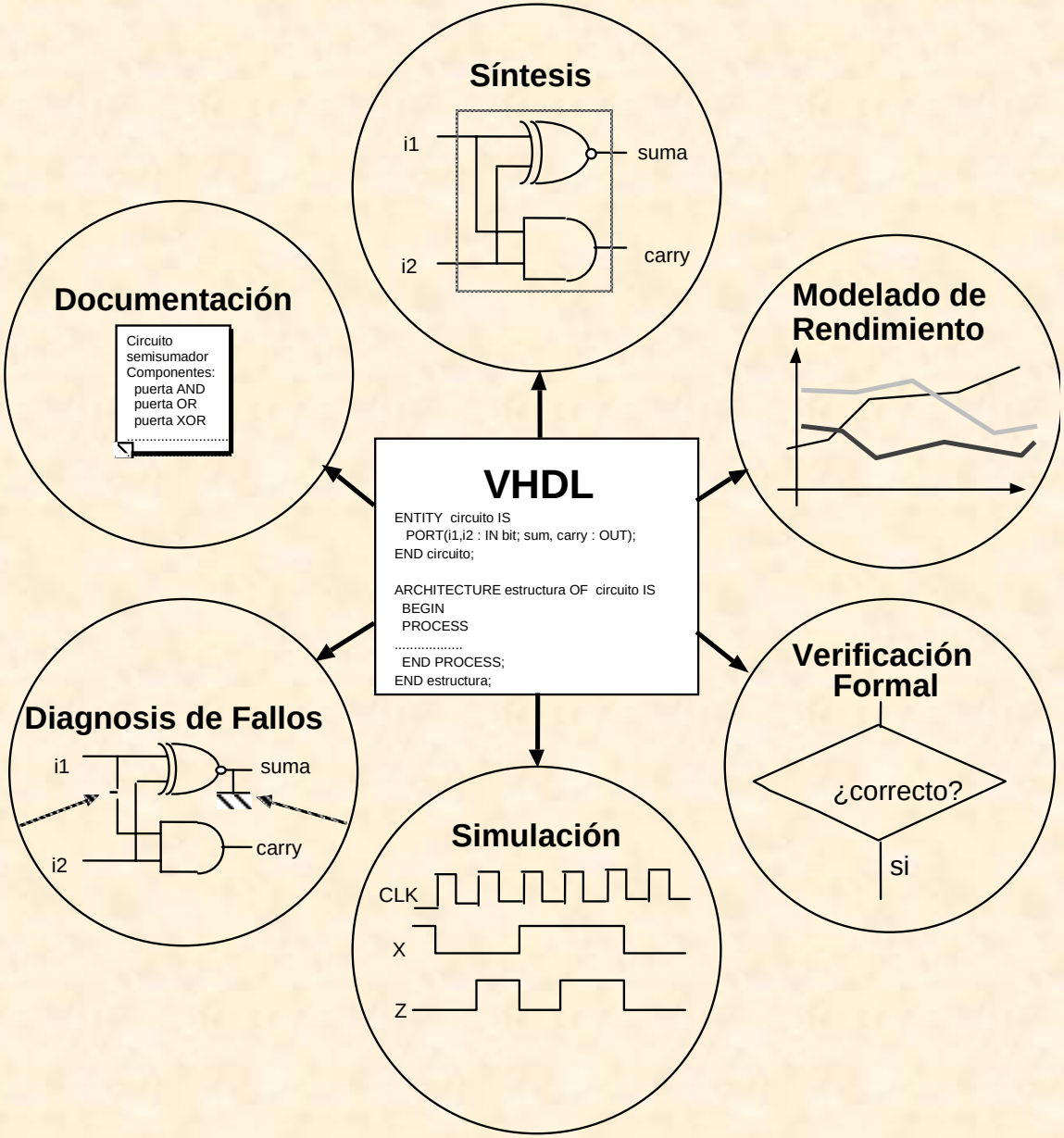
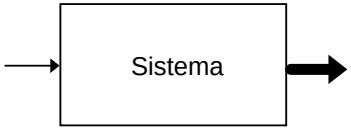
Sesión 1: Introducción al lenguaje VHDL

