UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
FACULTAD DE INGENIERIA ELECTRONICA
ESCUELA DE ELECTRONICA

LABORATORIO Nº2: "Descripciòn VHDL utilizando el Estilo Algoritmico"

1.-Implemente y simule el circuito latch:

```
entity latch is
   Port ( enable : in bit;
          data : in bit;
          q : out bit);
end latch;
architecture pld of latch is
begin
latch: PROCESS(enable,data)
                BEGIN
                        IF(enable='1')THEN
                                q <=data;--/ after 15 ns;
                        END IF;
                END PROCESS latch;
end pld;
```

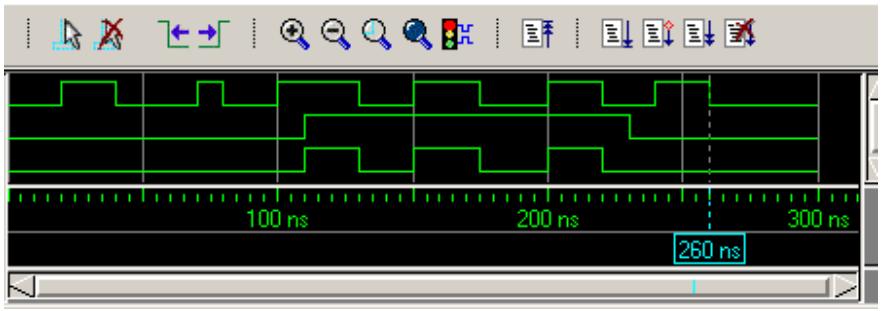
1.1 Dibuje la salida Q del circuito cuando:

Para lograr la simulacion le agregamos: a test_latch.vhd.....

```
-- *** Test Bench - User Defined Section ***
        data <= '0','1'after 20 ns,'0' after 40 ns,'1' after 70 ns,'0' after 80 ns,'1'
after 100 ns,
                '0' after 130 ns,'1' after 150 ns,'0' after 175 ns,'1'after 200 ns,'0'
        after 220 ns,
                '1' after 240 ns,'0' after 260 ns;
                tb : PROCESS
  BEGIN
                wait for 110 ns;
                enable <= '1';
                wait for 120 ns;
                enable <= '0';
    wait; -- will wait forever
  END PROCESS;
```

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

*** End Test Bench - User Defined Section ***



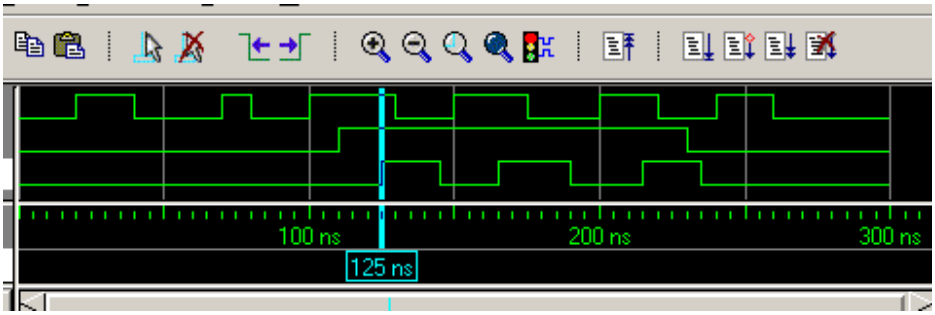
Como resultado obtuvimos: (En el orden data, enable, q)

1.2 Dibuje la salida del circuito cuando latch tiene un retardo de 15ns

Para obtener el retardo variamos el codigo en:

```
architecture pld of latch is
begin
latch: PROCESS(enable,data)
            BEGIN
                  IF(enable='1')THEN
                        q <=data after 15 ns;--// after 15 ns, le da el retardo que nos piden
                  END IF;
            END PROCESS latch;
    end pld;
```

* en el test_bench, no variamos nada, y el resultado que obtenemos es:**(En el orden data, enable, q)**



y antes la primera respuesta '1' estaba en 110ns y ahora esta en 125ns

2.- Implente y simule el ciruito reg:

```
entity reg is
   Port ( data,clk : in bit;
```

