2.5. Descripción de máquinas de estado (FSM) en VHDL

Objetivos

- Qué es una FSM
- Tipos de FSM:
  - Mealy
  - Moore
- Descripción de máquinas de estado en VHDL:
  - Un proceso
  - Dos procesos

2.4.1. Qué son las máquinas de estado

Máquinas de estado finita (FSM)

- Es un circuito donde su salida depende del valor de la entradas ahora y del valor de las entradas antes (estado actual).→Cualquier circuito secuencial

- **Ejemplo: ascensor, contador, etc.**

- Basan su funcionamiento en los AUTÓMATAS FINITOS (matemáticas)

- La FSM avanza hacia cada estado según una secuencia predeterminada (tabla o diagrama de transición de estados).

Máquinas de estado (FSM)

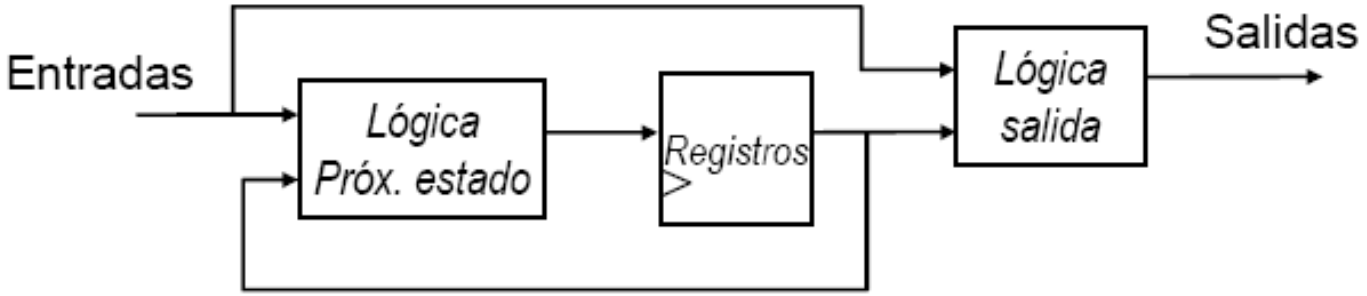
Proceso de diseño:

1. Análisis del problema para determinar entradas y estados posibles.
2. Decidir qué tipo de máquina se diseñará (Mealy o **Moore**)
3. Diseño del diagrama de transición de estados (grafo):
    - Transición entre estados
    - Salidas
4. Codificación VHDL

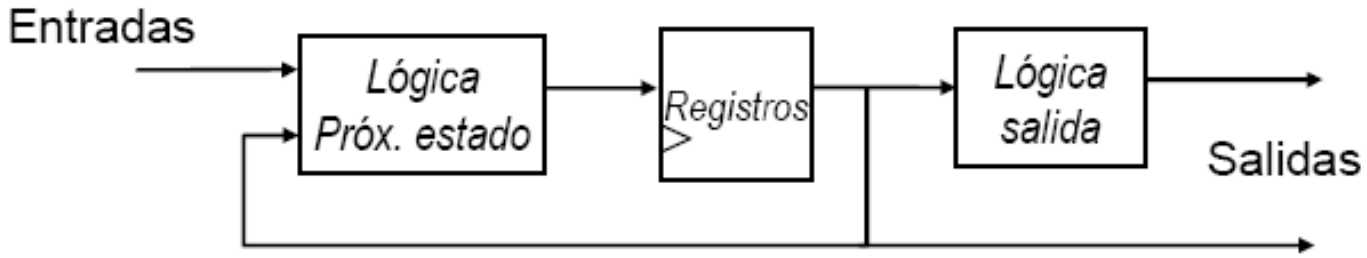
2.4.2. Tipos de máquinas de estado

Máquinas de estado (FSM)

- **Mealy**: las salidas son función del estado y entradas actuales



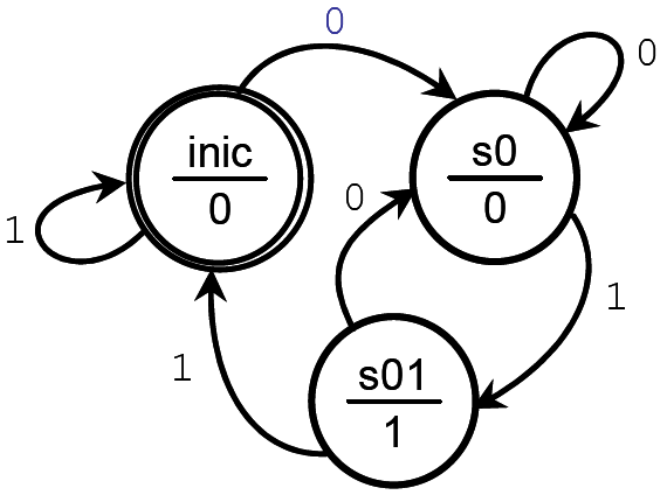
- **Moore**: las salidas son función sólo del estado actual



Máquinas de estado (FSM): ejemplo

Detector de flancos (Secuencia “01”)

PULSADOR (IN)	ESTADO ACTUAL	ESTADO SIGUIENTE	Salida
0	inic	S0	0
1	inic	inic	0
0	S0	S0	0
1	S0	S01	1
0	S01	S0	0
1	S01	inic	0



Inic → Esperando

S0 → Llegó un “0”