AREAS DE APLICACION DE VHDL

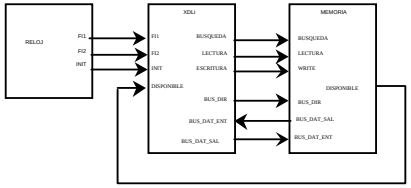


NIVELES DE RESOLUCION DE UN SISTEMA DIGITAL

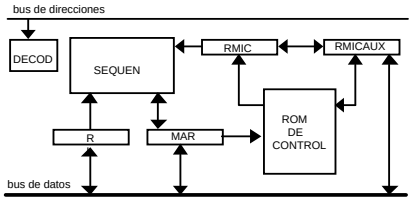
Sistema



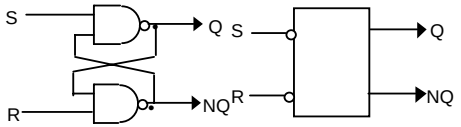
Chip



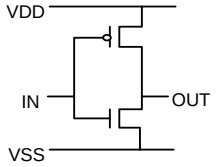
Registros



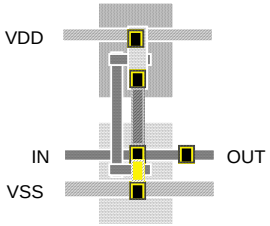
Puertas



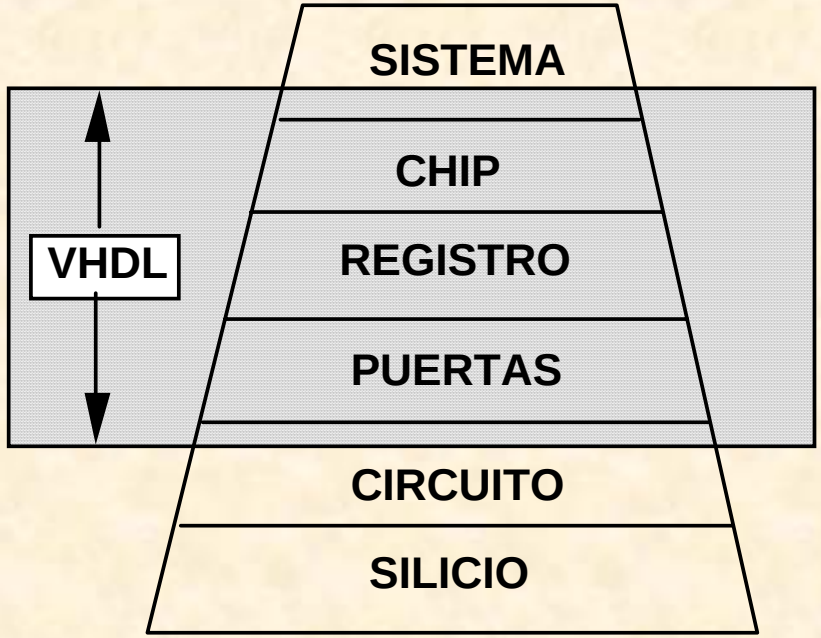
Circuito



