Universidad Nacional Autonoma de Mexico Facultad de ingeniería

Diseño Digital VLSI

Grupo : 4

Práctica: 2

DISEÑO DE REGISTROS DE CORRIMIENTO EN CASCADA

Edgar Daniel Barcenas Martinez

OBJETIVO:
Demostrar a los estudiantes mediante el diseño de registros de corrimiento en cascada, que las declaraciones secuenciales requieren de un orden para ser ejecutadas, utilizando las estructuras de control *if-then-else* o *case* dentro de un proceso.

ESPECIFICACIONES:
Utilizando un FPGA y 8 displays de 7 segmentos, diseñar un sistema digital que despliegue un mensaje que se vea recorrer en los displays.

DIAGRAMA DE BLOQUES:

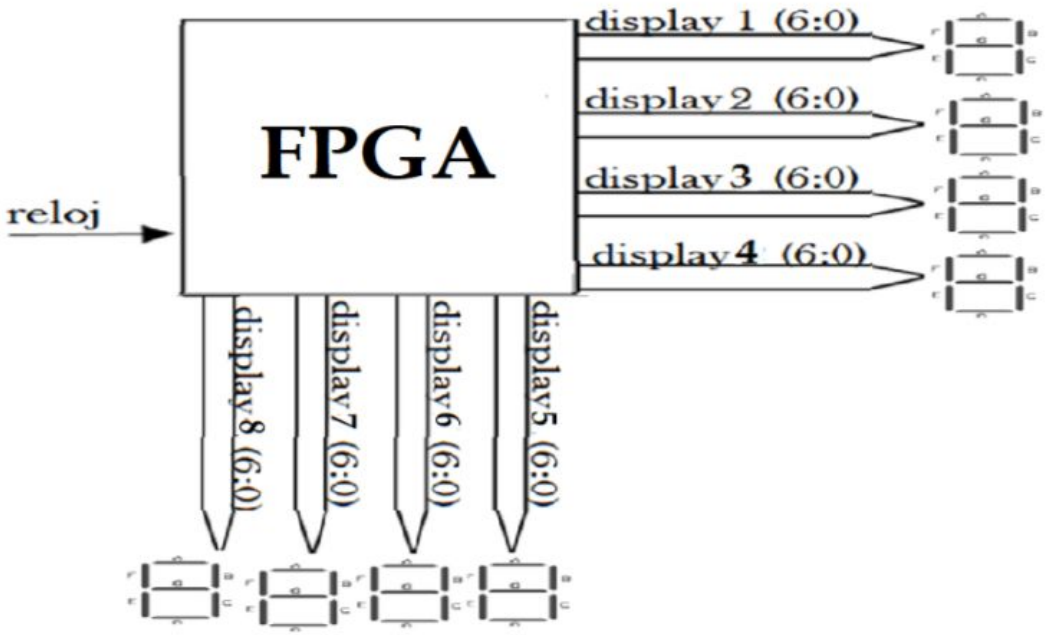

Figura 2.1. Diagrama de bloques del sistema Registros de Corrimiento en Cascada.

Dentro del sistema digital Registros de Corrimiento en Cascada, se tienen varios bloques funcionales, los cuales internamente ejecutan instrucciones en forma secuencial.

BLOQUES FUNCIONALES:

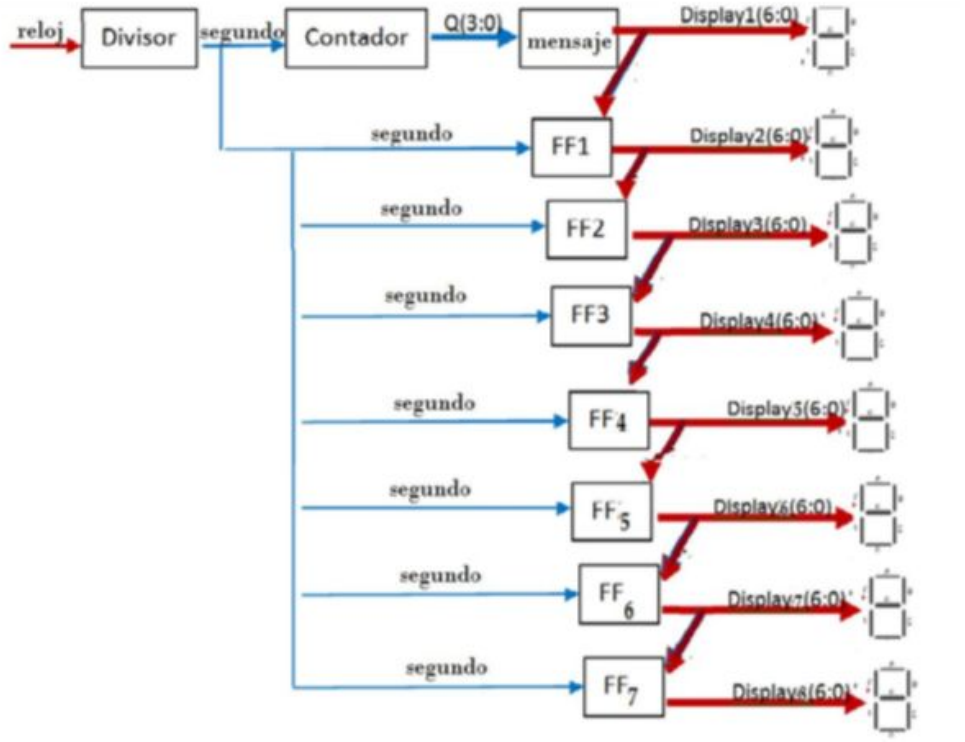

Figura 2.2. Diagrama de bloques funcionales del sistema Registros de Corrimiento en Cascada

Codigo:

Entidad de los registros de corrimiento en cascada.

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

entity corri is
   Port ( reloj : in  std_logic;
          display1, display2, display3, display4, display5, display6,
          display7, display8 : out std_logic_vector (6 downto 0));
end corri;
```

Parte declarativa de la arquitectura del sistema Registros de Corrimiento en Cascada.

```
architecture Behavioral of corri is
   signal segundo : std_logic;
   signal Q : std_logic_vector(3 downto 0):="0000";
```

Parte operatoria de la arquitectura del sistema Registros de Corrimiento en Cascada.

```
begin
   divisor : process (reloj)
      variable CUENTA: std_logic_vector(27 downto 0) :=  X"0000000";
   begin
      if rising_edge (reloj) then
         if CUENTA =X"48009E0" then
            cuenta :=  X"0000000";
         else
            cuenta  := cuenta+1;
         end if;
      end if;
      segundo  <=CUENTA(22);
   end process;

   contador : process (segundo)
   begin
      if rising_edge (segundo) then
         Q  <= Q +1;
      end if;
   end process;
```

```
    with Q select
       display1 <= "0000110" when "0000",  -- E
                   "0101011" when "0001",  -- n
                   "1111111" when "0010",  -- espacio
                   "1000111" when "0011",  -- L
                   "0001000" when "0100",  -- A
                   "1111111" when "0101",  -- espacio
                   "1000000" when "0110",  -- O
                   "1000111" when "0111",  -- L
                   "0001000" when "1000",  -- A
                   "1111111" when others;  -- espacios

    FF1 : process (segundo)
    begin
       if rising_edge (segundo) then
          display2 <= display1;
       end if;
    end process;

    FF2 : process (segundo)
    begin
       if rising_edge (segundo) then
          display3 <= display2;
       end if;
    end process;

    FF3 : process (segundo)
    begin
       if rising_edge (segundo) then
          display4 <= display3;
       end if;
    end process;
```

```
    FF4 : process (segundo)
    begin
       if rising_edge (segundo) then
          display5 <= display4;
       end if;
    end process;

    FF5 : process (segundo)
    begin
       if rising_edge (segundo) then
          display6 <= display5;
       end if;
    end process;

    FF6 : process (segundo)
    begin
       if rising_edge (segundo) then
          display7 <= display6;
       end if;
    end process;
```

```
   FF7 : process (segundo)
   begin
      if rising_edge (segundo) then
         display8 <= display7;
      end if;
   end process;
end Behavioral;
```

Actividad de la Clase:

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

entity p2 is
        Port ( reloj : in std_logic;
                        AN : out  STD_LOGIC_VECTOR (3 DOWNTO 0);
                        L : out  STD_LOGIC_VECTOR (6 DOWNTO 0));
end p2;

architecture Behavioral of p2 is
        signal display1, display2, display3, display4 : std_logic_vector (6 downto 0);
        signal segundo : std_logic;
        signal Q : std_logic_vector(3 downto 0) := "0000";
        SIGNAL rapido : STD_LOGIC;
        SIGNAL Qr : STD_LOGIC_VECTOR(1 DOWNTO 0);
begin
```

```
divisor : process (reloj)
	variable CUENTA: std_logic_vector(27 downto 0) := X"0000000";
begin
	if rising_edge (reloj) then
		if CUENTA =X"48009E0" then
			cuenta := X"0000000";
		else
			cuenta := cuenta+1;
		end if;
	end if;
	segundo <= CUENTA(22);
	rapido <= CUENTA(10);
end process;

contador : process (segundo)
begin
	if rising_edge (segundo) then
		Q <= Q +1;
	end if;
end process;

CONTRAPID: process(rapido)
	variable CUENTA: STD_LOGIC_VECTOR(1 DOWNTO 0) := "00";
begin
	if rising_edge (rapido) then
		CUENTA := CUENTA +1;
	end if;
	Qr <= CUENTA;
end process;

seledisplay: process(Qr)
begin
	case Qr is
		when "00" =>
			AN <= "1110";
		when "01" =>
			AN <= "1101";
		when "10" =>
			AN <= "1011";
		when others =>
			AN <= "0111";
	end case;
end process;
```

```
MUXY: process (Qr)
begin
	if  Qr = "00" then
		L <= display1;
	elsif Qr = "01" then
		L <= display2;
	elsif Qr = "10" then
		L <= display3;
	elsif Qr = "11" then
		L <= display4;
	end if;
end process;

with Q select
	display1 <= "0000110" when "0000", -- E
		    "0101011" when "0001", -- n
		    "1111111" when "0010", -- espacio
		    "1000111" when "0011", -- L
		    "0001000" when "0100", -- A
		"1111111" when "0101", -- espacio
		    "1000000" when "0110", -- O
		    "1000111" when "0111", -- L
		    "0001000" when "1000", -- A
		    "1111111" when others; -- espacios

FF1 : process (segundo)
begin
	if rising_edge (segundo) then
		display2 <= display1;
	end if;
end process;

FF2 : process (segundo)
begin
	if rising_edge (segundo) then
		display3 <= display2;
	end if;
end process;

FF3 : process (segundo)
begin
	if rising_edge (segundo) then
```

```
                display4 <= display3;
            end if;
        end process;

end Behavioral;
```

ACTIVIDAD COMPLEMENTARIA:

Diseñar un sistema que realice la venta de bebidas de 4 diferentes sabores, cada bebida vale $15, se aceptan billetes de $100, $50, $20 y monedas de $1, $2 $5 y $10 y da cambio. Cuando este prendido el sistema y nadie esté comprando se activará una grabación invitando a beber esas ricas bebidas.

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;
entity corri is
Port ( boton1, boton2, switch : in std_logic;
                reloj: in std_logic;
                display1, display2, display3, display4, display5, display6 : buffer std_logic_vector
(6 downto 0));
end corri;
architecture Behavioral of corri is
        signal segundo : std_logic;
        signal Q : std_logic_vector(6 downto 0):="0000000";
        signal s: std_logic_vector(2 downto 0):="000";
        signal cuenta: integer range 0 to 3500000;
        signal unHz, dcHz: STD_LOGIC;
begin
--Es un divisor que va a sincronizar cada uno de los push-button.
        divisorunHz: process(reloj,cuenta,unHz)
        begin
                if rising_edge(reloj) then
                        if(cuenta=3499999) then
                                cuenta<=0;
                                unHz<=not (unHz);
                        else
                                cuenta<=cuenta+1;
                        end if;
                end if;
        end process;
```

```
	divisor : process (reloj)
		variable CUENTA: std_logic_vector(27 downto 0) := X"0000000";
	begin
		if rising_edge (reloj) then
			if CUENTA =X"48009E0" then
				cuenta := X"0000000";
			else
				cuenta := cuenta+1;
			end if;
		end if;
	segundo <=CUENTA(22);
	end process;
--Proceso que va a simular una máquina expendedora de monedas de 5 y 10
	maquina : process (boton1)
		variable CONTADOR: std_logic_vector(7 downto 0) := X"00";
	begin
		Q<=s&Q(3 downto 0);
		if rising_edge(unHz) then
			if boton1='0' then
				if CONTADOR =X"19" or CONTADOR =X"1E" then
					Q<=s&Q(3 downto 0);
					s<=s+"000";
				else
					s<=s+"001";
					CONTADOR:=CONTADOR+X"5";
				end if;
			elsif boton2='0' then
				if CONTADOR =X"19" or CONTADOR =X"1E" or CONTADOR
=X"23" then
					Q<=s&Q(3 downto 0);
					s<=s+"000";
				else
					s<=s+"010";
					CONTADOR:=CONTADOR+X"A";
				end if;
			end if;
			if CONTADOR =X"19" and switch ='1' then
				s<="111";
				CONTADOR :=X"00";
			elsif CONTADOR =X"1E" and switch ='1' then
				s<="001";
				CONTADOR :=X"05";
```

```
                elsif CONTADOR =X"00" and s="111" and switch ='0' then
                        s<="000";
                end if;
        end if;
end process;

contador : process (segundo)
begin
        if rising_edge (segundo) then
                Q(3 downto 0) <= Q(3 downto 0) +1;
        end if;
end process;

with Q select
        display1 <= "0000110" when "0000000", -- E
                                    "0101011" when "0000001", -- n
                                    "1111111" when "0000010", -- espacio
                                    "0000110" when "0000011", -- E
                                    "0010010" when "0000100", -- S
                                    "0001100" when "0000101", -- P
                                    "0000110" when "0000110", -- E
                                    "0001111" when "0000111", -- R
                                    "0001000" when "0001000", -- A

                                    "0010010" when "0010000", -- 5
                                    "1000110" when "0010001", -- C
                                    "0001000" when "0010010", -- A
                                    "0010010" when "0010011", -- S
                                    "0001001" when "0010100", -- H

                                    "1111001" when "0100000", -- 10
                                    "1000000" when "0100001", -- 10
                                    "1000110" when "0100010", -- C
                                    "0001000" when "0100011", -- A
                                    "0010010" when "0100100", -- S
                                    "0001001" when "0100101", -- H


                                    "1111001" when "0110000", -- 15
                                    "0010010" when "0110001", -- 15
                                    "1000110" when "0110010", ---C
                                    "0001000" when "0110011", ---A
                                    "0010010" when "0110100", ---S
```

```
                                        "0001001" when "0110101", ---H


                                        "0100100" when "1000000", -- 20
                                        "1000000" when "1000001", -- 20
                                        "1000110" when "1000010", -- C
                                        "0001000" when "1000011", -- A
                                        "0010010" when "1000100", -- S
                                        "0001001" when "1000101", -- H

                                        "0100100" when "1010000", -- 25
                                        "0010010" when "1010001", -- 25
                                        "1000110" when "1010010", -- C
                                        "0001000" when "1010011", -- A
                                        "0010010" when "1010100", -- S
                                        "0001001" when "1010101", -- H

                                        "0110000" when "1100000", -- 30
                                        "1000000" when "1100001", -- 30
                                        "1000110" when "1100010", -- C
                                        "0001000" when "1100011", -- A
                                        "0010010" when "1100100", -- S
                                        "0001001" when "1100101", -- H

                                        "1000110" when "1110000", -- C
                                        "0001001" when "1110001", -- H
                                        "0001000" when "1110010", -- A
                                        "1000000" when "1110011", -- O
                                        "1111111" when others; -- espacios
FF1 : process (segundo)
begin
        if rising_edge (segundo) then
                display2 <= display1;
        end if;
end process;
FF2 : process (segundo)
begin
        if rising_edge (segundo) then
                display3 <= display2;
        end if;
end process;
FF3 : process (segundo)
begin
```

```
            if rising_edge (segundo) then
                display4 <= display3;
            end if;
        end process;
        FF4 : process (segundo)
        begin
            if rising_edge (segundo) then
                display5 <= display4;
            end if;
        end process;
        FF5 : process (segundo)
        begin
            if rising_edge (segundo) then
                display6 <= display5;
            end if;
        end process;
end Behavioral;
```

Conclusiones:

En esta practica se cumplio el objetivo el cual es diseñar y emplear registros de corrimiento en cascada para esto empleamos el lenguaje de descripción de hardware VHDL en el cual nos dimos cuenta que las declaraciones secuenciales requieren un orden para ser ejecutadas y de esta manera lograr el corrimiento en cascada, para lograr esto utilizamos estructuras de control if-then-else y case dentro de un proceso.

Respecto al FPGA utilizamos 8 displays de 7 segmentos en el cual desplegamos un mensaje en pantalla.Dentro del sistema digital de los registros de corrimiento en cascada se tienen bloques funcionales y estos internamente ejecutan instrucciones en forma secuencial.