Universidad Nacional Autónoma de México

Facultad de Ingeniería

Ingeniería en Computación

Asignatura: Diseño Digital VLSI

Grupo: 4

Practica 5: Diseño Del Control De Servomotores.

Profesor: M.I. Alberto Navarrete Hernández

Alumnos:

- Jiménez Gutiérrez Miguel
- Martínez Ortiz Carlos Daniel

Diseño Del Control De Servomotores.

Objetivo: El alumno aprenderá la manera de organizar un proyecto de manera modular y separarlo en diferentes archivos, con la finalidad de que vaya construyendo su propia biblioteca de módulos funcionales, y que pueda reutilizar los módulos generados en otros proyectos.

Introducción

Un sistema diseñado en VHDL debe simularse y probarse para la funcionalidad antes de que se convierta en hardware. En este pase de simulación, errores de diseño e incompatibilidad de los componentes utilizados en el diseño pueden ser detectados. La simulación de un diseño requiere la generación de datos de prueba y la observación de los resultados de la simulación.

Especificaciones:

Diseñar el control de un servomotor de modelismo utilizando en un FPGA, en el cual, por medio de cuatro interruptores de presión tipo push-button, se pueda controlar la posición del eje del motor. Dos de los interruptores permitirán llevar al eje a cada una de las posiciones extremas, mientras que los otros permitirán que el motor gire en cada dirección avanzando paso a paso a través de doce posiciones definidas cada vez que el interruptor es presionado. La determinación de la posición se realizará por medio de una señal PWM. La figura 1.1 muestra el diagrama de bloques de este sistema.

Diagrama de bloques:

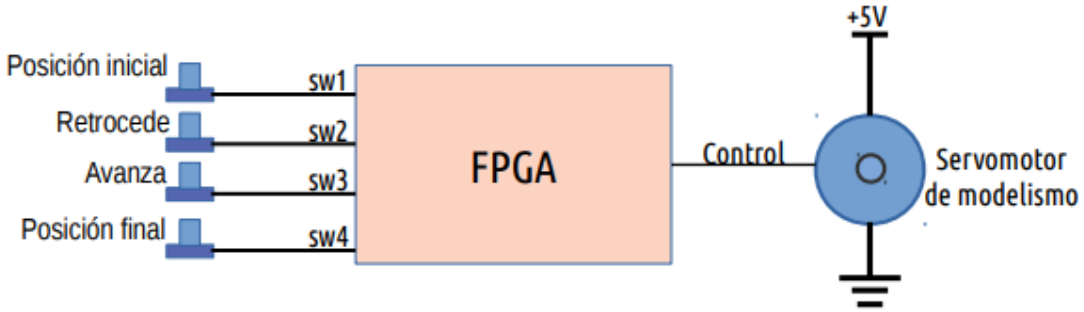


Figura 1.1 Diagrama de bloques del control de un servomotor de modelismo.

En la elaboración de un proyecto basado en un FPGA, comúnmente se desarrolla una gran cantidad de módulos funcionales para manejar las tareas necesarias en cierta aplicación. Una buena práctica de diseño es la de utilizar cada uno de esos módulos de manera independiente, ya que esto simplifica el proceso de diseño y permite distribuir las diferentes tareas entre varios grupos de trabajo. Además, si se hace una buena división de tareas, al final se contará con un conjunto de módulos funcionales que eventualmente podrán ser reutilizados en otros proyectos. De esta

manera, al aplicar la metodología de diseño, el alumno podrá ir construyendo su propia biblioteca de módulos funcionales, lo que en el futuro le permitirá reducir los tiempos de diseño al reutilizar estos módulos. Esto implica que cada módulo funcional debe estar contenido en un archivo diferente.

Para el desarrollo de esta práctica se aplicará este concepto de división en módulos funcionales, cada uno de ellos contenidos en un archivo diferente, que posteriormente serán integrados en un solo proyecto al ser instanciados en el módulo principal. La figura 1.2 muestra los bloques funcionales que componen al control del servomotor.

