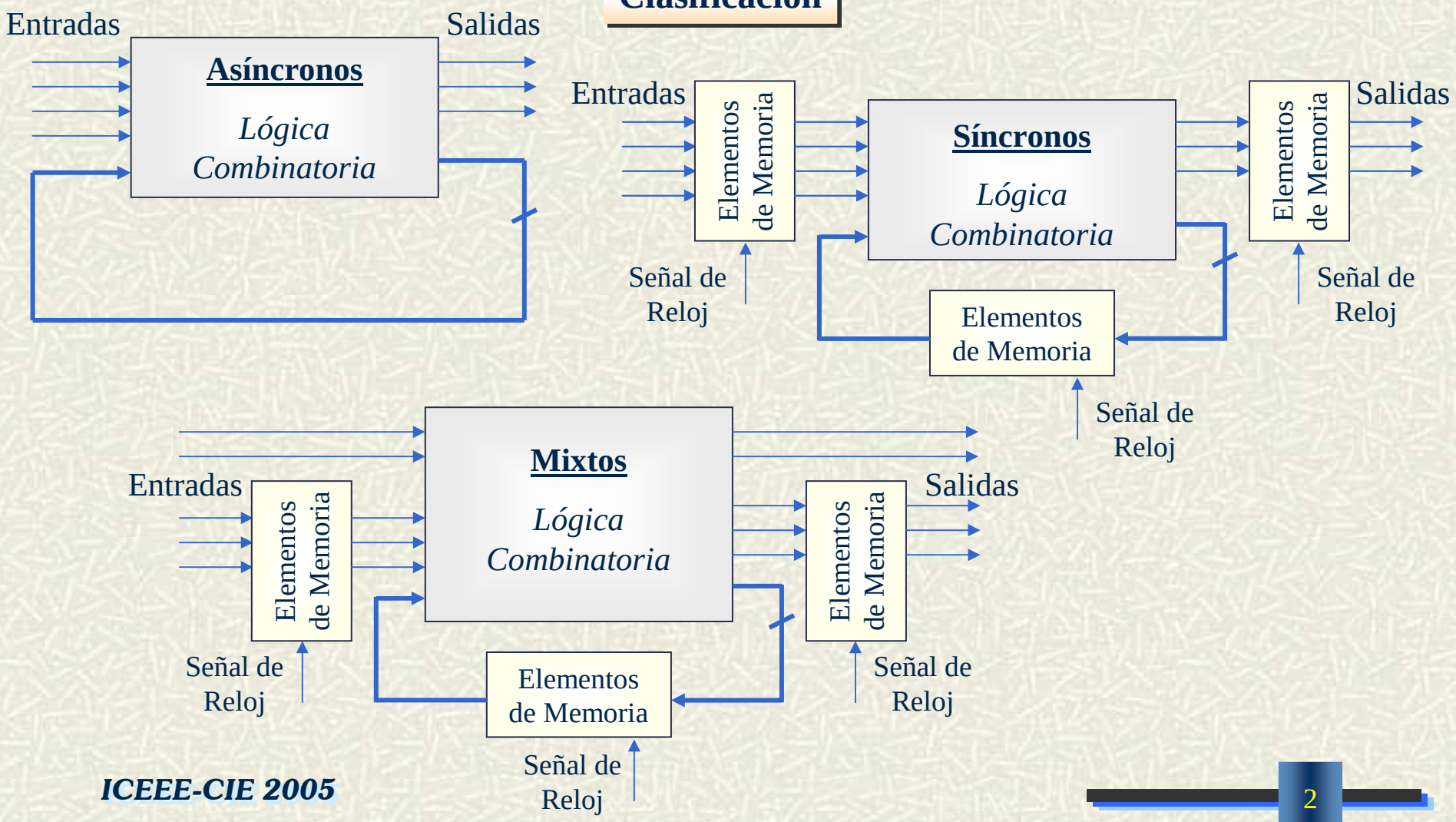
Capítulo 4

Circuitos Lógicos Secuenciales

Circuitos Lógicos Secuenciales

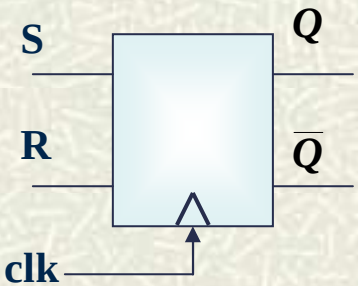
Estructura de un Sistema Lógico Secuencial

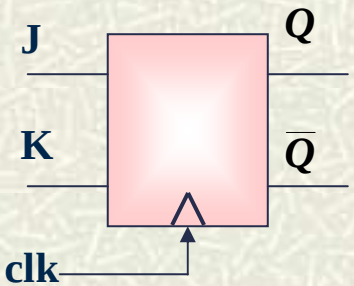
Clasificación

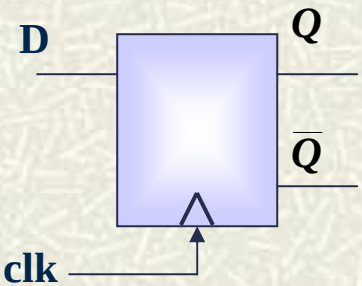


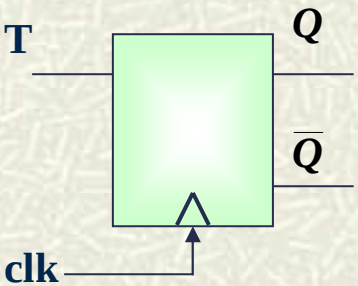
Elementos de Memoria: flip-flops

Elementos de Memoria: Flip-Flops









S	R	Q_t	Q_{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X

J	K	Q_t	Q_{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

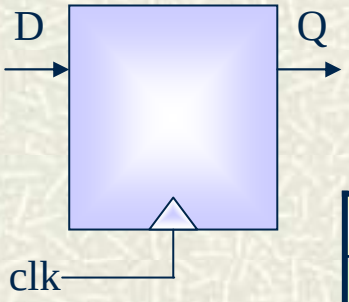
D	Q_t	Q_{t+1}
0	0	0
0	1	0
1	0	1
1	1	1

T	Q_t	Q_{t+1}
0	0	0
0	1	1
1	0	1
1	1	0

Elementos de Memoria: flip-flops

En el *Diseño Secuencial* con VHDL, las construcciones:

if-then-else / if-then-elsif-then son las más utilizadas.