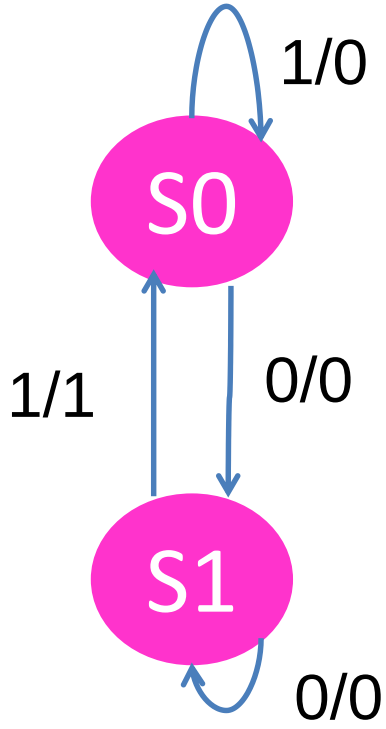
S01 → Llegó “01”

Solución A: Moore

Máquinas de estado (FSM): ejemplo

Detector de flancos (Secuencia “01”)

PULSADOR (IN)	ESTADO ACTUAL	ESTADO SIGUIENTE	Salida
0	S0	S1	0
1	S0	S0	0
0	S1	S1	0
1	S1	S0	1

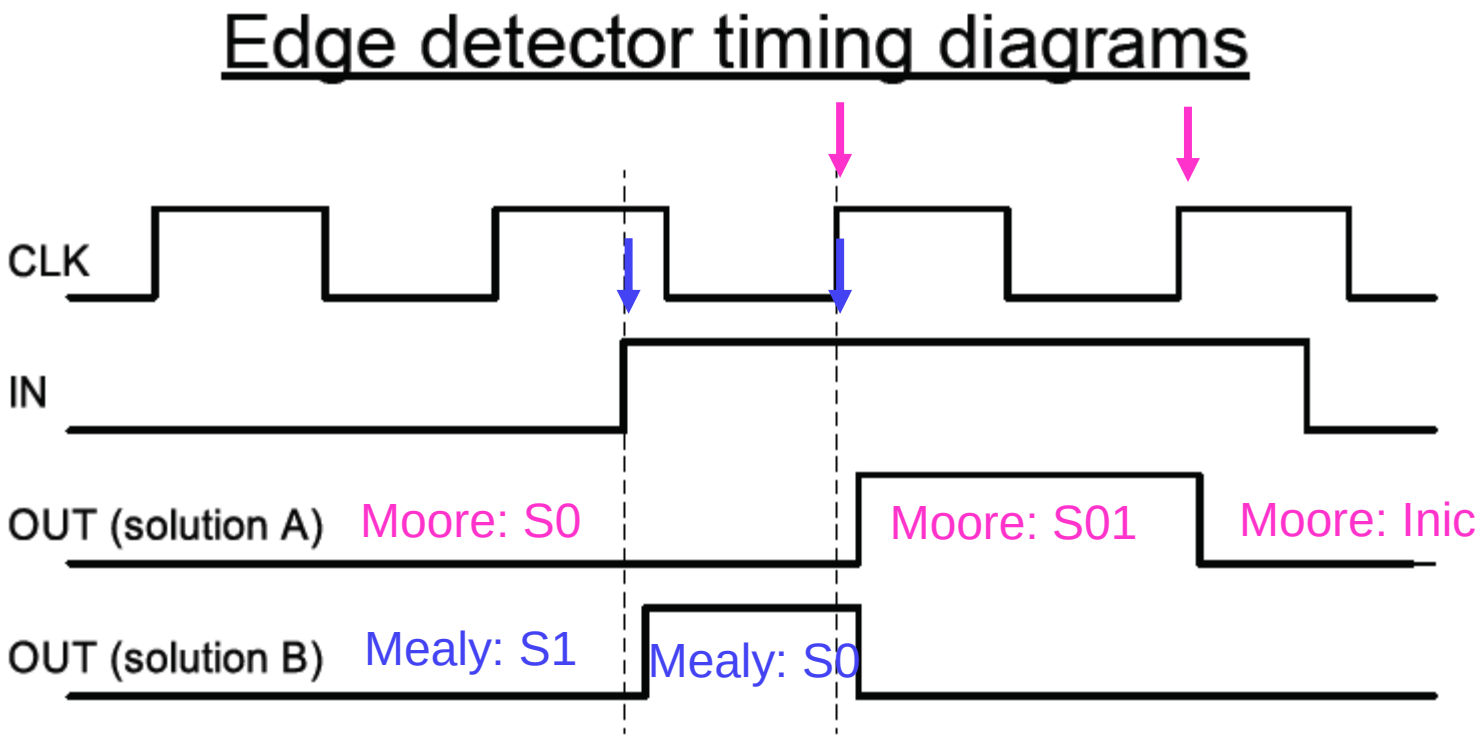


S0 → Esperando

S1 → Llegó un “0”

