Universidad Nacional Autónoma de México
Facultad de Ingeniería
División de Ingeniería Eléctrica

Diseño Digital VLSI (1535)

Semestre 2023-1, Grupo 5

Proyecto 3: Control de pluma ferroviaria con sensores ultrasónicos

Integrantes:
Angel Alvarado Campos
Luis Carlos Ramos Rosas

Profesora: M.I. Elizabeth Fonseca Chavez

Objetivos

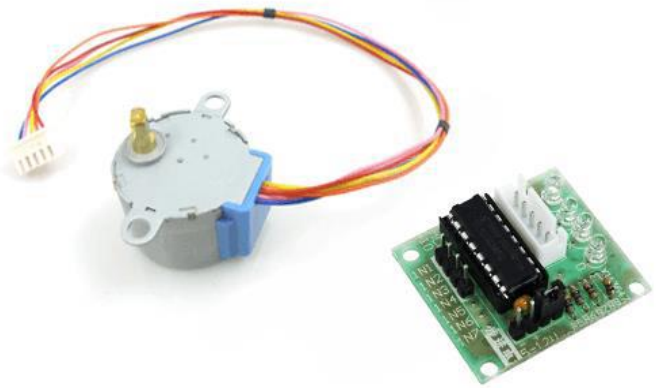
Crear un proyecto compuesto, con estructura TOP, a partir de proyectos previos realizados en la asignatura.

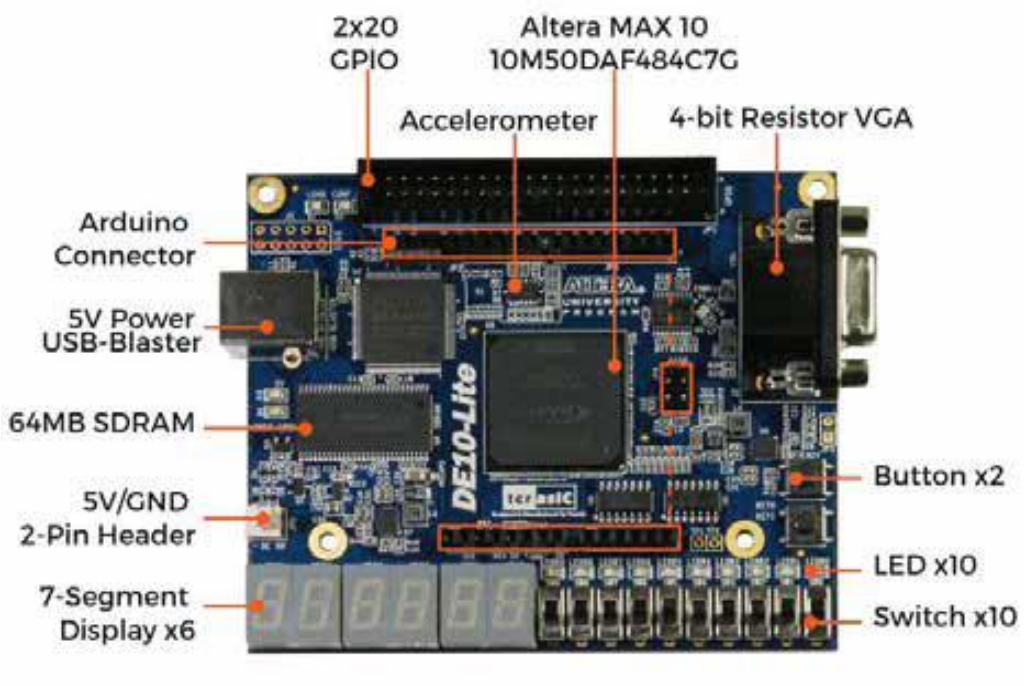
En particular, para este proyecto se busca implementar sensores ultrasónicos en el proyecto 2 entregado anteriormente. En nuestro caso, el proyecto 2 es el control de una pluma ferroviaria en un *cruce a nivel*.

A su vez, el proyecto 2 fue desarrollado a partir del proyecto 1 (ALU automatizada) y el módulo de memoria ROM.

