

CAPITULO 3: LENGUAJE DE DESCRIPCIÓN DE HARDWARE VHDL

EJEMPLOS EN VHDL

EJERCICIO # 1:

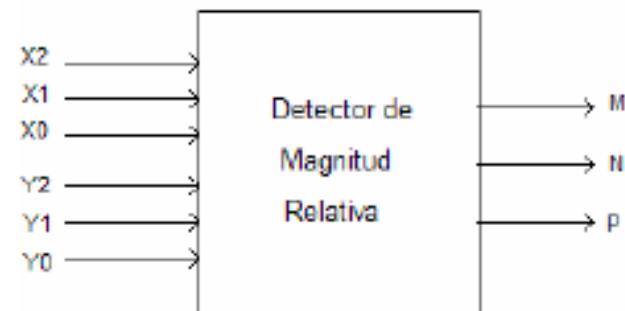
La figura representa un detector de magnitud relativa que toma dos números binarios (STD_LOGIC) de 3 bits, **X** ($X_2X_1X_0$) y **Y** ($Y_2Y_1Y_0$), y determina si son iguales y, si no lo son, cual de ellos es el mayor.

Hay 3 salidas que se definen como sigue:

$M=1$ sólo si los dos números de entrada son iguales.

$N=1$ sólo si **X** es mayor que **Y**

$P=1$ sólo si **Y** es mayor que **X**.



EJERCICIO 1: SOLUCIÓN

```
library ieee;
use ieee.std_logic_1164.all;

ENTITY eje1 IS

PORT(   X      : IN      STD_LOGIC_VECTOR(2 downto 0);
        Y      : IN      STD_LOGIC_VECTOR(2 downto 0);
        M,N,P   : OUT     STD_LOGIC);

END eje1;

ARCHITECTURE sol OF eje1 IS

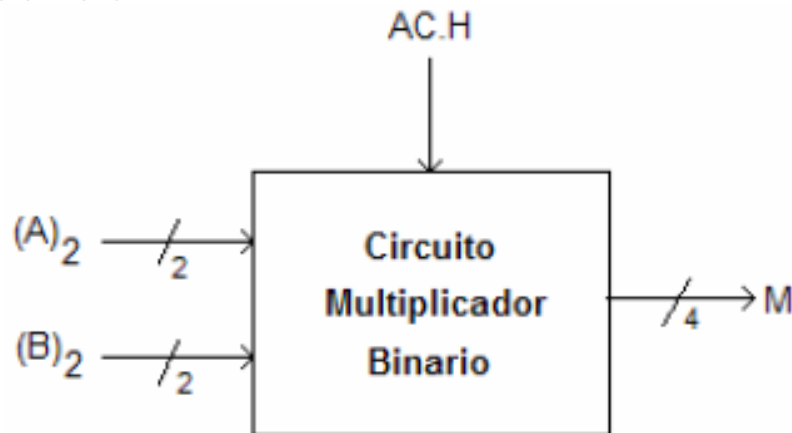
BEGIN
    M <= '1' when X=Y else '0';
    N <= '1' when X>Y else '0';
    P <= '1' when X<Y else '0';

END sol;
```

EJERCICIO # 2

Diseñar un circuito multiplicador Binario en VHDL utilizando la descripción RTL. El circuito multiplicador binario tiene 2 entradas de datos binarios, A y B de 2 dígitos cada una y la entrada habilitadora AC. Además el circuito tiene una salida M de 4 bits. El circuito funcionará según las siguientes especificaciones:

- Si $AC.H = H$ el circuito se encuentra activado y presenta en la salida M el producto matemático de $A \times B$.
- Si $AC.H = L$ el circuito se encuentra desactivado y presenta en la salida un nivel de alta impedancia.



