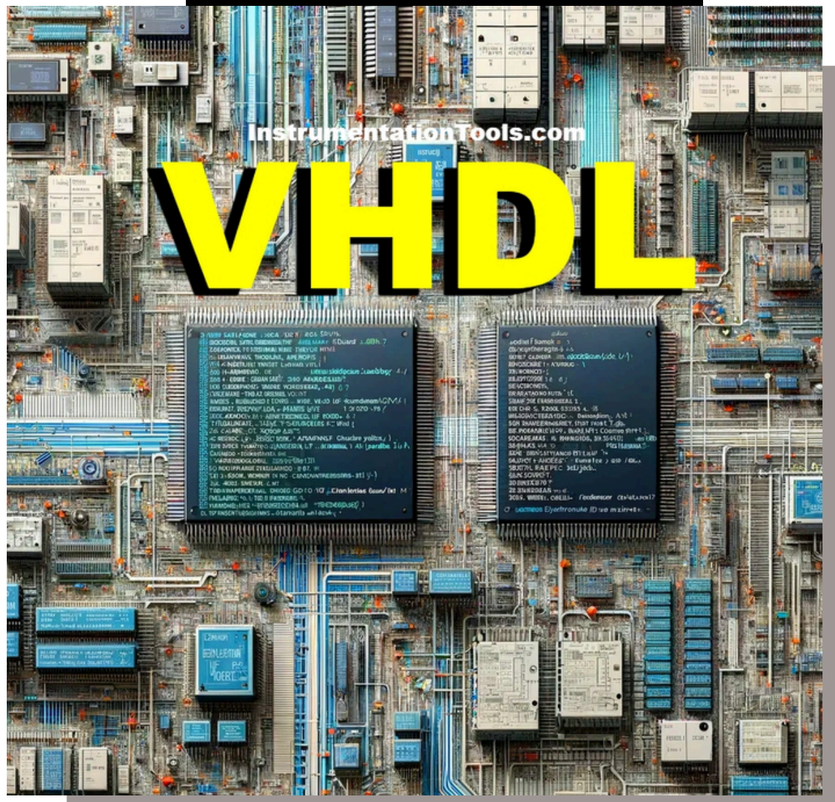
APLICACIÓN DE VHDL CON FPGA

SENSOR DE TEMPERATURA

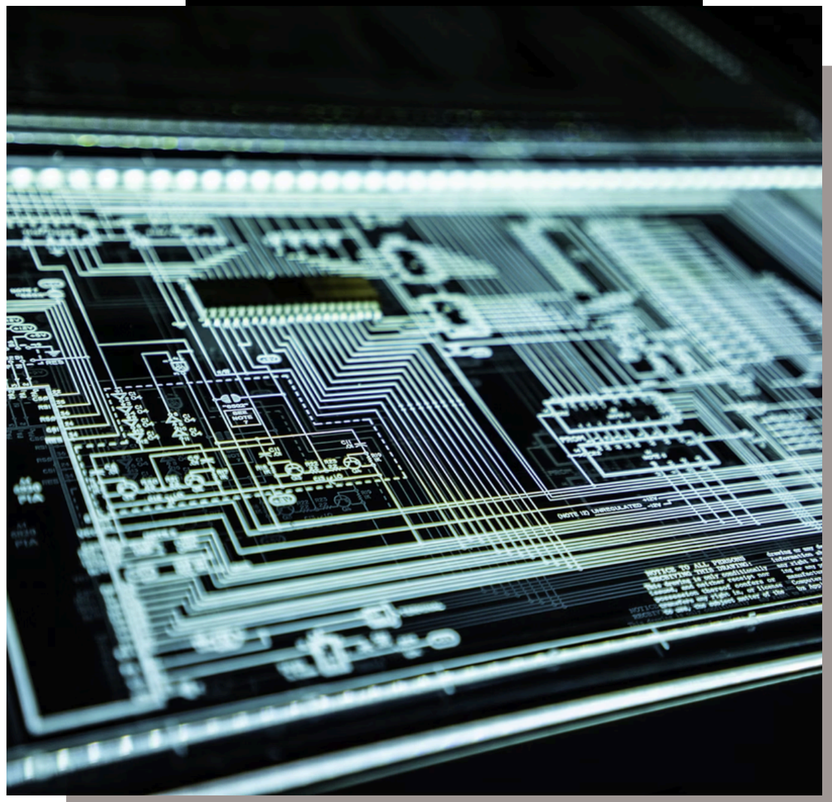


Presentada por Grupo C:

ANTHONY YAGUANA
STEFANO CRUZ
MARÍA DEL CISNE VALAREZO
GERARDO NAULA

AGENDA

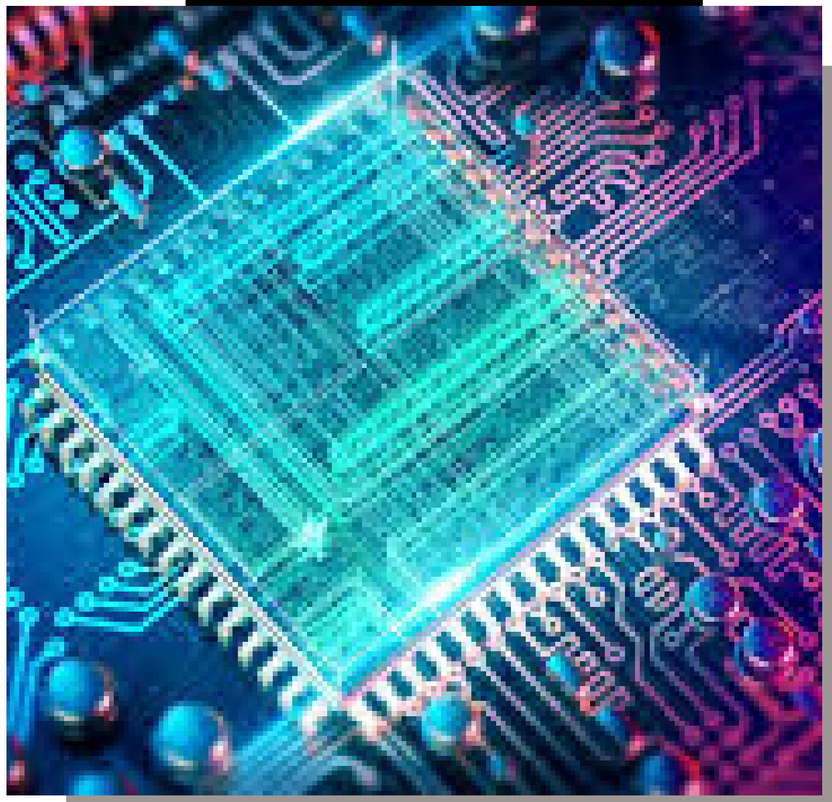
1	OBJETIVOS INTODUCCION:
2	PROYECTO



OBJETIVOS

Diseñar e implementar un sistema sensor de temperatura con la ayuda del procesador FPGA utilizando el lenguaje de hardware VHDL, capaz de procesar señales de temperatura provenientes de un sensor analógico LM35 mediante el uso de un convertidor analógico a digital (ADC0804).

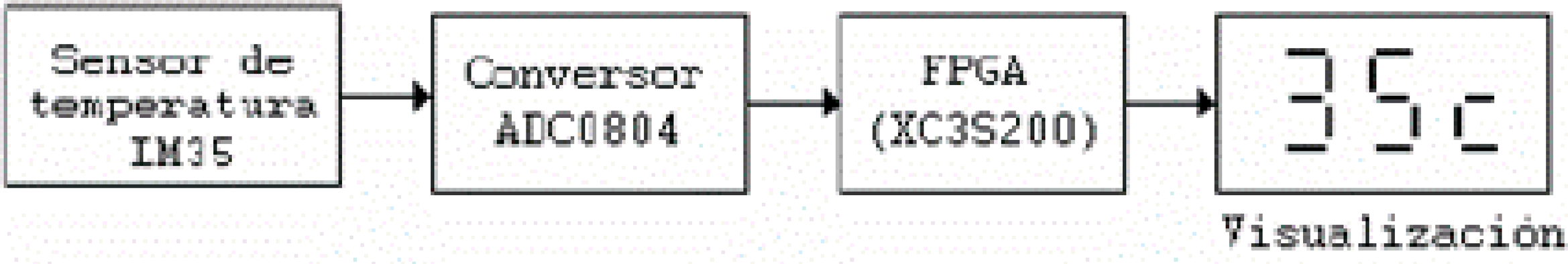
Optimizar la funcionalidad y precisión del termómetro garantizando una conversión eficiente de las señales analógicas a digitales y su correcta interpretación en la FPGA para mostrarlecturas de los valores adecuados en tiempo real.

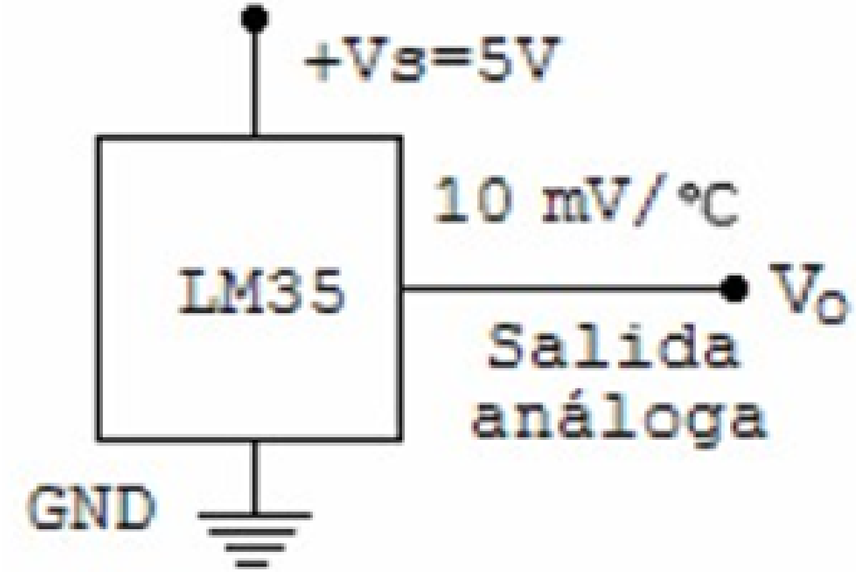


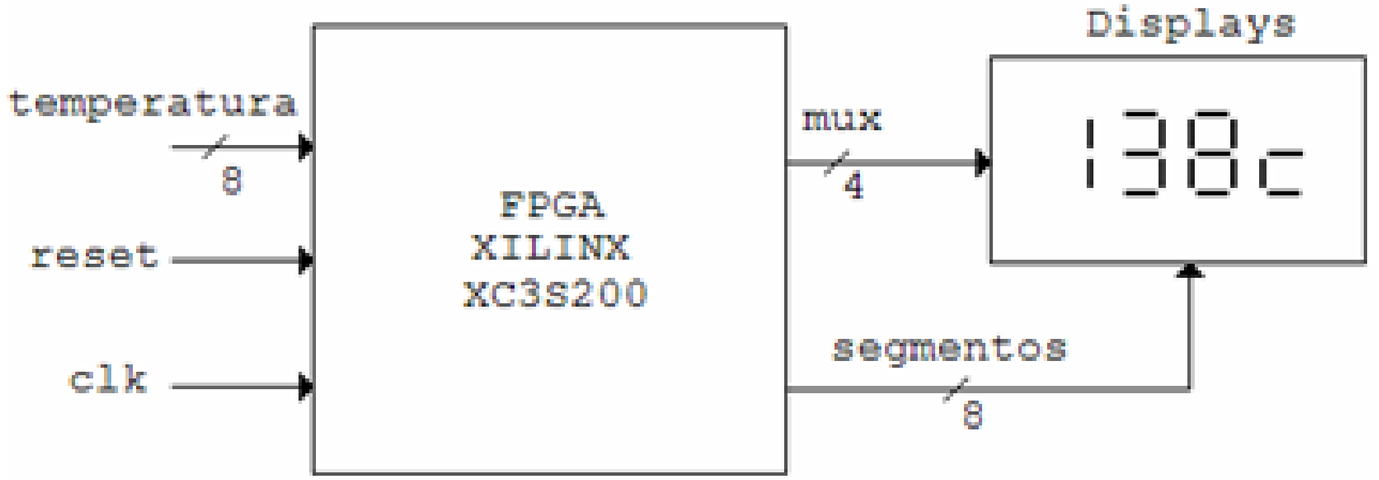
INTRODUCCIÓN

En este proyecto, se presenta el desarrollo de un termómetro digital implementado utilizando una FPGA. El trabajo se centra en el diseño y la programación de la parte digital del sistema utilizando el lenguaje VHDL, una herramienta fundamental para describir la estructura y el comportamiento de los circuitos digitales.

