

## DESCRIPCION DE MAQUINAS DE ESTADO

### A) ESTILO EXPLICITO E1:

Se utiliza un proceso combinacional para describir la función del próximo estado y las asignaciones de salida y un proceso secuencial para describir las asignaciones sobre el registro de estado en al transición activa de la señal de reloj.

1. Implementar y simular la siguiente máquina de estado utilizando el estilo explícito 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 nxt_estado:estados;

    begin
    combinac:    process(estado,x)
    begin
    nxt_estado<=estado;
    case estado is
    when s0=> Z<='0';
                if (x='0') then
                    nxt_estado<=s1;
                else
                    nxt_estado<=s2;
                end if;
    when s1=> Z<=X;
                nxt_estado<=s0;
    when s2=> Z<='0';
                if (x='0') then
                    nxt_estado<=s1;
                else
                    nxt_estado<=s3;
                end if;
    when s3=> Z<='1';
    
```

```

        if (x='1') then
            nxt_estado<=s1;
        else
            nxt_estado<=s3;
        end if;
    end case;
end process combinac;

secuencial: process(reset,reloj)
begin
    if (reset='0') then
        estado<=s0;
        elsif(reloj'event and reloj='1') then
            estado<=nxt_estado;
        end if;
    end process secuencial;
end Behavioral;

```

- La descripción de la arquitectura coincide con el flujo seguido en el diagrama de estado?: NO

La sentencia a cambiará es la siguiente:

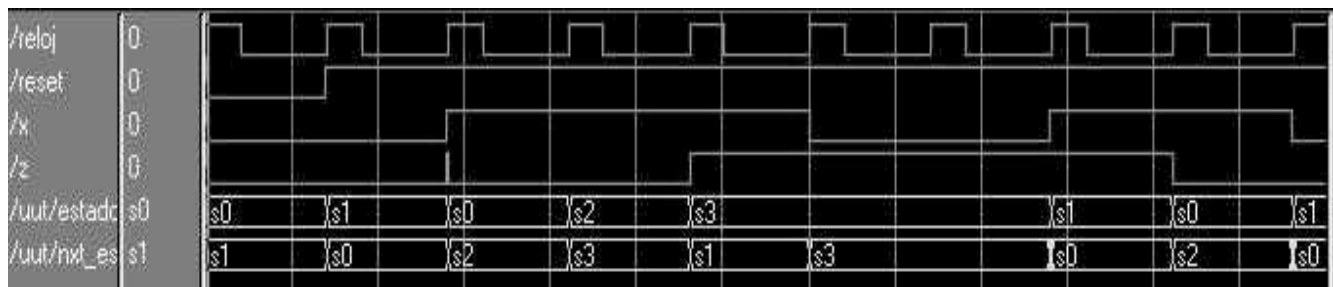
```

when s3=> Z<='1';
    if (x='1') then
        nxt_estado<=s1;
    else
        nxt_estado<=s3;
    end if;

```

- Indique cuanto es el porcentaje del recurso utilizado: 2 % de macrocelda.

Diagrama de simulación:



**B) ESTILO EXPLICITO E2:**

Se describe utilizando un único proceso secuencial con una sentencia de espera de reloj. Todas las acciones tienen lugar en sincronía con la señal de reloj. Por lo tanto, sólo permite la descripción de máquinas de MOORE.

2. Implementar y simular la siguiente máquina de estado utilizando el estilo explícito E2:

```

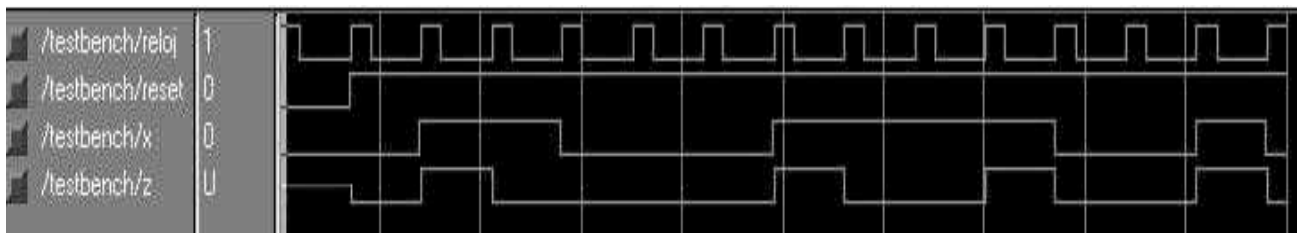
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity mef2 is
    Port ( reloj : in std_logic;
          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<='1';
                    if (x='1') then estado:=S1; end if;
            end case;
        end if;
    end process secuencial;
end algoritmo;

```

Diagrama de simulación:



3. Implementar un circuito para detectar la secuencia 1011(considere traslape).la salida normalmente es '0' y se activa al detectar la secuencia.

- Utilizando el estilo mef-e1

```

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

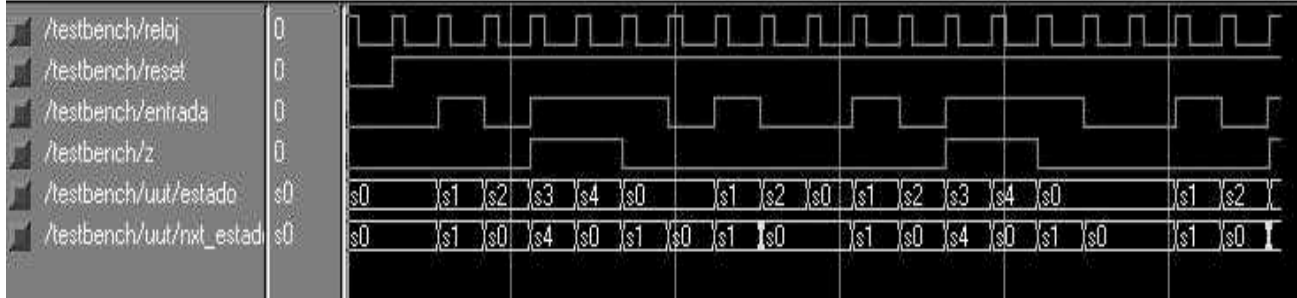
entity secuencia is
    Port ( reloj : in std_logic;
          reset  : in std_logic;
          entrada : in std_logic;
          z       : out std_logic);
end secuencia;

architecture algoritmo of secuencia is
    type estados is (s0,s1,s2,s3,s4);
    signal estado:estados;
    signal nxt_estado:estados;
begin
    combinac:process(estado,entrada)
    begin
        nxt_estado<=estado;
        case estado is
            when s0=>z<='0';
                if(entrada='1') then nxt_estado<=s1;           else
nxt_estado<=s0; end if;
            when s1=>z<='0';
                if(entrada='0') then nxt_estado<=s2;           else
nxt_estado<=s1; end if;
            when s2=>z<='0';
                if(entrada='1') then nxt_estado<=s3;           else
nxt_estado<=s0; end if;
            when s3=>z<=entrada;
                if(entrada='1') then nxt_estado<=s4;
            else nxt_estado<=s2; end if;
            when s4=>z<='1';
                nxt_estado<=s0;
        end case;
    end process combinac;

    secuencial:process(reloj,reset)
    begin
        if(reset='0')then
            estado<=s0;
        elsif(reloj'event and reloj='1') then
            estado<=nxt_estado;
        end if;
    
```

```
end process secuencial;
end algoritmo;
```

Diagrama de simulación:



- Utilizando estilo mef-e2

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

entity secuencia2 is
    Port ( reloj : in std_logic;
          reset : in std_logic;
          entrada : in std_logic;
          z : out std_logic);
end secuencia2;

architecture algoritmo of secuencia2 is
    type estados is (s0,s1,s2,s3,s4);
    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 (entrada='1') then estado:=s1; else estado:=s0;
                            end if;
                        when s1=>z<='0';
                            if (entrada='0') then estado:=s2; else estado:=s1;
                            end if;
                        when s2=>z<='0';
                            if (entrada='1') then estado:=s3; else estado:=s0;
                            end if;
                        when s3=>z<='0';
                            if (entrada='0') then estado:=s4; else estado:=s1;
                            end if;
                    end case;
                end if;
            end process;
        end;
    end;
end algoritmo;
```

```
        if (entrada='1') then estado:=s4; else estado:=s2;
        end if;
    when s4=>z<='1';
        estado:=s0;
end case;
end if;
end process secuencial;
end algoritmo;
```

Diagrama de simulación:

