

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

**Laboratorio N° 3:** Descripción VHDL utilizando Estilo Algorítmico-Parte II "

SENTENCIA LOOP

1.- Implementar y simular el siguiente circuito que cuenta el número de bits '1' que ingresan

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity contador_loop is
    Port ( dato : in          (7 downto 0);
          salida : out       (3 downto 0));
end contador_loop;

architecture Behavioral of contador_loop is
begin
process(dato)
variable num_bits :          (3 downto 0);
begin
    num_bits:= "0000";
    for i in 0 to 7 loop
    if dato (i) ='1' then
        num_bits := num_bits +1;
    end if;
    end loop;
    salida <= num_bits;
end process;
end Behavioral;
```

- Indique cuanto es el recurso utilizado 14/108 (17%)
- Utilizando VHDL Test Bench realice la simulación teniendo en consideración la siguiente sentencia concurrente:

Dato<=(dato+"0001") after 100 us;

Teniendo en cuenta los paquetes de la biblioteca IEEE que se usaron:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE ieee.std_logic_arith.ALL;
USE ieee.std_logic_unsigned.ALL;
```

El resultado de la simulación fue:

**TEST Contador de BITS**



Y como podemos notar la salida nos da la cantidad de unos que tiene nuestro dato.

## Laboratorio 3: Diseño Digital

- Modificar el programa para que el circuito contador de bits '1' aumente el número de entradas de 8 a 16 bits.

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity contador_loop_16 is
    Port ( dato : in          (15 downto 0);
          salida : out       (5 downto 0));
end contador_loop_16;

architecture Behavioral of contador_loop_16 is
begin
process(dato)
variable num_bits :          (5 downto 0);
begin
    num_bits:= "00000";
    for i in 0 to 15 loop
        if dato (i) ='1' then
            num_bits := num_bits +1;
        end if;
    end loop;
    salida <= num_bits;
end process;
end Behavioral;

```

- Indique cuanto es el porcentaje de recurso utilizado **32 /108 (29%)**

### DESCRIPCION DE MAQUINA DE ESTADO

#### A) ESTILO EXPLICITO E1:

- 2.- Implemente y simule la siguiente maquina de estado utilizando el estilo explicito E1.

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

```

```

entity MEF1 is
    Port ( reloj : in std_logic;
          reset : in std_logic;
          x : in std_logic;
          z : out std_logic);
end MEF1;

```

```

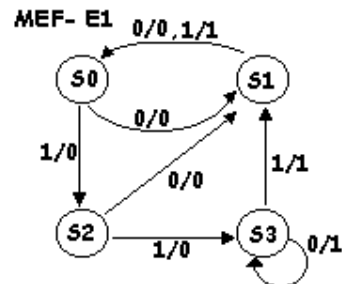
architecture Behavioral of MEF1 is
type estados is (S0,S1,S2,S3);
signal estado : estados ;
signal ntx_estado : estados ;
begin
combinacional: process(estado; x)
begin
    ntx_estado <= estado ;
    case estado is
        when S0 => z <= '0';

```

```

        if (x='0') then
            ntx_estado <= S1;
        else
            ntx_estado <= S2;
        end if;
    end case;
end process;
end Behavioral;

```



## Laboratorio 3: Diseño Digital

- ¿La descripción de la arquitectura coincide con el flujo seguido en el diagrama de estado?  
Rspta: SI
- Indique cuanto es el porcentaje del recurso utilizado **3/108 (2%)**
- Utilizando Test Bench Wave Form realice la simulación del circuito teniendo en consideración los siguientes valores de entrada.....

.....Se simulo y los resultado fueron tal y como se vio en la guia del laboratorio, y por considerar que estaban demás repetirlos, no pongo los resultados nuevamente.....

### **B) ESTILO EXPLICITO E2:**

#### 3.- Implemente y simule la siguiente maquina

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity MEF2 is
  Port ( RELOJ,RESET : in std_logic;
        X : in std_logic;
        Z : out std_logic);
end MEF2;

architecture algoritmo of MEF2 is
  type estados is(S0,S1,S2,S3);
begin
  secuencial:process
  variable estado:estados;
  begin
  wait until reloj='1';
  if(reset='0')then estado:=S0;
  else
  case estado is
  when S0=>Z<='0';if(x='0') then estado:=S1; else estado:=S2;end if;
  when S1=>Z<=X;estado:=S0;
  when S2=>Z<='0';if(x='0') then estado:=S1; else estado:=S3;end if;
  when S3=>Z<='0';if(x='1') then estado:=S1; end if;
  end case;
  end if;
  end process secuencial;
end algoritmo;
```

#### 4.- (TRABAJO) Implementar un decodificador de teclado matricial 4x4.

Utilice MEF.

Sol

Desarrollare el Problema 1 del examen, ya que a parte de ser un decodificador de teclado matricial, su enunciado esta mejor explicado.

### Laboratorio 3: Diseño Digital

PROB.- Se desea diseñar un circuito detector de teclado matricial 4x4. El circuito a implementar debe generar una secuencia por los pines de la columna C[4...1] como se muestra en la figura adjunta. Cada secuencia debe generarse cada 5 ms aprox. Si no se ha pulsado ninguna tecla los datos que se leen en las filas F[4...1] es "1111".

Si se pulsa una tecla debe codificarla de acuerdo al dato que ingresa por las filas de acuerdo a la tabla. Además la señal DV(dato valido) debe activarse 5 ms aprox.

NOTA: La señal de reloj que llega al CPLD es de 1 MHz.

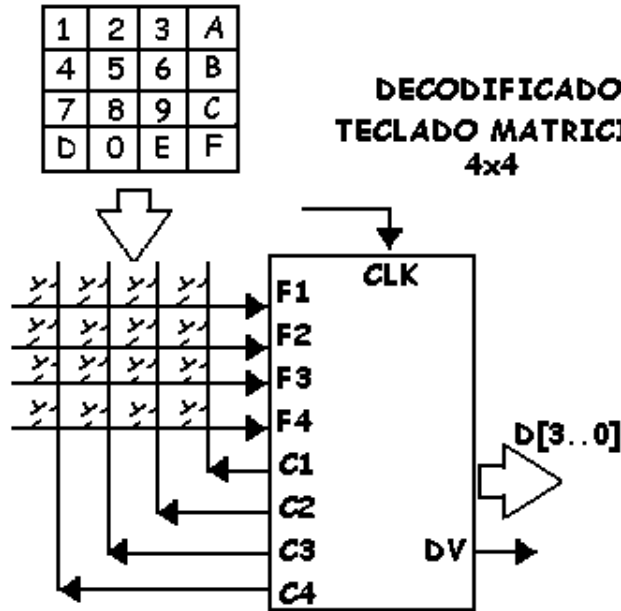


TABLA DE CONVERSIÓN TECLADO 4x4

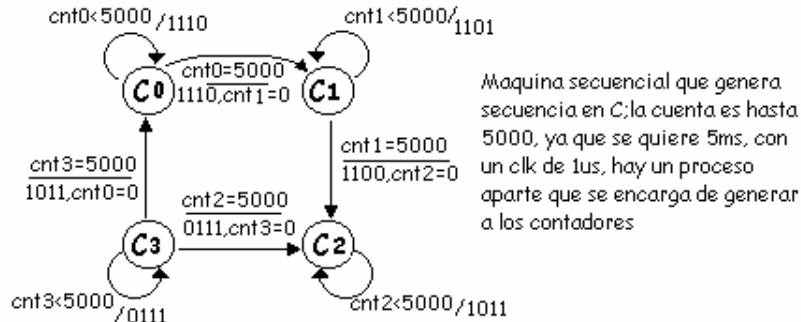
C4	C3	C2	C1	F4	F3	F2	F1	CARACTER
1	1	1	0	1	1	1	0	A=1010
1	1	1	0	1	1	0	1	B=1011
1	1	1	0	1	0	1	1	C=1100
1	1	1	0	0	1	1	1	F=1111
1	1	0	1	1	1	1	0	3=0011
1	1	0	1	1	1	0	1	6=0110
1	1	0	1	1	0	1	1	9=1001
1	1	0	1	0	1	1	1	E=1110
1	0	1	1	1	1	1	0	2=0010
1	0	1	1	1	1	0	1	5=0101
1	0	1	1	1	0	1	1	8=1000
1	0	1	1	0	1	1	1	0=0000
0	1	1	1	1	1	1	0	1=0001
0	1	1	1	1	1	0	1	4=0100
0	1	1	1	1	0	1	1	7=0111
0	1	1	1	0	1	1	1	D=1101

## Laboratorio 3: Diseño Digital

### Solucion:

Para resolver este ejercicio, lo voy a descomponer en 2 partes, cada una de ellas son procesos concurrentes, Asi que lo voy a trabajar como si fueran dos maquinas separadas.

1ra maquina: Esta va a ser la encargada de generarme la secuencia en C[4...0], y que cada una de estas secuencias dure 5 ms; C va alternar en los valores '1110', '1101', '1011' y '0111', nuestra secuencia seguira el siguiente diagrama de estados:



Y para su más facil codificación lo hacemos con el estilo MEF-E2:

Hasta ahora su codificación seria.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity secuencia is
    Port ( Clk : in bit;
          C : out std_logic_vector(3 downto 0));
end secuencia;

architecture algoritmo of secuencia is
    signal rstcnt, rstcnt0, rstcnt1, rstcnt2, rstcnt3: boolean:=true;
    signal cnt, cnt0, cnt1, cnt2, cnt3: integer:=0;
begin
    secuencia: process
        type estados is (C0, C1, C2, C3);
        variable estado: estados:=C0;
        begin
            wait until clk='1' and clk'event;
            case estado is
                when C0=>C<="1110";
                    if(cnt0>=5000) then
                        estado:=C1; rstcnt0<=true; rstcnt1<=true;
                    else rstcnt0<=false; estado:=C0; end if;
                when C1=>C<="1101";
                    if(cnt1>=5000) then
                        estado:=C2; rstcnt1<=true; rstcnt2<=true;
                    else rstcnt1<=false; estado:=C1; end if;
                when C2=>C<="1011";
                    if(cnt2>=5000) then
                        estado:=C3; rstcnt2<=true; rstcnt3<=true;
                    else rstcnt2<=false; estado:=C2; end if;
                when C3=>C<="0111";
                    if(cnt3>=5000) then
                        estado:=C0; rstcnt3<=true; rstcnt0<=true;
                    else rstcnt3<=false; estado:=C3; end if;
            end case;
        end process secuencia;
```

