CODIGO

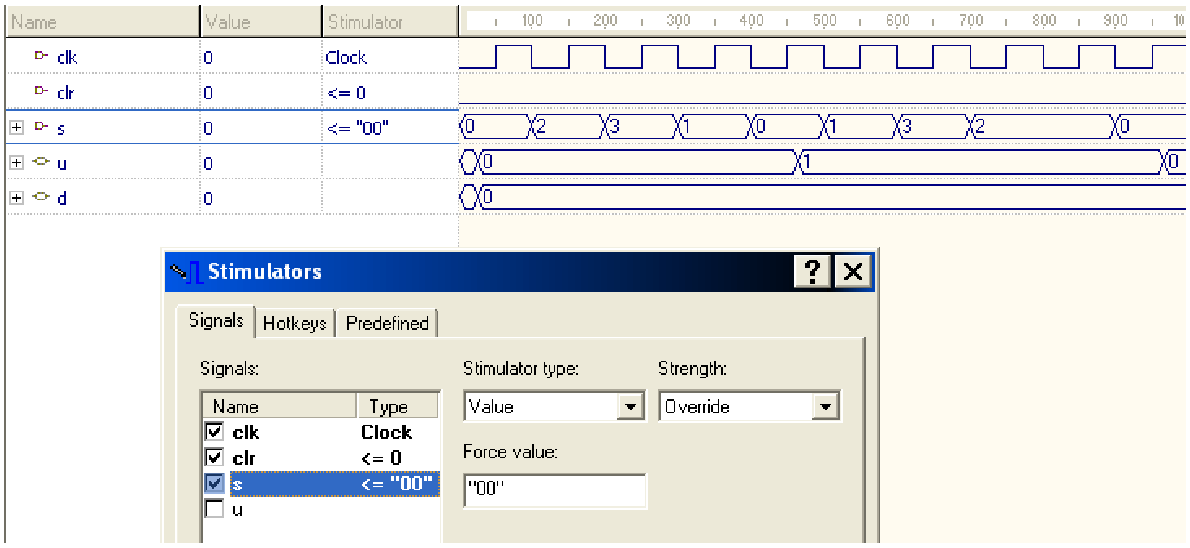
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity pract9 is port (
s : in std_logic_vector (1 downto 0);
clk, clr:in std_logic;
u : out std_logic_vector(3 downto 0);
d : out std_logic_vector(2 downto 0)
);
end pract9;

architecture apract9 of pract9 is
type estado is (q0, q1, q2, q3, q4, q5, q6, q7);
signal eAnt, eSig : estado;
signal salida : std_logic_vector(1 downto 0);
begin
    process(clk,clr) begin
        if(clr = '1') then
            eAnt <= q0;
        elsif rising_edge(clk) then
            eAnt <= eSig;
        end if;
    end process;
    process (eAnt,s) begin
        case eAnt is
            when q0 =>
                if (s = "00") then
                    salida <= "00";
                    eSig <= q0;
                elsif (s = "01") then
                    salida <= "00";
                    eSig <= q4;
                elsif (s = "10") then
                    salida <= "00";
                    eSig <= q1;
                else
                    salida <= "00";
                    eSig <= q7;
                end if;
            when q1 =>
                if (s = "00") then
                    salida <= "00";
                    eSig <= q0;
                elsif (s = "01") then
                    salida <= "00";
                    eSig <= q4;
                elsif (s = "10") then
                    salida <= "00";
                    eSig <= q1;
                else
                    salida <= "00";
                    eSig <= q2;
                end if;
            when q2 =>
                if (s = "00") then
                    salida <= "00";
                    eSig <= q0;
                elsif (s = "01") then
                    salida <= "00";
                    eSig <= q3;
                elsif (s = "10") then
                    salida <= "00";
                    eSig <= q1;
                else
                    salida <= "00";
                    eSig <= q2;
                end if;
            when q3 =>
                if (s = "00") then
                    salida <= "01";
                    eSig <= q0;
                elsif (s = "01") then
                    salida <= "00";
                    eSig <= q3;
                elsif (s = "10") then
                    salida <= "01";
                    eSig <= q1;
                else
                    salida <= "00";
                    eSig <= q2;
                end if;
            when q4 =>
                if (s = "00") then
                    salida <= "00";
                    eSig <= q0;
                elsif (s = "01") then
                    salida <= "00";
```

```
                    eSig <= q4;
                elsif (s = "10") then
                    salida <= "00";
                    eSig <= q1;
                else
                    salida <= "00";
                    eSig <= q5;
                end if;
            when q5 =>
                if (s = "00") then
                    salida <= "00";
                    eSig <= q0;
                elsif (s = "01") then
                    salida <= "00";
                    eSig <= q4;
                elsif (s = "10") then
                    salida <= "00";
                    eSig <= q6;
                else
                    salida <= "00";
                    eSig <= q5;
                end if;
            when q6 =>
                if (s = "00") then
                    salida <= "10";
                    eSig <= q0;
                elsif (s = "01") then
                    salida <= "10";
                    eSig <= q4;
                elsif (s = "10") then
                    salida <= "00";
                    eSig <= q6;
                else
                    salida <= "00";
                    eSig <= q5;
                end if;
            when q7 =>
                if (s = "00") then
                    salida <= "00";
                    eSig <= q0;
                elsif (s = "01") then
                    salida <= "00";
                    eSig <= q4;
                elsif (s = "10") then
                    salida <= "00";
                    eSig <= q1;
                else
                    salida <= "00";
                    eSig <= q7;
                end if;
        end case;
    end process;
    process (clk, clr) begin
        if (clr = '1') then
            u <= "0000";
            d <= "000";
        elsif rising_edge(clk) then
            case salida is
            when "00" =>
                u <= u;
                d <= d;
            when "01" =>
                if (u = "1001") then
                    u <= "0000";
                    d <= d + 1;
                else
                    u <= u + 1;
                    end if;
                when "10" =>
                    if (u = "0000") then
                    u <= "1001";
                    d <= d - 1;
                else
                    u <= u - 1;
                end if;
            when others =>
                u <= "----";
                d <= "---";
            end case;
        end if;
    end process;
end architecture;
```

SIMULACION

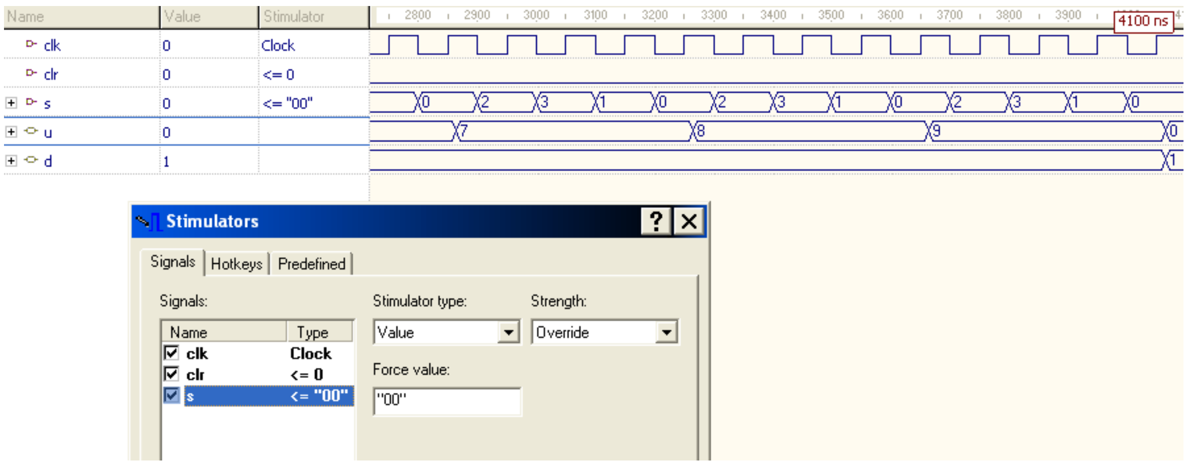


Name	Value	Stimulator
clk	0	Clock
clr	0	<= 0
s	0	<= "00"
u	0	
d	0	

Signals:

Name	Type
☑ clk	Clock
☑ clr	<= 0
☑ s	<= "00"
☐ u	

Stimulator type: Value
Strength: Override
Force value: "00"



Name	Value	Stimulator
clk	0	Clock
clr	0	<= 0
s	0	<= "00"
u	0	
d	1	

Signals:

Name	Type
☑ clk	Clock
☑ clr	<= 0
☑ s	<= "00"

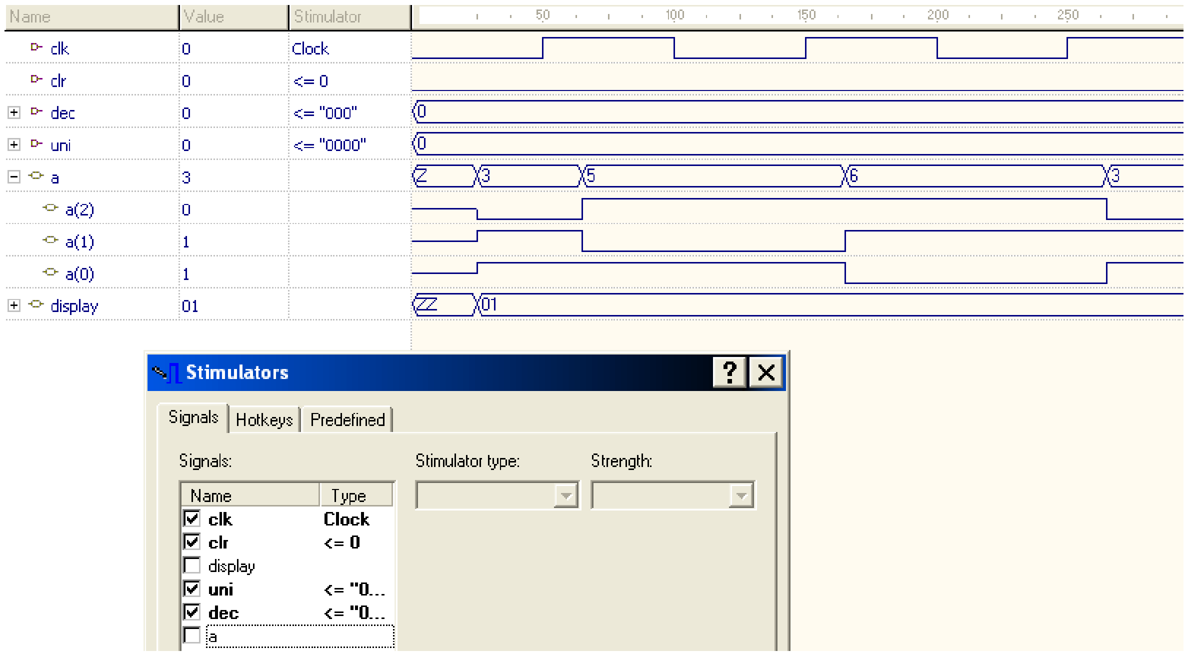
Stimulator type: Value
Strength: Override
Force value: "00"

MULTIPLEXOR

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity multi is port (
dec: in std_logic_vector(2 downto 0);
uni: in std_logic_vector(3 downto 0);
a : out std_logic_vector (2 downto 0);
display: out std_logic_vector(6 downto 0);
clk, clr:in std_logic
);
end multi;
```

```vhdl
architecture amulti of multi is
signal an: std_logic_vector(2 downto 0);
signal entrada: std_logic_vector(3 downto 0);
begin
    process(clk,clr)
    begin
        if(clr = '1') then an <= "011";
        elsif rising_edge(clk) then
            an <= to_stdlogicvector(to_bitvector(an)rol 2);
        end if;
    end process;
    a <= an;
    entrada <= '0'&dec when an = "101" else
    uni when an = "110" else "0000";
      display <= "0000001" when entrada = "0000" else
      "1001111" when entrada = "0001" else
      "0010010" when entrada = "0010" else
      "0000110" when entrada = "0011" else
      "1001100" when entrada = "0100" else
      "0100100" when entrada = "0101" else
      "0100000" when entrada = "0110" else
      "0001111" when entrada = "0111" else
      "0000000" when entrada = "1000" else
      "0001100"; --9 0C
end architecture;
```



