Pre-Informe 3
Simulando con SystemVerilog

Cristian Bernal R.
201773026-9

Pregunta 3.1

a) Los lenguajes de descripción de Hardware, mas conocidos como HDL (*Hardware Description Language*) nos permiten explicar las interconexiones, el comportamiento y del diseño de un circuito electrónico, sin usar diagramas esquemáticos. Estos lenguajes, describen a circuitos electrónicos, por otro lado los lenguajes de programación describen los pasos necesarios que debe realizar el procesador para ejercer una tarea.

b) Todas estas variables entregan valores lógicos, primero ***wire*** permite conectar diferentes elementos, usado en asignamientos continuos y no almacena valores. ***reg*** permite el almacenado de estados, pudiéndose cambiar este valor por otro asignándoselo, ademas no cambia con el tiempo. Con ***logic*** la variable puede almacenar asignaciones continuas como una de bloqueo, en pocas palabras, funcionar como un wire o un reg.

c) Existen dos tipos de asignaciones. ***assign*** es del tipo continua, que solo permite modelar lógica combinacional, permitiendonos solo declararla como un wire. Las dos asignaciones restantes son del tipo procesural, se les puede asignar cualquier tipo, se les puede poner un delay, la diferencia es que $=$ (bloqueo) la asignación se realiza antes de la siguiente asignación y la $<=$ (sin bloqueo) se evalúa el termino, pero se asigna al final de la evaluación.

d) ***reg*** la variable es de asignación continua, ***[15:0]*** es el tamaño de la variable (16 bits), ***g*** es el nombre de esta variable, $=$ se uso una asignación procedural con bloqueo y ***'hA6B2*** es un numero hexadecimal sin tamaño designado.

e) Al aplicar un $<=$ tenemos un delay, ahora el valor de firt que tenemos dentro del bloque always_ff es "local", por lo que los cambios que tenga globalmente serán en los cantos de subida, esto explicaría porque después quedaría first 1, second 0 y el arr 0111.

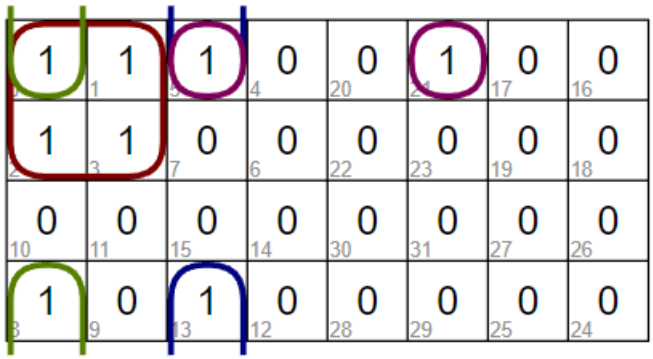
Pregunta 3.2

Usando el valor 0 en r (0000), se obtiene el siguiente ruteo (los valores encerrados en casillas son los valores que tienen al inicio de cada canto de subida)

x	y	z	n	m	s
0	119	0	119	1	0
		0			1
		0			1
		1			
0	238	1	118	2	1
		2			
0	116	2	116	3	1
		3			
0	116	3	116	4	1
		4			
16	116	4	132	5	1
		5			
16	100	5	116	6	1
		6			
16	68	6	84	7	1
		7			
16	68	7	84	0	1
		0			
17	196	0	213	1	1
		1			
19	197	1	216	2	1
		2			
23	197	2	220	3	1

Pregunta 3.3

Primero, necesitamos obtener la ecuación lógica que nos de resultado 1 si el número binario $n = x_0x_1x_2x_3x_4$ es de la serie de fibonacci. Usamos el mapa de Karnaugh pensando que los únicos números que son de la serie hasta el $2^5 - 1$ son 0,1,2,3,5,8,13,21:



Se obtiene y = $\bar{x_0}\bar{x_1}\bar{x_2} + \bar{x_0}\bar{x_2}\bar{x_3}\bar{x_4} + \bar{x_0}x_2\bar{x_3}x_4 + \bar{x_1}x_2\bar{x_3}x_4$. Pasando a SystemVerilog, tendríamos el siguiente código

```
Ln#   1 Driver
 1  module fibo(clk,n,fib);
 2          input logic clk;
 3          output logic[4:0] n;
 4          output logic fib;
 5          reg [4:0]aux=0;
 6          assign n=aux;
 7          assign fib=((~n[4])&(~n[3])&(~n[2]))|
 8                     ((~n[4])&(~n[2])&(~n[1])&(~n[0]))|
 9                     ((~n[4])&(n[2])&(~n[1])&(n[0]))|
10                     ((~n[3])&(n[2])&(~n[1])&(n[0]));
11          always_ff@(posedge clk) begin
12                  aux <= aux+1'b1;
13          end
14  endmodule
15
16
```

Tenemos en las entradas del módulo a clk (el clock), salida n que será nuestro número binario y salida fib que será un binario 1 si el numero es de fibonacci. Finalmente, usando un módulo de clock, procedemos a simular y tenemos lo siguiente: