UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

INGENIERÍA EN COMPUTACIÓN

Laboratorio de diseño de sistemas digitales

Practica 1:

Reporte

MARIACA VAZQUEZ ENRIQUE

Grupo: 8457

2024-I I

TURNO MATUTINO

PROFESOR. FRANCISCO ESTRIBERO GONZALES

Fecha de realización 16 de febrero de 2024

Hora de realización: 11:00am. – 01:00pm.

Previo Práctica 1.

1) Defina con sus palabras los niveles de abstracción en Verilog.

- **Nivel de puerta:** En este nivel, definimos las compuertas básicas bajo un modelo estructural, or, not, and… etc, siendo lo que parece el nivel de diseño más bajo.
- **Nivel de transferencia de registro:** En este nivel definimos los datos de registros o guardados de datos, subiendo un nivel más en comparación con las compuertas básicas.
- **Nivel de comportamiento:** En este nivel diseñaremos el comportamiento de nuestro sistema con las bases de los niveles anteriores.

2) Describa los siguientes tipos de procesos.

- **Initial**: Es un proceso que solo se ejecuta una vez, comenzando en el tiempo cero y no es sintetizable.
- **Always**: Es un proceso que se puede ejecutar continuamente en forma de bucle, es sintetizable y puede ser controlada su ejecución por medio de un temporizador, también puede ser ejecutado por eventos.

3) Defina y escriba la sintaxis para las asignaciones continuas en Verilog.

Estos son formas de asignación de valores, debe de estar declarada del tipo wire y tiene que ser declarada fuera de cualquier proceso, es decir, fuera de bloques always e initial. Son ejecutadas secuencialmente, pero pueden ser llamadas con temporizadores o eventos.

La sintaxis es:

Assign variable #<delay> = f(wire, reg, constante numérica)

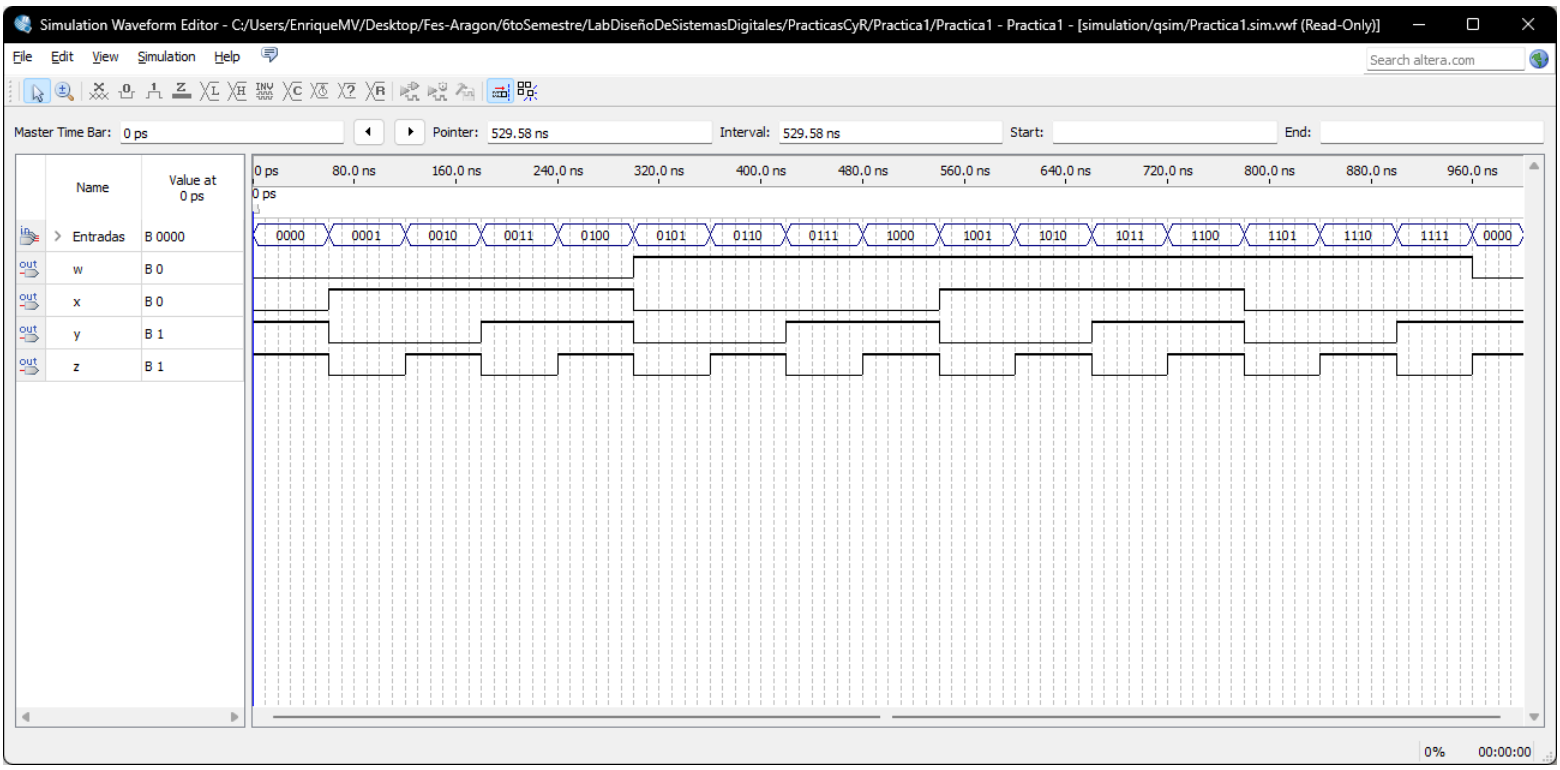
Reporte

- **Código:**

```
module myAND(input a, input b, output z); //Modulo para generar la compuerta AND entre 2 valores
   assign z = a & b;
endmodule

module myOR(input a, input b, output z); //Modulo para generar la compuerta OR entre 2 valores
   assign z = a|b;
endmodule

module myNOT(input a, output z); //Modulo para generar la compuerta NOT de 1 valor
   assign z = ~a;
endmodule

module myNAND(input a, input b, output z); //Modulo para generar la compuerta NAND entre 2 valores
   assign z = ~a & ~b;
endmodule


module Exceso3 (input systemA, input systemB, input systemC, input systemD, output w, output x, output y, output z);
   wire ORtoAND;
   wire ANDtoOR;
   wire NOTtoAND1;
   wire NOTtoAND2;
   wire ANDtoOR2;
   wire ANDtoOR3;  //Declaramos las coneciones o cables, variables wire para almacenar datos entre operaciones
   wire NANDtoAND;
   wire ANDtoOR5;
   wire ORtoAND5;
   wire NOTtoAND5;
   wire ANDtoOR52;
   wire NANDtoOR;

//Aqui empieza W
myOR OR1(
.a(systemD),
.b(systemC),
.z(ORtoAND)
);

myAND AND1(
.a(ORtoAND),
.b(systemB),
.z(ANDtoOR)
);

myOR OR2(
.a(ANDtoOR),
.b(systemA),
.z(w)
);
```

```
    //Aqui empieza X
    myNAND NAND2(
    .a(systemC),
    .b(systemD),
    .z(NANDtoAND)
    );

    myAND AND4(
    .a(systemB),
    .b(NANDtoAND),
    .z(ANDtoOR5)
    );

    myOR OR4(
    .a(systemD),
    .b(systemC),
    .z(ORtoAND5)
    );

    myNOT NOT2(
    .a(systemB),
    .z(NOTtoAND5)
    );

    myAND AND5(
    .a(NOTtoAND5),
    .b(ORtoAND5),
    .z(ANDtoOR52)
    );

    myOR OR5(
    .a(ANDtoOR5),
    .b(ANDtoOR52),
    .z(x)
    );

  //Aqui empieza Y

  myNAND NAND1(
  .a(systemC),
  .b(systemD),
  .z(NANDtoOR)
  );

  myAND AND3(
  .a(systemC),
  .b(systemD),
  .z(ANDtoOR3)
  );

  myOR OR3(
  .a(NANDtoOR),
  .b(ANDtoOR3),
  .z(y)
  );

  //Aqui empieza Z
  myNOT NOT1(
  .a(systemD),
  .z(z)
  );


endmodule
```