Solución B: Mealy

Máquinas de estado (FSM): ejemplo



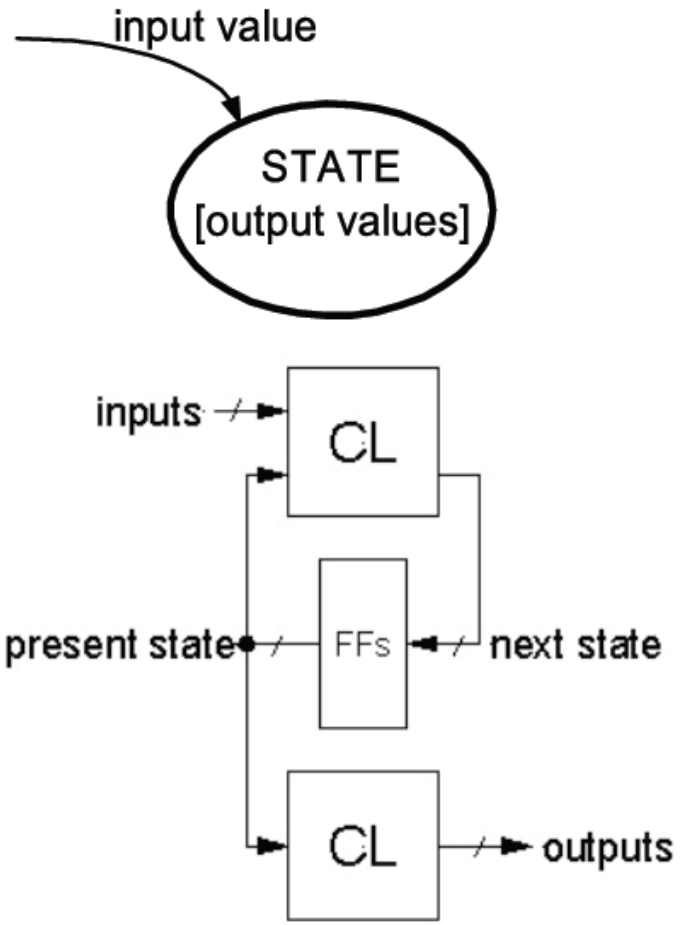
- Solution A: output follows the clock
- Solution B: output changes with input rising edge and is asynchronous wrt the clock.

Máquinas de estado (FSM)

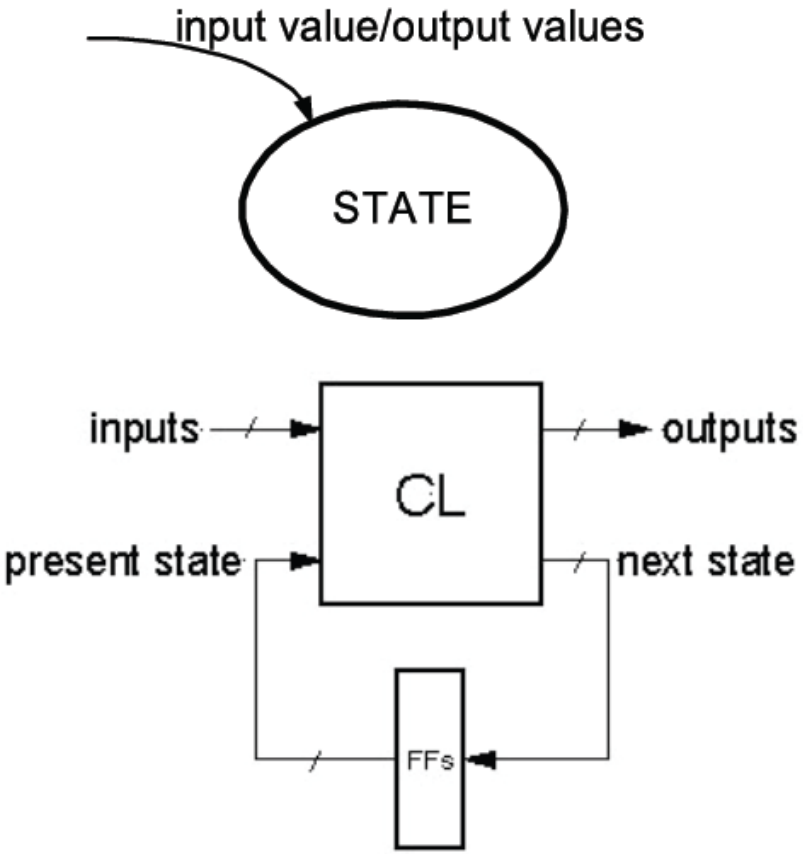
Moore	Mealy
Tiene **más estados**	Tiene **menos estados**
Tarda un **ciclo más** en dar la salida	Tarda **un ciclo menos** en dar la salida
Salida estable durante **un ciclo** completo de **reloj**	**Salida NO estable** durante **un ciclo** completo de **reloj**
Salida síncrona	**Salida asíncrona**

Máquinas de estado (FSM)

