

LCSE. Diseño de Circuitos Síncronos

Laboratorio de Diseño de Circuitos y
Sistemas Electrónicos

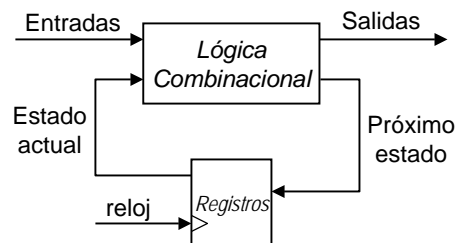
Curso 2005-2006

DIE-LSI

M.L. López-Vallejo

Sistemas secuenciales síncronos

- ✓ Sistema en el que las salidas dependen de entradas previas y actuales
- ✓ **Síncrono**: el estado siguiente se actualiza cuando cambia la señal de reloj



DIE-LSI

M.L. López-Vallejo



Ventajas de los sistemas síncronos

- ✓ Se muestrea las señales en instantes de tiempo conocidos y bien definidos
- ✓ Inmune a *glitches*
- ✓ Se simplifican los problemas de variación de retardos en los diferentes caminos de lógica
- ✓ Estable frente a cambios de temperatura, tensión o proceso
- ✓ Interfaces sencillas
- ✓ Gran disponibilidad de herramientas de síntesis y simulación

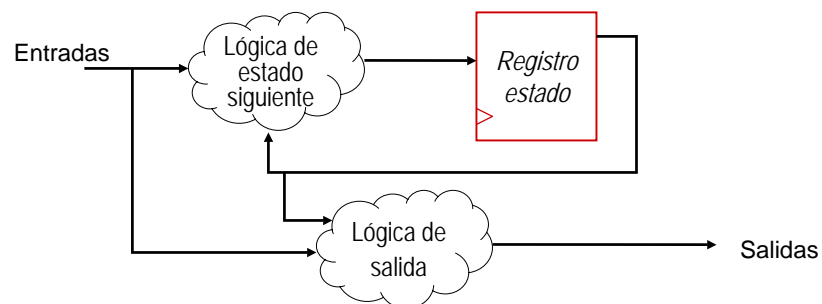
DIE-LSI

M.L. López-Vallejo

Máquinas de estados finitos

Tres bloques típicos:

- Lógica de estado siguiente
- Registro de estado actual
- Lógica de salida

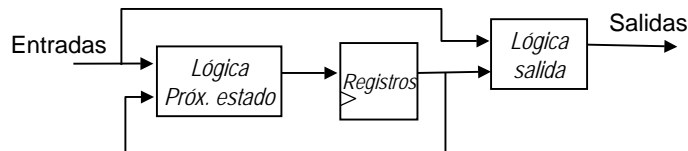


DIE-LSI

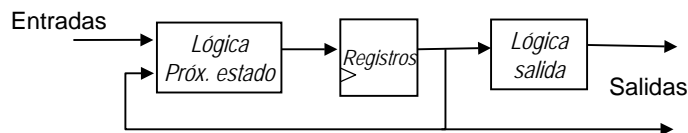
M.L. López-Vallejo

Máquinas de estados finitos

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



- ✓ Moore: las salidas son función del estado actual

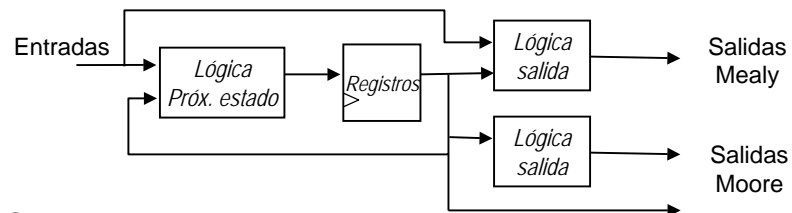


DIE-LSI

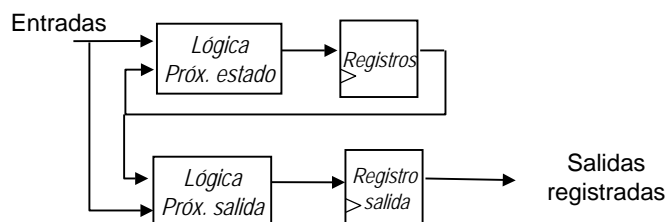
M.L. López-Vallejo

Máquinas de estados finitos

- ✓ Mealy/Moore combinadas



- ✓ Con salidas registradas



DIE-LSI

M.L. López-Vallejo

Diseño mediante VHDL

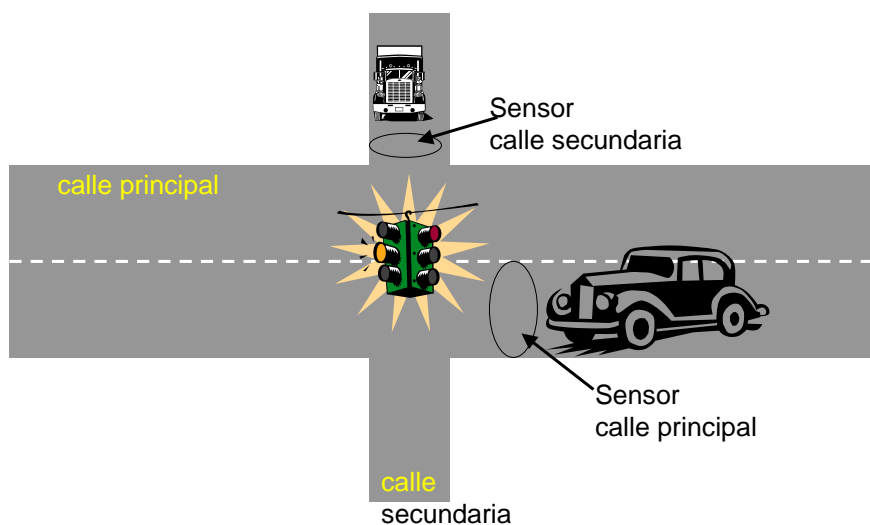
1. Realización del diagrama de estados
2. Codificación de estados: creación de un tipo (type)
3. Realización de la lógica de “Próximo estado”
4. Registro del vector “Estado actual”
5. Realización de la lógica de generación de salidas



DIE-LSI

M.L. López-Vallejo

Controlador de un semáforo

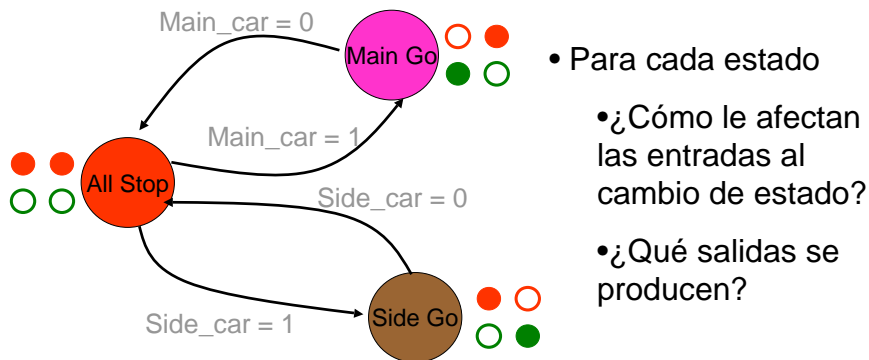


DIE-LSI

M.L. López-Vallejo

Diseño del control

- ✓ Entradas, salidas, estados, transiciones



DIE-LSI

M.L. López-Vallejo

Codificación de estados

- ✓ Es la forma en que se asignan números binarios a estados
- ✓ En VHDL podemos utilizar un tipo *enumerado* con lo que se deja a la herramienta de síntesis decidir la codificación