Elver Yoel OCMIN GRANDEZ

[yoelocmin@hotmail.com](mailto:yoelocmin@hotmail.com)

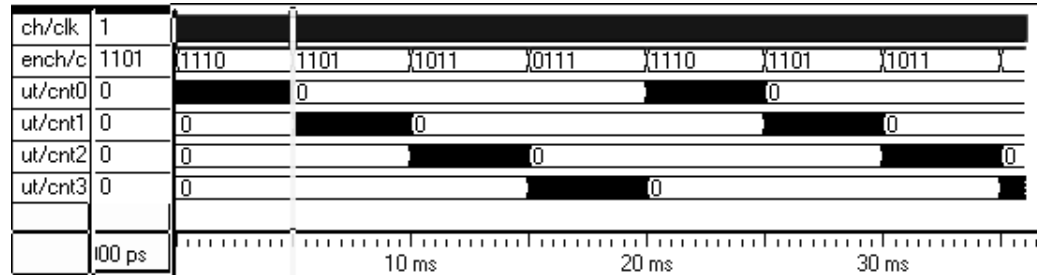
## Laboratorio 3: Diseño Digital

```

contador: process
begin
wait until Clk='1'and clk'event;
if rstcnt0 then cnt0<=0;else cnt0<=cnt0+1;end if;
if rstcnt1 then cnt1<=0;else cnt1<=cnt1+1;end if;
if rstcnt2 then cnt2<=0;else cnt2<=cnt2+1;end if;
if rstcnt3 then cnt3<=0;else cnt3<=cnt3+1;end if;
end process contador;
end algoritmo;

