Universidad Autónoma de Baja California

Ingeniería en Computación

Facultad de Ciencias Químicas e Ingeniería

Diseño Digital

Práctica 10: Implementación de circuitos secuenciales en VHDL

Zavala Román Irvin Eduardo

Grupo: 551

21/11/2021

Periodo 2021-2

Instrucciones

Implementar circuitos secuenciales en dispositivos programables, empleando la sintaxis adecuada de un lenguaje de descripción de hardware, para facilitar su construcción, mostrando una actitud analítica y responsable.

El alumno:

-Diseña máquinas de estado utilizando un lenguaje de descripción de hardware. Implementa una máquina de estado en lenguaje de descripción de hardware.

-Elabora un reporte de actividades que describa el diseño e implementación del circuito secuencial.

Circuito secuencial de Moore con una entrada X y una salida Z. La salida Z debe ser 1 si el número total de unos recibidos es impar y se han recibido al menos dos ceros consecutivos.

Procedimiento

El diagrama de estados del circuito que se pide es el siguiente:

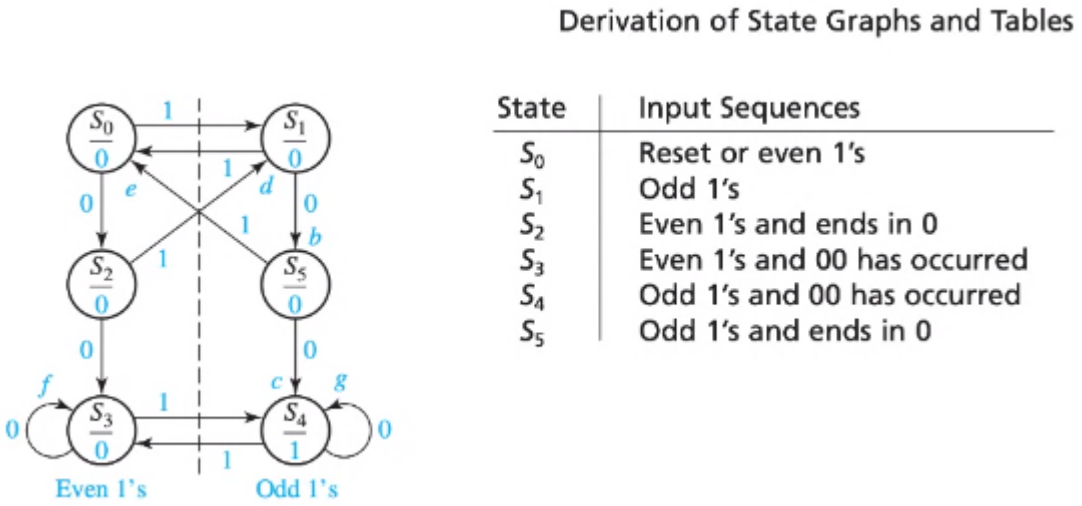

Figura 1. Diagrama de estados del circuito a implementar en VHDL

Para implementar esto en VHDL podemos ver los siguientes elementos: X, Z, CLK, estado actual y siguiente. El circuito va a recibir X cuando el CLK esté en alto, en ese momento realizará las operaciones necesarias para cambiar entre los estados y la salida Z. Por lo que quedaria asi:

```
-- Practica 10: Implementacion de circuito secuencial de Moore
library IEEE;
use IEEE.std_logic_1164.all;
entity detector is
        port(
        X: in std_logic;
        CLK: in std_logic;
        Z: out std_logic);
end detector;

architecture Behavorial of detector is
        signal state, nextState: integer range 0 to 5;
begin
   process (CLK) is
   begin
      if(CLK'event and CLK = '1') then
        Z <= '0';

        case state is
          when 0 => if X = '0' then nextState <= 2;
                    else nextState <= 1;
                    end if;
          when 1 => if X = '0' then nextState <= 5;
                    else nextState <= 0;
                    end if;
          when 2 => if X = '0' then nextState <= 3;
                    else nextState <= 1;
                    end if;
          when 3 => if X = '0' then nextState <= 3;
                    else nextState <= 4; Z<='1';
                    end if;
          when 4 => if X = '0' then nextState <= 4; Z<='1';
                    else nextState <= 3;
                    end if;
          when 5 => if X = '0' then nextState <= 4;Z<='1';
                    else nextState <= 0;
                    end if;
        end case;
      end if;
   end process;
   state <= nextState;
end Behavorial;
```