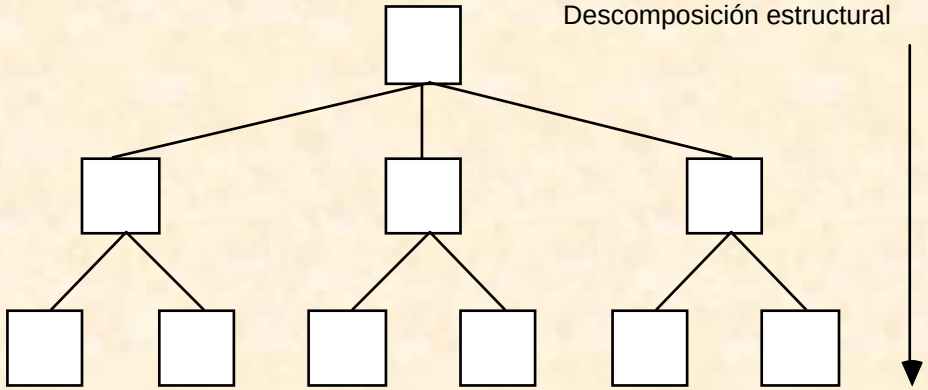
Silicio



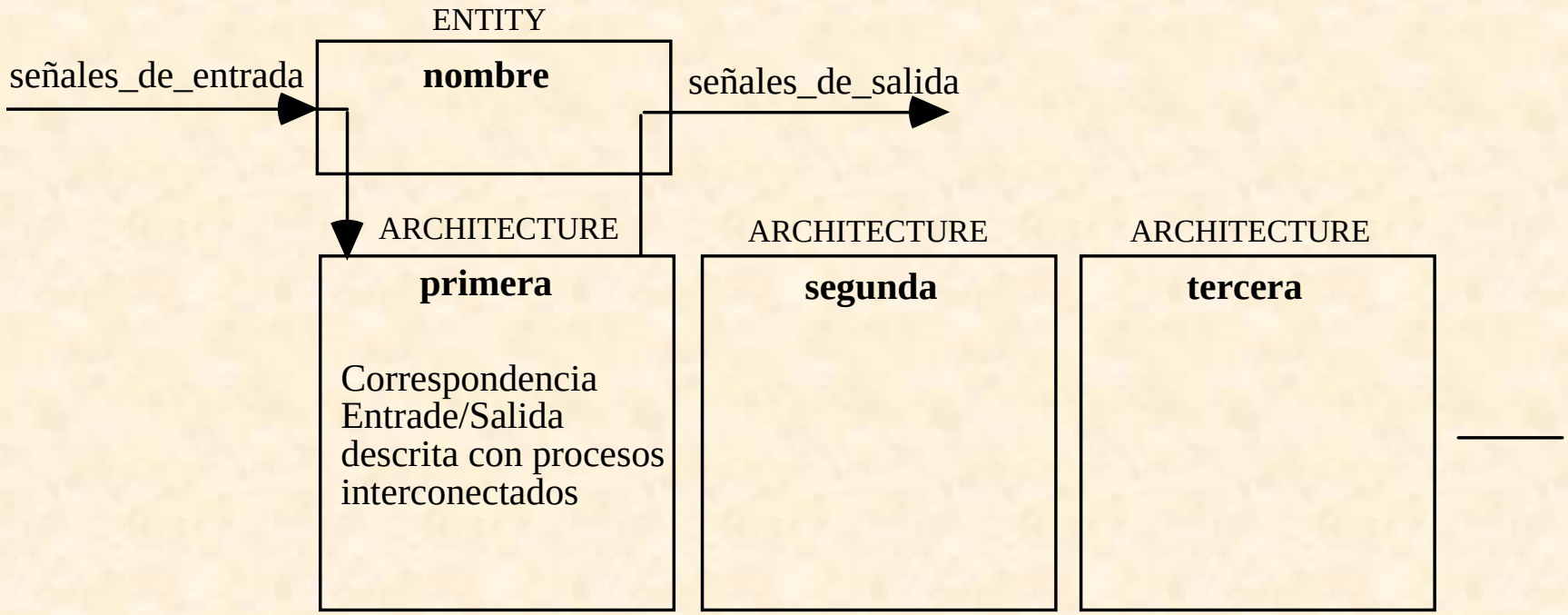
CAPACIDAD EXPRESIVA DE VHDL



DESCOMPOSICION ESTRUCTURAL DEL DISEÑO



UNIDAD DE DISEÑO EN VHDL



UNIDAD DE DISEÑO EN VHDL: SINTAXIS

```
ENTITY  nombre  IS
        PORT ( señales_de_entrada : IN tipo; señales_de_salida : OUT tipo);
END  nombre;
```

```
ARCHITECTURE  primera  OF  nombre  IS
 BEGIN
  PROCESS
```

