Pre-Informe 3

Sebastian Gutierrez
201773006-4

Pregunta 1 Investigacion de conceptos

a.

Es un lenguaje usado para describir , modelar y en algunos casos simular circuitos logicos. Se diferencia de los lenjuajes de programacion es que el lenguaje de descripcion de hardware modela circuitos mientras que un lenjuaje de programacion crea programas los cuales son ejecutables y tienen alguna utilidad practicamente inmediata en cambio el otro a lo mas nos permite hacer simulaciones del modelo.Otra diferencia es que el HDL puede modelar multiples procesos paralelos que se ejecutan de forma independiente.

b.

El tipo reg hace referencia a variables que almacenan informacion, en contraste las variables del tipo wire representan conexiones entre componentes (no almacenan informacion).Cabe destacar a wire se le hacen asignaciones fuera de bloques always y a reg se le pueden hacer dentro de bloques always.Logic es de cierta manera la combinacion de estas ya que en system verylog se puede identificar de forma automatica de que forma se esta usando una variable de tipo logic.

c.

El assign se usa fuera de cualquier bloque (ej:always) ya que este define logica , osea es una asignacion continua si los valores en el lado derecho cambian cambiaran los del lado isquierdo.El = se usa dentro de bloques always,if,initial,etc estos asignan valores a una variable pero lo mas importante es que varias sentencias = seguidas se ejecutan secuencialmente osea uno despues del otro(blocking assignment).En contraste el ¡= se usa igual que el = pero varias sentencias de estas se ejecutan como si pasaran al mismo tiempo(non blocking assignment).

d.

La expresion sirve para declarar y asignar un valor a la variable g, la primera parte señala el tipo de la variable esta dice reg por ende g es del tipo reg , lo siguiente que esta entre corchetes sirve para declarar el tamaño de nuestra variable osea que g es un reg de 16 bits siendo el 0 el bit menos significativo y 15 el mas significativo,el = obviamente señala una assignacion ,por ultimo el valor 16'hA6B2(en la guia dice 'hA6B2 pero por moodle se aviso que este era un claro error y se cambiaba por ese) corresponde a un numero donde lo que va antes del ' es el numero de bits en que se almacena el numero ,el h representa la base en que esta escrito en este caso es hexadecimal y lo que queda es el numero osea A6B2 que en binario equivale a 1010 0110 1011 0010 y en base 10 a 42674.

e.

Se supone que el valor de second es el negado del valor de first por ende al cambiar el valor de first a 1 el de second deberia ser 0 si este fuera el caso en la sentencia $arr[first] <= -arr[second]$ deberia cambiar el bit 1 de arr por el negado del bit 0 lo que significaria que en 0101 el bit 0 seria un 1 cuyo negado es 0 pero el bit 1 ya es 0 por lo que no hay cambios.Aun asi se nos dice que arr paso de ser 0101 a 0111 lo que no tiene sentido segun lo que dijimos,esto paso probablemente por el delay osea que cuando se evaluo $arr[first] <= -arr[second]$ el valor de first ya habia cambiado pero el de second aun no osea second seguia siendo 1 causando que el bit 1 se remplazara por el negado del mismo pasando de 0101 a 0111.

Pregunta 2 Analisis de codigo

La tabla siguiente representa el ruteo del codigo, en esta cada columna es una variable y cada fila una subida del clock.La primera subida es especial pues es la unica en la que se "ejecuta" el if por eso en la tabla esta tiene 2 filas la primera tiene los cambios del if y la segunda los cambios restantes.Los - denotan que no hubo cambio. y el valor de entrada r fue 000.

	x	y	z	m	n	s	r
init	00000000	01110111	000	001	01110111	0	000
1	-	-	-	-	-	1	-
	-	01110110	001	010	01110110	-	-
2	-	01110100	010	011	01110100	-	-
3	-	-	011	100	-	-	-
4	00010000	-	100	101	10000100	-	-
5	-	01100100	101	110	01110100	-	-
6	-	01000100	110	111	01010100	-	-
7	-	-	111	000	-	-	-
8	00010001	11000100	000	001	11010101	-	-
9	00010011	11000101	001	010	11011000	-	-
10	00010111	-	010	011	11011100	-	-

Con esto determinamos que el valor final de n, empezando con r = 000 , es 11011100.