```

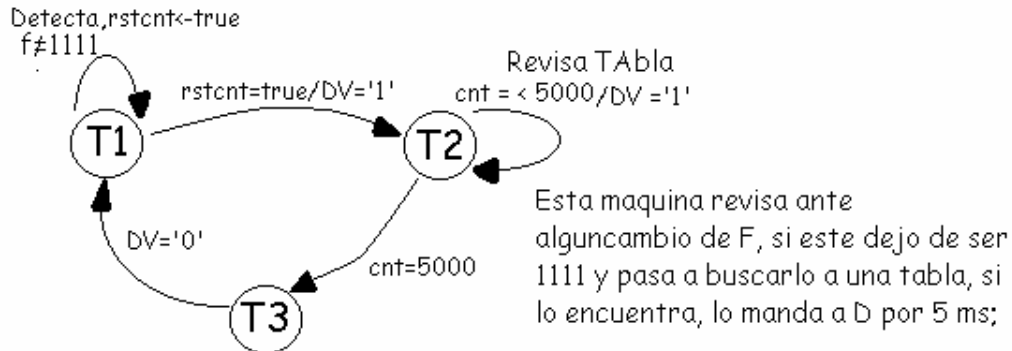
Y su simulacion seria :



Donde la parte ensobrecida es cuando los contadores cuentan 5000 c/u, para así, genera secuencia después de cada 5 ms;

Hasta ahora tenemos la secuencia, pero nos falta la tabla de conversión, o mejor dicho, la maquina que nos diga que teclas se presiono.

Esta maquina tiene mas o menos el siguiente diagrama de estado:



y la codificación total de esta maquina y la secuencial seria:

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity teclado is
    Port ( Clk : in bit;
          C : buffer std_logic_vector(3 downto 0);
          F : in std_logic_vector(3 downto 0);
          D : out std_logic_vector(3 downto 0);
          DV: out std_logic);
end teclado;

```

## Laboratorio 3: Diseño Digital

```
architecture algoritmo of teclado is
signal rstcnt,rstcnt0,rstcnt1,rstcnt2,rstcnt3:boolean:=true;
signal cnt,cnt0,cnt1,cnt2,cnt3:integer:=0;
begin

secuencia: process
type estados is(C0,C1,C2,C3);
variable estado:estados:=C0;
begin
wait until clk='1'and clk'event;
case estado is
when C0=>C<="1110";
if(cnt0>=5000) then estado:=C1;rstcnt0<=true;rstcnt1<=true;
else rstcnt0<=false;estado:=C0;end if;
when C1=>C<="1101";
if(cnt1>=5000) then estado:=C2;rstcnt1<=true;rstcnt2<=true;
else rstcnt1<=false;estado:=C1;end if;
when C2=>C<="1011";
if(cnt2>=5000) then estado:=C3;rstcnt2<=true;rstcnt3<=true;
else rstcnt2<=false;estado:=C2;end if;
when C3=>C<="0111";
if(cnt3>=5000) then estado:=C0;rstcnt3<=true;rstcnt0<=true;
else rstcnt3<=false;estado:=C3;end if;
end case;
end process secuencia;
tabla: process(F)
type estadostab is (T0,T1,T2);
variable tab:estadostab:=t0;
variable t: std_logic_vector(7 downto 0);
begin
T:=C&F;
primer: case tab is
when T0 =>if(F="1111") then rstcnt<=true;
else tab:=T1;rstcnt<=false;end if;DV<='0';
when T1 =>if(cnt>=5000) then rstcnt<=true;DV<='0';tab:=T2;
else segundo :case T is
when "11101110"=>D<="1010";DV<='1';--A="E&E"
when "11101101"=>D<="1011";DV<='1';--B="E&D"
when "11101011"=>D<="1100";DV<='1';--C="E&B"
when "11100111"=>D<="1111";DV<='1';--F="E&7"
when "11011110"=>D<="0011";DV<='1';--3="D&E"
when "11011101"=>D<="0110";DV<='1';--6="D&D"
when "11011011"=>D<="1001";DV<='1';--9="D&B"
when "11010111"=>D<="1110";DV<='1';--E="D&7"
when "10111110"=>D<="0010";DV<='1';--2="B&E"
when "10111101"=>D<="0101";DV<='1';--5="B&D"
when "10111011"=>D<="1000";DV<='1';--8="B&B"
when "10110111"=>D<="0000";DV<='1';--0="B&7"
when "01111110"=>D<="0001";DV<='1';--1="7&E"
when "01111101"=>D<="0100";DV<='1';--4="7&D"
when "01111011"=>D<="0111";DV<='1';--7="7&B"
when "01110111"=>D<="1101";DV<='1';--D="7&7"
when others=>D<="XXXX";DV<='0';
end case segundo;
end if;
when T2=>DV<='0';tab:=T0;
end case primer;
end process tabla;

contador: process
begin
wait until Clk='1'and clk'event;
if rstcnt0 then cnt0<=0;else cnt0<=cnt0+1;end if;
if rstcnt1 then cnt1<=0;else cnt1<=cnt1+1;end if;
if rstcnt2 then cnt2<=0;else cnt2<=cnt2+1;end if;
if rstcnt3 then cnt3<=0;else cnt3<=cnt3+1;end if;
end process contador;
end algoritmo;
```

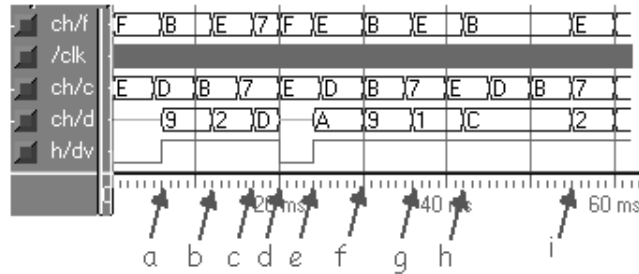
### Laboratorio 3: Diseño Digital

Y su simulación en el test-Bench lo hice teniendo en cuenta:

clk<=not clk after 500 ns; --//reloj de 1 us

f<="1111","1011"after 6 ms,"1110"after 12 ms,"0111"after 17 ms,"1111"after 20 ms,"1110"after 24 ms,"1011"after 30 ms,"1110"after 36 ms,"1011"after 42 ms,"1011"after 50 ms,"1110"after 55 ms,"1011"after 60 ms;

y la respuesta a la simulacion fue:



Donde podemos comprobar:

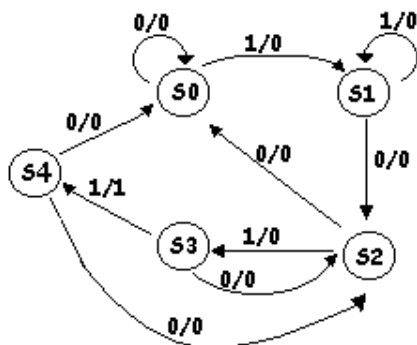
- En a: D&B = 9
- En b: B&E = 2
- En c: 7&7 = D
- En d: F-> no hay dato, no se presiono tecla
- En e: E&E =A
- En f: D&B=9
- En g: 7&E=1
- En h: E&B=C
- En i: B&E=2

5.-(TRABAJO) Implementar un circuito para detectar la siguiente secuencia: 1 0 1 1 (Considere Traslape). La salida Z normalmente es '0' y se activa cuando se detecta secuencia.



➤ Utilice el estilo MEF-E1

Su diagrama de estado seria:



Donde los estados son:

- S0: Cuando detecta el 1er numero, si encuentra al '1' pasa a S1, sino sigue esperando.
- S1: tiene un 1 y espera ahora un 0, para tener ahora '10' si lo encuentra pasa a S2, en caso de que no lo encuentre se queda ahí, ya que tendría un 11X- y si x=0, tendría parte de la serie en 110- tendría 10.



### Laboratorio 3: Diseño Digital

S2: Espera un 1 para tener "101", en caso de no tenerlo regresa a S0, ya que tendría 00X- y no le sirve.

S3: Espera un 1 para completar su serie "1011", en caso de no hacerlo tendría "1010" y le conviene regresar a S2 y esperar un "11" para terminar serie

S4: Ya completo la serie, ahora se pone a 1 y regresa a S0;

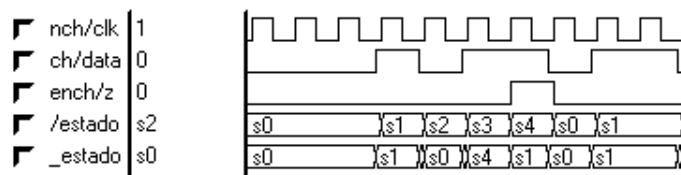
El programa sería:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity sec1011_E1 is
    Port (clk,data:in std_logic;
          z:out std_logic);
end sec1011_E1;

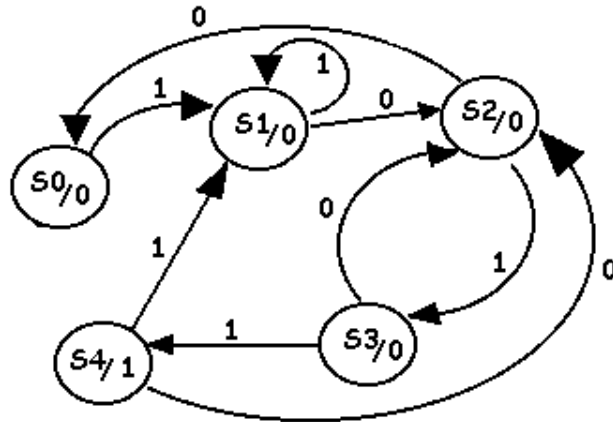
architecture Behavioral of sec1011_E1 is
    type estados is(S0,S1,S2,S3,S4);
    SIGNAL ESTADO:ESTADOS:=S0;