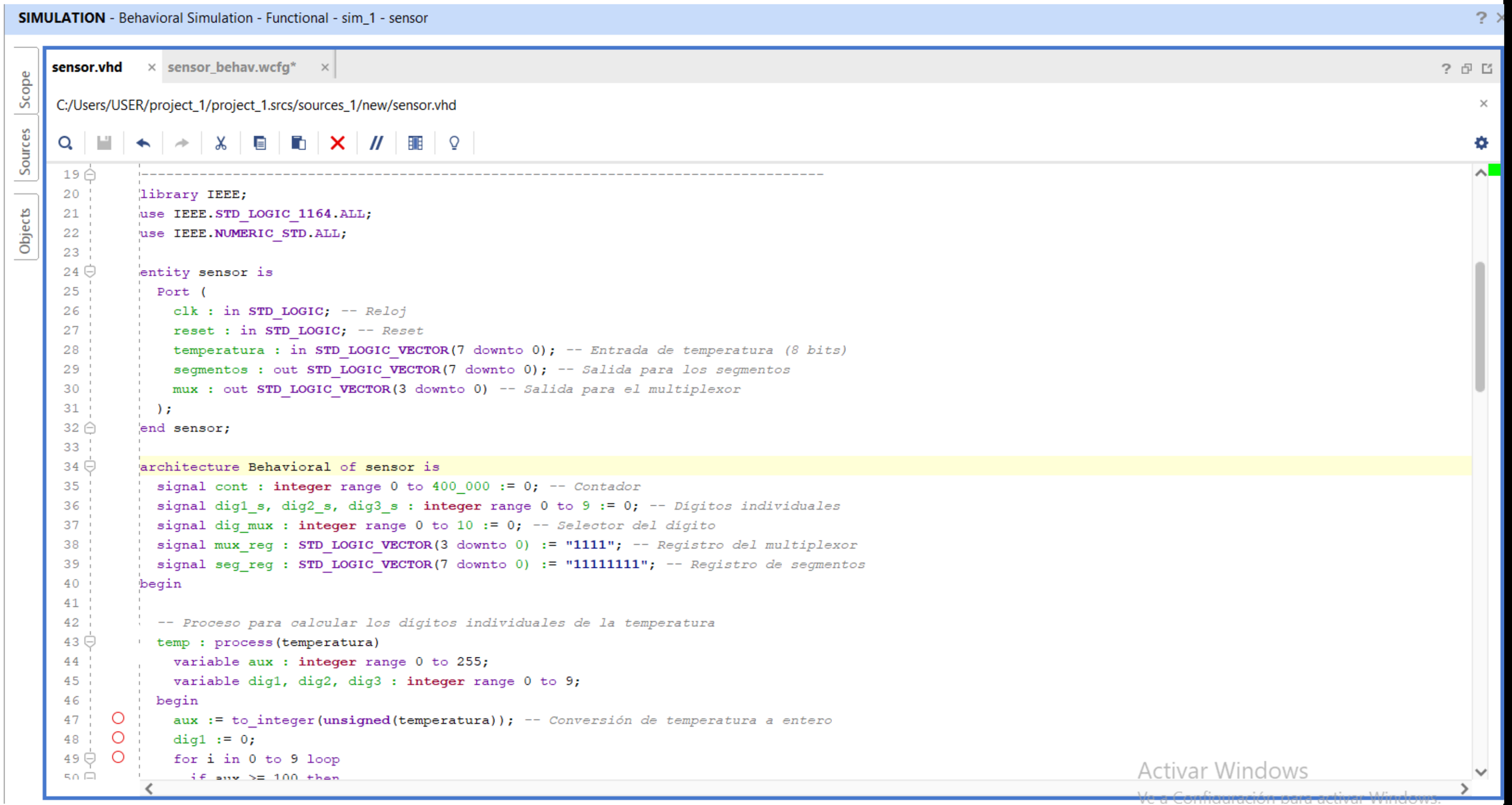
DIAGRAMA PROYECTO:



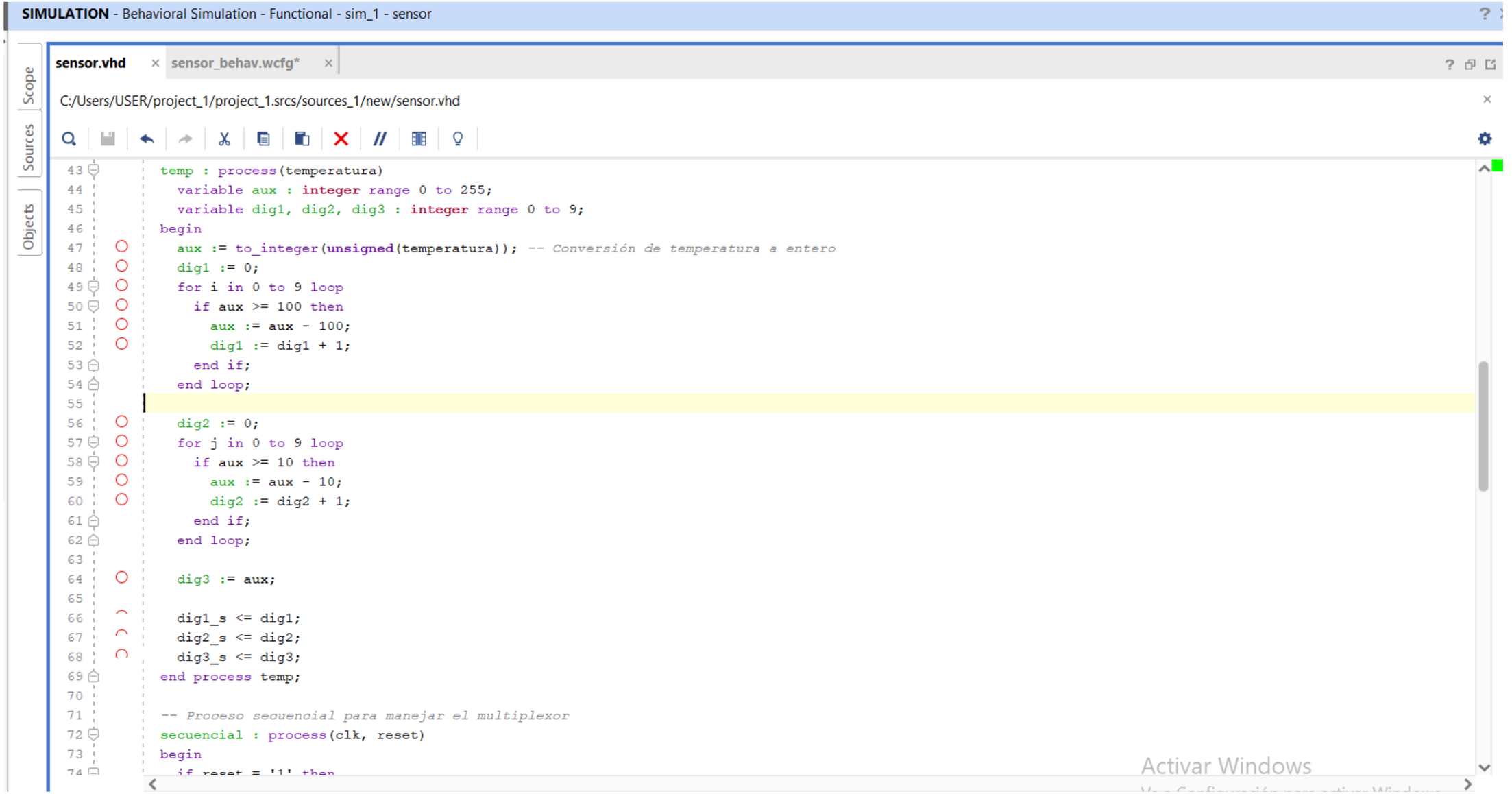




CODIGO VHDL


SIMULATION - Behavioral Simulation - Functional - sim_1 - sensor
Scope
Sources
Objects
sensor.vhd
sensor_behav.wcfg*
C:/Users/USER/project_1/project_1.srcs/sources_1/new/sensor.vhd
19 ----------
20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL;
22 use IEEE.NUMERIC_STD.ALL;
23
24 entity sensor is
25 Port (
26 clk : in STD_LOGIC; -- Reloj
27 reset : in STD_LOGIC; -- Reset
28 temperatura : in STD_LOGIC_VECTOR(7 downto 0); -- Entrada de temperatura (8 bits)
29 segmentos : out STD_LOGIC_VECTOR(7 downto 0); -- Salida para los segmentos
30 mux : out STD_LOGIC_VECTOR(3 downto 0) -- Salida para el multiplexor
31);
32 end sensor;
33
34 architecture Behavioral of sensor is
35 signal cont : integer range 0 to 400_000 := 0; -- Contador
36 signal dig1_s, dig2_s, dig3_s : integer range 0 to 9 := 0; -- Digitos individuales
37 signal dig_mux : integer range 0 to 10 := 0; -- Selector del digito
38 signal mux_reg : STD_LOGIC_VECTOR(3 downto 0) := "1111"; -- Registro del multiplexor
39 signal seg_reg : STD_LOGIC_VECTOR(7 downto 0) := "11111111"; -- Registro de segmentos
40 begin
41
42 -- Proceso para calcular los digitos individuales de la temperatura
43 temp : process(temperatura)
44 variable aux : integer range 0 to 255;
45 variable dig1, dig2, dig3 : integer range 0 to 9;
46 begin
47 aux := to_integer(unsigned(temperatura)); -- Conversión de temperatura a entero
48 dig1 := 0;
49 for i in 0 to 9 loop
Activar Windows
Ve a Configuración para activar Windows.