EJERCICIO 2: Tabla de Verdad

AC	A1	A0	B1	B0	M3	M2	M1	M0
0	0	0	0	0	Z	Z	Z	Z
0	0	0	0	1	Z	Z	Z	Z
0	0	0	1	0	Z	Z	Z	Z
0	0	0	1	1	Z	Z	Z	Z
0	0	1	0	0	Z	Z	Z	Z
0	0	1	0	1	Z	Z	Z	Z
0	0	1	1	0	Z	Z	Z	Z
0	0	1	1	1	Z	Z	Z	Z
0	1	0	0	0	Z	Z	Z	Z
0	1	0	0	1	Z	Z	Z	Z
0	1	0	1	0	Z	Z	Z	Z
0	1	0	1	1	Z	Z	Z	Z
0	1	1	0	0	Z	Z	Z	Z
0	1	1	0	1	Z	Z	Z	Z
0	1	1	1	0	Z	Z	Z	Z
0	1	1	1	1	Z	Z	Z	Z
1	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0
1	0	0	1	0	0	0	0	0
1	0	0	1	1	0	0	0	0
1	0	1	0	0	0	0	0	0
1	0	1	0	1	0	0	0	0
1	0	1	1	0	0	0	1	0
1	0	1	1	1	0	0	1	1
1	1	0	0	0	0	0	0	0
1	1	0	0	1	0	0	1	0
1	1	0	1	0	0	1	0	0
1	1	0	1	1	0	1	1	0
1	1	1	0	0	0	0	0	0
1	1	1	0	1	0	0	1	1
1	1	1	1	0	0	1	1	0
1	1	1	1	1	1	0	0	1

EJERCICIO 2 : FUNCIONES LOGICAS

$$M3 = AC(A1.A0.B1.B0)$$

$$M2 = AC(A1.B1.(\overline{A0} + \overline{B0}))$$

$$M1 = AC(A0.B1.\overline{B0} + A0.B1.\overline{A1} + B0.A1.\overline{B1} + B0.A1.\overline{A0})$$

$$M0 = AC.A0.B0$$

EJERCICIO 2: SOLUCION a

```

library ieee;
use ieee.std_logic_1164.all;

ENTITY eje2a IS
    PORT(    A,B : IN STD_LOGIC_VECTOR(1 downto 0);
            AC  : IN STD_LOGIC;
            M   : OUT STD_LOGIC_VECTOR(3 downto 0));
END eje2a;

ARCHITECTURE sol OF eje2a IS

BEGIN
    M(3) <= ( A(1) and A(0) and B(1) and B(0) ) when AC='1' else 'Z';
    M(2) <= ( A(1) and B(1) and (not A(0) or not B(0)) ) when AC='1' else 'Z';
    M(1) <= ( (A(0) and B(1) and ( not B(0) )) or ( A(0) and B(1) and (not A(1)) )
              or (B(0) and A(1) and (not B(1)) ) or (B(0) and A(1) and (not A(0))) )
              when AC='1' else 'Z';
    M(0) <= ( A(0) and B(0)) when AC='1' else 'Z';
END sol;

```

EJERCICIO 2: SOLUCION b

```

library ieee;
use ieee.std_logic_1164.all;

ENTITY eje2b IS
    PORT(    A,B : IN STD_LOGIC_VECTOR(1 downto 0);
            AC  : IN STD_LOGIC;
            M   : OUT STD_LOGIC_VECTOR(3 downto 0));
END eje2b;

ARCHITECTURE sol OF eje2b IS

    SIGNAL w : STD_LOGIC_VECTOR(4 downto 0);

BEGIN
    w <= AC & A & B;
    with w select
    M <=
        "0000" when "10000" | "10001" | "10010" | "10011" | "10100" | "11000" | "11100",
        "0001" when "10101",
        "0010" when "10110" | "11001",
        "0011" when "10111" | "11101",
        "0100" when "11010",
        "0110" when "11011" | "11110",
        "1001" when "11111",
        "ZZZZ" when others;

END sol;

```

EJERCICIO 2: SOLUCION c

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;