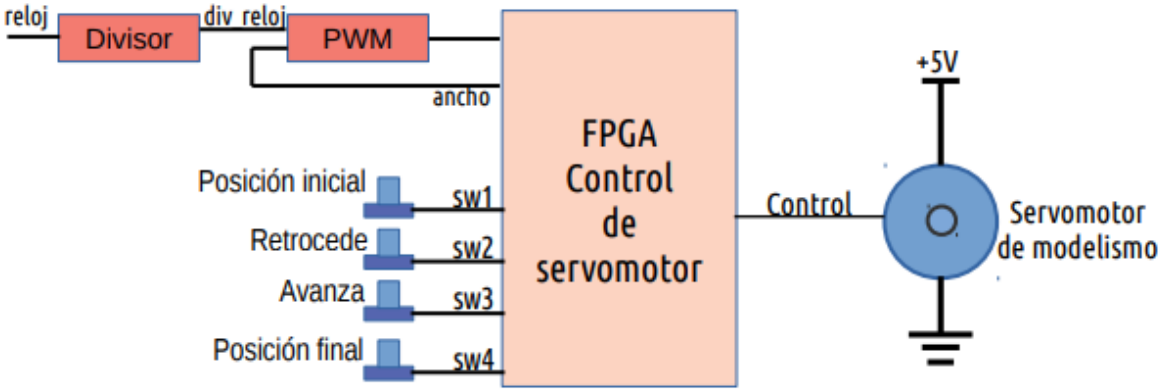


Figura 1.2 Bloques funcionales del control del servomotor.

Para la elaboración de este proyecto, se diseñarán dos módulos funcionales de aplicación genérica, el módulo Divisor y el módulo PWM, que podrán ser los dos primeros módulos funcionales de la biblioteca del alumno, además el módulo principal dedicado a la aplicación específica del control del servomotor manipulado por cuatro interruptores, en donde se instanciarán los dos módulos de uso general.

El primer módulo para diseñar es el correspondiente al divisor, el cual generará, a partir de la señal de reloj de 50 MHz de la tarjeta de desarrollo, una señal de salida cuya frecuencia corresponde a dividir la señal de entrada, entre una potencia de dos. La frecuencia de salida estará definida por el valor de la constante N. El código muestra para este módulo, se presenta a continuación y se deberá guardar en un archivo.

Desarrollo

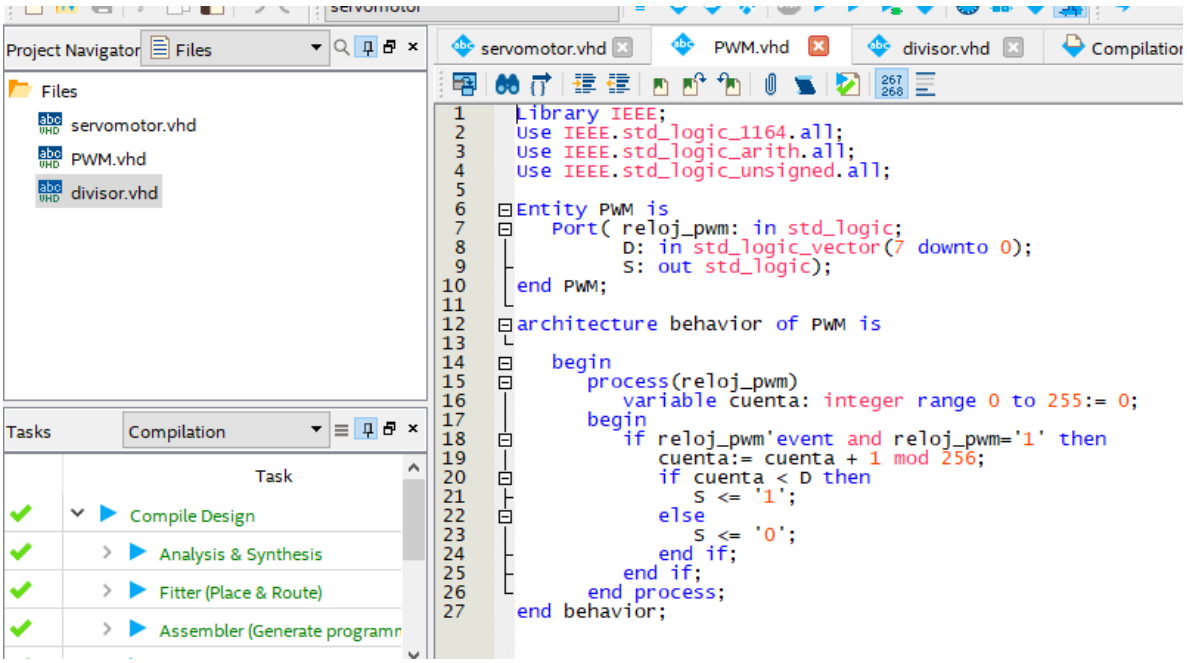
1.- Siguiendo la metodología de diseño presentada, el alumno elaborará un módulo funcional genérico para controlar un servomotor de modelismo, que complementará la biblioteca de módulos del alumno.