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

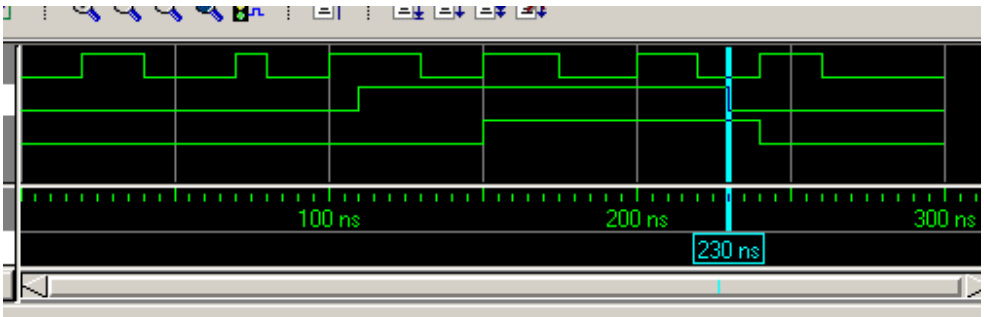
```
        q : out bit);
end reg;
architecture pld of reg is
begin
    PROCESS(clk)
                BEGIN
                    if clk='0' then
                                q<=data after 1 ns;
                        end if;
                END PROCESS;
end pld;
```

2.1 Dibuje la salida cuando:

Para lograr la simulacion hacemos en el test bench:

```
-- *** Test Bench - User Defined Section ***
        clk <= '0','1'after 20 ns,'0' after 40 ns,'1' after 70 ns,'0' after 80 ns,'1'
after 100 ns,
                '0' after 130 ns,'1' after 150 ns,'0' after 175 ns,'1'after 200 ns,'0'
        after 220 ns,
                '1' after 240 ns,'0' after 260 ns;
  tb : PROCESS
  BEGIN
                wait for 110 ns;
                data <= '1';
                wait for 120 ns;
                data <= '0';
    wait; -- will wait forever
  END PROCESS;
-- *** End Test Bench - User Defined Section ***
```

el resultado del test_reg en el modelsim:(en el orden CLK, DATA, Q)



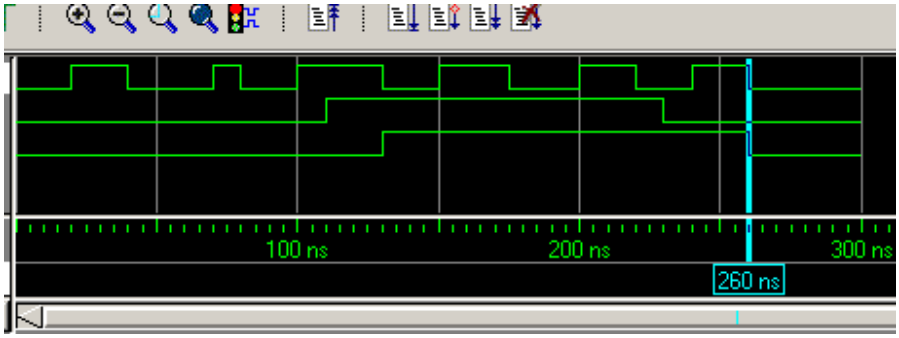
Explique como responde el ciruito:

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

La respuesta del cicuito es:
Cada vez que el **clk** sufre un evento, el es decir pasa a '1' o '0', entra en el proceso, en donde si el **clk** es '1', es decir esta en el flanco de subida, en el estado que este **dato** se queda **q** hasta el proximo evento de **clk**.

¿Que sentencias tendría que cambiar en el programa para que el ciruito responda en el otro flanco?

```
PROCESS(clk)
        BEGIN
           if clk='0' then
                    q<=data;
                end if;
        END PROCESS;
```



Y el resultado en el modelsim es:(El orden es Clk, DATA, Q)

Donde podemos observar que los cambios en los puntos mas criticos: data varia en 2 oportunidades cuando el **clk**, esta en 130ns, y ya no como antes que estaba en 150ns, y el ultimo cambio es 260ns y ya no en 240ns, y estos son respectivamente cuando clk, esta en '0', es decir en el flanco de bajada.

3.-Implementar y simular los siguientes contadores

```
entity conta1 is
   Port ( clk : in bit;
          enable : in bit;
          qa : out integer range 0 to 15);
end conta1;

architecture algoritmo of conta1 is
begin
        process(clk)
```

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

```
        variable cnt:integer range 0 to 15;
        BEGIN
                IF (clk='1') then
                        if enable='1' then
                                        cnt:=cnt+1;
                        end if;
                end if;
                        qa<=cnt;
         end process;
end algoritmo;
```

NOTA: Para la simulacion la señal de reoloj es periòdica y tiene una frecuencia de 1 KHz. La señal se habilita durante 5 ms a partir deltiempo 12 ms.