ENTITY eje2c IS
    PORT(    A,B : IN STD_LOGIC_VECTOR(1 downto 0);
            AC  : IN STD_LOGIC;
            M   : OUT STD_LOGIC_VECTOR(3 downto 0));
END eje2c;

ARCHITECTURE sol OF eje2c IS

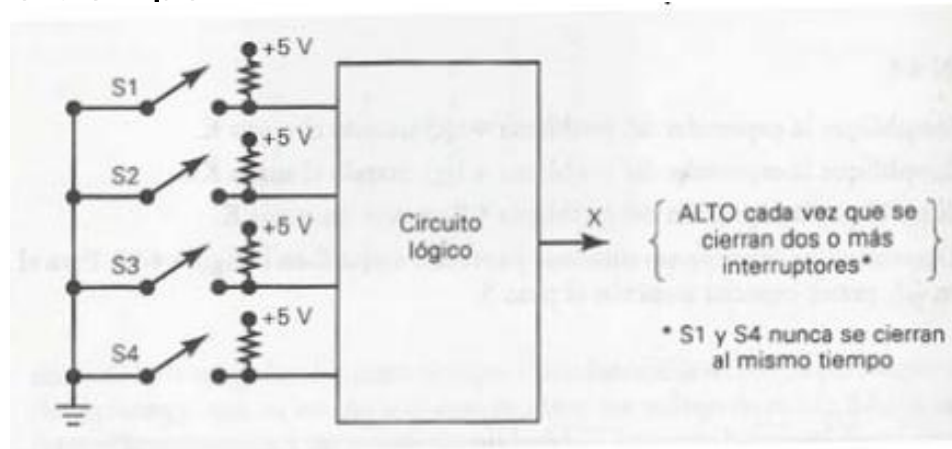
BEGIN

    M <= A*B when AC='1' else "ZZZZ";

END sol;
```

EJERCICIO # 3

La figura muestra cuatro interruptores que son parte de la circuitería de control de una máquina copiadora. Los interruptores se encuentran en distintos puntos consecutivos a lo largo del camino que recorre el papel dentro de la máquina. Cada interruptor está normalmente abierto y, cuando el papel pasa sobre el interruptor, éste se cierra. Es imposible que los interruptores S1 y S4 se cierran al mismo tiempo, además asuma que solo puede ingresar un papel a la vez. Diseñe un circuito lógico que genere una salida de voltaje ALTA (H) cada vez que dos o más interruptores están cerrados al mismo tiempo.



EJERCICIO 3 : TABLA DE VERDAD

S4	S3	S2	S1	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	Φ
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	Φ
1	0	1	0	Φ
1	0	1	1	Φ
1	1	0	0	1
1	1	0	1	Φ
1	1	1	0	1
1	1	1	1	Φ

S4,S3

S2,S1

	00	01	11	10
00	0	0	1	0
01	0	Φ	Φ	Φ
11	1	1	Φ	Φ
10	0	1	1	Φ

$$X = S_1 S_2 + S_3 S_4 + S_2 S_3$$

EJERCICIO 3 : SOLUCION a

```
library ieee;
use ieee.std_logic_1164.all;

ENTITY eje3a IS

    PORT(  S      : IN      STD_LOGIC_VECTOR(4 downto 1);
          X      : OUT     STD_LOGIC);

END eje3a;

ARCHITECTURE sol OF eje3a IS

BEGIN

    X <= (S(1) and S(2)) or (S(3) and S(4)) or (S(2) and S(3));

END sol;
```

Nota : Esta solución asume que todas las señales son de lógica positiva. Para considerar que las señales son de lógica negativa, es necesario agregar el operador “NOT” delante de cada variable.

EJERCICIO 3 : SOLUCION b