CODIGO VHDL


SIMULATION - Behavioral Simulation - Functional - sim_1 - sensor
Scope
Sources
Objects
sensor.vhd
sensor_behav.wcfg*
C:/Users/USER/project_1/project_1.srcs/sources_1/new/sensor.vhd
43 temp : process(temperatura)
44 variable aux : integer range 0 to 255;
45 variable dig1, dig2, dig3 : integer range 0 to 9;
46 begin
47 aux := to_integer(unsigned(temperatura)); -- Conversión de temperatura a entero
48 dig1 := 0;
49 for i in 0 to 9 loop
50 if aux >= 100 then
51 aux := aux - 100;
52 dig1 := dig1 + 1;
53 end if;
54 end loop;
55
56 dig2 := 0;
57 for j in 0 to 9 loop
58 if aux >= 10 then
59 aux := aux - 10;
60 dig2 := dig2 + 1;
61 end if;
62 end loop;
63
64 dig3 := aux;
65
66 dig1_s <= dig1;
67 dig2_s <= dig2;
68 dig3_s <= dig3;
69 end process temp;
70
71 -- Proceso secuencial para manejar el multiplexor
72 secuencial : process(clk, reset)
73 begin
Activar Windows

CODIGO VHDL

ensor.vhd × sensor_behav.wcfg* ×

:/Users/USER/project_1/project_1.srcs/sources_1/new/sensor.vhd ×

```
70
71      -- Proceso secuencial para manejar el multiplexor
72      secuencial : process(clk, reset)
73      begin
74        if reset = '1' then
75          cont <= 0;
76          mux_reg <= "1111";
77        elsif rising_edge(clk) then
78          if cont >= 400_000 then
79            cont <= 0;
80          else
81            cont <= cont + 1;
82          end if;
83
84          case cont is
85            when 100_000 => mux_reg <= "0111";
86            when 200_000 => mux_reg <= "1011";
87            when 300_000 => mux_reg <= "1101";
88            when 400_000 => mux_reg <= "1110";
89            when others => mux_reg <= mux_reg;
90          end case;
91        end if;
92      end process secuencial;
93
94      -- Proceso para decodificar los digitos a segmentos
95      d  ificador : process(dig_mux)
96      beg
97          e dig_mux is
98          when 0 => seg_reg <= "10000001";
99          when 1 => seg_reg <= "11001111";
00          when 2 => seg_reg <= "10010010";
01          when 3 => seg_reg <= "10000110";
```