D	Q_t	Q_{t+1}
0	0	0
0	1	0
1	0	1
1	1	1

Instrucciones equivalentes:

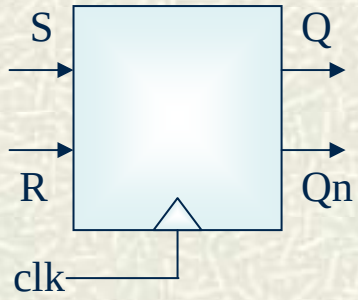
```
if rising_edge(clk)   – verdadero con el flanco de subida
if ( clk'event and clk= '1' and clk'last_value= '0' )

 if falling_edge(clk)   – verdadero con el flanco de bajada
if ( clk'event and clk= '0' and clk'last_value= '1' )
```

Ejemplo Nº 1 – Flip-Flop tipo D

```
library ieee;
use ieee.std_logic_1164.all;
entity ffd is
        port (D, clk: in std_logic;
                Q: out std_logic);
end ffd;
architecture arq_ffd of ffd is
begin
        process (clk)
        begin
        if (clk'event and clk='1') then
                Q <= D;
        end if;
        end process;
end arq_ffd;
```

Elementos de Memoria: flip-flops

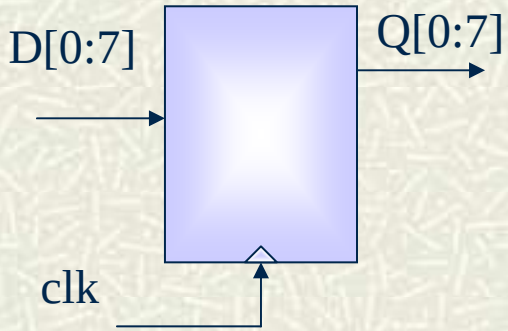
S	R	Q_t	Q_{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X



Ejemplo Nº 2 – Flip-Flop tipo RS

```
library ieee;
use ieee.std_logic_1164.all;
entity ffsr is
        port (S, R, clk: in std_logic;
                Q, Qn: inout std_logic);
end ffsr;
architecture a_ffsr of ffsr is
begin
        process (clk, S, R)
        begin
        if (clk'event and clk='1') then
                if (S = '0' and R = '1') then
                        Q <= '0';
                        Qn <= '1';
                elsif (S = '1' and R = '0') then
                        Q <= '1';
                        Qn <= '0';
                elsif (S = '0' and R = '0') then
                        Q <= Q;
                        Qn <= Qn;
                else
                        Q <= '-';
                        Qn <= '-';
                end if;
        end if;
        end process;
end a_ffsr;
```

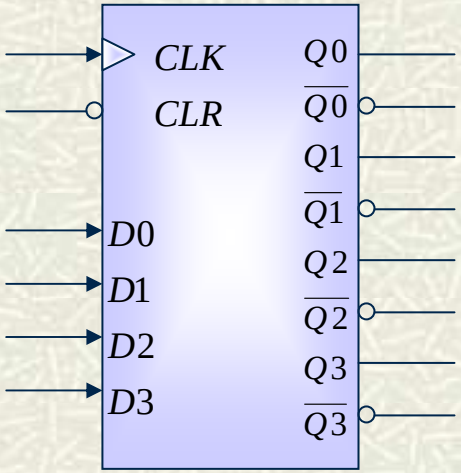
Registros



Ejemplo Nº 3 – Registro Paralelo de 8-Bits

```
library ieee;
use ieee.std_logic_1164.all;
entity reg is
        port (D: in std_logic_vector (0 to 7);
                clk: in std_logic;
                Q: out std_logic_vector (0 to 7));
end reg;
architecture arqreg of reg is
begin
        process (clk)
        begin
        if (clk'event and clk='1') then
                Q <= D;
        end if;
        end process;
end arqreg;
```

Registros

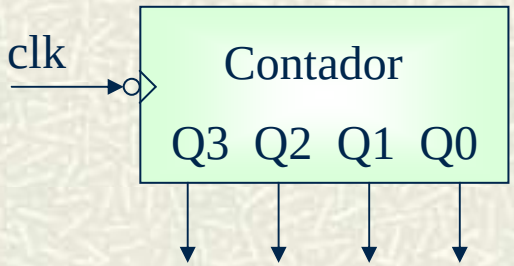


CLR	D	Q	Qn
0	'-'	0	1
1	D	D	Dn

Ejemplo Nº 4 – Registro Paralelo de 4-Bits con 'Clear'

```
library ieee;
use ieee.std_logic_1164.all;
entity reg4 is
        port (D: in std_logic_vector (3 downto 0);
                CLK, CLR: in std_logic;
                Q, Qn: out std_logic_vector (3 downto 0));
end reg4;
architecture a_reg4 of reg4 is
begin
        process (CLK, CLR) begin
        if (CLK'event and CLK='1') then
                if (CLR = '1') then
                        Q <= D;
                        Qn <= not D;
                else
                        Q <= "0000";
                        Qn <= "1111";
                end if;
        end if;
        end process;
end a_reg4;
```

Contadores



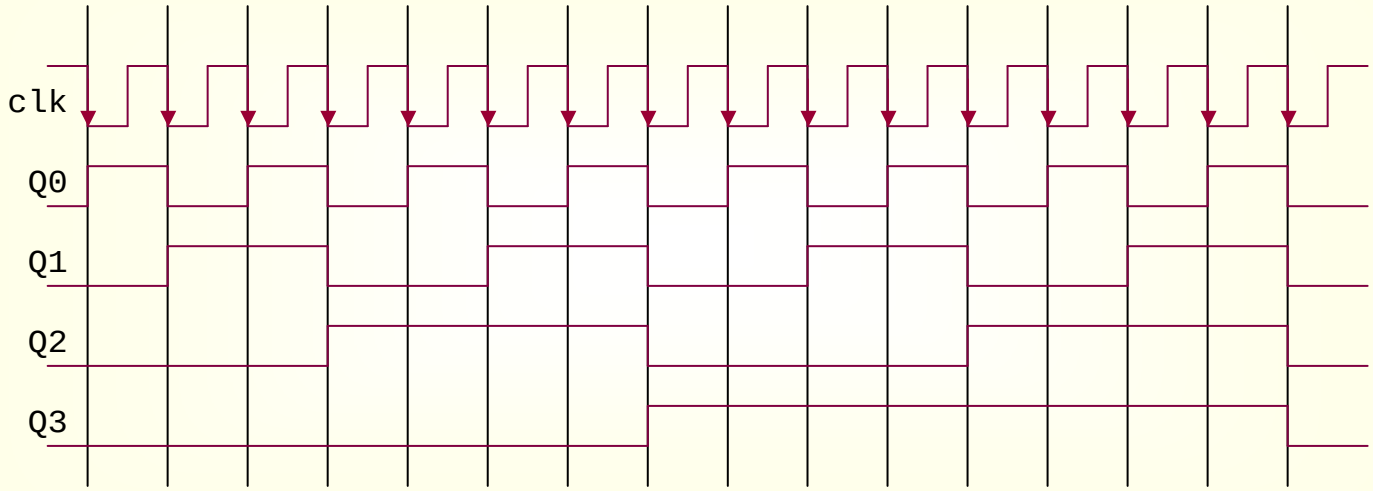


Diagrama de tiempo del contador de 4 bits

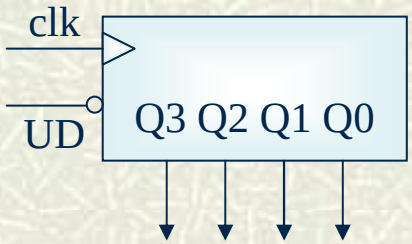
Contadores

Ejemplo Nº 5 – Contador de 4-Bits

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity cont4 is
        port (clk: in std_logic;
                Q: inout std_logic_vector (3 downto 0));
end cont4;
architecture arqcont of cont4 is
begin
        process (clk)
                begin
                if (clk'event and clk = '0') then
                        Q <= Q +1;
                end if;
        end process;
end arqcont;
```

Es verdadera con el flanco de bajada de clk

Contadores

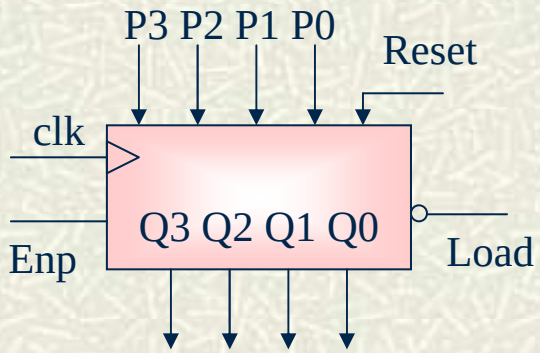


UD	Acción
0	Cuenta Ascendente
1	Cuenta Descendente

Ejemplo Nº 6 – Contador Ascendente/Descendente de 4-Bits

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity contador is
        port (clk: in std_logic;
                UD: in std_logic;
                Q: inout std_logic_vector (3 downto 0));
end contador;
architecture a_contador of contador is
begin
        process (UD, clk) begin
                if (clk'event and clk = '1') then
                        if (UD = '0') then
                                Q <= Q +1;
                        else
                                Q <= Q -1;
                        end if;
                end if;
        end process;
end a_contador;
```

Contadores



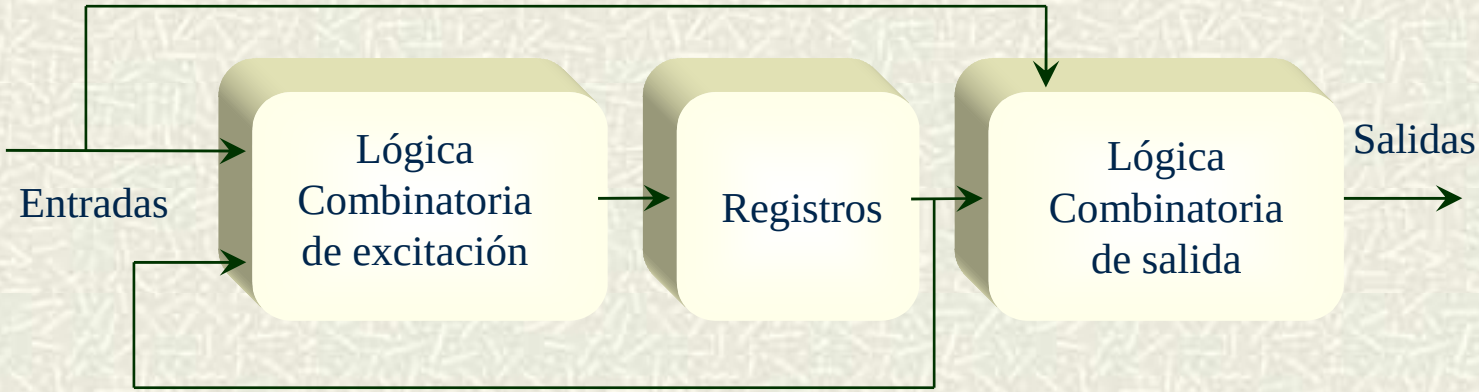
Enp	Load	Acción
0	0	Carga
0	1	Mantiene Estado
1	0	Carga
1	1	Cuenta