```
library ieee;
use ieee.std_logic_1164.all;
```

```
ENTITY eje3b IS
```

```
    PORT(  S      : IN      STD_LOGIC_VECTOR(4 downto 1);
           X      : OUT     STD_LOGIC);
```

```
END eje3b;
```

```
ARCHITECTURE sol OF eje3b IS
```

```
BEGIN
```

```
with S select
```

```
    X <=  '1' when "0011" | "0110" | "0111" | "1100" | "1110",
          '-' when "0101" | "1001" | "1010" | "1011" | "1101" | "1111",
          '0' when others;
```

```
END sol;
```

EJERCICIO # 4

Diseñe el circuito de control que comande el apagado automático de un televisor. El controlador recibe las siguientes señales:

Automático.L : es baja (L) cuando se desea activar el apagado automático.

TTV.H : es alta (H) cuando se presiona una tecla en el televisor

TCR.H : es alta (H) cuando se presiona alguna tecla en el control remoto.

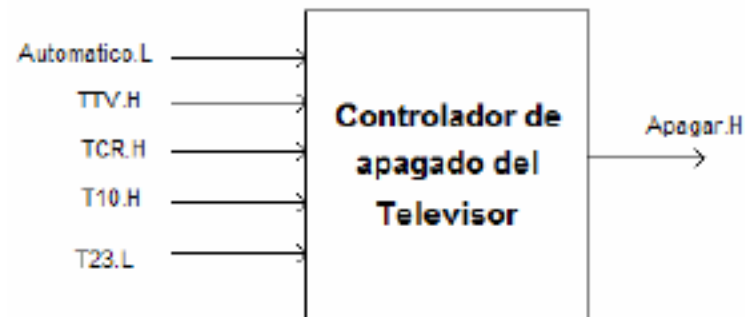
T10.H : es alta (H) cuando han pasado 10 minutos desde el último movimiento de teclas en el TV.

T23.L : es baja (L) cuando el reloj del TV marca mas de las 23h00.

El circuito debe activar la salida (Apagar.H=H) cuando este activado el automático, no se está moviendo ninguna tecla, sean mas de las 23h00 y hayan pasado mas de 10 minutos desde el último movimiento.

Si el automático no está activado la salida debe tomar un valor de alta impedancia.

Asuma que no es posible que la señal de 10 minutos se active y al mismo tiempo se presiona alguna tecla.



Ejercicio 4 : Tabla de verdad

Auto	TTV	TCR	T10	T23	Apagar
0	0	0	0	0	Z
0	0	0	0	1	Z
0	0	0	1	0	Z
0	0	0	1	1	Z
0	0	1	0	0	Z
0	0	1	0	1	Z
0	0	1	1	1	Z
0	1	0	0	0	Z
0	1	0	0	1	Z
0	1	0	1	0	Z
0	1	0	1	1	Z
0	1	1	0	0	Z
0	1	1	0	1	Z
0	1	1	1	0	Z
0	1	1	1	1	Z
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	1
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	Φ
1	0	1	1	1	Φ
1	1	0	0	0	0
1	1	0	0	1	0
1	1	0	1	0	Φ
1	1	0	1	1	Φ
1	1	1	0	0	0
1	1	1	0	1	0
1	1	1	1	0	Φ
1	1	1	1	1	Φ

TTV,TCR					
T10,T23		00	01	11	10
	00	0	0	0	0
	01	0	0	0	0
	11	1	Φ	Φ	Φ
	10	0	Φ	Φ	Φ

$$Apagar = T10.T23$$

EJERCICIO # 4 : SOLUCION a

```
library ieee;
use ieee.std_logic_1164.all;

ENTITY eje4a IS
    PORT(    automatico,T10,TCR,T10,T23    : IN        STD_LOGIC;
           apagar                          : OUT       STD_LOGIC);
END eje4a;

ARCHITECTURE sol OF eje4a IS

BEGIN
    apagar <= (T10 and T23) when automatico='1' else '2';
END sol;
```

EJERCICIO # 4 : SOLUCION b

