Prueba 1 - EdC - Grupo 2 - Curso 19/20

Memorias

1. En la memoria de acceso aleatorio el tiempo de acceso es independiente de la dirección del dato a que se accede.
   En la memoria de acceso secuencial el tiempo de acceso puede ser muy diferente según la dirección a la que se accede.
2. Una memoria volátil es aquella que para mantener almacenada la información necesita estar alimentada.
3. Las memorias de mayor velocidad son las que tienen más coste por bit. Las de mayor capacidad son las que tienen menor coste/bit.
4. a) Todas las memorias tienen bus de dirección de 10 bits y bus de datos de 8 bits.
   Por tanto son memorias de capacidad $2^{10} \times 8$, es decir 1K x 8 o 1 kbyte. Son 1024 palabras de 8 bits.
   b) Si consideramos M1 y M5, como ambas reciben el mismo bus de direcciones y tienen buses de datos independientes, se tiene como resultante una memoria de capacidad $2^{10} \times 16$. Son 1024 palabras de 16 bits.

	Tema	Fecha	Página	

c) Si consideramos el decodificador junto a las memorias M1, M2, M3 y M4, tenemos un bus de direcciones de 12 bits: A_{11} y A_{10} seleccionan una entre las 4 memorias y $A_9 A_8 \cdots A_0$ seleccionan la palabra dentro de cada memoria. El bus de datos es de 8 bits, unión de todos los buses de datos en un único bus compartido que se usa por turnos. El decodificador hace que no pueda haber conflicto de uso. En definitiva, tenemos una capacidad de $2^{12} \times 8 = 4$ kbytes → son 4096 palabras de 8 bits.

d) Considerando el circuito completo se tiene que M5, M6, M7 y M8 vienen a ampliar el bus de datos de la configuración del apartado c, con lo que se tienen las mismas palabras en número pero de doble longitud. La capacidad será $2^{12} \times 16$, es decir, 4096 palabras de 16 bits.

	Tema	Fecha	Página	
>				>

Verilog

1. e) dos 3 tipos son funcional, estructural y procedimental

f) las variables tipo tienen memoria, es decir, almacenan su valor hasta que se les asigna uno nuevo. Por contra, las variables tipo wire pueden ser leídas o asignadas pero no almacenan su valor, tienen que ser establecidas de forma continua, bien por un assign o conectándose a la salida de algún módulo.
En un proceso always, si queremos asignar un valor a una variable, esta debe ser tipo reg.
Un assign se utilizará con variables tipo wire.
En una descripción estructural usaremos tipo wire.

g) En la conexión posicional es necesario respetar el orden que se utilizó al definir las entradas y salidas del módulo. En la conexión nombrada esto no es necesario ya que nos referiremos a cada entrada o salida citando explícitamente su nombre.

h) + representa la suma aritmética.

	Tema	Fecha	Página	
>				>

2. La señal in1 controla la puesta a 0 del contador.
La señal in2 controla la cuenta ascendente.
La lista de sensibilidad tiene que contener
"negedge ck" por ser disparado por flanco de bajada
"posedge in1" por ser la puesta a 0 asíncrona y activa en alta.
Quedaría: always @ (negedge ck, posedge in1)

3. Para que el código corresponda a un registro de las características que se piden basta con cambiar la lista de sensibilidad de esta forma:

always @ (negedge ck)

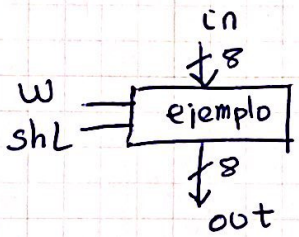
es decir, eliminamos la referencia a in1 porque ahora la puesta a 0 ha de ser síncrona.
El otro cambio que hay que hacer es introducir en la lista de entradas y salidas, las nuevas entradas para la carga en paralelo:

module nombre (input ck, in1, in2, input [3:0] x, output [3:0] z);

y también cambiar el 2º if:

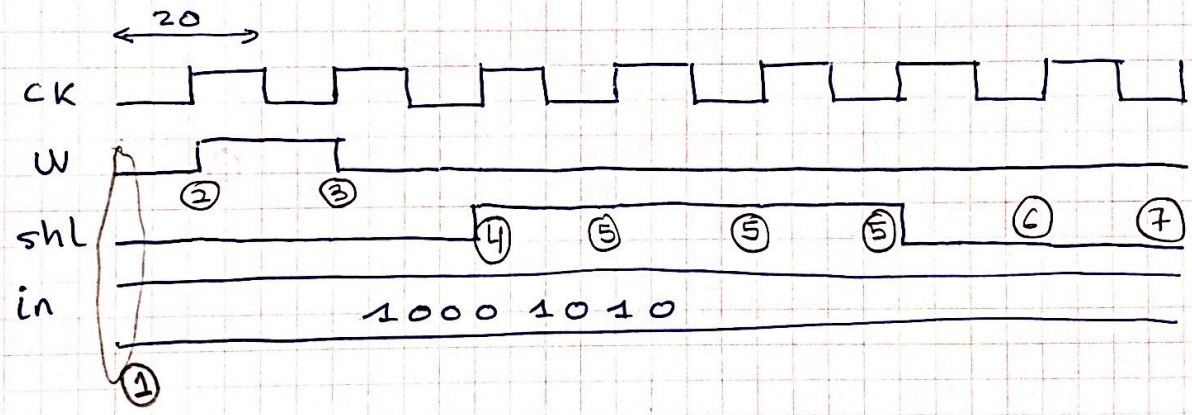
else if (in2==1)
z<=x;

4. Se trata de un registro de 8 bits con carga en paralelo, desplazamiento circulas a la izquierda, disparado por flanco de bajada. → porque se introduce el bit que sale, no hace falta decir circular



w	shL	ejemplo ←	out=
0	0	ejemplo	[ejemplo]
0	1	shL(ejemplo, ejemplo7)	[ejemplo]
1	X	in	[ejemplo]

5.




```
initial begin
  ck=0, w=0; shl=0; in = 8'h8a  (1)
  @(posedge ck) w=1;  (2)
  @(posedge ck) w=0;  (3)
  @(posedge ck) shl=1;  (4)
  repeat3 @(posedge ck) shl=0;  (5)
  @(posedge ck);  (6)
  @(posedge ck);  (7)
  $finish;
```

Tema | Fecha | Página