Universidad Nacional Autónoma de México
Facultad de Ingeniería

Laboratorio de Diseño de Sistemas Digitales

Reporte Práctica 9:
Diseño y construcción de un registro de corrimiento

Alumno:
Barranco Serrano Miguel Angel

Profesora:
M.I. Norma Elva Chávez

Fecha de entrega:
17/10/13

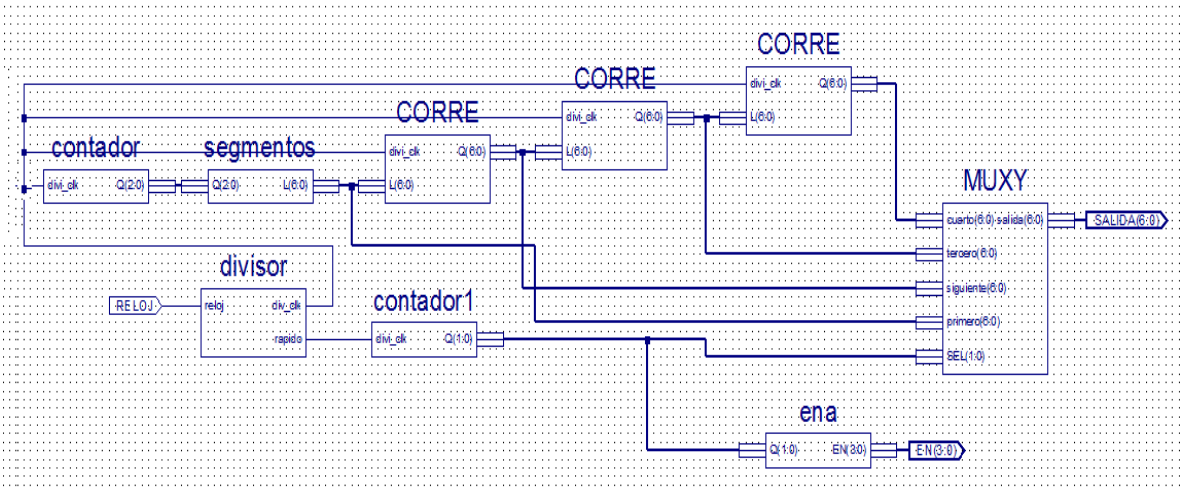
Introducción

Un registro de corrimiento paralelo-paralelo, es un conjunto de flip-flops conectados de tal forma que el vector de números binarios almacenados en ellos, son desplazados de un grupo de flip-flops al siguiente grupo de flip-flops con cada pulso de reloj aplicado.

Objetivo

El alumno comprenderá el funcionamiento y manejo de los registros de corrimiento utilizando lenguaje VHDL.
Además de aprender el manejo de 4 displays de 7 segmentos conectados en paralelo mediante hardware y mediante software hacer que cada uno muestre distinta información al mismo tiempo.

Diagrama de bloques



Procedimiento:

Para procedimiento y diseño de esta práctica se va a hacer uso de la plataforma ISE de Xilinx y la tarjeta SPARTAN 3.Creando una carpeta con cualquier nombre que en esta ocasión se le pondrá “Registro” en esta carpeta es en donde se va a realizar toda esta practica

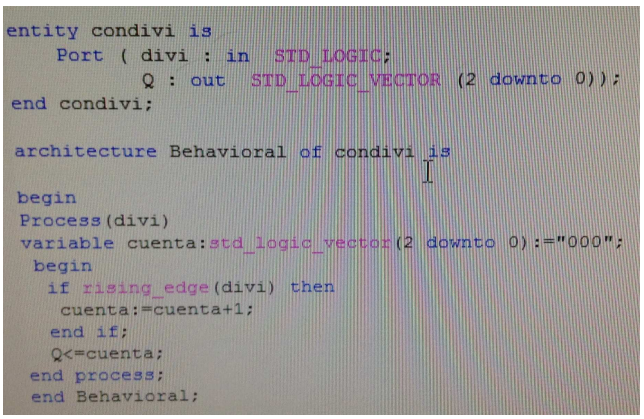
Después configuramos cambiando a estas opciones las siguientes propiedades:

Family: Spartan 3
Device: XC3S200
Package: FT256

Y marcamos con una Paloma todas las demás opciones (EnableMessageFiltering, etc…)
Al estar configurado empezamos con la creación del código