```

library ieee;
use ieee.std_logic_1164.all;

ENTITY eje4b IS
    PORT(    automatico,TTU,TCR,T10,T23    : IN    STD_LOGIC;
            apagar                          : OUT   STD_LOGIC);
END eje4b;

ARCHITECTURE sol OF eje4b IS

    SIGNAL w : std_logic_vector (4 downto 0);

BEGIN
    w <= automatico & TTU & TCR & T10 & T23;

    with w select
        apagar <=
            '1' when "10011",
            '-' when "10110" | "10111" | "11010" | "11011" | "11110" | "11111",
            '0' when "10000" | "10001" | "10010" | "10100" | "10101" | "11000" |
                  "11001" | "11100" | "11101",
            '2' when others;

END sol;

```

EJERCICIO # 5

Se desea disponer de un circuito generador de paridad.

Este dispositivo recibe una palabra de información de 4 bits, a través de la entrada "DATO.H".

Por medio del interruptor "SELECTOR DE PARIDAD" se escoge la paridad que se desea generar en la salida "BIT DE PARIDAD.H".

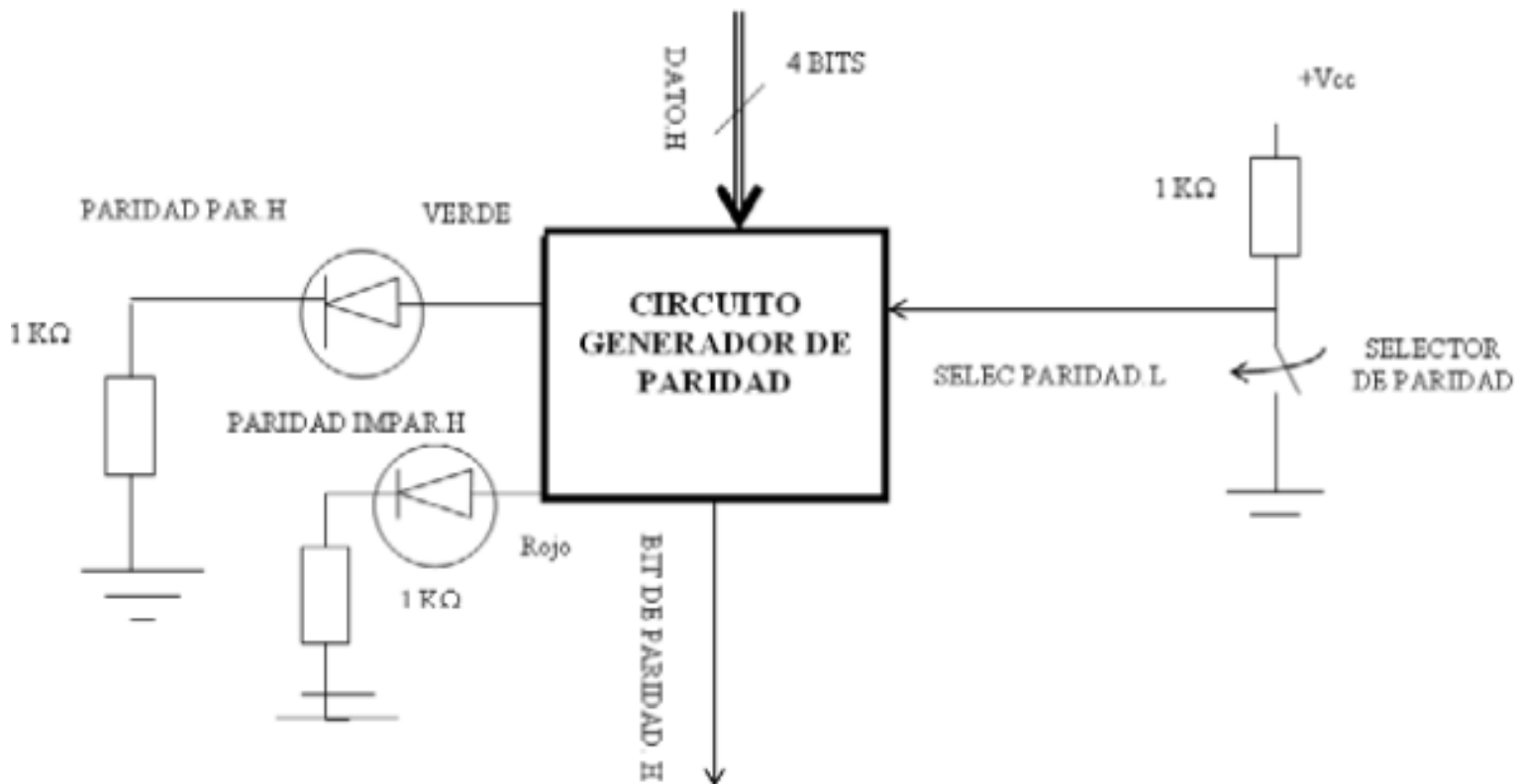
Cuando este interruptor está cerrado (SELEC PARIDAD.L = L), se generará la paridad par, esto es, si el número de unos en la palabra "DATO" es impar, la señal BIT DE PARIDAD.H se fijará en nivel alto (BIT DE PARIDAD.H = H); en caso contrario, esta salida se fijará en nivel bajo (BIT DE PARIDAD.H = L).