Para probar este código, creamos un testbench para ver si el comportamiento es el deseado. Para complementar esto miramos el diagrama de tiempos generado por EDAPlayGround.

```
-- Code your testbench here
library IEEE;
use IEEE.std_logic_1164.all;
entity testbench is
-- empty
end testbench;
architecture tb of testbench is
component detector is
port(
  X: in std_logic;
  CLK: in std_logic;
  Z: out std_logic);
end component;
signal X_in, Z_out,CLK_in: std_logic;
begin
        DUT: detector port map(X_in,CLK_in, Z_out);

process
begin
    X_in <= '0';
    CLK_in <='1';
    wait for 5 ns;

    X_in <= '0';
    CLK_in <='0';
    wait for 5 ns;

    X_in <= '0';
    CLK_in <='1';
    wait for 5 ns;

    X_in <= '0';
    CLK_in <='0';
    wait for 5 ns;

    X_in <= '1';
    CLK_in <='1';
    wait for 5 ns;

    X_in <= '0';
    CLK_in <='0';
    wait for 5 ns;

    X_in <= '0';
    CLK_in <='1';
    wait for 5 ns;
    wait;

    end process;
end tb;
```

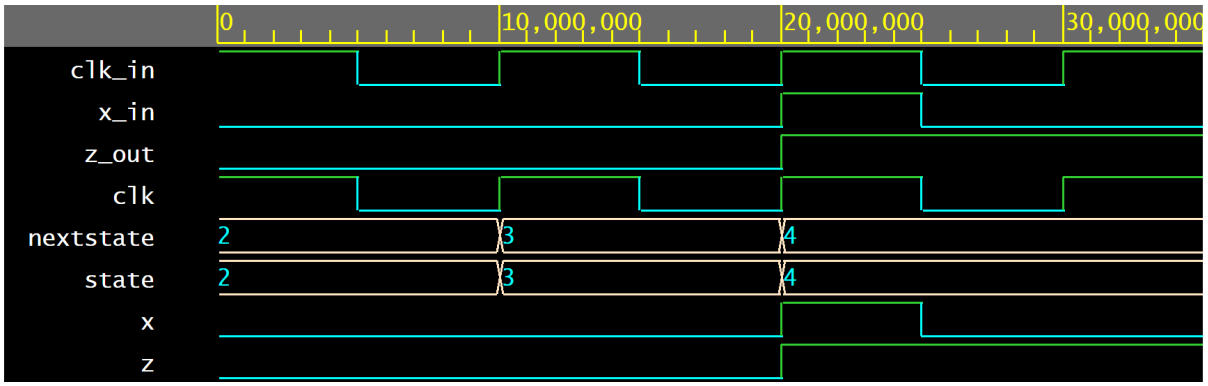

Figura 2. Diagrama de tiempos generado por VHDL

Como vemos, se insertan 2 ceros, 1 uno y 1 cero nuevamente. El recorrido de los estados sería S0, S2, S3, S4, S4 con Z = 1 en el estado 4. Vemos que el diagrama de tiempos coincide tanto en los estados como en la salida.

Para pasar este codigo de VHDL a un GAL, usamos el ispLever y nos genera la siguiente configuración de pines:

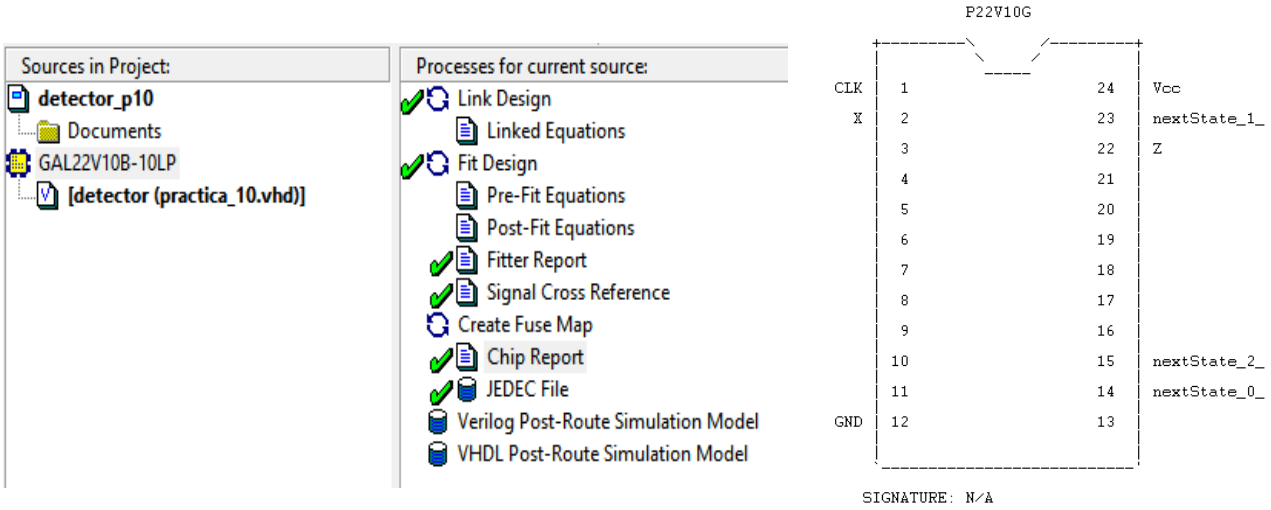

Figura 3. Resultado ispLever

Agarramos el archivo .jedec y lo implementamos en Proteus con un AM22V10.

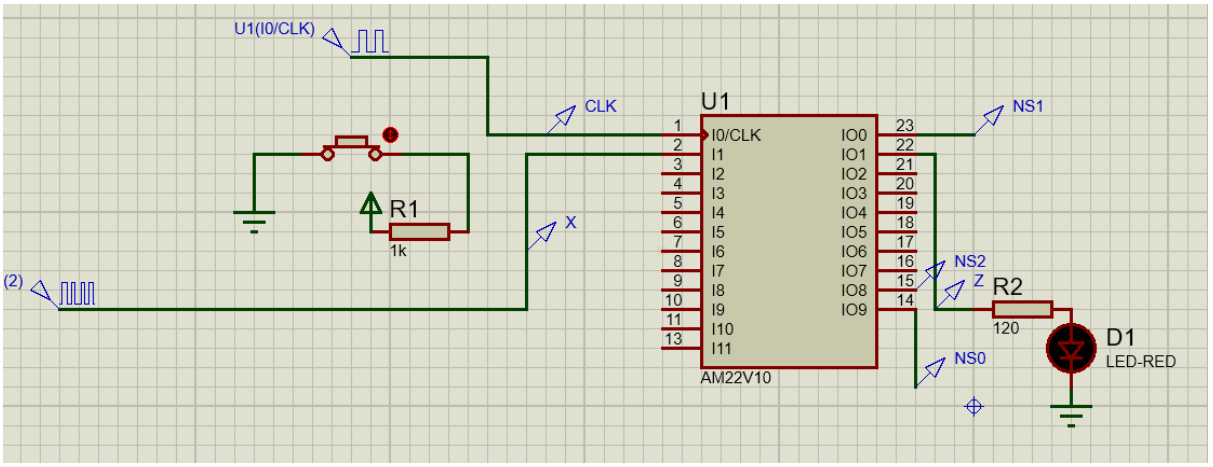

Figura 4. Implementación en Proteus del circuito secuencial en VHDL

Podemos ver que hay un interruptor al aire, esto es por si queremos meter X de manera manual. Pero lo que se va a hacer es usar un generador de pulsos para realizar una secuencia de entrada visible en un diagrama digital en Proteus.