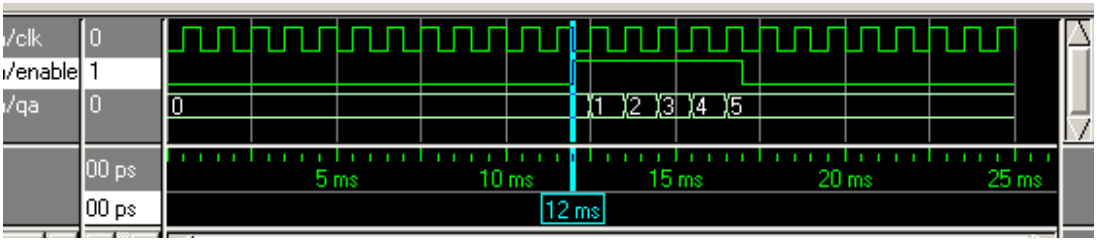
Para la simulacion en el Test Bench
Debo aumentar:

```
-- *** Test Bench - User Defined Section ***

                clk<=not clk after 0.5 ms;
                enable<='0','1' after 12 ms , '0' after 17 ms;

  tb : PROCESS
  BEGIN
    wait; -- will wait forever
  END PROCESS;

-- *** End Test Bench - User Defined Section ***
```



--Contador con Clear sincrono

Entity conta2 is

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

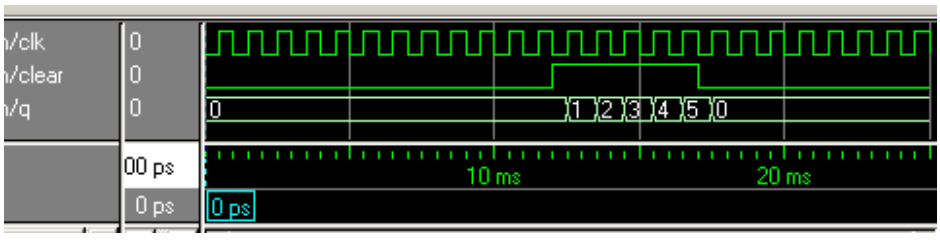
```
        Port(  clk     : in bit;
               Clear   : in bit;
               q       : out integer range 0 to 15);
end conta2;

ARCHITECTURE ALGORIT2 of conta2 IS
BEGIN
        PROCESS(clk)
        VARIABLE cnt        :INTEGER RANGE 0 TO 15;
        BEGIN
               IF clk='1' THEN
                      IF clear ='0' THEN
                             Cnt:=0;
                      ELSE
                             Cnt:=cnt+1;
                      END IF;
               END IF;
               Q <=cnt;
        END PROCESS;
END ALGORIT2
```

Para la simulacion hacemos el cambio:

```
-- *** Test Bench - User Defined Section ***
               clk<=not clk after 0.5 ms;
               clear<='0','1' after 12 ms , '0' after 17 ms;

  tb : PROCESS
  BEGIN
    wait; -- will wait forever
  END PROCESS;
-- *** End Test Bench - User Defined Section ***
```



3.1.- Indique que que diferencias encontró entre las simulaciones de conta1 y conta2

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

una vez que acaba la simulacion notamos que los valores finales en ambos son:

Para conta1 -> 5

Para conta2 -> 0

Es decir que si para conta1 ingresamos un '1' en enable, deja de contar, pero hasta que enable sea '0'; y esto no sucede con cont2, donde cuando ingresamos clear ='1', borra nuestro contador

4. Implementar un contador modulo 16 con control de cuenta up/down.

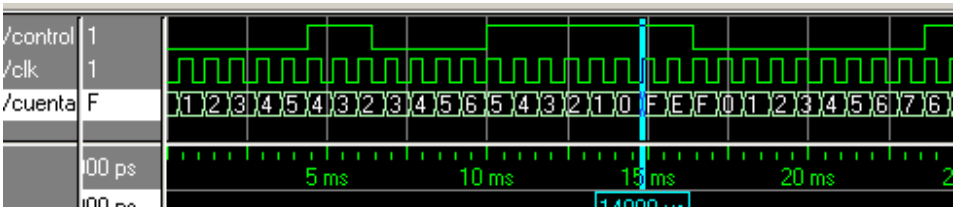
Preg4.vhd

```
entity preg4 is
   Port (CONTROL,CLK:in bit;
                        CUENTA:out integer range 0 to 15);
end preg4;

architecture contador_UP_DOWN of preg4 is
begin
PRocess(clk)
        variable cnt:integer range 0 to 15;
        begin
                if (clk='1') then
                        if control='0' then
                                if cnt=15 then
                                        cnt:=0;
                                else
                                cnt:= cnt +1;
                                end if;
                        else
                                if cnt=0 then
                                        cnt:=15;
                                else
                                cnt:= cnt -1;
                                end if;
                        end if;
                end if;
                        cuenta<=cnt;
        end process;
end contador_UP_DOWN;
```

Para la simulacion usamos el test_wave_form para crear las señales de los datos y el resultado en el MODELSIM fue:

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

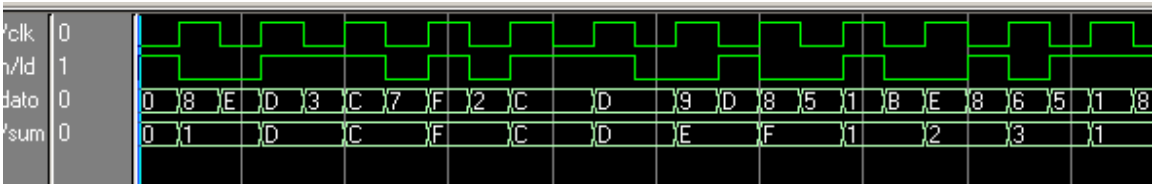


5. Implementar un contador modulo 16 con carga de adatos externa

Preg5.vhd

```
entity preg5 is
   Port ( Clk,LD: in bit;
          DATO:in integer range 0 to 15;
          Sum:out integer range 0 to 15);
end preg5;
architecture carga of preg5 is
begin
process
variable cnt:integer range 0 to 15;
begin
wait until clk='1';
if LD='0' then
        if cnt=15 then
                cnt:=0;
                else
                        cnt:=cnt+1;
        end if;
else
        cnt:=DATO;
                end if;
        sum<=cnt;
        end process;
end carga
```

Para simular usamos el test_wave_form para general las señales de los datos y el resultado en el MODELSIM fue:



6.-Simular el siguiente registro entrada paralela/salida paralela

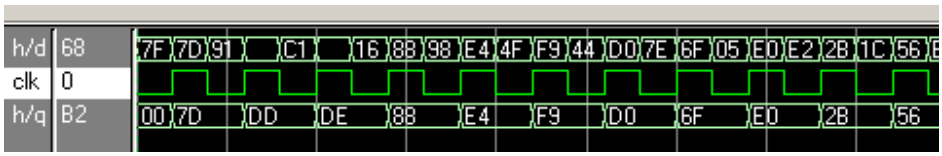
Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

Preg6.vhd

```
entity pipo is
   Port (d      :in bit_vector(7 downto 0);
         clk    :in bit;
         q      :out  bit_vector(7 downto 0)
         );
end pipo;
architecture algoritmo of pipo is
begin
        process
        begin
                wait until clk='1';
                q <=d;
        end process;
end algoritmo;
```

Nota:deben cargarse como minimo 10 datos.

Creamos las señales en el Test_Wave_Form y los resultados en el MODELSIM fue



7.- Implementar el siguiente código VHDL y explique como funcionan las entradas de CLEAR (clr) y PRESET(pre)

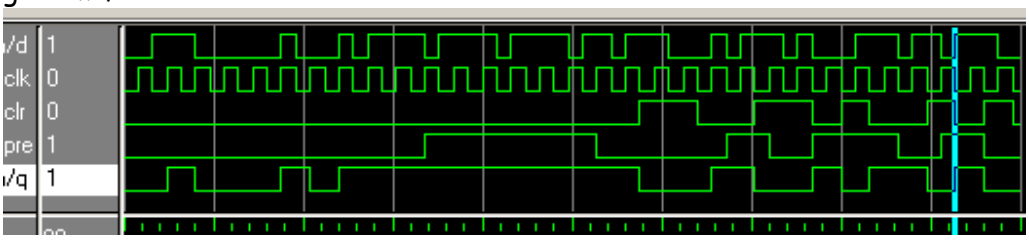
preg7.vhd

```
entity registro is
   Port (d,clk,clr,pre :in bit;
         q             :out bit
         );
end registro;

architecture algoritmo of registro is
begin
        process(clk,clr,pre)
        begin
                if clr='1' then
                        q<='0';
```

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

```
            elsif pre='1' then
                    q<='1';
            elsif clk='1' then
                    q<=d;
        end if;
        end process;
end algoritmo;
```



