A

library ieee;

use ieee.std\_logic\_1164.all;

entity Prac7A is port(

clk, clr, e : in std\_logic;

display : inout std\_logic\_vector(6 downto 0)

);

end Prac7A;

architecture aPrac7A of Prac7A is

begin

process(clk, clr)

begin

if (clr = '1') then

display <= "0110000";

elsif(rising\_edge(clk)) then

if(e = '1') then

case display is

when "0110000" =>

display <= "1101101";

when "1101101" =>

display <= "1111001";

when "1111001" =>

display <= "0110011";

when "0110011" =>

display <= "1011011";

when "1011011" =>

display <= "1011111";

when "1011111" =>

display <= "0110000";

when others => display <= "-------";

end case;

else

display <= display;

end if;

end if;

end process;

end architecture;

Graphical user interface, application

Description automatically generatedDiagram, schematic

Description automatically generatedDiagram, schematic

Description automatically generatedDiagram, schematic

Description automatically generatedDiagram, schematic

Description automatically generated

B

library ieee;

use ieee.std\_logic\_1164.all;

entity Prac7B is port(

clk, clr, e : in std\_logic;

display : inout std\_logic\_vector(6 downto 0)

);

end Prac7B;

architecture aPrac7B of Prac7B is

begin

process(clk, clr)

begin

if (clr = '1') then

display <= "1111110"; --0

elsif(rising\_edge(clk)) then

if(e = '1') then

case display is

when "1111110" => --1

display <= "0110000";

when "0110000" => --2

display <= "1101101";

when "1101101" => --3

display <= "1111001";

when "1111001" => --4

display <= "0110011";

when "0110011" => --5

display <= "1011011";

when "1011011" => --6

display <= "1011111";

when "1011111" => --7

display <= "1110000";

when "1110000" => --8

display <= "1111111";

when "1111111" => --9

display <= "1111011";

when "1111011" => --A

display <= "1110111";

when "1110111" => --B

display <= "0011111";

when "0011111" => --C

display <= "1001110";

when "1001110" => --D

display <= "0111101";

when "0111101" => --E

display <= "1001111";

when "1001111" => --F

display <= "1000111";

when "1000111" => --0

display <= "1111110";

when others => display <= "-------";

end case;

else

display <= display;

end if;

end if;

end process;

end architecture;

Graphical user interface, application, Word

Description automatically generatedDiagram

Description automatically generatedDiagram, schematic

Description automatically generatedA close up of a piece of paper

Description automatically generatedCalendar

Description automatically generatedDiagram

Description automatically generated

C

library ieee;

use ieee.std\_logic\_1164.all;

entity Prac7B is port(

clk, clr, en : in std\_logic;

display : out std\_logic\_vector(6 downto 0)

);

end Prac7B;

architecture aPrac7B of Prac7B is

constant l0 : std\_logic\_vector(1 downto 0) := "00";

constant l1 : std\_logic\_vector(1 downto 0) := "01";

constant l2 : std\_logic\_vector(1 downto 0) := "10";

constant i : std\_logic\_vector(6 downto 0) := "0000110";

constant n : std\_logic\_vector(6 downto 0) := "0010101";

constant f : std\_logic\_vector(6 downto 0) := "1000111";

constant e : std\_logic\_vector(6 downto 0) := "1001111";

constant r : std\_logic\_vector(6 downto 0) := "0000111";

constant o : std\_logic\_vector(6 downto 0) := "1111110";

constant s : std\_logic\_vector(6 downto 0) := "1011011";

signal estado : std\_logic\_vector(8 downto 0);

begin

process(clk, clr)

begin

if (clr = '1') then

estado <= l0&i;

elsif(rising\_edge(clk)) then

if(en = '1') then

case estado is

when "000000110" =>

estado <= "000010101";

when "000010101" =>

estado <= "001000111";

when "001000111" =>

estado <= "010000110";

when "010000110" =>

estado <= "001001111";

when "001001111" =>

estado <= "000000111";

when "000000111" =>

estado <= "010010101";

when "010010101" =>

estado <= "001111110";

when "001111110" =>

estado <= "001011011";

when "001011011" =>

estado <= "000000110";

when others => estado <= "---------";

end case;

else

estado <= estado;

end if;

end if;

end process;

display <= estado(6 downto 0); -- para solo tomar los bits que pertenecen a la letra e ignorar la etiqueta de repeticion

end architecture;