Materiales

- Tarjeta FPGA: DE10-Lite
- *Jumpers* macho-hembra y *jumpers* macho-macho
- Un motor a pasos 28BYJ-48
  - Controlador ULN2003AN
- Dos sensores ultrasónicos HC-SR04





¿Qué es un cruce a nivel?

Es una intersección al mismo nivel entre una vía ferroviaria y un camino o carrera.

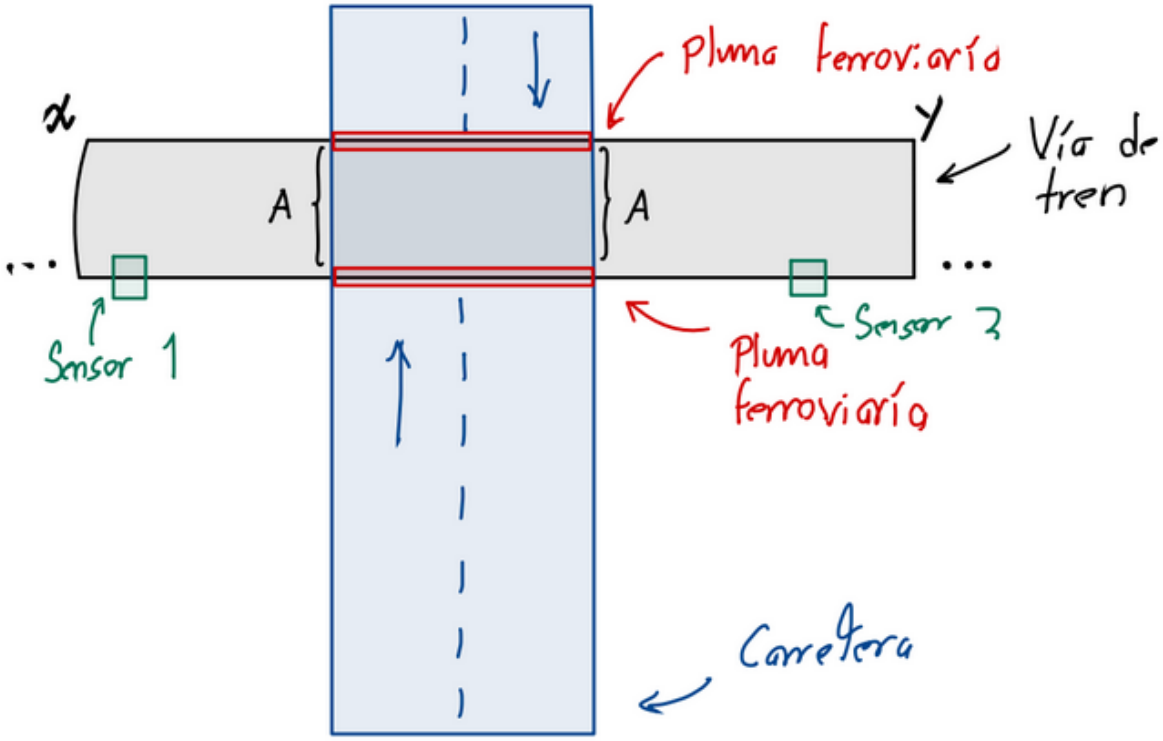
En un cruce a nivel **los trenes siempre tienen prioridad**.

Planteamiento del problema

El control de una pluma ferroviaria consiste en dos partes:

- Control del movimiento de la pluma
- Control de las órdenes para indicar movimiento en la pluma

En este proyecto buscamos controlar el movimiento de la pluma en función de los sensores que detectan el movimiento del tren.



Desarrollo

Requisitos:

- Utilizar el proyecto 2.
- Añadir sensores a lo realizado en el proyecto 2.

En nuestro proyecto 2, la pluma ferroviaria rotaba en función de un único pulso dato por la activación de un *switch* o botón, y cambiaba la dirección de rotación de manera automática y secuencial.

→ Ahora buscamos conectar los sensores a dicho pulso para que el movimiento de rotación tenga sentido en el contexto del cruce a nivel.