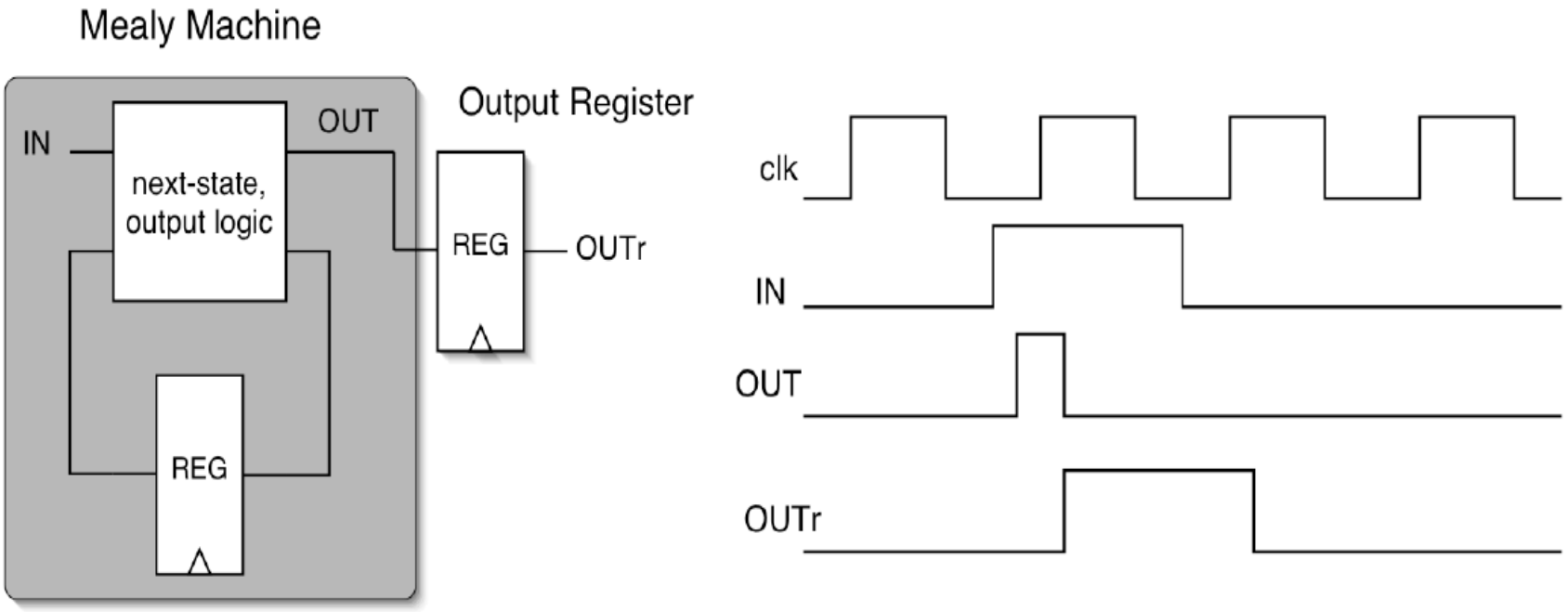
Moore Machine



Mealy Machine



Máquinas de estado (FSM): ejemplo



2.4.3. Descripción VHDL de FSM

Descripción VHDL de FSM: ejemplo secuencia “01”

Descripción de FSM MOORE

Máquina de estados en la que las salidas cambian sólo cuando cambia el estado, la salida no depende del valor de las entradas.

Estilos de descripción VHDL:

1. Dos procesos (o un proceso y sentencias concurrentes):
   - Combinacional → Salidas según estado
   - Secuencial → Estado siguiente

2. Un proceso:
   - Estado y salida

Tipo de dato enumerado definido por el usuario

- El usuario establece el NOMBRE del tipo y los elementos que lo forman
- Sintaxis:

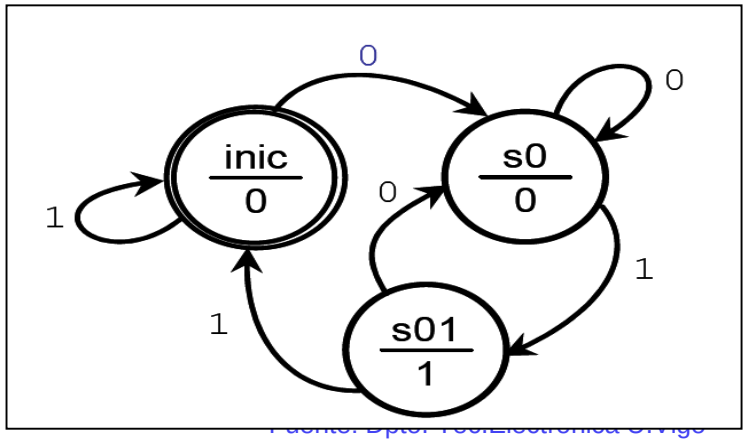
```
Type <nombre tipo de dato> is (<elementos del tipo de dato que se define>);
```

- Luego podrá utilizarse este tipo de dato con cualquier objeto

```
Signal  <nombre del dato> : < nombre tipo de dato >;
```

Descripción VHDL de FSM: ejemplo secuencia “01”

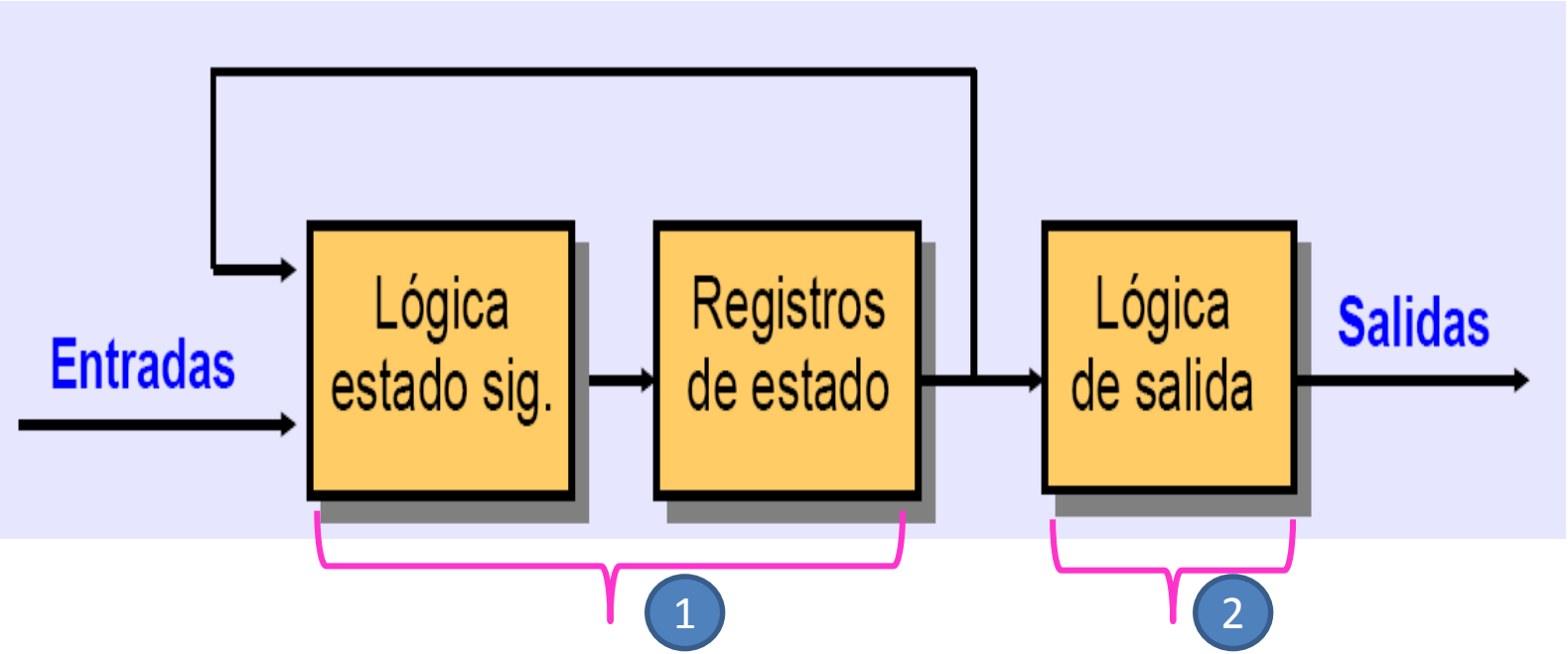
```
entity RisingEdge_FSM is
      Port ( Reset,CLK,Push : in STD_LOGIC;
             Pulse : out STD_LOGIC);
      end RisingEdge_FSM;
architecture Behavioral of RisingEdge_FSM is
-----------------------------------------------
-- USER ENUMERATED TYPE FOR FSM
-----------------------------------------------
type RisingEdge_States is (inic,S0,S01);
signal Next_State: RisingEdge_States ;

Begin
…
```