Name	Value	Stimulator
clk	0	Clock
clr	0	<= 0
dec	0	<= "000"
uni	0	<= "0000"
a	3	
a(2)	0	
a(1)	1	
a(0)	1	
display	01	

Name	Type
✓ clk	Clock
✓ clr	<= 0
☐ display	
✓ uni	<= "0...
✓ dec	<= "0...
☐ a	

Dominguez Martínez Braulio Sebastián 2CV5

CUESTIONARIO

1. ¿Cuántos dispositivos PLD 22V10 son necesarios para el desarrollo de esta práctica? R= 2

2. ¿Cuántos dispositivos de la serie 74xx (TTL) ó 40xx (CMOS) hubieras necesitado para el desarrollo de esta práctica? R= entre 20 y 30

3. ¿Cuántos pines de entrada/salida del PLD1 22V10 y PLD2 22V10 se usan en el diseño?
R= 19/20 y 14/22

4. ¿Cuántos términos producto ocupan las ecuaciones para cada señal de salida y que porcentaje se usa en total del PLD1 22V10 y PLD2 22V10? R= 19/121 (32%) y 88/121 (72%)

5. ¿Qué puedes concluir de esta práctica?

R= En esta practica se pudo observar otra utilidad para las maquinas finitas, que al completar su recorrido por un autómata definido permiten mantener un control de cantidad que en la realidad es gracias a dos sensores que se puede realizar.