Universidad del Valle de Guatemala
Facultad de Ingeniería
Departamento de Ingeniería Electrónica
IE2011 - Electrónica Digital I
Kurt Kellner

Laboratorio #09

Este laboratorio será trabajado de forma individual y se entregará de forma digital de acuerdo a la fecha de entrega en Canvas. Deberá identificar su entrega con su nombre, carné y sección. Deberá adjuntar su código (los archivos `.v`), los diagramas de timing (tanto imágenes como los archivos `.vcd`), breves explicaciones en un archivo PDF y un link a su repositorio (donde debe haber cargado su código también). Para cada ejercicio deberá hacer un folder separado. Las simulaciones en CircuitVerse puede hacerlas en su cuenta ya que la intención es verificar el esquemático. Por eso **tiene** que incluir las imágenes en los ejercicios que se le requiera.

Ejercicio 01

Implemente un Flip Flop tipo D de 1 bit que tenga ***Enable*** y ***Reset***. Luego utilice ese módulo para implementar un Flip Flop tipo D de 2 bits y uno de 4 bits. Implemente un **Testbench** para probar todos sus Flip Flops.

Ejercicio 02

El Flip Flop tipo T (*Toggle Flip Flop*) toma el valor de su entrada y, en cada flanco de reloj, lo invierte.

Implemente un Flip Flop tipo T con ***Enable*** y ***Reset*** utilizando el Flip Flop tipo D que hizo anteriormente. Haga la simulación en CircuitVerse (puede utilizar el componente ***DflipFlop***) y también el código en Verilog junto con un testbench. En su reporte incluya imágenes de la simulación en CircuitVerse. Recuerde incluir los diagramas de timing, código etc. que está indicado en las instrucciones.

Ejercicio 03

El Flip Flop tipo JK (*JK Flip Flop*) tiene dos entradas: **J** y **K** además del **CLK**.

Si **J** y **K** son 0 entonces la salida Q mantiene su valor anterior. Si **J** = 1 y **K** = 0 entonces **Q** = 1. Si **J** = 0 y **K** = 1 entonces **Q** = 0. Si **J** y **K** son = 1 entonces la salida **Q** = **!Q** (se invierte).

Implemente un Flip Flop JK con ***Enable*** y ***Reset*** tanto en CircuitVerse como en Verilog (junto con un testbench). Incluya en su reporte imágenes de la simulación así como sus diagramas de timing.

Ejercicio 04

Implemente en Verilog un buffer Tri-estado de 4 bits. Haga un testbench para probarlo.

Ejercicio 05

Implemente en Verilog la siguiente tabla de verdad como una memoria ROM utilizando la instrucción `case`. Implemente un testbench para probar cada opción (recuerde también probar algunas de las opciones con ***don't cares***). ***NO*** es necesario probar ***todas*** las posibles opciones (serían 2^7), pero sí debe probar las mostradas en la tabla (21) y unas 5-10 adicionales.

Inputs							Outputs											
x	x	x	x	x	x	0	1	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	1	x	1	0	1	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	x	1	1	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	1	1	x	1	1	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	1	0	x	1	0	1	0	0	0	0	0	0	0	1	0	0	0
0	0	1	0	x	x	1	0	0	0	1	0	0	1	0	0	0	0	1	0
0	0	1	1	x	x	1	1	0	0	1	0	0	1	1	0	0	0	0	0
0	1	0	0	x	x	1	0	0	1	1	0	1	0	0	0	0	0	1	0
0	1	0	1	x	x	1	0	0	1	1	0	1	0	0	0	0	1	0	0
0	1	1	0	x	x	1	1	0	1	1	0	1	0	1	0	0	0	0	0
0	1	1	1	x	x	1	1	0	0	0	0	0	0	1	1	1	0	0	0
1	0	0	0	x	1	1	0	1	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	x	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	1	x	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	1	x	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0
1	0	1	0	x	x	1	0	0	1	1	0	1	1	0	0	0	0	1	0
1	0	1	1	x	x	1	1	0	1	1	0	1	1	1	0	0	0	0	0
1	1	0	0	x	x	1	0	1	0	0	0	0	0	0	0	1	0	0	0
1	1	0	1	x	x	1	0	0	0	0	0	0	0	0	0	1	0	0	1
1	1	1	0	x	x	1	0	0	1	1	1	0	0	0	0	0	0	1	0
1	1	1	1	x	x	1	1	0	1	1	1	0	0	1	0	0	0	0	0

Figure 1: Tabla de Verdad Ejercicio 05