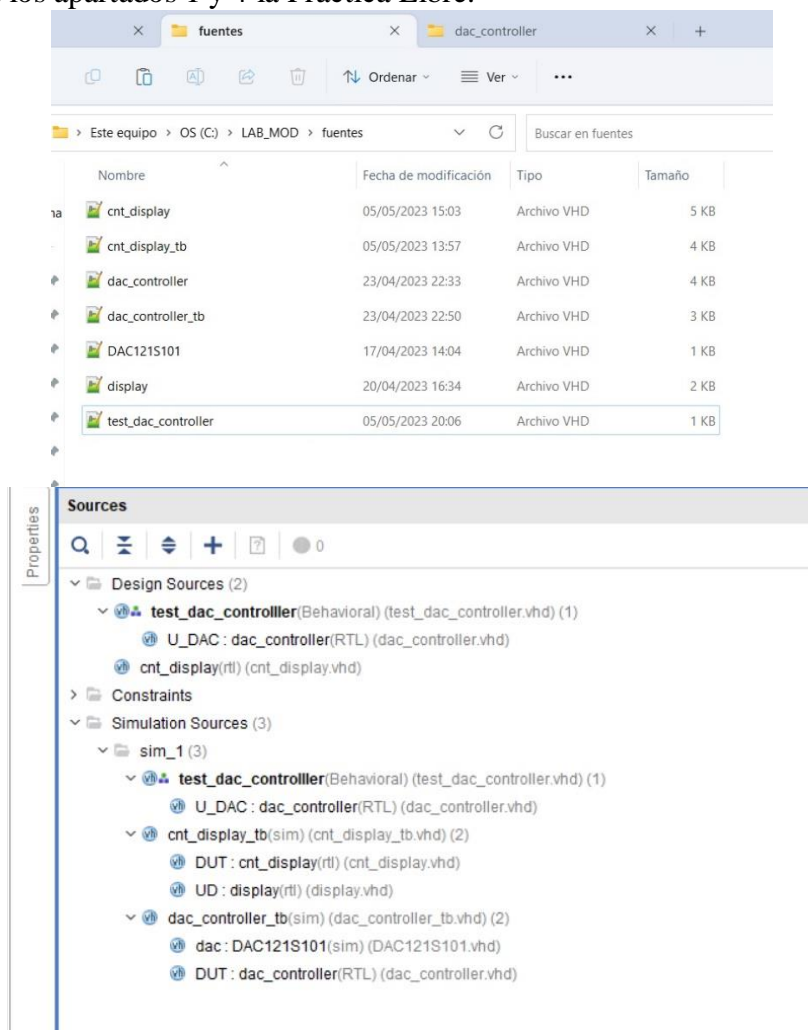


LABORATORIO DE MODELADO DE SISTEMAS COMPUTACIONALES		FECHA:	08-05-2023
APELLIDOS	FERREEROSA TRUQUE	NOMBRE	JHON JAMES

## EXAMEN INDIVIDUAL DE LA PRÁCTICA LIBRE

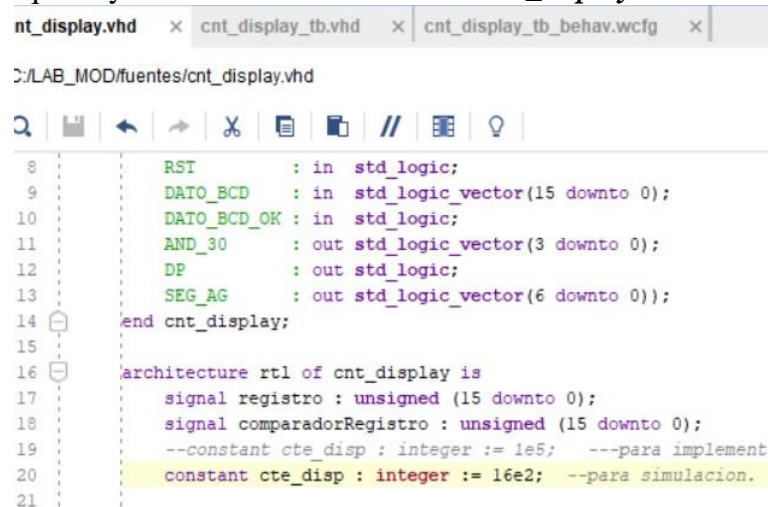
Duración del examen: 100 minutos.

1. Crea un directorio en **C:/** de nombre **LAB\_MOD**. En este, crea una carpeta con nombre **sources** donde almacenarás **sólo** los ficheros fuente (**sin incluir** los ficheros de formas de ondas \*.wcfg) correspondientes a los apartados 1 y 4 la Práctica Libre.