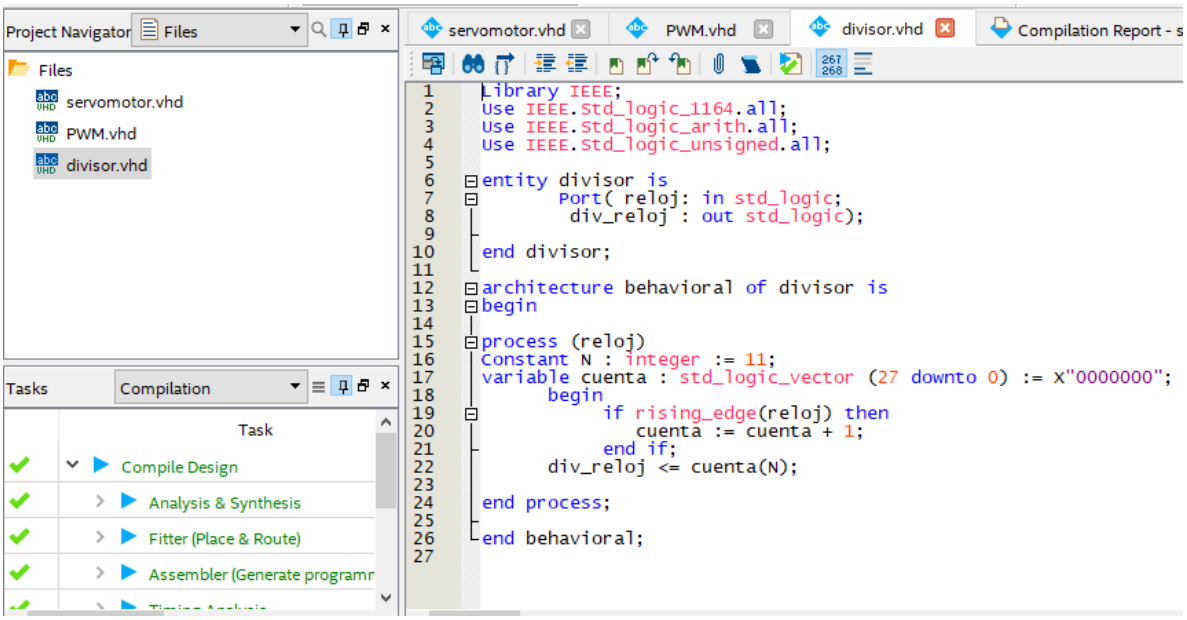
Código en VHDL

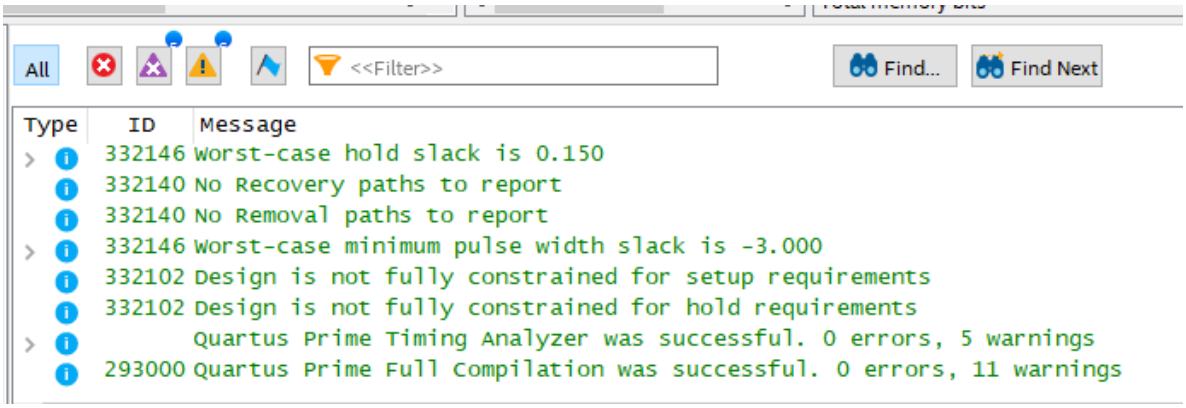
```
Library IEEE;
Use IEEE.Std_logic_1164.all;
Use IEEE.Std_logic_arith.all;
Use IEEE.Std_logic_unsigned.all;

Entity servomotor is
     Port (reloj_sv : in std_logic;
Pini, Pfin, Inc, Dec : in std_logic;
           control : out std_logic);


End entity;

Architecture Behavioral of servomotor is
Component divisor is
    Port (reloj : in std_logic;
    reloj_divclk: out std_logic);

End Component;

Component pwm is
Port (reloj_pwm : in std_logic;
             D: in std_logic_vector(7 downto 0);
             S: out std_logic);

End Component;

Signal reloj_serv: std_logic;
```

```
Signal ancho : std_logic_vector(7 downto 0) := X"0F";

Begin
    U1: Entity work.divisor(behavioral) Port map(reloj_sv, reloj_serv);
    U2: Entity work.pwm(behavior) Port map(reloj_serv, ancho, control);

            Process(reloj_serv, Pini, Pfin, Inc, Dec)
    Variable valor : std_logic_vector(7 downto 0) := X"0F";
    Variable cuenta : integer range 0 to 4023 := 0;

    Begin
      If reloj_serv= '1' and reloj_serv'event then
        If (cuenta > 0) then
          cuenta := cuenta - 1;
        Else
          If Pini = '1' then  --0
            valor := X"0D";
            Elsif Pfin = '1' then  --0
                valor := X"18";
            Elsif Inc = '1' and valor < X"18" then  --0
                valor := valor + 1;
            Elsif Dec = '1' and valor > X"0D" then   --0
                valor := valor - 1;
          End if;
                                          cuenta := 4023;
        End if;
      End if;
```

```
        --cuenta := 1023
        ancho <= valor;
      End process;

End behavioral;
```