- **Simulación**



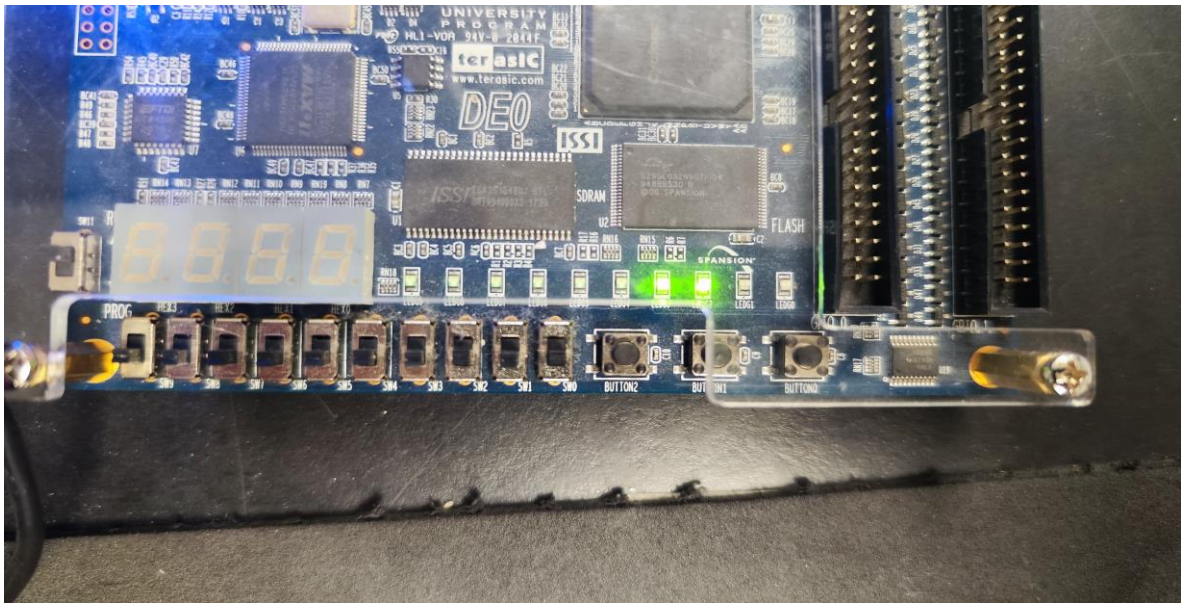
- **Conclusiones**

El lenguaje descriptivo Verilog nos permite una gran versatilidad de acciones en cuestión a la programación y configuración de software, es importante como ingenieros conocer su funcionamiento y las implicaciones que tiene, por ejemplo, el diseño de sistemas para dar solución a múltiples problemas.

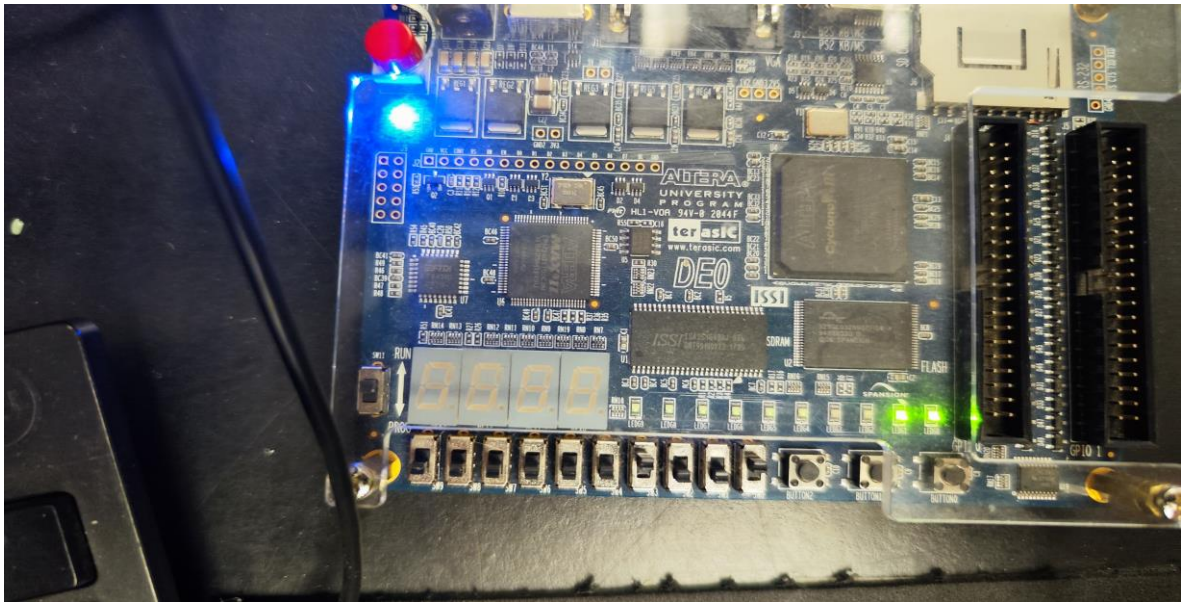
A través de lo desarrollado en clase me he dado cuenta de algunos usos que tiene este lenguaje y su posterior implementación en una FPGA, queda registrado a través de esta práctica mi primer acercamiento concientizado al programa y al lenguaje, dejando una clara impresión sobre cómo debo informarme del uso, configuración, programación e implementación de este software para el control de hardware.

Regístrese en las siguientes fotografías los resultados obtenidos post implementación del código en la FPGA, siendo las configuraciones de entrada:

1. 0000



2. 0100



3. 1001