Ejemplo Nº 7 – Contador de 4-bits con reset y carga en paralelo

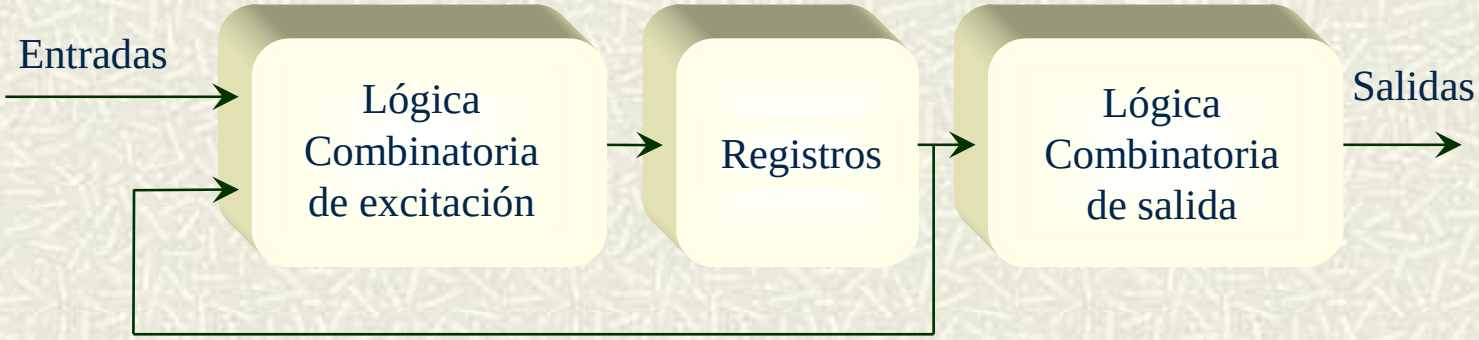
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all; use ieee.std_logic_unsigned.all;
entity cont is
        port (P: in std_logic_vector (3 downto 0);
                clk, Load, Enp, Reset: in std_logic;
                Q: inout std_logic_vector (3 downto 0));
end cont;
architecture arq_cont of cont is
begin
        process (clk, Reset, Load, Enp) begin
        if (Reset = '1') then
                Q <= "0000";
        elsif (clk'event and clk = '1') then
                        if (Load = '0' and Enp = '-') then
                                Q <= P;
                        elsif (Load = '1' and Enp = '0') then
                                Q <= Q;
                        elsif (Load = '1' and Enp = '1') then
                                Q <= Q + 1;
                        end if;
        end if;
        end process;
end arq_cont;
```

Operación Asíncrona

Máquinas de estados



Máquina de Mealy

