Taller de Lógica Digital - Parte 2

Organización del Computador 1

Primer Cuatrimestre 2023

Ejercicios

1. **Componentes de 3 estados**

a) Completar la siguiente tabla:

A	A_{en}	B	B_{en}	C	C_{en}	Estimado	Obtenido
0	0	0	0	0	0	Hi-Z	Hi-Z
0	1	1	1	0	0	error	error
1	0	1	0	1	0	Hi-Z	Hi-Z
1	1	0	0	0	1	error	error
0	1	0	1	0	1	0	0
0	1	1	1	1	1	error	error
1	0	1	1	1	0	1	1

b) Completar la siguiente tabla:

Color	Interpretación
Gris	Cable no conectado a nada
Verde claro	1
Verder oscuro	0
Azul	Hi-Z
Rojo	error

c) Enunciar la regla:

No se deben enviar un 0 y un 1 en dos entradas diferentes (A, B, C) cuando sus entradas de control son 1.

d) Explicar cuáles son y por qué:

Los combinaciones basura son aquellas en las que las entradas de control son 1 y los valores de entrada son iguales ya que se estan enviando a la vez varias señales.

2. **Transferencia entre registros**

a) Detallar entradas y salidas:

entradas: clk representa el clock, Force_Input es una entrada de data, en_Force_Input es la entrada de control que permite que pase el Force_input, w es la entrada de write y en_out la entrada que permite la salida del registro
salidas: R0, R1 y R2 son las salidas de debug de los registros y la salida de la derecha es la salida de los registrtos

clk, en_Force_Input, w y en_out son las entradas de control del circuito

b) Secuencia de señales:

Suponiendo que el clk se mueve constantemente:
w del registro R1 en 1 -> Force_Input en 1 -> en_Force_Input en 1 -> esperar flanco ascendente de clk -> w del registro R1 en 0 -> en_Force_Input en 0 -> Force_Input en 0

c) Secuencia de señales:

Suponiendo que el clk se mueve constantemente:
activo w de R0 -> pongo Force_Input en el valor deseado -> activo en_Force_Input -> espero clk -> desactivo w de R0
-> desactivo en_Force_Input -> activo en_out de R0 -> activo w de R1 -> espero clk -> desactivo w de R1 -> desactivo en_out de R0
-> activo en_out de R2 -> activo w de R0 -> espero clk -> desactivo w de R0 -> desactivo en_out de R2 -> activo en_out de R1
-> activo w de R2 -> espero clk -> desactivo w de R2 -> desactivo en_out de R1

3. **Máquina de 4 registros con suma y resta**.

a) Detallar entradas y salidas:

Con respecto a las entradas, ALU_A_Write es la entrada de control que permite la entrada de data al registro A del ALU, ALU_B_Write es la entrada de control que permite la entrada de data al registro B del ALU, OP es una entrada que permite controlar lo que hace la ALU con la data (suma, resta, or, and), ALU_enableOut es la entrada que permite que salga la data del ALU, los Reg_Write permiten escribir en los diferentes registros, los Reg_enableOut permiten la salida de cada registro, Force_Input es la entrada para ingresar un dato, en_Force_Input es la entrada para dejar pasar el dato de entrada a los registros, y clk es la entrada de clock.
Las entradas de control son todas menos el Force_Input.
Con respecto a las salidas, N es la salida que dice cuando es un numero negativo, Z es la salida que dice cuando es 0, V es la salida que dice cuando hay overflow, y C es la salida que dice cuando hay carry o borrow.

b) Detallar el contenido de cada display:

El display de abajo del todo es el que indica el valor del Force_Input o el output del ALU, los 4 displays de los Reg_Debug muestran la salida debug de cada registro, el display del A_Debug muestra el valor de A en el ALU,
el display del B_Debug muestra el valor de B en el ALU, y el display de S_Debug muestra en todo momento el resultado del ALU.

c) Secuencia de señales:

Suponiendo que el clk se mueve constantemente:
ingreso el 4 (0100) en el Force_Input -> activo el en_Force_Input -> activo Reg2_Write -> espero clk -> desactivo Reg2_Write
-> ingreo el -3 (1101) en el Force_Input -> activo Reg3_Write -> espero clk -> desactivo Reg3_Write -> desactivo en_Force_Input

d) Completar la siguiente tabla:

d) secuencia 1er caso "(4,0), OR, SUB":
Force_Input = 0100 -> en_Force_Input = 1
-> Reg0_Write = 1 -> clk = 1
-> Reg0_Write = 0 -> en_Force_Input = 0
-> Reg0_enableOut = 1 -> ALU_A_Write = 1
-> clk = 1 -> ALU_A_Write = 0
-> Reg0_enableOut = 0 -> Force_Input = 0000
-> en_Force_Input = 1 -> Reg1_Write = 1
-> clk = 1 -> Reg1_Write = 0
-> en_Force_Input = 0 -> Reg1_enableOut = 1
-> ALU_B_Write = 1 -> clk = 1
-> ALU_B_Write = 0 -> Reg1_enableOut = 0
-> OP = 11 -> ALU_enableOut = 1
-> Reg2_Write = 1 -> clk = 1 -> Reg2_Write = 0
-> OP = 01 -> Reg3_Write = 1 -> clk = 1
-> Reg3_Write = 0

Valor inicial	Resultado operación 1	Flags	Resultado operación 2	Flags
(4, 0)	0100 / 4 / 4	-	0100 / 4 / 4	-
(7, -1)	1000 / 8 / -8	N V C	0111 / 7 / 7	-
(-8, -2)	0110 / 6 / 6	V C	1010 / 10 / -6	N C
(8, -9)	no son representables, el circuito toma en complemento a 2	-	-	-

Los resultados interpretados en sin signo y en complemento a 2.

e) Explicar

Se niega la señal clk en la ALU antes de pasarle el valor al registro de salida porque queremos que el registro de salida se actualice después de hacer los cambios en la ALU, no al mismo tiempo. Si usáramos la misma señal clk entonces el ALU se actualizaría al mismo tiempo que el registro de salida, y este último tomaría el valor que tenia el ALU antes del flanco ascendente del clock (que no es lo que queremos). En este caso, necesitaría un tick (ciclo de clock) más para obtener el valor de salida esperado.

Corrección

Integrantes:

Nombre y Apellido: Lucio Schraier LU: 756/24
Nombre y Apellido: Mia Modini LU: 416/24
Nombre y Apellido: César Agustín Ruarte Sarapura LU: 149/24

Para uso de los docentes:

1	2	3