    signal nxt_estado:estados:=S0;
begin
    combinac: process(estado,data)
        begin
            nxt_estado<=estado;
            case estado is
                when S0=>Z<='0';
                    if(data='1') then nxt_estado<=S1;end if;
                when S1=> Z<='0';
                    if(data='0') then nxt_estado<=S2;end if;
                when S2=> Z<='0';
                    if(data='1') then nxt_estado<=S3;
                    else nxt_estado<=S0;end if;
                when S3=> z<='0';
                    if(data='1') then nxt_estado<=S4;
                    else nxt_estado<=S2;z<='0';end if;
                when S4=>z<='1';
                    if(data='0') then nxt_estado<=S0;
                    else nxt_estado<=S1;end if;
            end case;
        end process combinac;
    secuencial:process(clk)
        begin
            if(clk'event and clk='1') then estado<=nxt_estado;
            end if;
        end process secuencial;
end Behavioral;
```

La simulación con el Test Bench WaveForm:



Z es 1 cuando han pasado sus estados por S0 ,S1,S2,S3,S4 y ha reconocido que hubo la señal "1011" y en S4 pasa Z a '1'

➤ Utilice el estilo MEF-E2: Nuestro diagrama de estado es:



Y la programación sería:

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity sec1011_E2 is
    Port (clk,data:in std_logic;
          z:out std_logic);
end sec1011_E2;

architecture Behavioral of sec1011_E2 is
    type estados is(S0,S1,S2,S3,S4);
    begin
    secuencial: process
        variable estado:estados:=S0;
        begin
            wait until clk='1';
            case estado is
                when S0=>Z<='0';
                    if(data='1') then estado:=S1;end if;
                when S1=> Z<='0';
                    if(data='0') then estado:=S2;end if;
                when S2=> Z<='0';
                    if(data='1') then estado:=S3;else estado:=S0;end if;
                when S3=> Z<='0';
                    if(data='1') then estado:=S4;else estado:=S2;end if;
                when S4=> Z<='1';
                    if(data='1') then estado:=S1;else estado:=S4;end if;
            end case;
        end process secuencial;
    end Behavioral;
    
```

Y su simulación con el Test Bench WaveForm:

