**PRIMERA SESIÓN REMOTA DE LÓGICA PROGRAMABLE**

**17 DE MARZO DEL 2020**

**DESCRIPCIÓN EN VHDL DEL CONTROL DE UN SERVO PARA SITUARLO A 90°**

Proyectos a completar

* Diseñar el servo con tres estados (si quieres). Incluso hacer uno de tres estados distinto al del autómata.
* Usar 4 interruptores como selector de posición final.
* Usar un contador de dedo (ya visto) para que el usuario elija entre 10º y 170º. Usar botones de arriba y abajo, btnU y btnD.
* Hacer que el servo se mueva solo de 10º a 170 y de 170º a 10º al ritmo de 1 segundo. Como un limpiaparabrisas ¿cierto?
* Mediante interruptores (tenemos 16) se pueda cambiar el tiempo de cambio de la posición del servo.
* Mostrar la posición del servo en los 7-segmentos.

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.numeric\_std.all;

use IEEE.std\_logic\_unsigned.all;

entity main is

port (

clk : in std\_logic;

sw : in STD\_LOGIC\_VECTOR(15 DOWNTO 0); --interruptores

btnU : in STD\_LOGIC; --boton arriba

btnD : in STD\_LOGIC; --boton abajo

btnL : in STD\_LOGIC; --boton izquierda

btnR : in STD\_LOGIC; --boton derecha

btnC : in STD\_LOGIC; --boton central

led : out STD\_LOGIC\_VECTOR(15 DOWNTO 0); --leds

seg : out STD\_LOGIC\_VECTOR(6 DOWNTO 0); --siete segmentos

dp : out STD\_LOGIC; --punto decimal del seite segmentos

an : out STD\_LOGIC\_VECTOR(3 DOWNTO 0); -- control de 7-seg

servo : out std\_logic;

stepper : out std\_logic\_vector (3 downto 0)

);

end main;

architecture Behavioral of main is

--aqui comienzan las signals

signal estado: std\_logic\_vector (1 downto 0);

signal cont\_flancos: integer range 0 to 2000000;

signal inicio: std\_logic;

signal selector: std\_logic\_vector (3 downto 0);

signal grados: integer range 0 to 180;

signal tope\_flancos: integer range 0 to 200000;

begin

inicio<=btnC;

tope\_flancos<=grados\*1111;

selector<=sw(3 downto 0);

process(selector)

begin

case selector is

when "0000" => grados<=10;

when "0001" => grados<=20;

when "0010" => grados<=30;

when "0011" => grados<=40;

when "0100" => grados<=50;

when "0101" => grados<=60;

when "0110" => grados<=70;

when "0111" => grados<=80;

when "1000" => grados<=90;

when "1001" => grados<=100;

when "1010" => grados<=110;

when "1011" => grados<=120;

when "1100" => grados<=130;

when "1101" => grados<=140;

when "1110" => grados<=150;

when "1111" => grados<=170;

when others => grados<=10;

end case;

end process;

process(clk, inicio)

begin

if inicio='1' then

estado<="00";

cont\_flancos<=0;

elsif rising\_edge(clk) then

case estado is

when "00" => cont\_flancos<=0;

estado<="01";

when "01" => cont\_flancos<=1;

estado<="10";

when "10" => cont\_flancos<=cont\_flancos+1;

if cont\_flancos=tope\_flancos then

estado<="11";

else

estado<="10";

end if;

when "11" => cont\_flancos<=cont\_flancos+1;

if cont\_flancos=2000000 then

estado<="01";

else

estado<="11";

end if;

when others => cont\_flancos<=0;

estado<="00";

end case;

end if;

end process;

process(estado)

begin

case estado is

when "00" => servo<='0';

when "01" => servo<='1';

when "10" => servo<='1';

when "11" => servo<='0';

when others => servo<='0';

end case;

end process;

end Behavioral;