2.4.3.1. Dos procesos

Descripción VHDL de FSM: ejemplo secuencia “01”

Dos procesos



Diferenciamos dos bloques:

1- Proceso secuencial→ Estado futuro/siguiente

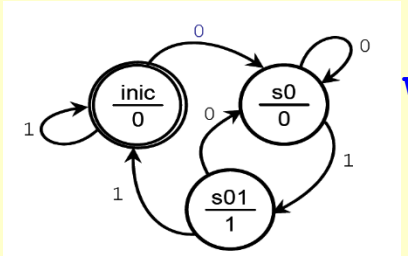
2- Proceso combinacional o Sentencias concurrentes → Valor de las salidas según el estado.

Descripción VHDL de FSM: ejemplo secuencia “01”

Proceso 1

```
Process (RESET, CLK)
 begin
        if RESET = '1' then
                Next_State<= inic; -- INICIO CON RESET
        elsif rising_edge(CLK) then
         case Next_State is
                when inic => if Push= '0' then
                                Next_State <= S0; --llega "0-"
                             end if;
                when S0 => if Push = '1' then
                                Next_State <= S01; --llega "01"
                           end if;
                when S01 => if Push = '0' then
                                 Next_State <= S0;--"0" para "0-"
                            else
                                Next_State <= inic; --llega un "1",
                            end if;
                when others => Next_State <= inic;
         end case;
        end if;
end process;
```

Estado Actual
Entradas

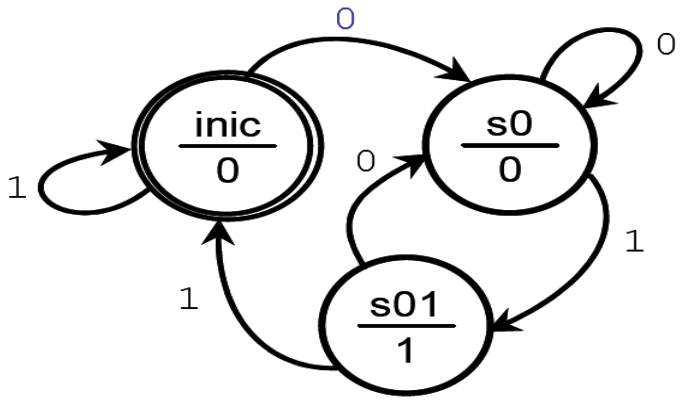


Descripción VHDL de FSM: ejemplo secuencia “01”

Dos procesos: Sentencias concurrentes

“Proceso 2”

```
-------------------------------------------------
-- TWO PROCESS: CONCURRENTS ASSIGNMENTS FOR SET OUTPUTS
-------------------------------------------------- ---------
with Next_State select
      Pulse <= '0'      when inic,
               '0'      when S0,
               '1'      when S01,
               '0'      when others; -- catch all end
Behavioral;
```

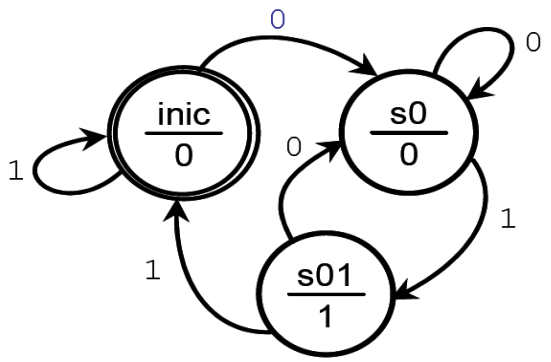


Descripción VHDL de FSM: ejemplo secuencia “01”