2. En **Vivado**, crea un proyecto denominado **proy\_mod**, para la FPGA de la familia **Artix 7 XC7A35TICPG236-1L** y añade al proyecto los ficheros fuente correspondientes a los apartados 1 y 4 de la práctica libre. **Haz una captura de pantalla y pégala a continuación** donde se vea la jerarquía de todos los ficheros tanto para síntesis e implementación (**Design Sources**) como para simulación (**Simulation Sources**). Asimismo, **incluye otra captura de pantalla** de la estructura de directorios (carpetas) resultante una vez creado el proyecto.

3. Utilizando los modelos VHDL del **apartado 1** correspondientes al módulo *cnt\_display*:
- Modifica el código fuente para que periodo de refresco (*Refresh period*) sea igual a 16  $\mu$ s. Indica justificadamente el porqué de las modificaciones realizadas.
  - Captura las líneas que hayas modificado en la entidad *cnt\_display* e insértalas a continuación.



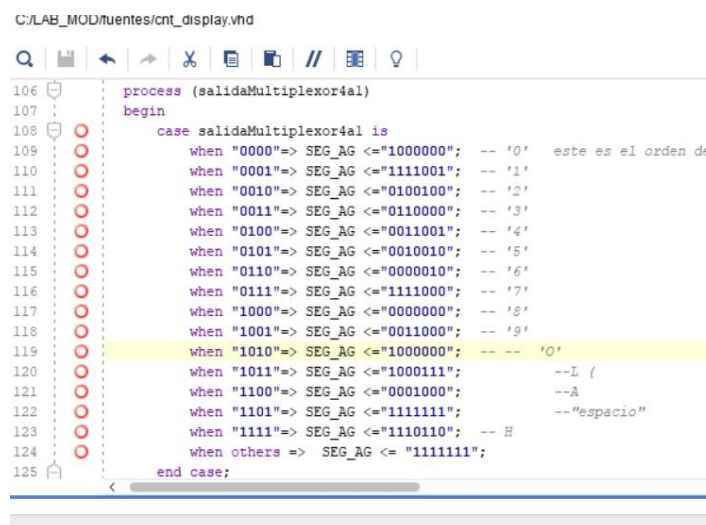
```

nt_display.vhd x cnt_display_tb.vhd x cnt_display_tb_behav.wcfg x
C:/LAB_MOD/fuentes/cnt_display.vhd

8      RST      : in  std_logic;
9      DATO_BCD : in  std_logic_vector(15 downto 0);
10     DATO_BCD_OK : in  std_logic;
11     AND_30    : out std_logic_vector(3 downto 0);
12     DP       : out std_logic;
13     SEG_AG    : out std_logic_vector(6 downto 0);
14 end cnt_display;
15
16 architecture rtl of cnt_display is
17     signal registro : unsigned (15 downto 0);
18     signal comparadorRegistro : unsigned (15 downto 0);
19     --constant cte_disp : integer := 1e5; ---para implement
20     constant cte_disp : integer := 16e2; --para simulacion.
21

```

- Modifica el código fuente teniendo en cuenta que cuando se reciba un dato no BCD se visualice en los displays la palabra **HOLA**.
- Captura las líneas que has modificado en la entidad *cnt\_display* e insértalas a continuación.



```

C:/LAB_MOD/fuentes/cnt_display.vhd

106 process (salidaMultiplexor4al)
107 begin
108     case salidaMultiplexor4al is
109     when "0000" => SEG_AG <= "1000000"; -- '0' este es el orden de
110     when "0001" => SEG_AG <= "1111001"; -- '1'
111     when "0010" => SEG_AG <= "0100100"; -- '2'
112     when "0011" => SEG_AG <= "0110000"; -- '3'
113     when "0100" => SEG_AG <= "0011001"; -- '4'
114     when "0101" => SEG_AG <= "0010010"; -- '5'
115     when "0110" => SEG_AG <= "0000010"; -- '6'
116     when "0111" => SEG_AG <= "1111000"; -- '7'
117     when "1000" => SEG_AG <= "0000000"; -- '8'
118     when "1001" => SEG_AG <= "0011000"; -- '9'
119     when "1010" => SEG_AG <= "1000000"; -- -- '0'
120     when "1011" => SEG_AG <= "1000111"; -- L (
121     when "1100" => SEG_AG <= "0001000"; -- A
122     when "1101" => SEG_AG <= "1111111"; -- "espacio"
123     when "1111" => SEG_AG <= "1110110"; -- H
124     when others => SEG_AG <= "1111111";
125 end case;

```

- Modifica el archivo necesario para que en la simulación sólo se vea el procesamiento/respuesta a tres datos: uno BCD sin ceros a la izquierda, otro BCD con 2 ceros a la izquierda y uno que tenga sólo un dígito no BCD.
- Captura las líneas que hayas modificado del archivo correspondiente, indicando el nombre de este, e insértalas a continuación.