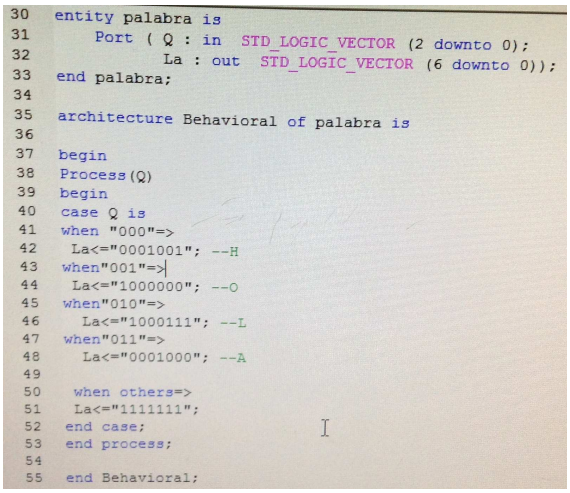
Código

Se crea “condivi” el cual basándonos en el diagrama tiene “divi” de entrada “Q(2:0)” de salida



```
entity condivi is
    Port ( divi : in  STD_LOGIC;
           Q : out  STD_LOGIC_VECTOR (2 downto 0));
end condivi;

architecture Behavioral of condivi is

begin
Process(divi)
variable cuenta:std_logic_vector(2 downto 0):="000";
 begin
  if rising_edge(divi) then
   cuenta:=cuenta+1;
  end if;
  Q<=cuenta;
 end process;
 end Behavioral;
```

Se crea “palabra” el cual basándonos en el diagrama tiene “Q(2:0)” de entrada y “La” de salida



```
30  entity palabra is
31      Port ( Q : in  STD_LOGIC_VECTOR (2 downto 0);
32             La : out  STD_LOGIC_VECTOR (6 downto 0));
33  end palabra;
34
35  architecture Behavioral of palabra is
36
37  begin
38  Process(Q)
39  begin
40  case Q is
41  when "000"=>
42   La<="0001001"; --H
43  when"001"=>
44   La<="1000000"; --O
45  when"010"=>
46    La<="1000111"; --L
47  when"011"=>
48    La<="0001000"; --A
49
50   when others=>
51   La<="1111111";
52  end case;
53  end process;
54
55  end Behavioral;
```

d

Se crea “corre” con entradas “La(6:0)” y “divi”, salidas Lb(6:0)

```
entity corre is
    Port ( La : in  STD_LOGIC_VECTOR (6 downto 0);
           Lb : out  STD_LOGIC_VECTOR (6 downto 0);
           divi : in  STD_LOGIC);
end corre;

architecture Behavioral of corre is

begin
Process(divi)
 begin
  if rising_edge(divi)then
  Lb<=La;
  end if;
end process;

end Behavioral;
```

Se crea Muxy con entradas “La(6:0)”, Lb(6:0) Lc(6:0) Lx(6:0)Sel(1:0) y salida L(6:0)

```
entity muxy is
    Port ( La : in  STD_LOGIC_VECTOR (6 downto 0);
           Lb : in  STD_LOGIC_VECTOR (6 downto 0);
           Lc : in  STD_LOGIC_VECTOR (6 downto 0);
           Lx : in  STD_LOGIC_VECTOR (6 downto 0);
           Sel : in  STD_LOGIC_VECTOR (1 downto 0);
           L : out  STD_LOGIC_VECTOR (6 downto 0));
end muxy;

architecture Behavioral of muxy is

begin
Process(Sel)
 begin
  case Sel is
  when"00"=>
   L<=La;
  when"01"=>
   L<=Lb;
  when"10"=>
   L<=Lc;
  when Others=>
   L<=Lx;
  end case;
  end process;

 end Behavioral;
```

Se crea “conrap” con entradas “Rapido” y salida Sel(1:0)

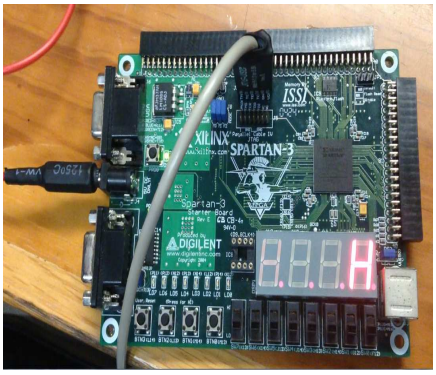
```
entity conrap is
    Port ( Rapido : in  STD_LOGIC;
           Sel : out  STD_LOGIC_VECTOR (1 downto 0));
          -- Sel : inout  STD_LOGIC_VECTOR (1 downto 0));
end conrap;

architecture Behavioral of conrap is

begin
Process(rapido)
variable cuenta:STD_LOGIC_VECTOR (1 downto 0):="00";
 begin
  if rising_edge(rapido) then
   cuenta:=cuenta+1;
  end if;
  sel<=cuenta
 end process;

end Behavioral;
```

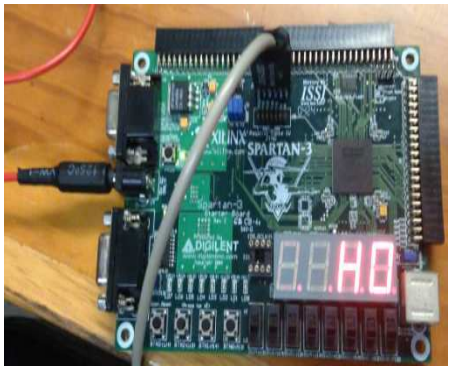
Se crea “ena” con estrada Sel(1:0) y salida En(3:0)

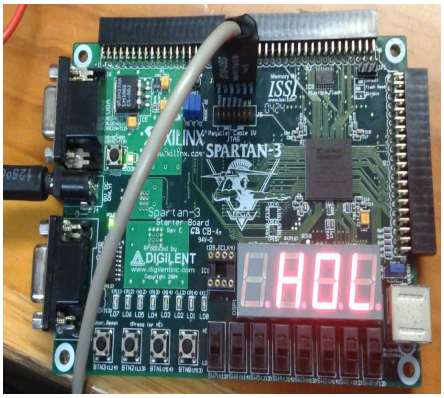
```
entity ena is
    Port ( Sel : in  STD_LOGIC_VECTOR (1 downto 0);
           En : out  STD_LOGIC_VECTOR (3 downto 0));
end ena;

architecture Behavioral of ena is

begin
  Process(Sel)
   begin
    case Sel is
     when "00"=>
     En<="1110";
     when "01"=>
     En<="1101";
     when "10"=>
     En<="1011";
     when others=>
     En<="0111";
    end case;
  end process;
 end Behavioral;
```

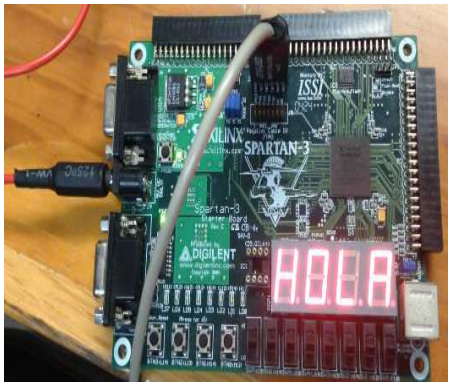
Resultado

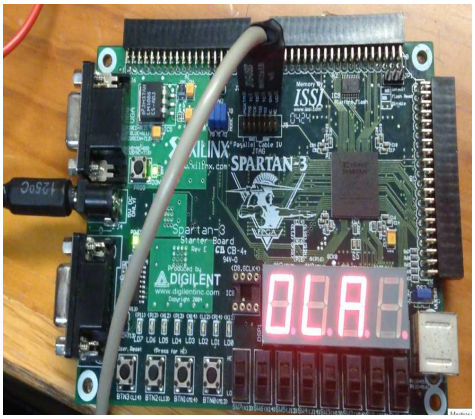
Al realizar este procedimiento crear su símbolo esquemático armar el diagrama esto es lo que nos muestra

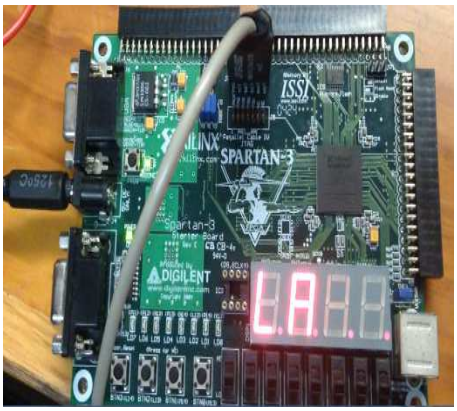


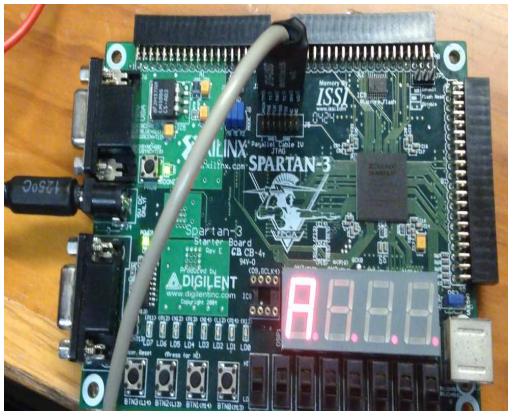












Conclusión:

La práctica fue muy interesante, ya que aprendimos que es un flip-flop para que se utiliza y como se desplazaba con cada pulso de reloj, a su vez que es un registro de corrimiento, mostrándonos de forma simple con un ejemplo muy entretenido ya que pudimos apreciar en cuatro display como aparecía en el SPARTAN 3, y de forma secuencial la palabra “HOLA” sin la necesidad de alambrar solo usando el lenguaje VHDL y teniendo los recursos adecuados.