Dos procesos: Proceso combinacional

Proceso 2

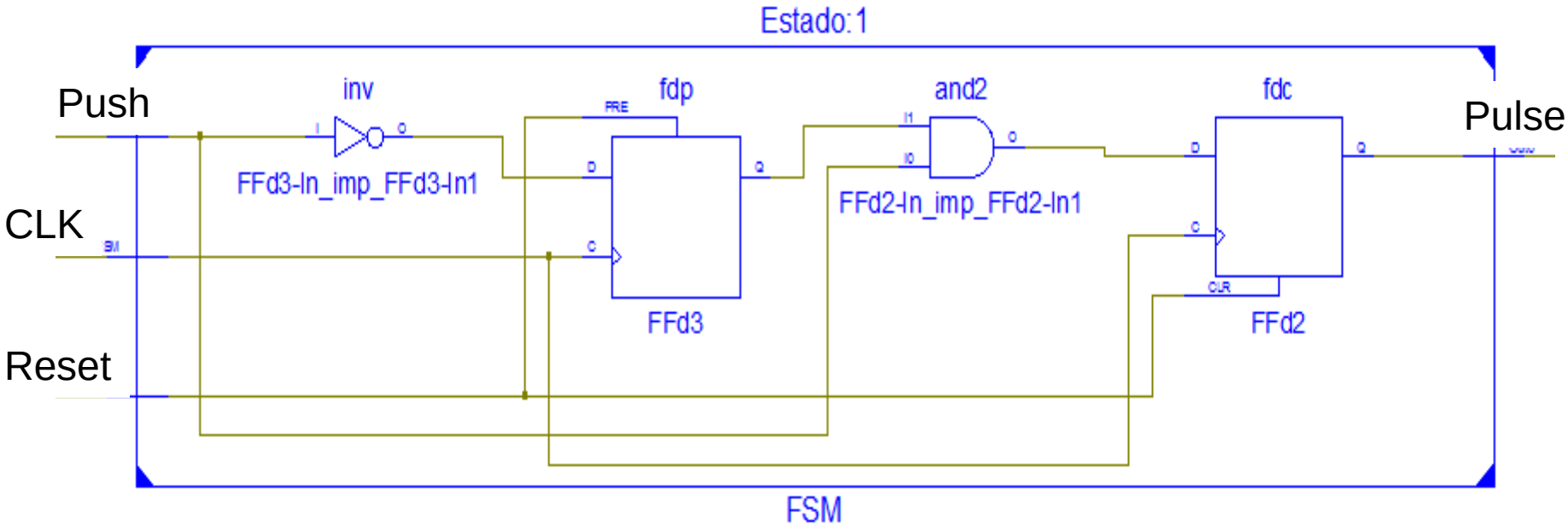
```
-------------------------------------------------
-- TWO PROCESS: COMBINATIONAL PROCESS FOR SET OUTPUTS
-------------------------------------------------
process (Next_State)
 begin
       case Next_State is
             when inic     => Pulse <= '0';
             when S0       => Pulse <= '0';
             when S01      => Pulse <= '1';
             when others   => Pulse <= '0';
       end case;
 end process;
```



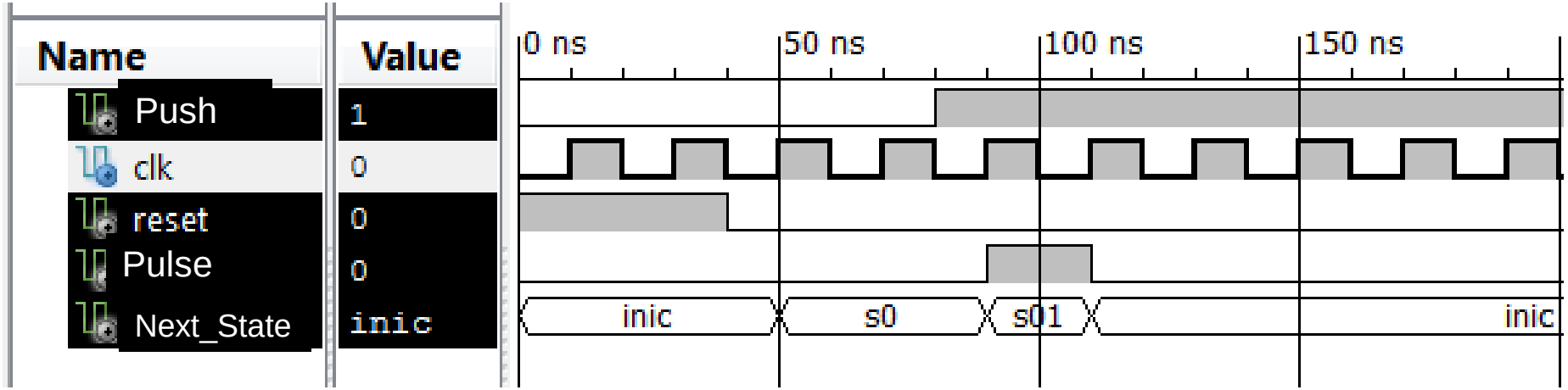
Descripción VHDL de FSM: ejemplo secuencia “01”

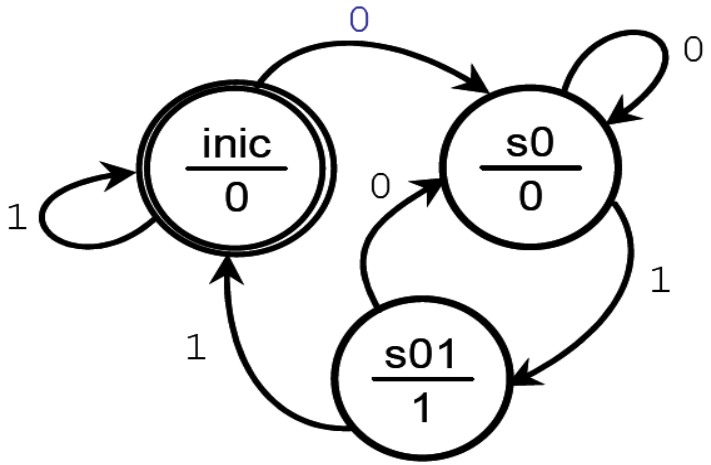
```
=========================================================================
*                            HDL Synthesis
=========================================================================

Performing bidirectional port resolution...

Synthesizing Unit <DetectorFlanco_FSM>.
    Related source file is "C:/PoyectosTDC/Proyecto29/DetectorFla
    Found finite state machine <FSM_0> for signal <Estado>.
    -----------------------------------------------------------------
    | States             | 3
    | Transitions        | 6
    | Inputs             | 1
    | Outputs            | 1
    | Clock              | CLK                          (rising_edge
    | Reset              | RESET                        (positive)
    | Reset type         | asynchronous
    | Reset State        | zero
    | Power Up State     | zero
    | Encoding           | automatic
    | Implementation     | LUT
    -----------------------------------------------------------------
    Summary:
        inferred   1 Finite State Machine(s).
Unit <DetectorFlanco_FSM> synthesized.
```