Diagrama RTL del proyecto 2 (Versión 1)

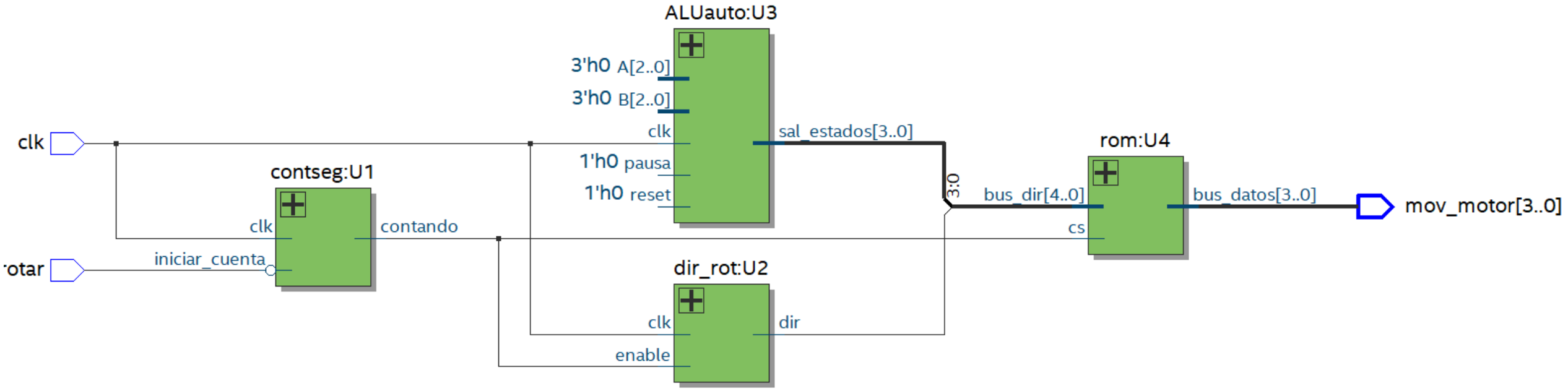
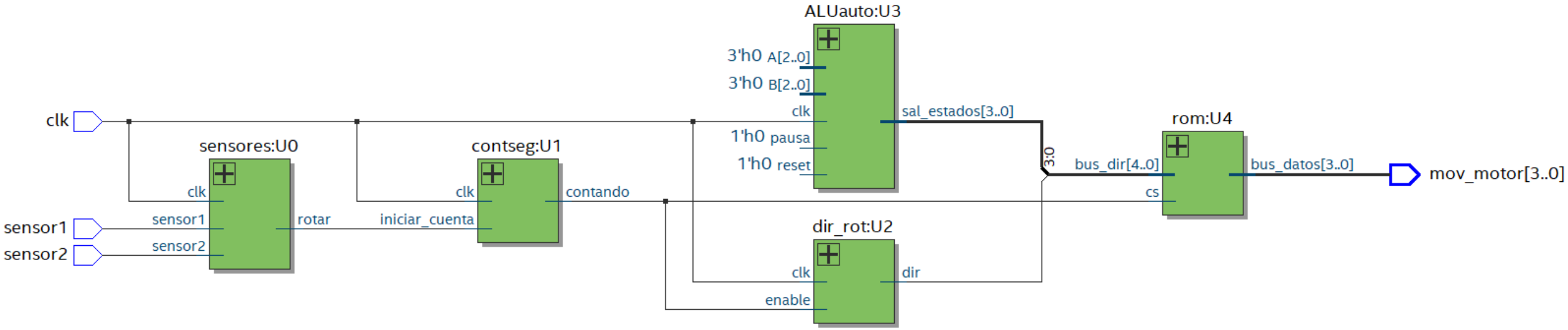


Diagrama RTL del proyecto 2 (Versión 2)