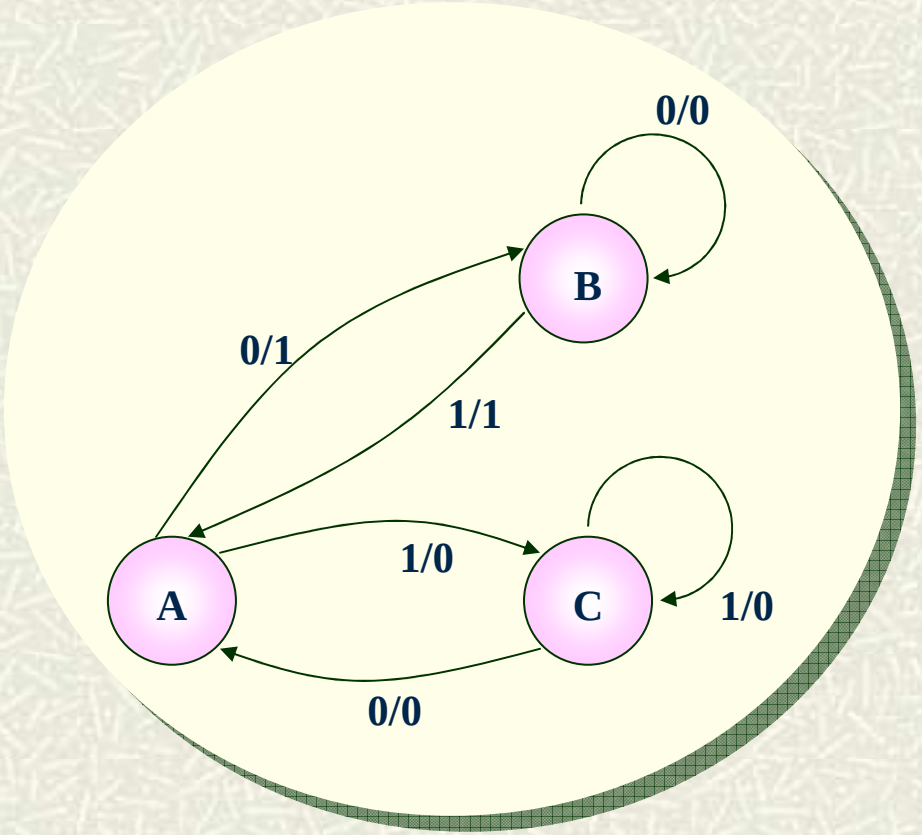


Máquina de Moore

Máquinas Secuenciales

Ejemplo Nº 8

Representación de una Máquina de Mealy



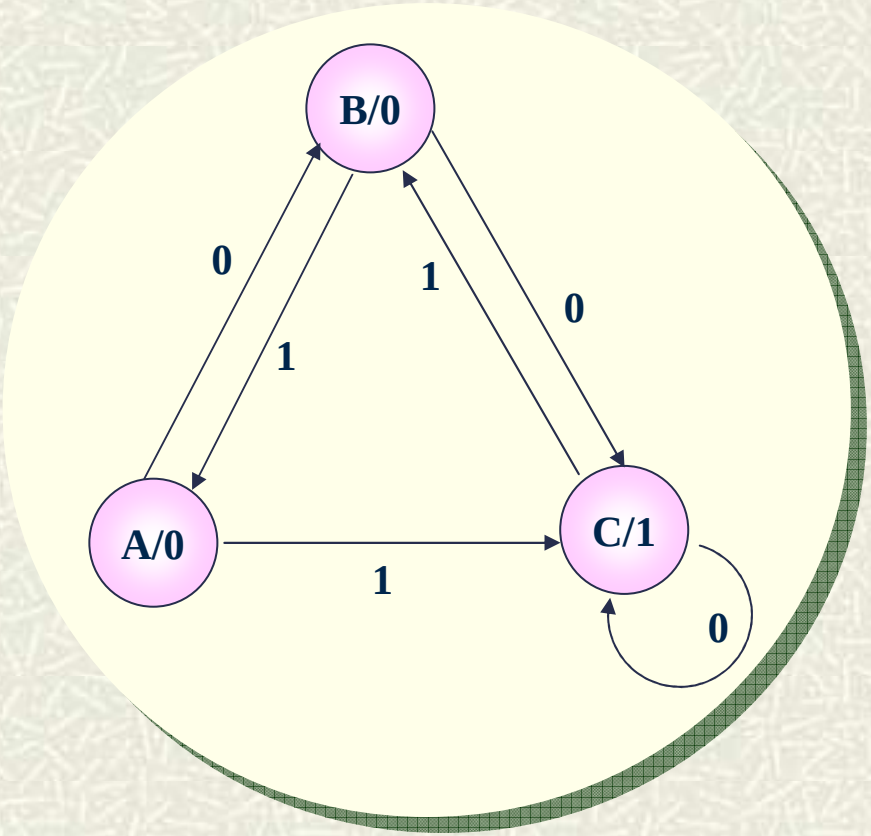
Estado Presente	Entrada X	
	0	1
A	B/1	C/0
B	B/0	A/1
C	A/0	C/0

Próximo Estado / Salida Y

Máquinas Secuenciales

Ejemplo Nº 9

Representación de una Máquina de Moore



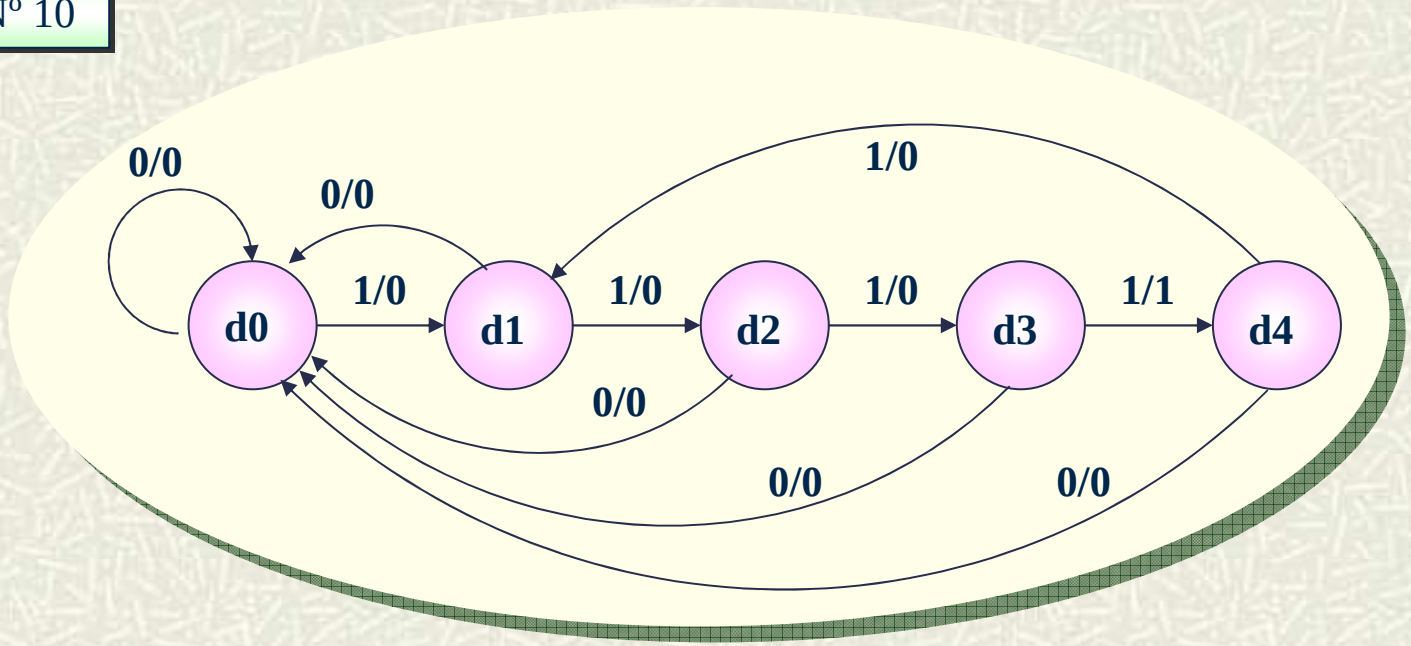
Estado Presente	Entrada X		Salida Y (Para el Estado Presente)
	0	1	
A	B	C	0
B	C	A	0
C	C	B	1

Próximo Estado

Diseño de Sistemas Secuenciales Síncronos

Ejemplo Nº 10

Circuito Secuencial que detecta 4-Unos (1's) consecutivos



Edo. Presente	Edo. Futuro		z (Salida)	
	x = 0	x = 1	x = 0	x = 1
d0	d0	d1	0	0
d1	d0	d2	0	0
d2	d0	d3	0	0
d3	d0	d4	0	1
d4	d0	d1	0	0

Diseño de Sistemas Secuenciales Síncronos

Para declarar los estados de una Máquina de Estados Finitos se realiza lo siguiente:

Valores que pueden tener el *edo.presente* y el *edo.futuro* en el ejemplo Nº 10:	
edo_presente	*edo_futuro*
d0	d0
d1	d1
d2	d2
d3	d3
d4	d4

estados

Declaración de Estados en una FSM

```
type estados is (d0, d1, d2, d3, d4);
signal edo_presente, edo_futuro: estados;
```

estados es el nombre o identificador dado por el usuario al conjunto de datos conformado por *d0, d1,d2 d3, d4*. A este tipo de datos se le conoce como Tipo de Datos *Enumerados*

edo_presente, edo_futuro son señales (signal) que pueden adquirir cualquiera de los valores (*d0, d1, d2, d3, d4*) que describen al tipo de dato enumerado identificado con el nombre de *estados*. ***edo_presente*** y ***edo_futuro*** son también datos del tipo enumerado

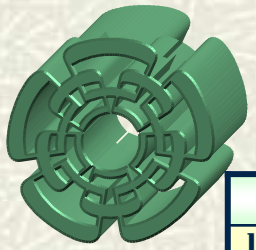
¿Cómo son codificados: d0, d1, d2, d3, d4?

Tipos de Codificación utilizados:

- One-Hot
- Compact
- Secuencial
- Gray
- Johnson
- Definido por Usuario

000	d0
001	d1
010	d2
011	d3
100	d4
101	101
110	110
111	111

El Número de Bits utilizados en la codificación está en función del Número de Estados **(Tarea realizada por el Compilador o Sintetizador)***



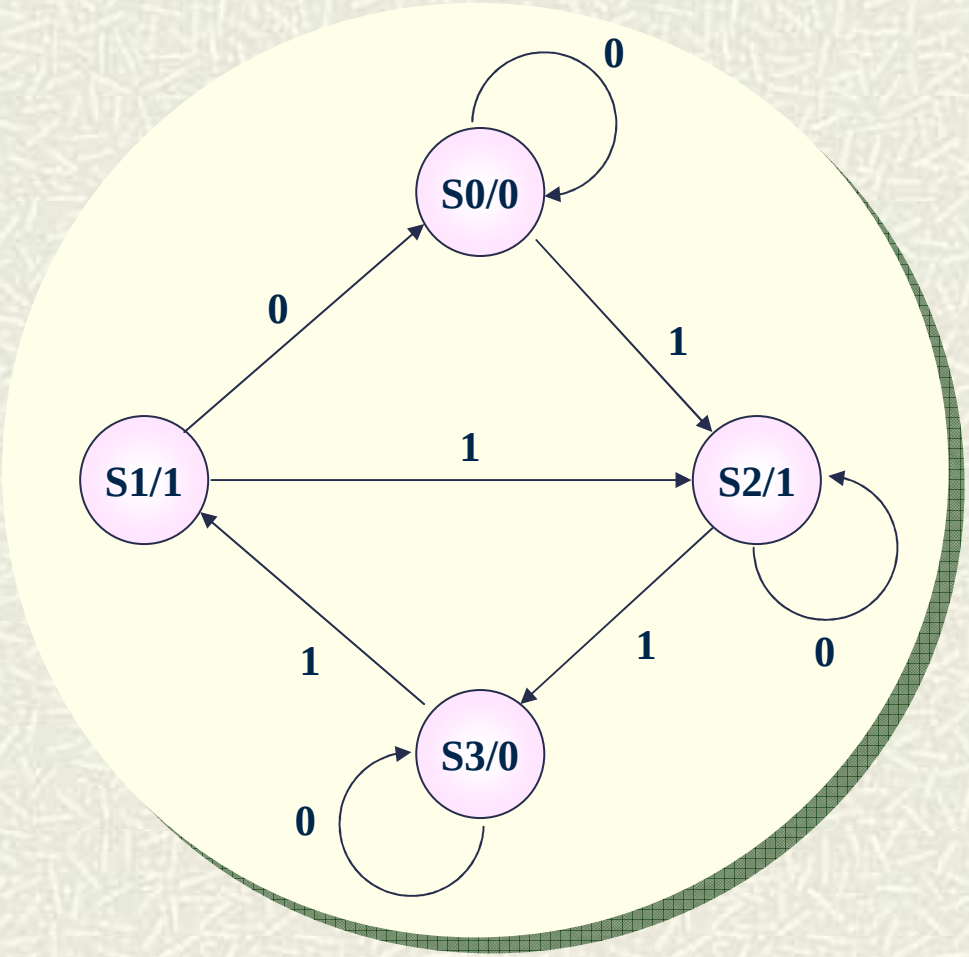
Diseño de Sistemas Secuenciales Síncronos

Ejemplo Nº 10 – Detector de Secuencia

```
library ieee;
use ieee.std_logic_1164.all;
entity diagrama is
        port (clk, x: in std_logic;
                z: out std_logic);
end diagrama;

architecture arq_diagrama of diagrama is
type estados is (d0, d1, d2, d3, d4);
signal edo_presente, edo_futuro: estados;

begin
        proceso1: process (edo_presente, x) begin
        case edo_presente is
                when d0 =>
                if x = '1' then
                        edo_futuro <= d1;
                        z <= '0';
                else
                        edo_futuro <= d0;
                        z <= '0';
                end if;
                when d1 =>
                if x = '1' then
                        edo_futuro <= d2;
                        z <= '0';
                else
                        edo_futuro <= d0;
                        z <= '0';
                end if;
                when d2=>
                if x = '1' then
                        edo_futuro <= d3;
                        z <= '0';
                else
                        edo_futuro <= d0;
                        z <= '0';
                end if;
                when d3 =>
                if x = '1' then
                        edo_futuro <= d4;
                        z <= '1';
                else
                        edo_futuro <= d0;
                        z <= '0';
                end if;
                when d4 =>
                if x = '1' then
                        edo_futuro <= d1;
                        z <= '0';
                else
                        edo_futuro <= d0;
                        z <= '0';
                end if;
end case;
end process  proceso1;
        proceso2: process (clk) begin
                if (clk'event and clk = '1') then
                        edo_presente <= edo_futuro;
                end if;
        end process proceso2;
 end arq_diagrama;
```

Ejemplo Nº 11

Máquina de Moore



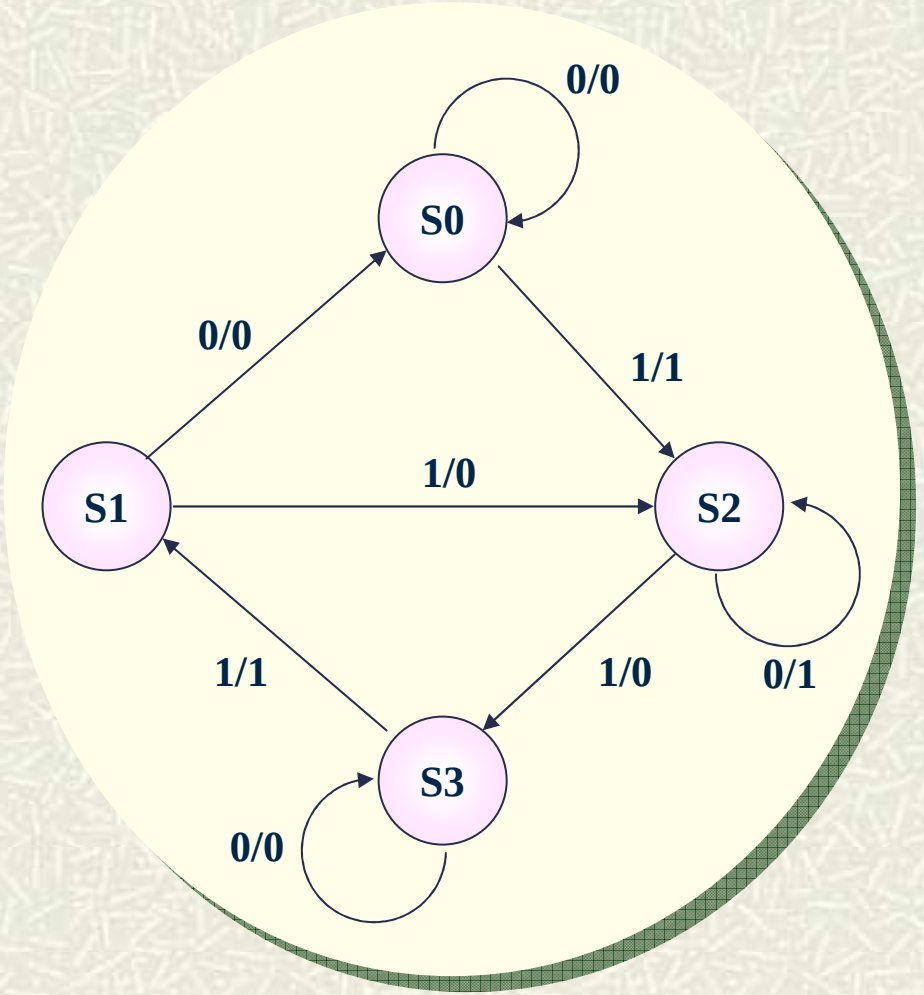
Diseño de Sistemas Secuenciales Síncronos

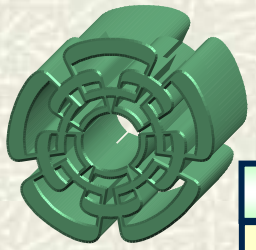
Ejemplo Nº 11 – Máquina de Moore