Pregunta 3 Diseño de modulo

Se escribio el siguiente modulo ,en systemverilog usando modelsim, donde q es nuestro numero de 5 bits y fib es el booleano que nos dice si el numero actual es parte de la serie de fibonacci:

```
1  module conta_fibo(input logic clk,
2                    output reg [4:0] q,
3                   output logic fib);
4          initial begin
5          q <= 5'b0;
6          end
7          assign fib = (~q[4]&~q[3]&~q[2])|(~q[4]&q[2]&~q[1]&q[0])|(~q[4]&~q[2]&~q[1]&~q[0])|(~q[3]&q[2]&~q[1]&q[0]);
8          always @ (posedge clk) begin
9               q++;
10         end
11 endmodule
12
```

Figure 1: Modulo

Cabe destacar la expresion usada en el assign fue encontrada usando mapas de karnaugh.La siguiente imagen es el resultado de la simulacion para todos los valores de q posibles:

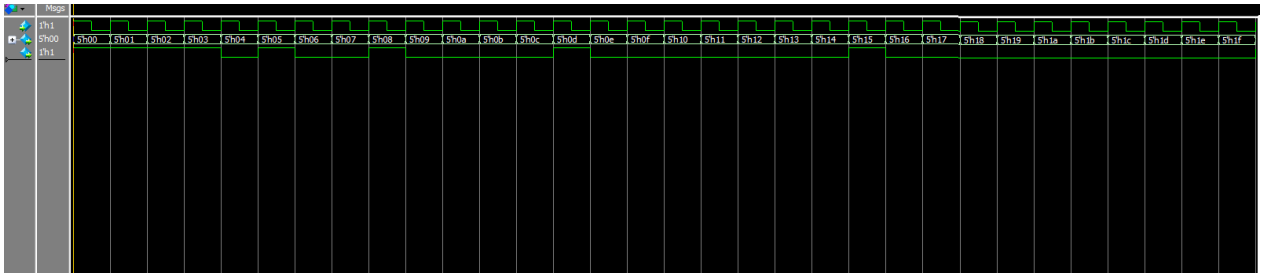

Figure 2: Simulacion

Como se puede ver solo el 0,1,2,3,5,8,13,21 son parte de la serie de fibonacci y en estos el valor de fib sube. Un pequeño dato adicional la figura 2 se ve un poco rara por que fue editada para que se vieran todos los valores de q ya que me fue imposible tomar un pantallazo de todo al mismo tiempo , de cualquier manera al final de este informe adjunto los dos pantallazos originales.

Extra

Aqui adjunto lo prometido en la seccion 3:

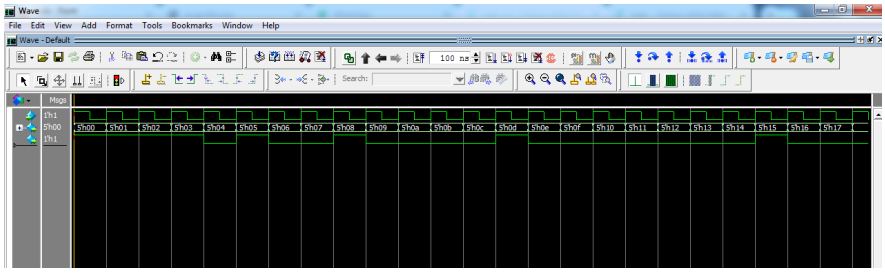

Figure 3: adicional1

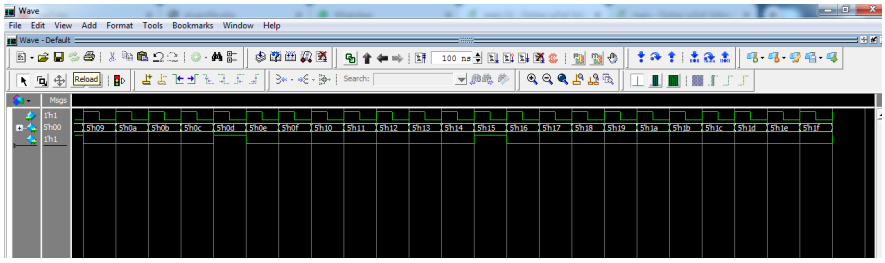

Figure 4: adicional2