Se añadió el módulo “sensores”, el cual es una máquina de estados

Máquina de estados de sensores

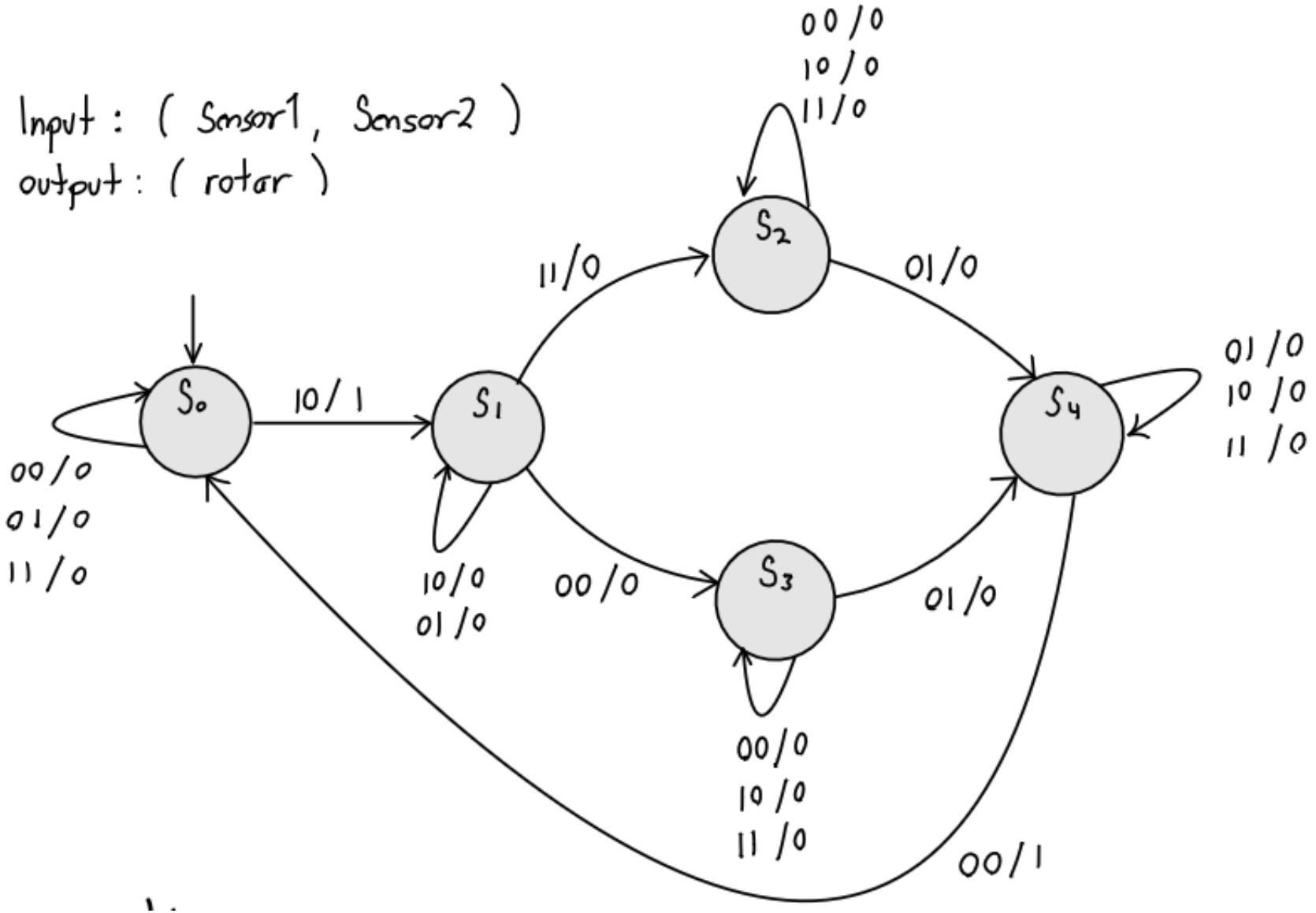
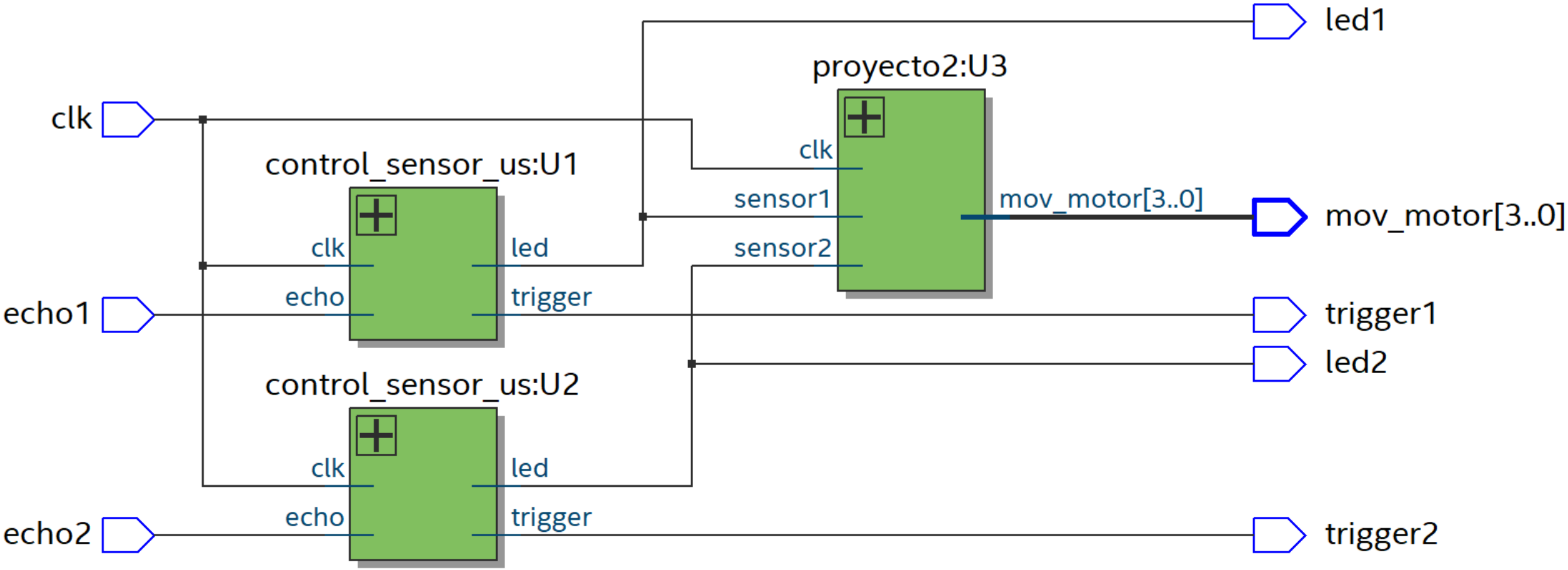