8.- implementar un decodificador a 7 segmentos para un diplay de ánodo común utilizando la sentencia CASE

preg8.vhd

```
entity decoder is
        port(data :  in bit_vector(3 downto 0);
             sal  :  out bit_vector(6 downto 0));
end decoder;
architecture a7segmt of decoder is
begin
conv: process (data)
begin
case data is
  when "0000" => SAL <= "1111110";--// 0
  when "0001" => SAL <= "1100000";--// 1
  when "0010" => SAL <= "1011011";--// 2
  when "0011" => SAL <= "1110011";--// 3
  when "0100" => SAL <= "1100101";--// 4
  when "0101" => SAL <= "0110111";--// 5
  when "0110" => SAL <= "0111111";--// 6
  when "0111" => SAL <= "1100010";--// 7
  when "1000" => SAL <= "1111111";--// 8
  when "1001" => SAL <= "1110111";--// 9
  when "1010" => SAL <= "1000000";--// A
  when "1011" => SAL <= "0111111";--// B
  when "1100" => SAL <= "1100010";--// C
  when "1101" => SAL <= "1111111";--// D
```

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

```
  when "1110" => SAL <= "1110111";--// E
  when "1111" => SAL <= "1100010";--// F
end case;
end process conv;
end a7segmt;
```

La simulacion de este problema no es posible verse mediante señales, pero

podemos comprobar que su sintaxis es la correcta

9.-Escriba el codigo VHDL para implementar un contador:0,1,2,3,4,0,1,2,3,4,0,1,...

preg9.vhd

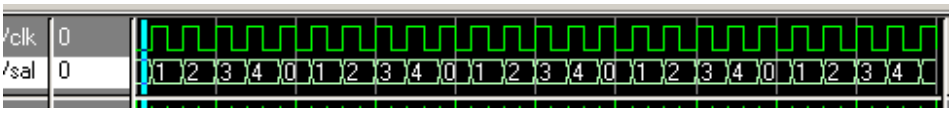
```
entity de0_4 is
   Port (clk    :in bit;
                        sal     :out integer range 0 to 4
         );
end de0_4;
architecture contador of de0_4 is
begin
        process
        variable cnt: integer range 0 to 4;
        begin
                wait until clk='1';
                if cnt=4 then
                cnt:=0;
                else
                cnt:=cnt+1;
                end if;
                sal<=cnt;
                end process;
end contador;
```

Para la simulacion en el test_beanch hacemos agregamos

```
-- *** Test Bench - User Defined Section ***
        clk<=not clk after 0.5 ms;
  tb : PROCESS
  BEGIN
    wait; -- will wait forever
  END PROCESS;
-- *** End Test Bench - User Defined Section ***
```

Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk

El resultado en el MODELSIM fue:



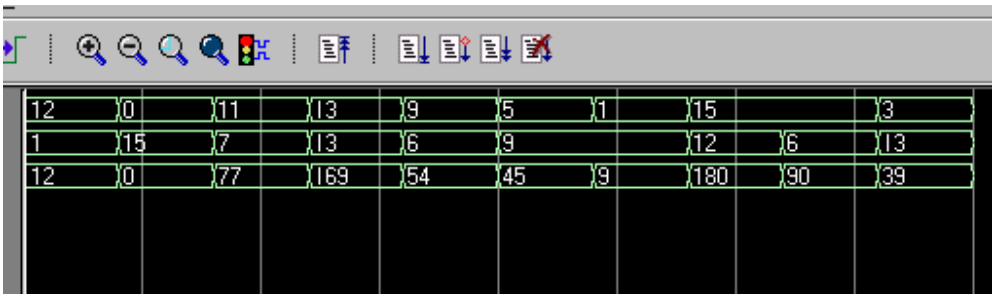
10.- Escriba el codigo en VHDL para implmentar un ciruito que realice la multiplicación de 2 números de 4 bits c/u utilizando el método de sumas sucesivas.

Preg10.vhd

```
entity multiplicador is
   Port (A,B:in integer range 0 to 15;
                              mul:out integer range 0 to 255
                              );
end multiplicador;
architecture sumas_sucesivas of multiplicador is
begin
        Process(A,B)
        Variable cnt:integer range 0 to 255;
        variable b2: integer range 0 to 15 :=B;
        Begin
        cnt:=0;
        For i in 0 to 15 loop
        if B>i then
                cnt:=cnt+A;
                end if;
        end loop;
        mul<=cnt;
        end process;
end sumas_sucesivas;
```

Para la simulacion:

Use el Test Beanch Waveform para crear las señanles y el resultado en el ModelSim fue el siguiente (el orden fue: A, B, mul)



Elver Yoel Ocmin Grandez
yoelocmin@hotmail.com
http://proyectos-fie-tk