```
library ieee;
use ieee.std_logic_1164.all;
entity MOORE is
        port (X, CLK: in std_logic;
                Z: out std_logic);
end MOORE;
architecture ARQ_MOORE of MOORE is
type Estados is (S0, S1, S2, S3);
signal Edo_Pres, Edo_Fut: Estados;
begin
        proceso1: process (Edo_Pres, X) begin
        case Edo_Pres is
                when S0 => Z<= '0';
                if X = '0' then
                        Edo_Fut <= S0;
                else
                        Edo_Fut <= S2;
                end if;
                when S1 => Z <= '1';
                if X = '0' then
                        Edo_Fut <= S0;
                else
                        Edo_Fut <= S2;
                end if;
                when S2=> Z <= '1';
                if X = '0' then
                        Edo_Fut <= S2;
                else
                        Edo_Fut <= S3;
                end if;
                when S3 => Z <= '0';
                if X = '0' then
                        Edo_Fut <= S3;
                else
                        Edo_Fut <= S1;
                end if;
        end case;
        end process  proceso1;
        proceso2: process (CLK) begin
                if (CLK'event and CLK = '1') then
                        Edo_Pres <= Edo_Fut;
                end if;
        end process proceso2;
end ARQ_MOORE;
```

Ejemplo Nº 12

Máquina de Mealy





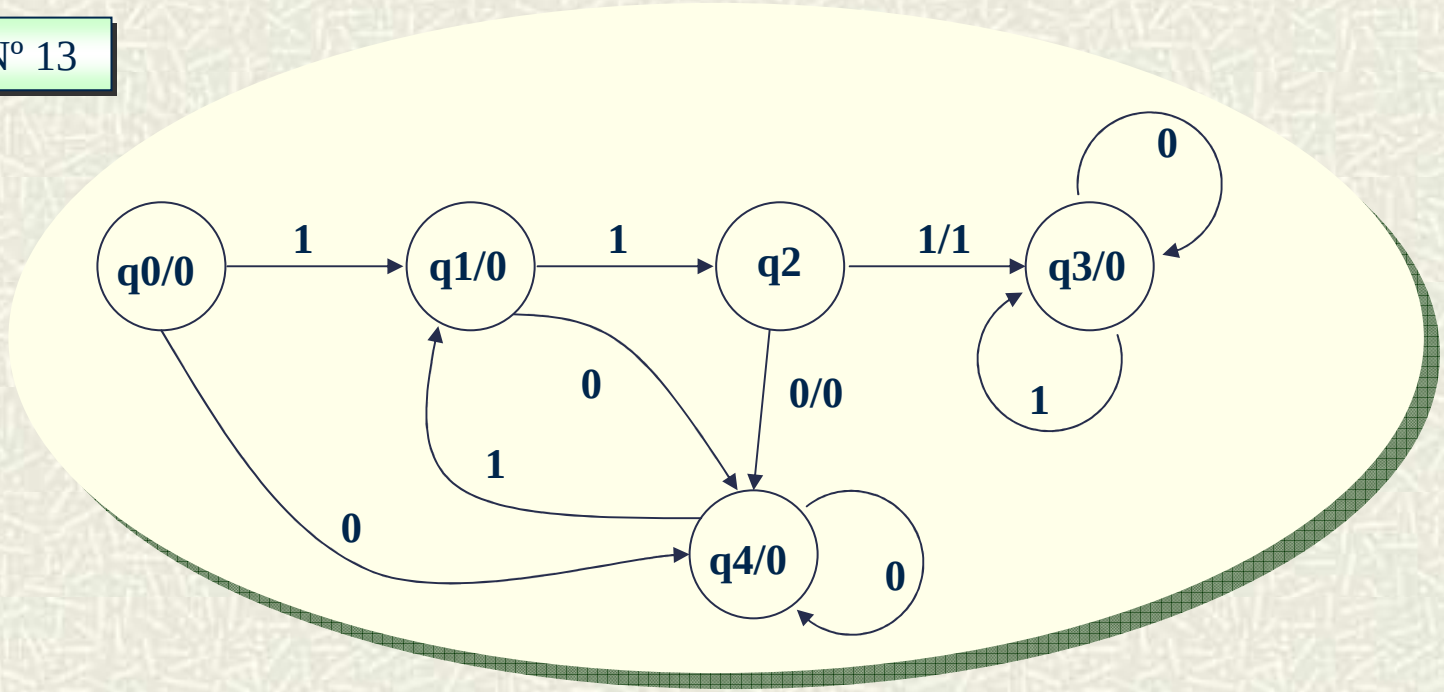
Diseño de Sistemas Secuenciales Síncronos

Ejemplo Nº 12 – Máquina de Mealy

```
library ieee;
use ieee.std_logic_1164.all;
entity MEALY is
        port (clk, x: in std_logic;
                z: out std_logic);
end MEALY;
architecture ARQ_MEALY of MEALY is
type Estados is (S0, S1, S2, S3);
signal Edo_Pres, Edo_Fut: Estados;
begin
        proceso1: process (Edo_Pres, x) begin
        case Edo_Pres is
                when S0 =>
                if x = '0' then
                        z <= '0';
                        Edo_Fut <= S0;
                else
                        z <= '1';
                        Edo_Fut <= S2;
                end if;
                when S1 =>
                if x = '0' then
                        z <= '0';
                        Edo_Fut <= S0;
                else
                        z <= '0';
                        Edo_Fut <= S2;
                end if;
                when S2=>
                if x = '0' then
                        z <= '1';
                        Edo_Fut <= S2;
                else
                        z <= '0';
                        Edo_Fut <= S3;
                end if;
                when S3 =>
                if x = '0' then
                        z <= '0';
                        Edo_Fut <= S3;
                else
                        z <= '1';
                        Edo_Fut <= S1;
                end if;
        end case;
        end process  proceso1;
        proceso2: process (clk) begin
                if (clk'event and clk = '1') then
                        Edo_Pres <= Edo_Fut;
                end if;
        end process proceso2;
end ARQ_MEALY;
```

Diseño de Sistemas Secuenciales Síncronos

Ejemplo Nº 13



¿Qué tipo de Máquina es?

¿Es correcta su Descripción en VHDL?

¿Cumple las Reglas de Verificación de un Diagrama de Estados?

Diseño de Sistemas Secuenciales Síncronos

Ejemplo Nº 13 - Máquina Mixta

```
library ieee;
use ieee.std_logic_1164.all;
entity diag is
        port (clk, x: in std_logic;
                z: out std_logic);
end diag;
architecture arq_diag of diag is
type estados is (q0, q1, q2, q3, q4);
signal edo_pres, edo_fut: estados;
begin
        proceso1: process (edo_pres, x) begin
        case edo_pres is
                when q0 => z<= '0';
                if x = '0' then
                        edo_fut <= q4;
                else
                        edo_fut <= q1;
                end if;
                when q1 => z <= '0';
                if x = '0' then
                        edo_fut <= q4;
                else
                        edo_fut <= q2;
                end if;
                when q2=>
                if x = '0' then
                        edo_fut <= q4;
                        z <= '0';
                else
                        edo_fut <= q3;
                        z <= '1';
                end if;
                when q3 => z <= '0';
                if x = '0' then
                        edo_fut <= q3;
                else
                        edo_fut <= q3;
                end if;
                when q4 => z <= '0';
                if x = '0' then
                        edo_fut <= q4;
                else
                        edo_fut <= q1;
                end if;
        end case;
        end process  proceso1;
        proceso2: process (clk) begin
                if (clk'event and clk = '1') then
                        edo_pres <= edo_fut;
                end if;
        end process proceso2;
end arq_diag;
```