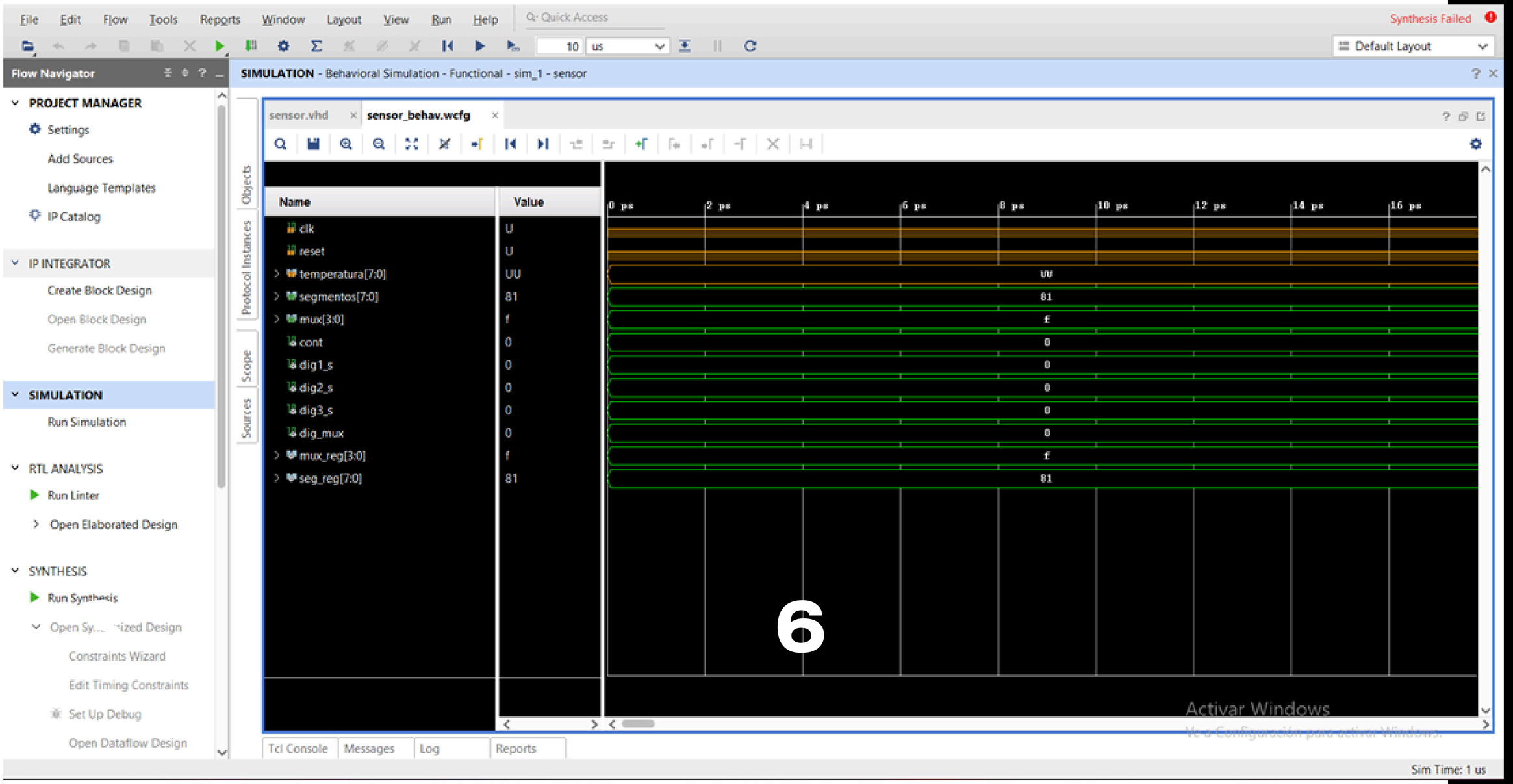
CODIGO VHDL

sensor.vhd | sensor_behav.wcfg*

C:/Users/USER/project_1/project_1.srcs/sources_1/new/sensor.vhd

```
98      when 0 => seg_reg <= "10000001";
99      when 1 => seg_reg <= "11001111";
100     when 2 => seg_reg <= "10010010";
101     when 3 => seg_reg <= "10000110";
102     when 4 => seg_reg <= "11001100";
103     when 5 => seg_reg <= "10100100";
104     when 6 => seg_reg <= "10100000";
105     when 7 => seg_reg <= "10001111";
106     when 8 => seg_reg <= "10000000";
107     when 9 => seg_reg <= "10000100";
108     when 10 => seg_reg <= "11110010"; -- Representación de "-"
109     when others => seg_reg <= "11111111"; -- Vacío
110   end case;
111 end process decodificador;
112
113 -- Proceso para seleccionar el dígito que se muestra
114 selector : process(dig1_s, dig2_s, dig3_s, mux_reg)
115 begin
116   case mux_reg is
117     when "0111" => dig_mux <= dig1_s;
118     when "1011" => dig_mux <= dig2_s;
119     when "1101" => dig_mux <= dig3_s;
120     when "1110" => dig_mux <= 10; -- Separador decimal
121     when others => dig_mux <= 0;
122    nd case;
123 ' process selector;
124
125 -- Asignaciones finales a las salidas
126 mux <= mux_reg;
127 segmentos <= seg_reg;
128
129 end Behavioral;
```

SIMULACION VHDL



SIMULACION VHDL