PRÁCTICA 4

Implementación de un multiplicador secuencial de 16bits con UP y UC específicas.

Objetivos

Después de realizar esta práctica, el alumno deberá:

1) Conocer y comprender un algoritmo de multiplicación secuencial de números naturales basado en sumas.
2) Comprender el funcionamiento de un multiplicador secuencial binario construido diferenciando entre unidad de proceso y unidad de control.
    a) Saber implementar la unidad de proceso y todos los bloques que la componen.
    b) Saber implementar la unidad de control, a partir de la unidad de proceso y de acuerdo a las especificaciones de funcionamiento del multiplicador.
3) Saber obtener el tiempo de ciclo mínimo del multiplicador secuencial y el tiempo de cálculo de una multiplicación.
4) Realizar la simulación necesaria para comprobar un funcionamiento especificado

Desarrollo 1

Algoritmo de multiplicación:

Dados dos vectores de 16 bits, $X = x15..... x2\ x1\ x0$ e $Y = y15.... y3\ y2\ y1\ y0$ debemos obtener un algoritmo que produzca el vector de bits W, que representa en binario al número natural $W_u = X_u$ x Y_u.
Como X_u e Y_u se encuentran en el rango de valores $0 \le X_u, Y_u \le 2^{16}-1$, su multiplicación puede valer $0 \le X_u$ x $Y_u \le 2^{32} - 2^{17} - 1$. Esto nos indica que necesitamos 32 bits para representar los posibles valores que resulten de la multiplicación (en general, para represntar el resultado de multiplicar operandos de n bits, necesitamos 2n bits).

El algoritmo más sencillo para la multiplicación consiste en sumar uno de los operandos consigo mismo tantas veces como el valor del otro operando. Supongamos que el valor de X determina el número de sumas e Y es el dato a sumar: el número de veces que debemos sumar Y es X-1 (si X=1 el resultado es Y, si X=2 el resultado es Y+Y, etc.). Una forma de escribir este algoritmo es:

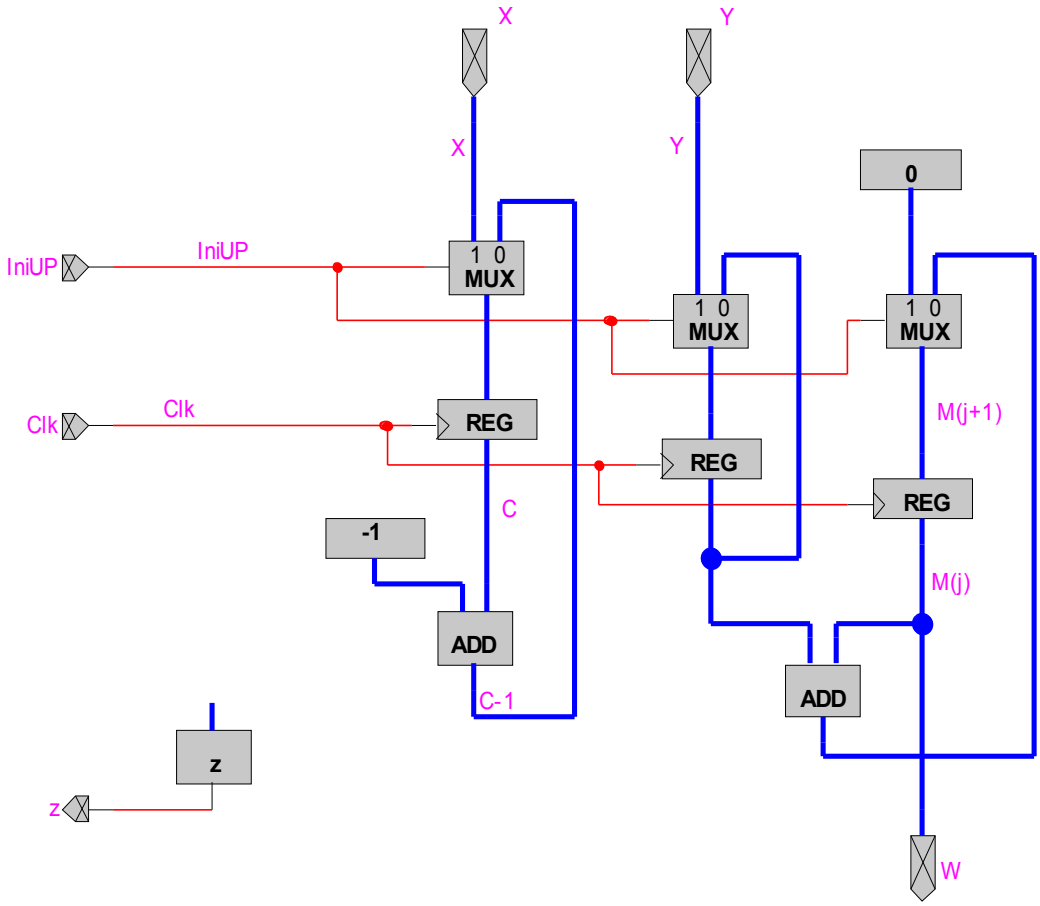
```
M(0) = 0
para j = 0  hasta   j = X – 1  hacer:
        M(j+1) = M(j) + Y
fin_para
W = M(X)
```

donde la variable M va acumulando los valores de la suma.

Implementación de la unidad de proceso:

Nos vamos a ocupar solo de los 16 bits de menor peso del resultado de la multiplicación, sin preocuparnos por la detección del desbordamiento. Por tanto, todos los elementos de la UP trabajan con datos de 16 bits.
En principio necesitamos tres registros (para X, Y, y la variable $M(j)$) y un sumador (bloque ADD: sumador combinacional con propagación de acarreo).
Para el contador de vueltas del bucle, lo más efectivo es restar uno en cada ciclo al valor de X : esto lo hacemos con otro sumador cuyas entradas son X y -1 (codificado en 16 bits). Para saber cuando el resultado de esta suma es cero tenemos el dispositivo *Zero* (*z*), cuya salida de un bit vale 1 solo cuando sus 16 bits de entrada valen 0.
También se necesitan tres multiplexores de buses, para encaminar los datos iniciales X e Y, y para inicializar a cero la variable $M(j)$. Comprobaremos que los tres se pueden controlar con la misma señal (*IniUP*)

En la librería de la práctica4 (*Lib-SD-4*) se encuentra el bloque *UPmul*, en cuyo interior tenemos una posible implementación de la unidad de proceso:



Comprobación de la UP: A la vista del circuito observamos lo siguiente:

1. Si un ciclo de reloj empieza con la señal *IniUP* a 1, se graban los datos *X* e *Y* en los registros y se inicia *M(j)* a cero
2. Si los ciclos siguientes empiezan con *IniUP* a 0, el valor de *X* disminuye una unidad, el valor de *Y* permanece, y a la variable *M(j)* se le suma el valor de *Y*
3. El dispositivo *z* está aún sin conectar

Vamos a comprobar este funcionamiento mediante el simulador. Para simplificar, trabajamos solo con los 4 bits menos significativos. Visualizaremos las variables *C, C-1, M(j)* y *M(j+1),* conectando en los buses respectivos dispositivos *HexDisplay.* Para los datos *X* e *Y,* conectamos a los buses de entrada los dispositivos *HexKeyboard* (para conectar los cables internos de un bus: *Schematic* → *New Breakout* → lista de cables, en nuestro caso *b0..3*). También conectamos *BinarySwitch* en las señales *IniUP* y *Clk*

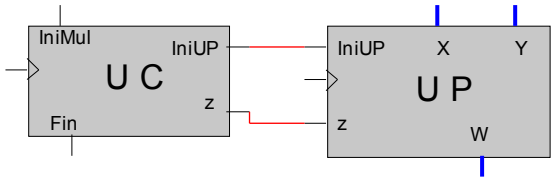
Completa la tabla siguiente para los datos ***X = 0003h, Y = 0004h*** recordando que con la señal *IniUP* a 1, el primer flanco ascendente de reloj da comienzo al proceso (*j=0*), que continúa si en los siguientes ciclos *IniUP* es 0:

	j = 0	***j = 1***	***j = 2***	***j = 3***	***j = 4***
C					
C-1					
M (j)					
M (j+1)					