Declaraciones

```
  BEGIN
```

Cuerpo del Proceso

(Algorítmo Secuencial)

```
  END PROCESS;
END primera;
```

SENTENCIAS SECUENCIALES DE VHDL

1) Sentencia de asignación de variable

```
variable := expresión;
```

2) Sentencias de control del flujo de ejecución

a) Condicional

```
IF condición THEN sentencias_1 ELSE sentencias_2;
```

b) Alternativa

```
CASE expresión IS
        WHEN valor_1 => sentencias_1;
        WHEN valor_2 => sentencias_2;
        ---------------------------------------
        WHEN valor_n => sentencias_n;
END CASE;
```

c) Bucle iterativo

```
FOR índice IN rango LOOP sentencias END LOOP;
```

3) Sentencias de interacción con el exterior

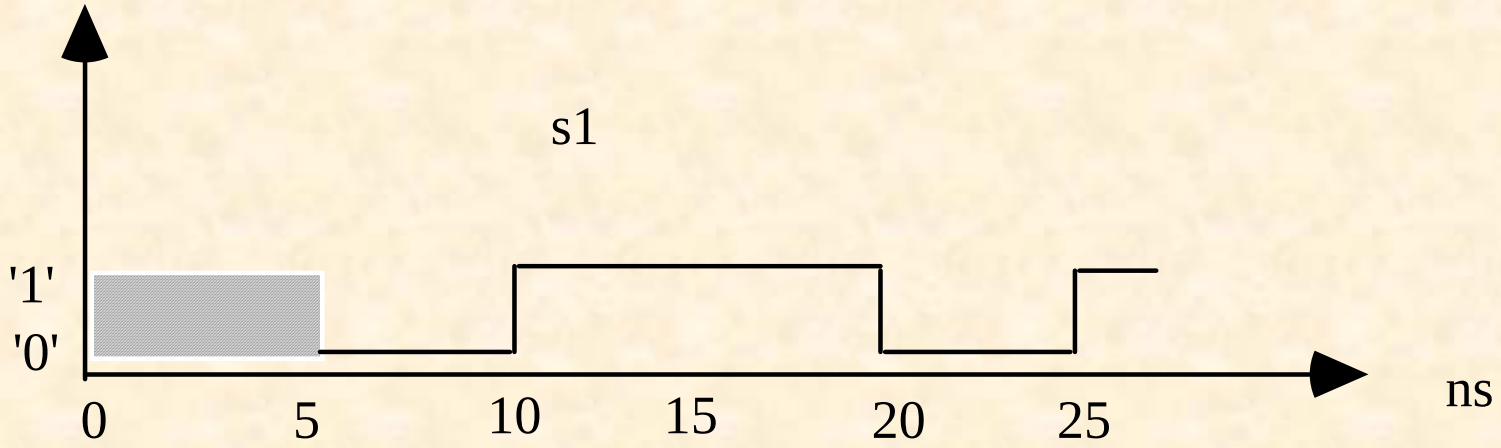
a) Asignación de señal

```
señal <= expresión AFTER retardo;
```