Binario	Gray	One hot
000	000	00000001
001	001	00000010
010	011	00000100
011	010	00001000
100	110	00010000
101	111	00100000
110	101	01000000
111	100	10000000

DIE-LSI

M.L. López-Vallejo

enum_encoding

- ✓ Atributo para especificar los valores de cada estado toma en el proceso de síntesis.

```
architecture Encl of StateMachine is
  type State is
    ( Idle, S1, S2, S3, S4, S5 );
  signal PresentState, NextState : State;
begin
  . . .
end architecture Encl;
```

- ✓ Se puede definir en un package

```
package estados is
  type tipo_estado is (Q0,Q1,Q2,Q3)
  attribute enum_encoding : string;
  attribute enum_encoding of tipo_estado is "00 01 10 11"
end estados;
```

DIE-LSI

M.L. López-Vallejo

Tipo enumerado

- ✓ Atributo para especificar los valores de cada estado toma en el proceso de síntesis.

```
architecture Encl of StateMachine is
  type State is
    ( Idle, S1, S2, S3, S4, S5 );
  signal PresentState, NextState : State;
begin
  . . .
end architecture Encl;
```

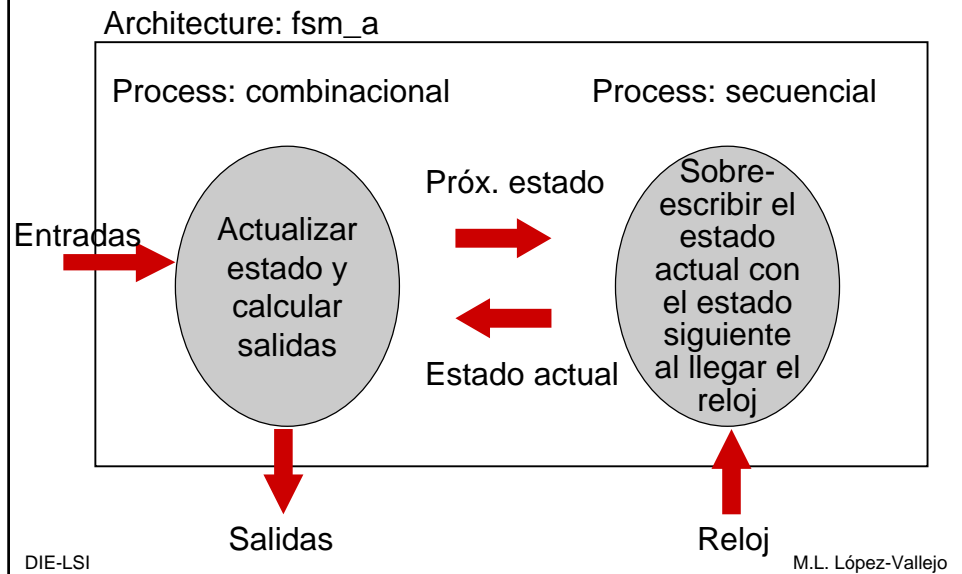
- Se puede definir en un package

```
package estados is
  type tipo_estado is (Q0,Q1,Q2,Q3)
end estados;
```