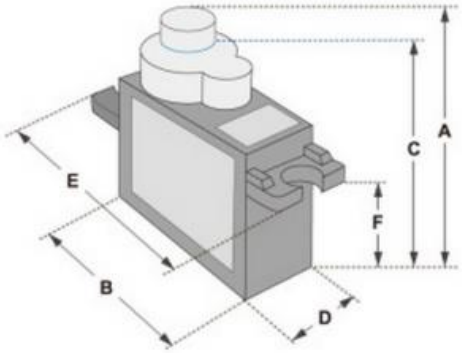






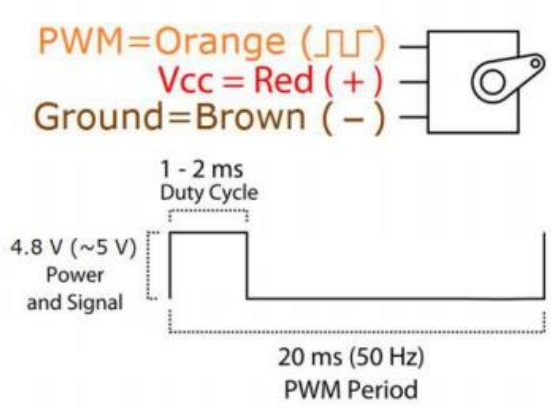
Asignación del Pin Planner

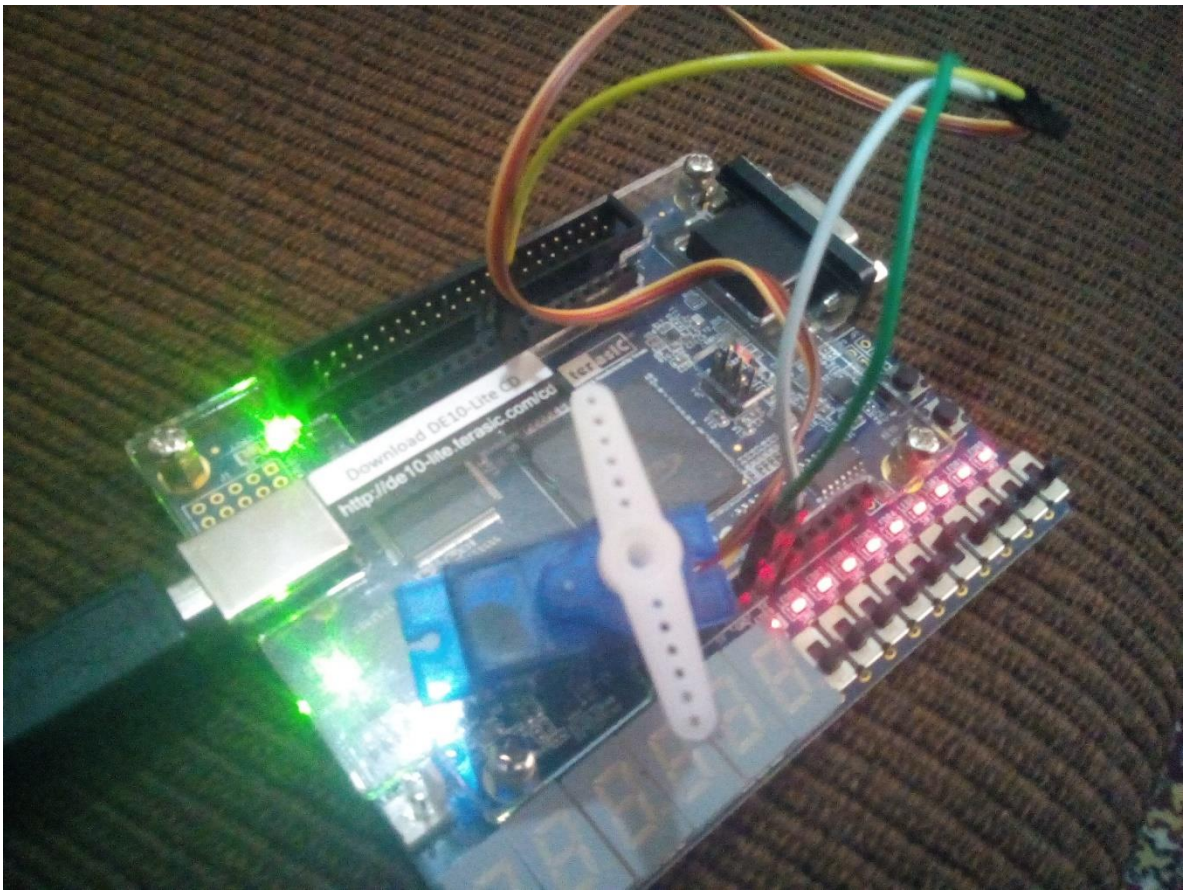
Node Name	Direction	Location	
out control	Output	PIN_AB17	4
in Dec	Input	PIN_C10	7
in Inc	Input	PIN_C11	7
in Pfin	Input	PIN_C12	7
in Pini	Input	PIN_A12	7
in reloj_sv	Input	PIN_P11	3
<<new node>>			



Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is middle, is all the way to the right, "-90" (~1ms pulse) is all the way to the left.





2.- Utilizando el módulo genérico para controlar un servomotor diseñado, construir un sistema que maneje dos servomotores de modelismo de forma complementaria, es decir, se moverán de la misma forma, pero girando en la dirección opuesta.

Para esta actividad yo sugiero implementar otra variable tipo std_logic llamada control 1, ya que ya tengo una llamada control, en conjunto con otro motor y su respectiva entidad PWM.

La variables relacionadas al giro tendrán parámetros diferentes a las originales , es decir en reversa

Conclusión

Para realizar esta práctica fue importante poner atención a las conexiones hechas en la FPGA, es de mucha utilidad dividir un sistema grande en pequeñas partes de él, así con los módulos si hay un problema de funcionalidad es más fácil corregirlo en un único módulo que estar buscando errores en un programa grande de varias líneas de código

Video demostrativo:

https://drive.google.com/drive/folders/1_fil1dsxC2neLzFlPJC7Y6oCwMuHPtuK?usp=sharing

Referencia:

Navabi, Z. (2007). VHDL:Modular desing ans synthesis of cores and systems. USA: McGraw-Hill

http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf
Consultado el 12 de noviembre de 2020