b) Espera de eventos

```
WAIT ON lista_de_señales;
```

CONCEPTO BASICO DE SEÑAL

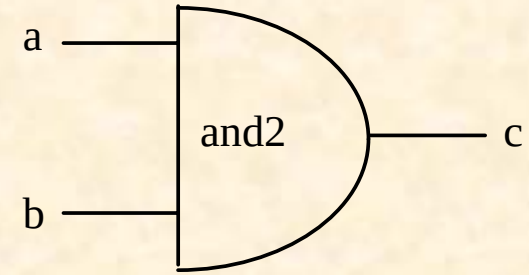


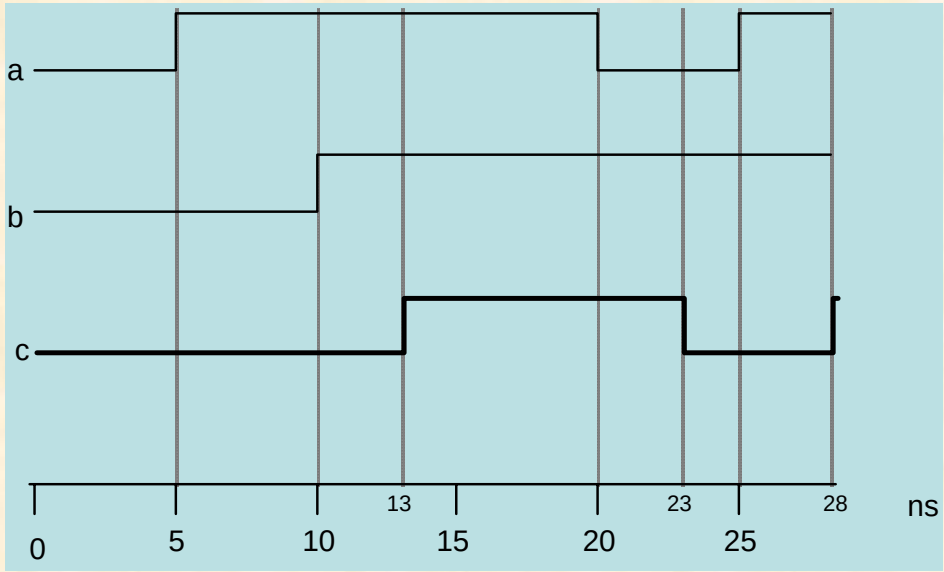
driver de la señal s1 : {('0', 5 ns), ('1', 10 ns), ('0', 20 ns, ('1', 25 ns)}

EJEMPLO 1: Puerta *and* de dos entradas

```
ENTITY  and2  IS
        PORT (a, b : IN bit;  c : OUT bit);
END and2;
```

```
ARCHITECTURE primera OF and2 IS
BEGIN
        PROCESS
        BEGIN
                c <= a AND b AFTER 3 ns;
                WAIT ON a, b;
        END PROCESS;
END primera;
```