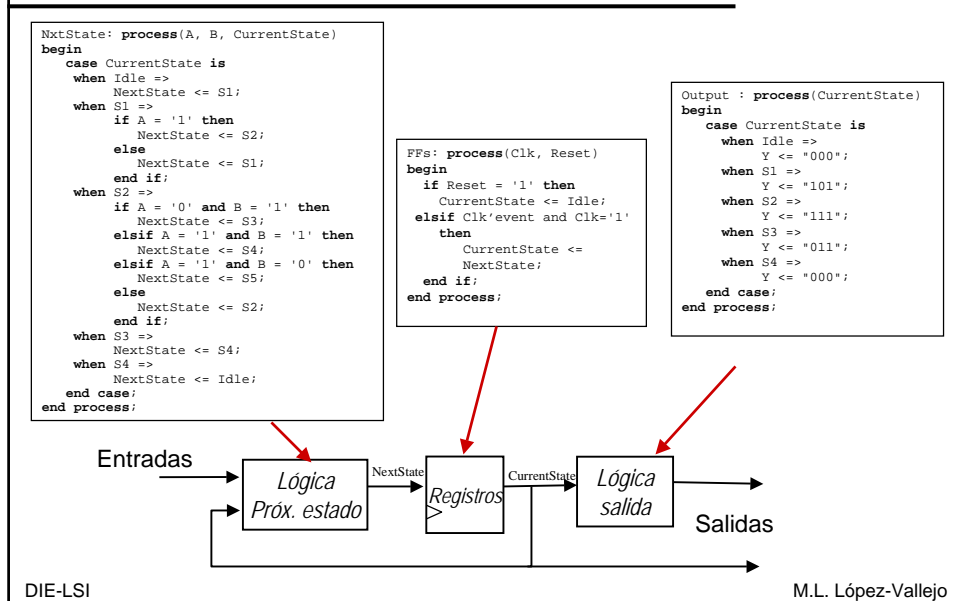
DIE-LSI

M.L. López-Vallejo

Máquinas de estados en VHDL



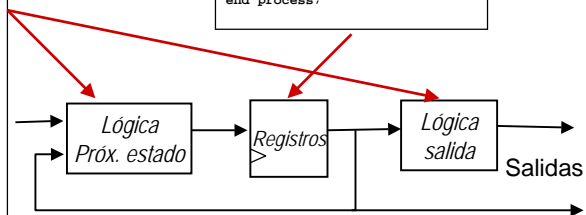
Máquina de estados de Moore



Máquina de estados de Moore, v.2

```
NxtState: process(A, B, CurrentState)
begin
  case CurrentState is
    when Idle =>
      NextState <= S1;
      Y <= "000";
    when S1 =>
      if A = '1' then
        NextState <= S2;
      else
        NextState <= S1;
      end if;
      Y <= "101";
    when S2 =>
      if A = '0' and B = '1' then
        NextState <= S3;
      elsif A = '1' and B = '1' then
        NextState <= S4;
      elsif A = '1' and B = '0' then
        NextState <= S5;
      else
        NextState <= S2;
      end if;
      Y <= "111";
    when S3 =>
      NextState <= S4;
      Y <= "011";
    when S4 =>
      NextState <= Idle;
      Y <= "000";
  end case;
end process;
```

```
FFs: process(Clk, Reset)
begin
  if Reset = '1' then
    CurrentState <= Idle;
  elsif clk'event and Clk = '1'
  then
    CurrentState <=
      NextState;
  end if;
end process;
```



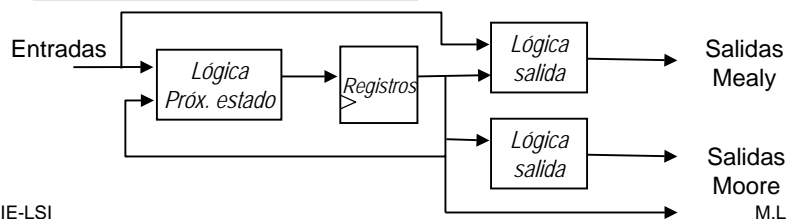
DIE-LSI

M.L. López-Vallejo

Mealy y Moore combinadas

```
process(Car, Timed, CurrentState)
begin
  StartTimer <= '0'; -- Default output
  case CurrentState is
    when S1 =>
      MajorGreen <= '1';
      MinorGreen <= '0';
      if Car = '1' then
        StartTimer <= '1'; -- Mealy output
        NextState <= S2;
      else
        NextState <= S1;
      end if;
    when S2 =>
      MajorGreen <= '0';
      MinorGreen <= '1';
      if Timed = '1' then
        NextState <= S1;
      else
        NextState <= S2;
      end if;
  end case;
end process;
```

```
FFs: process(Clk, Reset)
begin
  if Reset = '1' then
    CurrentState <= Idle;
  elsif clk'event and Clk = '1'
  then
    CurrentState <=
      NextState;
  end if;
end process;
```



DIE-LSI

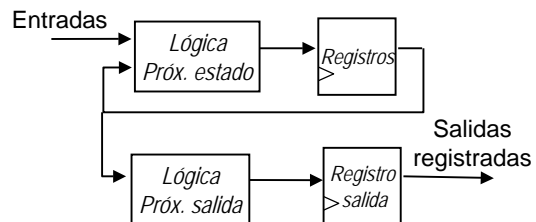
M.L. López-Vallejo

Con salidas registradas

```
NxtState: process(A, B, CurrentState)
begin
  case CurrentState is
    when Idle =>
      NextState <= S1;
    when S1 =>
      if A = '1' then
        NextState <= S2;
      else
        NextState <= S1;
      end if;
    when S2 =>
      if A = '0' and B = '1' then
        NextState <= S3;
      elsif A = '1' and B = '1' then
        NextState <= S4;
      elsif A = '1' and B = '0' then
        NextState <= S5;
      else
        NextState <= S2;
      end if;
    when S3 =>
      NextState <= S4;
    when S4 =>
      NextState <= Idle;
    end case;
  end process;
```

```
FFs: process(Clk, Reset)
begin
  if Reset = '1' then
    CurrentState <= Idle;
  elsif Clk'event and Clk='1'
    then
      CurrentState <=
        NextState;
    end if;
  end process;
```

```
Output : process(Clk, Reset)
begin
  if Reset = '1' then
    Y <= "000";
  elsif Clk'event and Clk='1'
    then
      case CurrentState is
        when Idle =>
          Y <= "000";
        when S1 =>
          Y <= "101";
        when S2 =>
          Y <= "111";
        when S3 =>
          Y <= "011";
        when S4 =>
          Y <= "000";
        end case;
      end process;
```



DIE-LSI

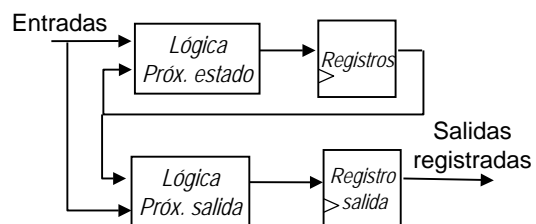
M.L. López-Vallejo

Con salidas registradas

```
NxtState: process(A, B, CurrentState)
begin
  case CurrentState is
    when Idle =>
      NextState <= S1;
    when S1 =>
      if A = '1' then
        NextState <= S2;
      else
        NextState <= S1;
      end if;
    when S2 =>
      if A = '0' and B = '1' then
        NextState <= S3;
      elsif A = '1' and B = '1' then
        NextState <= S4;
      elsif A = '1' and B = '0' then
        NextState <= S5;
      else
        NextState <= S2;
      end if;
    when S3 =>
      NextState <= S4;
    when S4 =>
      NextState <= Idle;
    end case;
  end process;
```

```
FFs: process(Clk, Reset)
begin
  if Reset = '1' then
    CurrentState <= Idle;
  elsif Clk'event and Clk='1'
    then
      CurrentState <=
        NextState;
    end if;
  end process;
```

```
Output : process(Clk, Reset)
begin
  if Reset = '1' then
    Y <= "000";
  elsif Clk'event and Clk='1'
    then
      case NextState is
        when Idle =>
          Y <= "000";
        when S1 =>
          Y <= "101";
        when S2 =>
          Y <= "111";
        when S3 =>
          Y <= "011";
        when S4 =>
          Y <= "000";
        end case;
      end process;
```



DIE-LSI

M.L. López-Vallejo