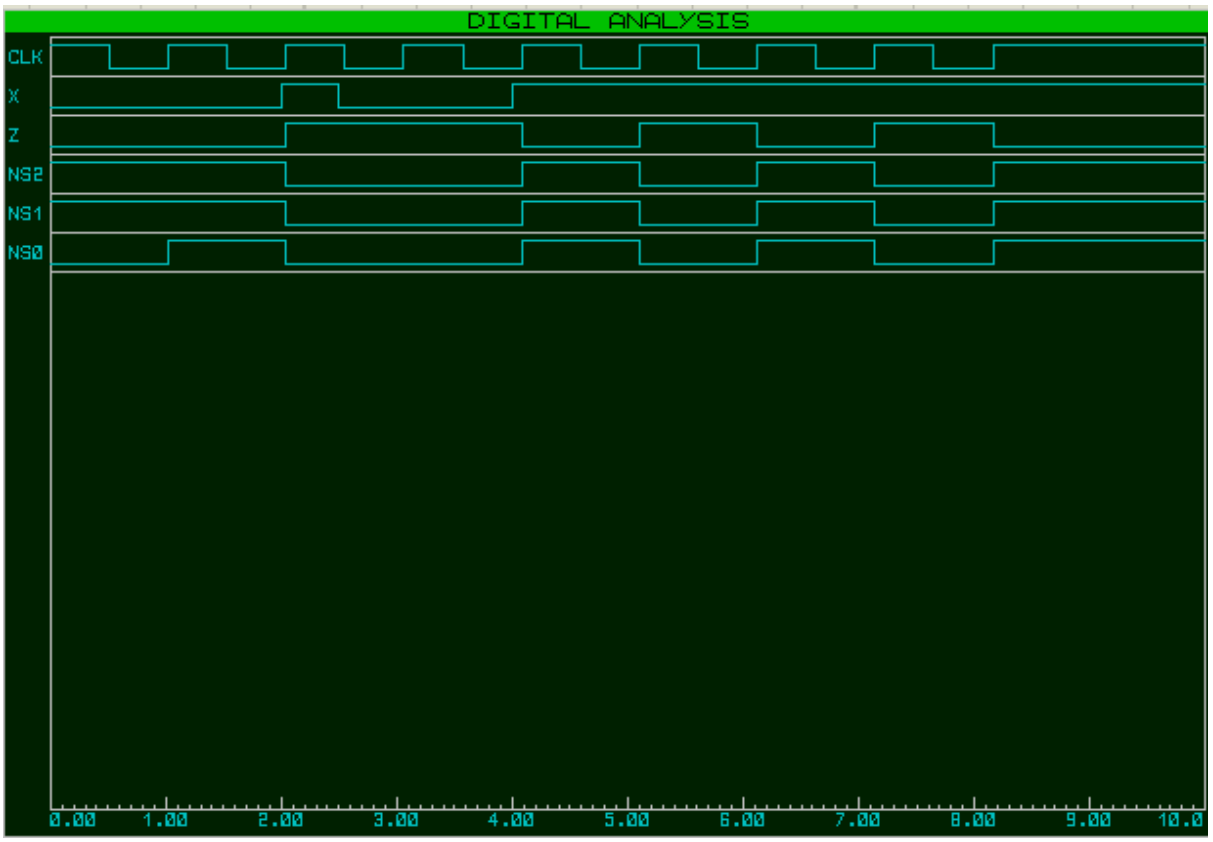

Figura 5. Diagrama de tiempos del GAL en Proteus

Podemos ver que los tiempos son los esperados del diagrama de estados, solo que los estados se muestran por bits, lo cual explica que hay un momento que cambia entre 000 y 111, es porque en realidad cambia entre los estados 1000 y 111 (4 y 3) e ispLever solo nos genero 3 bits para representar el estado.

Conclusión

Esta práctica fue bastante interesante ya que implica que con solo tener el diagrama de estados de un circuito secuencial, puedo implementarlo de forma sencilla en VHDL y de ahí a Proteus, que sería equivalente a hacerlo en un chip real. Esto simplifica procesos, aunque tiene limitaciones como la cantidad de pines y que es necesario VHDL.

Anexos

Practica 10: https://www.edaplayground.com/x/pjmq