Graphical user interface, application

Description automatically generatedDiagram, schematic

Description automatically generatedTable

Description automatically generatedCalendar

Description automatically generatedA close up of text on a whiteboard

Description automatically generated

D

library ieee;

use ieee.std\_logic\_1164.all;

entity Prac7B is port(

clk, clr, en : in std\_logic;

display : out std\_logic\_vector(6 downto 0)

);

end Prac7B;

architecture aPrac7B of Prac7B is

constant l0 : std\_logic\_vector(1 downto 0) := "00";

constant l1 : std\_logic\_vector(1 downto 0) := "01";

constant l2 : std\_logic\_vector(1 downto 0) := "10";

constant n2 : std\_logic\_vector(6 downto 0) := "1101101";

constant n0 : std\_logic\_vector(6 downto 0) := "1111110";

constant n1 : std\_logic\_vector(6 downto 0) := "0110000";

constant n7 : std\_logic\_vector(6 downto 0) := "1110000";

constant n6 : std\_logic\_vector(6 downto 0) := "1011111";

constant n3 : std\_logic\_vector(6 downto 0) := "1111001";

constant n8 : std\_logic\_vector(6 downto 0) := "1111111";

signal estado : std\_logic\_vector(8 downto 0);

begin

process(clk, clr)

begin

if (clr = '1') then

estado <= l0&"1101101";

elsif(rising\_edge(clk)) then

if(en = '1') then

case estado is

when "001101101" =>

estado <= "001111110";

when "001111110" =>

estado <= "000110000";

when "000110000" =>

estado <= "001110000";

when "001110000" =>

estado <= "001011111";

when "001011111" =>

estado <= "001111001";

when "001111001" =>

estado <= "011111110";

when "011111110" =>

estado <= "101111110";

when "101111110" =>

estado <= "001111111";

when "001111111" =>

estado <= "010110000";

when "010110000" =>

estado <= "001101101";

when others => estado <= "---------";

end case;

else

estado <= estado;

end if;

end if;

end process;

display <= estado(6 downto 0); -- para solo tomar los bits que pertenecen a la letra e ignorar la etiqueta de repeticion

end architecture;

Graphical user interface, application

Description automatically generatedDiagram, schematic

Description automatically generatedCalendar

Description automatically generatedCalendar

Description automatically generatedDiagram

Description automatically generated

CUESTIONARIO

1. ¿Cuántos dispositivos PLD 22V10 son necesarios para el desarrollo de

esta práctica?

**1 para cada contador, 4 en total**

2. ¿Cuántos dispositivos de la serie 74xx (TTL) ó 40xx (CMOS) hubieras

necesitado para el desarrollo de esta práctica?

**Mas de 100**

3. ¿Cuántos pines de entrada/salida del PLD 22V10 se usan en los

diseños?

**43 entre los 4 autómatas, la mayoría utiliza las macroceldas de input/output**

4. ¿Cuántos términos producto ocupan las ecuaciones para cada señal de

salida y que porcentaje se usa en total del PLD 22V10 en cada

aplicación?

**A – 17%**

**B – 28%**

**C - 18%**

**D – 29%**

5. ¿Es posible implementar los diseños usando cualquier tipo de

codificación en el PLD22V10?

**Sí, hay muchas maneras de codificar estos autómatas, pero se deben de tener en cuenta las limitaciones físicas de al GAL**

6. ¿Cuáles son las señales que funcionan de manera síncrona y cuales de

manera asíncrona?

**La señal asíncrona es el “CLR”, todas las demás son asíncronas**

7. ¿Qué puedes concluir de esta práctica?

**Mediante el uso de contadores se pueden crear autómatas que modelan una cantidad increíble de cosas, esta práctica fue pesada en el sentido de que el análisis a mano puede complicarse exponencialmente cuando se aumenta el número de bits necesarios para representar algo mediante un autómata.**