Descripción VHDL de FSM: ejemplo secuencia “01”



Descripción VHDL de FSM: ejemplo secuencia “01”

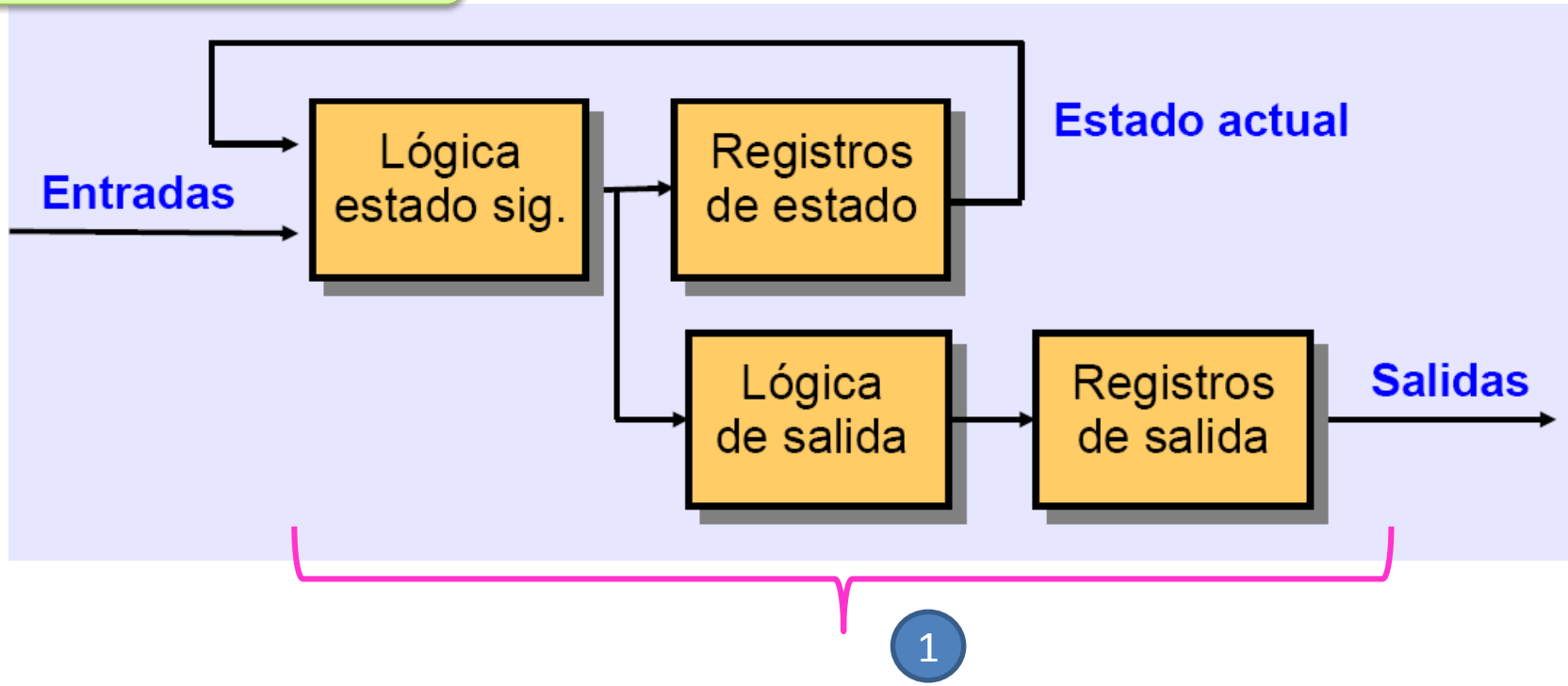




2.4.3.2. Un proceso

Descripción VHDL de FSM: ejemplo secuencia “01”

Un proceso



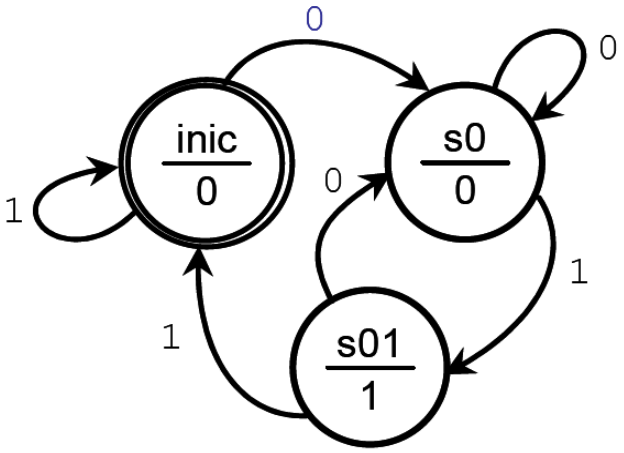
Diferenciamos un bloque:

- Proceso secuencial→ Estado futuro/siguiente + Salidas Registradas

Descripción VHDL de FSM: ejemplo secuencia “01”