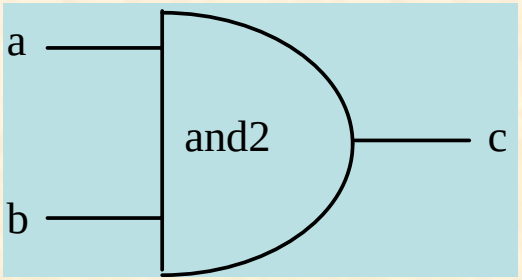


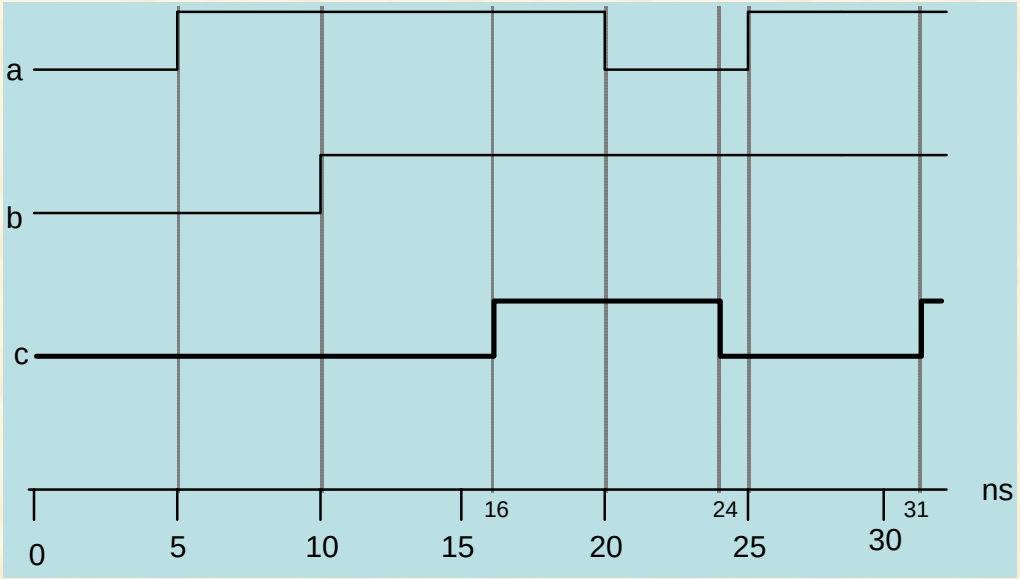


ns	a	b	c
0	0	0	**0**
5	1	0	**0**
10	1	1	**0**
13	1	1	**1**
20	0	1	**1**
23	0	1	**0**
25	1	1	**0**
28	1	1	**1**

SEGUNDA ARQUITECTURA para la Puerta *and* de dos entradas

```
ARCHITECTURE segunda OF and2 IS
BEGIN
        PROCESS
                VARIABLE  temp  :  BIT;
        BEGIN
                temp :=  a  AND  b;
                IF  (temp = '1')  THEN
                        c <=  temp  AFTER 6 ns;
                ELSE
                        c <=  temp  AFTER 4 ns;
                END IF;
                WAIT ON a, b;
        END PROCESS;
END segunda;
```



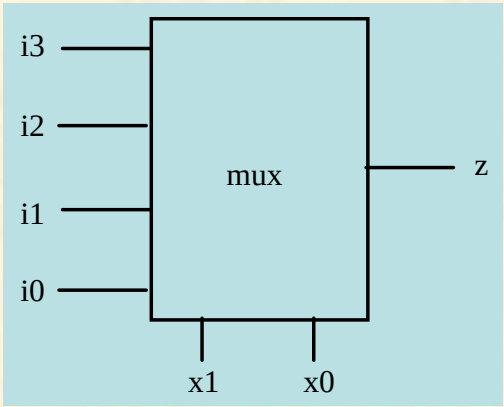


ns	a	b	**c**
0	0	0	**0**
5	1	0	**0**
10	1	1	**0**
16	1	1	**1**
20	0	1	**1**
24	0	1	**0**
25	1	1	**0**
31	1	1	**1**

EJEMPLO 2: Multiplexor

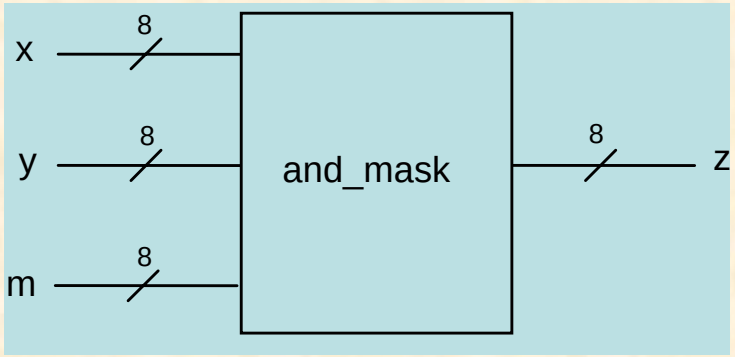
```
ENTITY  mux  IS
            PORT ( i3, i2, i1, i0, x1, x0  :  IN  BIT;  z  :  OUT  BIT );
END  mux;
```

```
ARCHITECTURE  mux  OF  mux IS
BEGIN
  PROCESS
   TYPE  peso  IS  RANGE  0  TO  3;
   VARIABLE  muxval  :  peso;
  BEGIN
   muxval  :=  0;
   IF  (x0 = '1')  THEN  muxval  :=  muxval  +  1;  END  IF;
   IF  (x1 = '1')  THEN  muxval  :=  muxval  +  2;  END  IF;
               CASE  muxval  IS
                          WHEN  0  =>
                                      z <= i0  AFTER  10 ns;
                          WHEN  1  =>
                                      z <= i1  AFTER  10  ns;
                          WHEN  2  =>
                                      z <= i2  AFTER  10  ns;
                          WHEN  3  =>
                                      z <= i3  AFTER  10 ns;
               END CASE;
  WAIT ON i3, i2, i1, i0, x1, x0;
  END PROCESS;
END mux;
```



EJEMPLO 3: Circuito *and* con máscara.

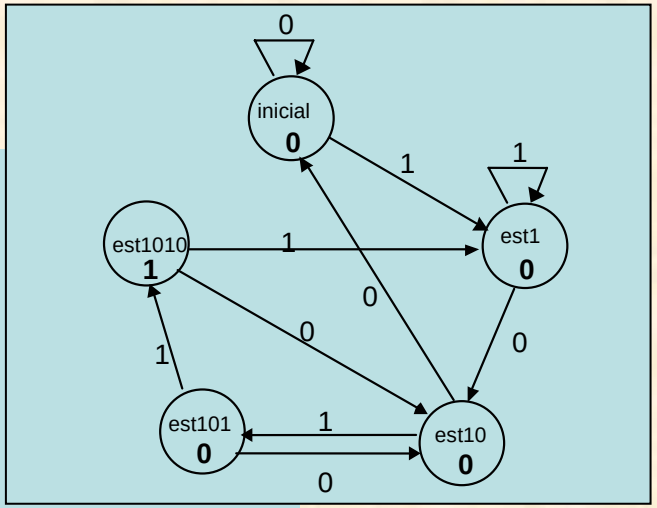
```
ENTITY  and_mask  IS
    PORT ( x, y, m  :  IN  bit_vector ( 7 DOWNTO  0);
                  z : OUT  bit_vector ( 7  DOWNTO  0));
END  and_mask;
```




```
ARCHITECTURE  and_mask  OF  and_mask  IS
BEGIN
 PROCESS
 BEGIN
            FOR  i  IN  0  TO  7  LOOP
                        IF  ( m(i) = '0')  THEN  NEXT;
                        ELSE
                                    z(i) <= x(i) AND y(i);
                        END IF;
             END  LOOP;
            WAIT ON x, y, m;
 END PROCESS;
END and_mask;
```

EJEMPLO 4: Máquina de Estados Finitos.

```
ENTITY mef IS
        PORT(x, reloj : IN BIT; z : OUT BIT);
END mef;
```




```
ARCHITECTURE unica OF mef IS
BEGIN
  PROCESS
  TYPE estados IS (inicial, est1, est10, est101, est1011);
  VARIABLE estado : estados := inicial;
  BEGIN
    WAIT ON reloj;
    IF reloj = '1' THEN
       CASE estado IS
          WHEN inicial =>
                IF    x = '1' THEN  estado := est1; ELSE estado := estado;  END IF;
                z <= '0';
          WHEN est1 =>
                IF    x = '1' THEN estado := est1;  ELSE estado := est10;  END IF;
                z <= '0';
          WHEN est10 =>
                IF    x = '1' THEN estado := est101; ELSE estado := inicial;  END IF;
                z <= '0';
          WHEN est101 =>
                IF    x = '1' THEN estado := est1011; ELSE estado := est10; END IF;
                z <= '0';
         WHEN est1011 =>
               IF x = '1'   THEN estado := est1; ELSE estado := est10; END IF;
               z <= '1';
       END CASE;
    END IF;
  END PROCESS;
END unica;
```

Practica 1: Conceptos básicos de VHDL

Objetivos

1. Utilización de las construcciones secuenciales básicas de VHDL.
2. Modelado del comportamiento de pequeños sistemas combinacionales y secuenciales.
2. Familiarización con el entorno VHDL para compilar, depurar, simular y trazar programas.

Práctica a realizar

Codificación de un programa VHDL *(entre 30 y 40 líneas de código) que utilice las sentencias secuenciales estudiadas hasta el momento.* El programa deberá modelar el comportamiento de un sistema digital previamente especificado (decodificador, decodificador con prioridad, codificador, demultiplexor, operador aritmético, máquina secuencial, etc)

Resultados a entregar

Documentación del programa en la que conste:

1. *Especificación* del sistema modelado.

2. *Listado* VHDL comentado del código del programa.

3. *Diagramas de tiempo* que muestren la respuesta del sistema frente a entradas significativas.