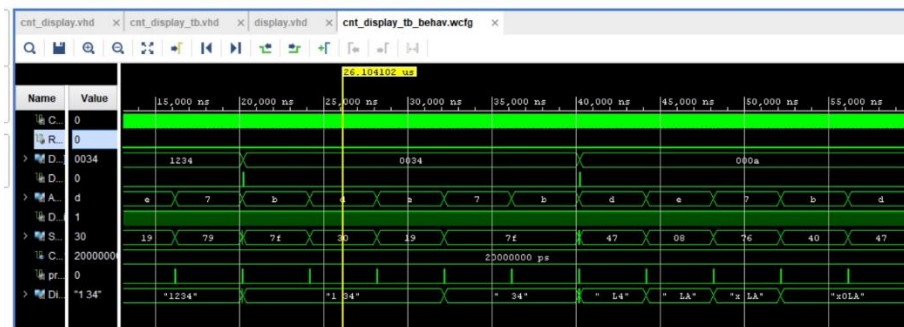
```

cnt_display.vhd x cnt_display_tb.vhd x display.vhd x cnt_display
C:/LAB_MOD/fuentes/cnt_display_tb.vhd

53 wait until CLK_i = '0';
54 DATO_BCD_OK_i <= '1';
55 DATO_BCD_i <= X"1234";-- Indicar valor
56 wait until CLK_i = '0';
57 DATO_BCD_OK_i <= '0';
58 wait for CNT1;
59
60
61 wait until CLK_i = '0';
62 DATO_BCD_OK_i <= '1';
63 DATO_BCD_i <= X"0034";-- Indicar valor
64 wait until CLK_i = '0';
65 DATO_BCD_OK_i <= '0';
66 wait for CNT1;
67
68 wait until CLK_i = '0';
69 DATO_BCD_OK_i <= '1';
70 DATO_BCD_i <= X"000A";-- Indicar valor
71 wait until CLK_i = '0';
72 DATO_BCD_OK_i <= '0';

```

- g. Ejecuta la **simulación funcional** y configura la ventana de formas de ondas del simulador para ver **todos** los puertos y **todas** las señales necesarias para comprobar que el modelo funciona correctamente. Se debe ajustar la ventana de formas de onda para que se vea la simulación completa. Recuerda que son tres datos.
- h. **Incluye una captura de pantalla** que muestre la simulación anterior y pégala en el hueco de abajo.



4 Utilizando los modelos VHDL del **apartado 4** correspondientes al controlador del DAC (*dac\_controller*):

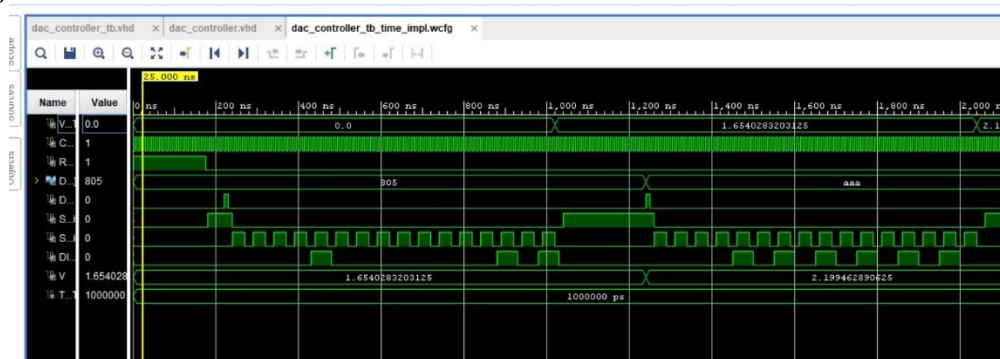
- a) Modifica el código teniendo en cuenta que la salida SCLK debe estar a nivel alto 30 ns y 20 ns a nivel bajo.
- b) Captura las líneas que hayas modificado en la entidad *dac\_controller* e insértalas a continuación.

```

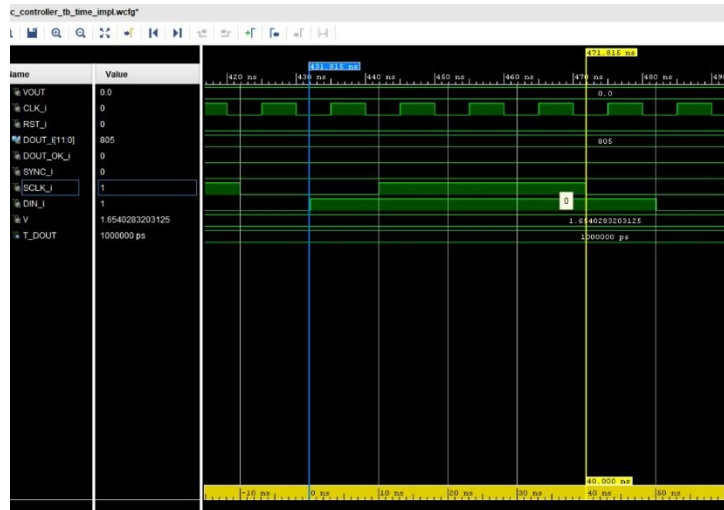
13     DIN      : out std_logic);
14 end dac_controller;
15
16 architecture RTL of dac_controller is
17     signal registroDesplazamiento : std_logic_vector (15 downto 0);
18     signal cnt_bits : unsigned (4 downto 0);
19     type fsm_type is (st0, st1, st2, st3, st4, stExtra);
20     signal estado, prox_estado : fsm_type;
21     signal miClockEnable : std_logic;
22     signal miShift : std_logic;
23     signal miSCLK : std_logic;
24     signal miSYNC : std_logic;
25 begin
26
27     --registro desplazamiento.
28     process (all)
29
30         prox_estado <= st1;
31     end if;
32     when st1 =>
33         prox_estado <= st2;
34     when st2 =>
35         prox_estado <= stExtra;
36     when stExtra =>
37
38     end if;
39 end process;
40 --maquina de estados, calculado de las salidas.
41 miClockEnable <= '1' when (estado = st4) else '0'; --antes estaba en el estado st2.
42 miShift <= '1' when (estado = st4) else '0';
43 miSCLK <= '1' when ((estado = st1) or (estado = st2) or (estado = stExtra)) else '0';
44 miSYNC <= '1' when (estado = st0) else '0';
45
46 --contador.
47 process (all)
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

- c) Modifica el testbench del módulo para que se envíen sólo dos datos: 805<sub>hex</sub> y AAA<sub>hex</sub>.
- d) Ejecuta la **simulación temporal** y configura la ventana de formas de onda del simulador para comprobar que el modelo funciona correctamente. Se debe ajustar la ventana de formas de onda para que se vea la totalidad de la simulación, es decir la transmisión completa de los dos datos mencionados anteriormente.
- e) **Incluye una captura de pantalla** que muestre la simulación anterior y pégala en el hueco de abajo.



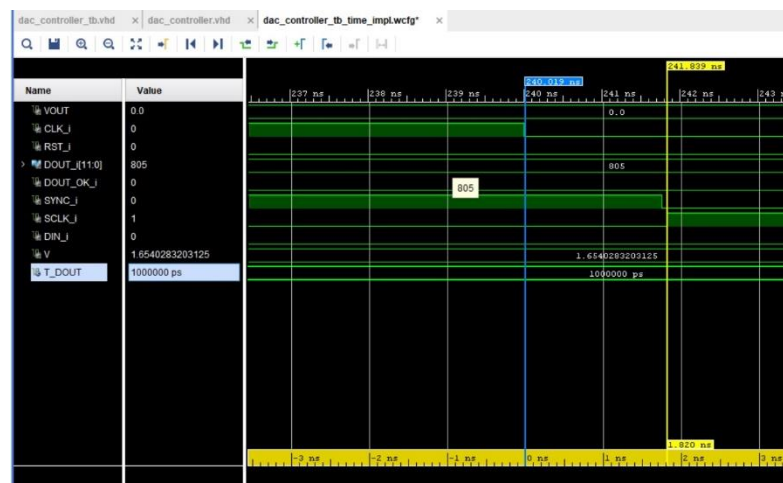
- f) Amplia una zona de la simulación y mide, utilizando cursores, el tiempo de *setup* de DIN respecto a SCLK.
- g) **Incluye una captura de pantalla** que muestre la medida anterior y pégala en el hueco de abajo.



- a) Vuelve a ampliar una zona de la simulación y mide, utilizando cursores, el retardo existente entre los flancos de subida de CLK y SCLK. Indica aquí cuál es ese valor de retardo medido.

1.820 ns

- b) **Incluye una captura de pantalla** que muestre la medida anterior y pégalas en el hueco de abajo.



5. Convierte este fichero a **pdf** y junto con el directorio **sources**, comprime todo a formato Zip y súbelo a la actividad de nombre **D\_indv\_may\_2023** creada en la BB

Nota. La carpeta **sources** deberá contener, también, todos los archivos de configuración de las formas de onda (cwfg) que se han utilizado para esta defensa.