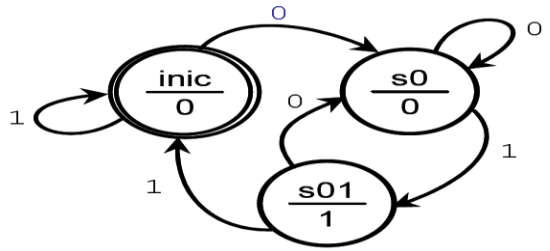
Un proceso

```
MEMORIA: Process (RESET, CLK)
begin

if RESET = '1' then
    Next_State<= inic; -- INICIO CON RES
    Pulse<='0';

elsif CLK'event and CLK='1' then
    case Next_State is
    when inic =>
        if Push= '0’ then
                Next_State <= S0; --llega “0-"
                Pulse <='0'; --Salida del estado S0
        else
                Next_State <= inic; -- esperando un "0-"
                Pulse <='0'; --Salida del estado inic
        end if;
```



Descripción VHDL de FSM: ejemplo secuencia “01”

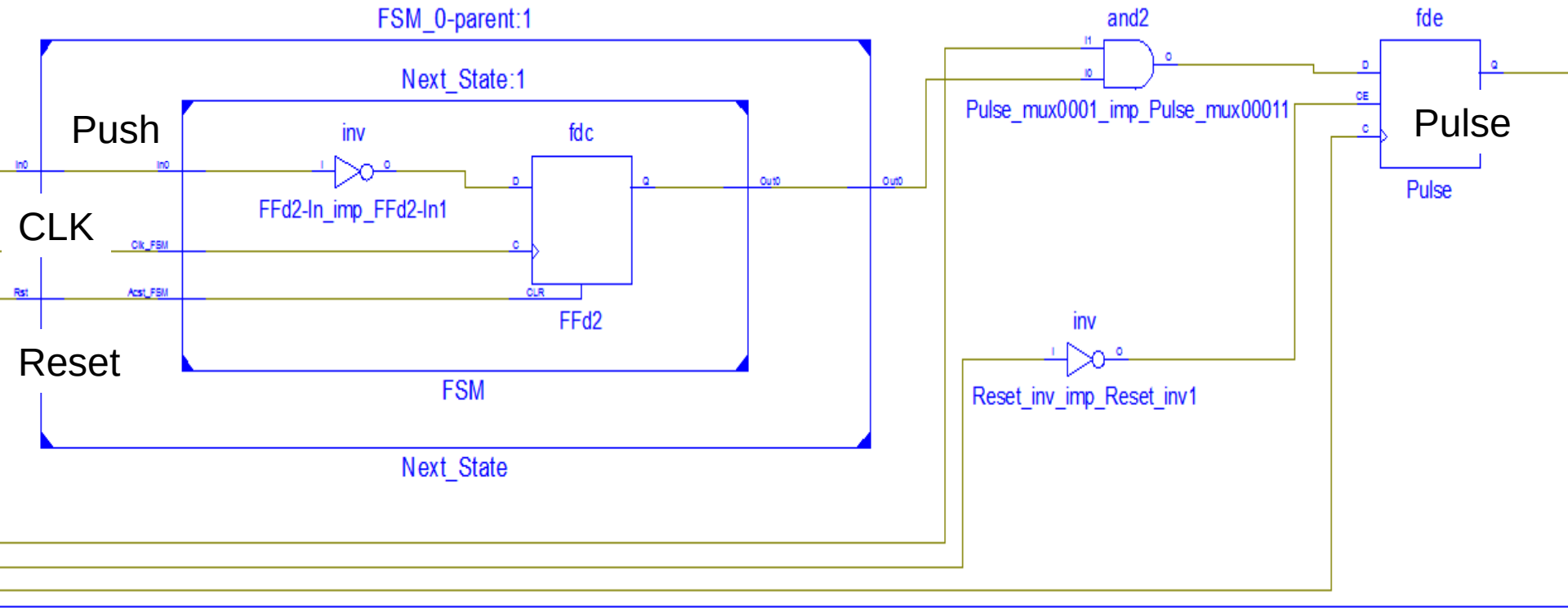
```
when S0 =>
        if Push= '1' then
                Next_State <= S01; --llegó "01"
                Pulse<='1'; --Salida del estado S01
        else
                 Next_State <= S0;
                 Pulse <='0'; --Salida del estado S0
        end if;
when S01 =>
        if Push= '0' then
                 Next_State <= S0;      --llega "0" para "0-"
                 Pulse <='0'; --Salida del estado S0
        else
                 Next_State <= inic; -- "1", espera "0"
                 Pulse<='0'; --Salida del estado inic
        end if;
when others => Next_State <= Inic;
               Pulse <='0';
end case;
end if;
end process;
end behavioral;
```



Descripción VHDL de FSM: ejemplo secuencia “01”

```
Synthesizing Unit <DetectorFlanco_FSM_Reg>.
    Related source file is "C:/PoyectosTDC/Proyecto29/Detec
    Found finite state machine <FSM_0> for signal <Estado>.
    -----------------------------------------------------
    | States             | 3                              
    | Transitions        | 6                              
    | Inputs             | 1                              
    | Outputs            | 1                              
    | Clock              | CLK                      (risin
    | Reset              | RESET                    (posit
    | Reset type         | asynchronous                   
    | Reset State        | zero                           
    | Power Up State     | zero                           
    | Encoding           | automatic                      
    | Implementation     | LUT                            
    -----------------------------------------------------
    Found 1-bit register for signal <Flanco>.
    Summary:
        inferred   1 Finite State Machine(s).
        inferred   1 D-type flip-flop(s).
Unit <DetectorFlanco_FSM_Reg> synthesized.
```

Descripción VHDL de FSM: ejemplo secuencia “01”



Descripción VHDL de FSM: ejemplo secuencia “01”