Desarrollo 2

Especificación del multiplicador como caja negra (especificación del circuito completo UP + UC desde el punto de vista de sus entradas y salidas, y sin hacer mención a su implementación interna):

Entradas: datos → *X, Y*
control → *IniMul*
Salidas: datos → *W*
control → *Fin*



Supongamos que nos exigen las siguientes especificaciones de funcionamiento:

1. En el ciclo en el que la señal *IniMul* vale 1, los datos de los buses de entrada *X* e *Y* son válidos y se debe comenzar su multiplicación.
2. Cuando el multiplicador tenga el resultado correcto en el bus de salida, lo indicará poniendo a 1 durante un ciclo la señal de control *Fin*.
3. El resultado correcto sólo estará presente en el bus de salida del multiplicador durante ese ciclo, así que el sistema exterior debe estar esperando este ciclo, para no perder el resultado.
4. Si mientras se está multiplicando un par de números (después del ciclo en el que *IniMul* vale 1 y antes de que *Fin* valga 1), se recibe por la entrada *IniMul* un 1, el multiplicador hará caso omiso de esta petición y continuará sin inmutarse hasta que ponga a 1 *Fin*.
5. En el ciclo en el que *Fin* vale 1, ya puede comenzar otra multiplicación, si *IniMul* vale 1 en ese mismo ciclo.

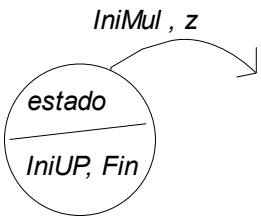
Para cumplir estas especificaciones, sabiendo que la señal de control *Fin* es generada por la UC (y no por la UP), y después de haber estudiado el funcionamiento de la UP en el apartado anterior :
¿Cómo podemos conectar el dispositivo z en la UP?

Desarrollo 3

Diseño de la unidad de control:

Se trata de obtener el grafo de estados de la UC, razonando a partir de las especificaciones anteriores y teniendo en cuenta la UP estudiada.

Grafo de estados de la UC:



Desarrollo 4

Comprobación del multiplicador:

El circuito *Mul16* es el resultado del proceso de diseño seguido en esta práctica. En él, podemos ver conectadas la UP y la UC. Además el circuito dispone de teclados y visores hexadecimales conectados a las entradas y salidas de datos. El bloque *Pulse* es un circuito que sirve para generar la señal de inicio, *IniMul*, de forma sincronizada con el reloj, cuando se pulsa el botón.
El problema es que el circuito no funciona, porque los bloques UP y UC no tienen todavía el contenido interno correcto:

Sustituye la ROM interna de la UC (hay que crear una nueva ROM a partir del grafo de estados de la UC: atención a la forma en que está conectada la ROM a las entradas salidas de la UC). Una vez hecha esta sustitución, hay que abrir el circuito interno de la unidad de proceso y conectar el bloque *z* tal y como se contestó la pregunta del desarrollo2.

Todavía nos falta definir la señal de reloj para que el funcionamiento del circuito sea correcto. Para determinar el camino crítico en un circuito secuencial como este, tendremos en cuenta que los dispositivos con mayor retardo se encuentran en la UP :

Especifica el camino crítico del multiplicador:

Especifica el cálculo del periodo mínimo de reloj:

Define la señal de reloj en el circuito de acuerdo al valor anterior. Ahora ya tenemos el multiplicador preparado para comprobar su funcionamiento: realiza las simulaciones necesarias para verificar que se cumplen las características que nos pedían en el desarrollo2, y contesta las preguntas:

¿Cuantos ciclos se necesitan para multiplicar *X* = *0012h* x *Y=0023h* ?

Comprueba que el multiplicador cumple la especificación de funcionamiento nº4