es: A = 66/42 = 1.5714

**3)**

Como podemos apreciar, el bit que más tarda en estabilizarse es el bit de salida 11, es último bit de salida del último bloque CSK

- Esto es debido a que dentro del segundo bloque CSK se esta produciendo una propagación interna total del bit de acarreo de entrada, por lo que para generar la última salida del bloque, se tiene que esperar a que el acarreo se propague por todos los sumadores elementales del bloque.