Diagrama RTL del proyecto 3



TOP de proyecto 3

proyecto3.vhd

```
1  -- TOP de proyecto 3
2  -- Integrantes:
3  --    Angel Alvarado Campos
4  --    Luis Carlos Ramos Rosas
5
6  library ieee;
7  use ieee.std_logic_1164.all;
8  use ieee.std_logic_unsigned.all;
9
10 entity proyecto3 is
11    port(
12       clk:  in std_logic;
13       echo1,echo2: in std_logic; -- echo sensor
14       trigger1,trigger2: buffer std_logic;   -- Trigger sensor
15       mov_motor: out std_logic_vector(3 downto 0); -- Vector de movimiento del motor
16       led1,led2: out std_logic
17    );
18 end entity;
19
20 architecture arch_proyecto3 of proyecto3 is
21 signal sensor1,sensor2: std_logic;  -- Distancia de lectura de sensores con T = 588 [µs] -> D ~ 5 [cm]
22 begin
23    U1: entity work.control_sensor_us(arch) generic map(588) port map(clk,echo1,trigger1,sensor1,open,open);
24    U2: entity work.control_sensor_us(arch) generic map(588) port map(clk,echo2,trigger2,sensor2,open,open);
25    U3: entity work.proyecto2(arch_proyecto2) port map(clk,sensor1,sensor2,mov_motor);
26    led1 <= sensor1;
27    led2 <= sensor2;
28 end architecture;
```

control_sensor_us.vhd

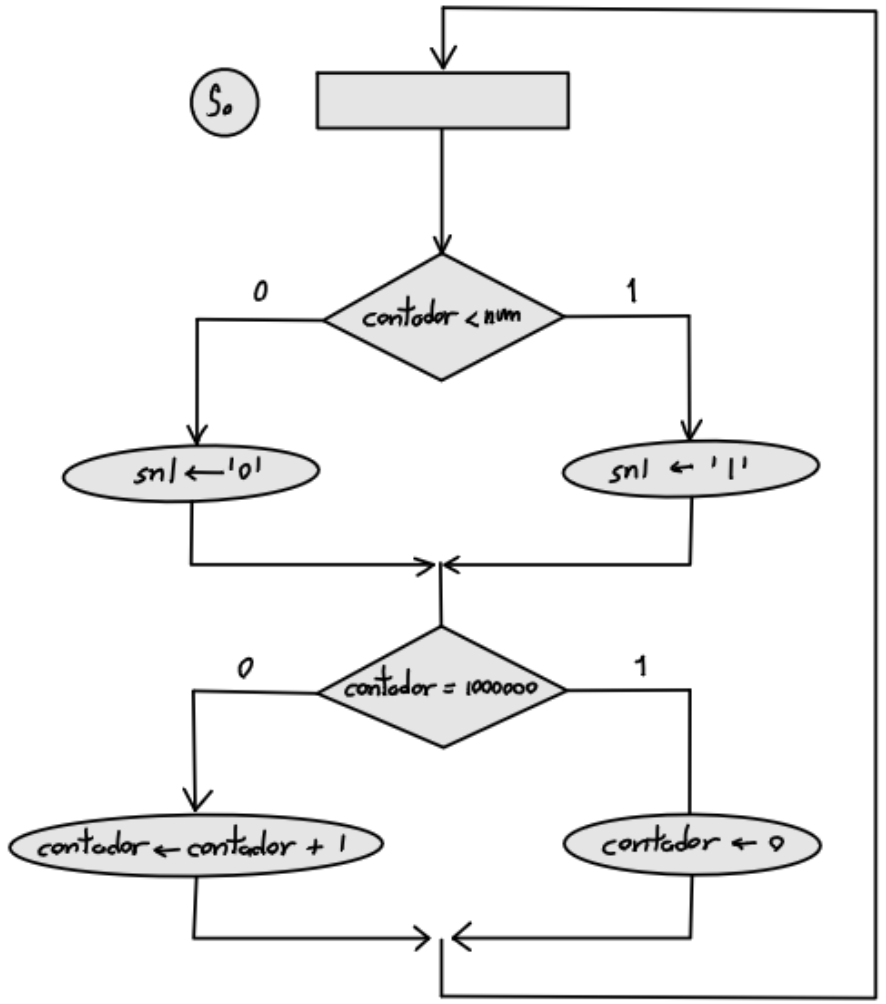
control_sensor_us.vhd

```vhdl
-- TOP del proyecto de control de sensor ultrasónico
-- Integrantes:
--    Angel Alvarado Campos
--    Luis Carlos Ramos Rosas

library ieee;
use ieee.std_logic_1164.all;

entity control_sensor_us is
   generic(
      lim: integer := 1235
   );
   port(
      clk:  in std_logic;
      echo: in std_logic;
      trigger: buffer std_logic;
      led: out std_logic;
      D1,D0: out std_logic_vector(6 downto 0)
   );
end entity;

architecture arch of control_sensor_us is
signal clk_aux: std_logic;
signal distancia: integer;

begin
   U1: entity work.divf(arch_divf) generic map(25) port map('0',clk,clk_aux); -- Genera una señal con frecuenci
   U2: entity work.senal_us(arch_senal_us) generic map(10) port map(clk_aux,trigger); -- Genera señal de entrad
   U3: entity work.contador_us(arch_contador_us) generic map(lim) port map(clk_aux,trigger,echo,led,distancia);
   U4: entity work.display_us(arch_display_us) port map(distancia,D1,D0); -- Imprime la distancia en display
end architecture;
```

senal_us.vhd

senal_us.vhd

```vhdl
-- Modulador de ancho de pulso
-- Integrantes:
--    Angel Alvarado Campos
--    Luis Carlos Ramos Rosas

library ieee;
use ieee.std_logic_1164.all;

entity senal_us is
   generic(
      num: integer := 500000
   );
   port(
      clk: in std_logic;
      snl: out std_logic
   );
end entity;

architecture arch_senal_us of senal_us is
signal contador: integer range 0 to 1000000 := 0;
begin
   process(clk)
   begin
      if (rising_edge(clk)) then
         if (contador < num) then
            snl <= '1'; -- Lanza pulso de 1 [us] respecto a iteraciones
         else
            snl <= '0';
         end if;

         if (contador = 1000000) then
            contador <= 0;
         else
            contador <= contador + 1;
         end if;

      end if;
   end process;
end architecture;
```



contador_us.vhd

contador_us.vhd*

```
1  -- contador de distancia
2  -- Integrantes:
3  --    Angel Alvarado Campos
4  --    Luis Carlos Ramos Rosas
5
6  library ieee;
7  use ieee.std_logic_1164.all;
8
9  entity contador_us is
10    generic(
11       lim: integer := 1235
12    );
13    port(
14       clk: in std_logic;    -- Reloj
15       trigger,echo: in std_logic;
16       salida: out std_logic;
17       distancia: out integer
18    );
19 end entity;
20
21 architecture arch_contador_us of contador_us is
22 signal conteo: integer range 0 to 12000;  -- Conteo de
23 signal contar: std_logic; -- '0' = no, '1' = si
24 begin
25    process(clk,trigger,echo)
26    begin
27       if (trigger = '1') then
28          conteo <= 0;
29          salida <= '0';
30          distancia <= 0;
31       elsif (rising_edge(clk)) then
32          if (echo = '1') then
33             conteo <= conteo + 1;
34          else
35             if (conteo >= 117 and conteo < lim) then
36                salida <= '1';
37                distancia <= conteo;
38             end if;
39             conteo <= 0;
40          end if;
41       end if;
42    end process;
43 end architecture;
```

Principales dificultades

Este proyecto fue más sencillo de aterrizar que el proyecto 2 porque en éste únicamente se tuvieron que implementar sensores a lo realizado previamente. Las dificultades que existieron fueron las siguientes.

1. Organizar físicamente a los dispositivos conectados a la tarjeta FPGA en el proyecto.
2. Manejo adecuado de señales de reloj.

Aprendizajes

1. Tener un conocimiento adecuado sobre el uso de máquinas de estado es fundamental al momento de desarrollar sistemas digitales.
2. Se debe tener cuidado al momento de conectar dispositivos electrónicos externos a la tarjeta FPGA, pues estos pueden sufrir daños si no se operan con las señales correctas. En nuestro caso, se daño un sensor ultrasónico.
3. Los sensores ultrasónicos usados no deben suponerse como dispositivos infalibles.

Conclusiones del proyecto

Concluimos este proyecto afirmando que se logró implementar en una FPGA un sistema digital con *aplicación en la vida real* usando sensores ultrasónicos, motor a pasos, y módulos de ALU automatizada y memoria ROM.

Al abstraer el problema en módulos, se pudo llegar a una solución con más facilidad.

Conclusiones del curso

A lo largo del curso aprendimos a implementar sistemas digitales complejo por medio de la conexión entre módulos independientes; es decir, aprendimos a hacer TOPs. Por otro lado, también logramos utilizar diferentes dispositivos externos para extender la funcionalidad de los sistemas implementados en la tarjeta FPGA.

Bibliografía

- Colaboradores de la Facultad de Ingeniería, DIE. (2021). *Prácticas de diseño digital VLSI 2022 para tarjeta FPGA Altera-Intel (DE10-Lite).*