Con el interruptor "SELECTOR DE PARIDAD" abierto (SELEC PARIDAD.L = H), se generará la paridad impar, y la salida "BIT DE PARIDAD.H" se irá a nivel alto (BIT DE PARIDAD.H = H), cada vez

que el número de unos en la entrada "DATO.H" es par y lo contrario cuando el número de unos es impar.

EJERCICIO # 5

Cuando se trabaja con paridad par, se enciende el led verde y cuando la paridad es impar, se prende el led rojo. El diagrama de bloques a continuación ilustra las entradas y salidas del circuito.



Spar	D3	D2	D1	D0	BP	Ppar	Pimp
0	0	0	0	0	1	0	1
0	0	0	0	1	0	0	1
0	0	0	1	0	0	0	1
0	0	0	1	1	1	0	1
0	0	1	0	0	0	0	1
0	0	1	0	1	1	0	1
0	0	1	1	0	1	0	1
0	0	1	1	1	0	0	1
0	1	0	0	0	0	0	1
0	1	0	0	1	1	0	1
0	1	0	1	0	1	0	1
0	1	0	1	1	0	0	1
0	1	1	0	0	1	0	1
0	1	1	0	1	0	0	1
0	1	1	1	0	0	0	1
0	1	1	1	1	1	0	1
1	0	0	0	0	0	1	0
1	0	0	0	1	1	1	0
1	0	0	1	0	1	1	0
1	0	0	1	1	0	1	0
1	0	1	0	0	1	1	0
1	0	1	0	1	0	1	0
1	0	1	1	0	0	1	0
1	0	1	1	1	1	1	0
1	1	0	0	0	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	0	0	1	0
1	1	0	1	1	1	1	0
1	1	1	0	0	1	1	0
1	1	1	0	1	0	1	0
1	1	1	1	0	1	1	0
1	1	1	1	1	0	1	0
1	1	1	1	1	1	1	0

EJERCICIO 5 : SOLUCION a

```
library ieee;
use ieee.std_logic_1164.all;
```

```
ENTITY eje5a IS
```

```
    PORT ( D           : IN    STD_LOGIC_VECTOR(3 DOWNT0 0);
           spar        : IN    STD_LOGIC;
           BP, ppar, pimp : OUT  STD_LOGIC);
```

```
END eje5a;
```

```
ARCHITECTURE sol OF eje5a IS
```

```
    SIGNAL w : std_logic_vector (4 downto 0);
```

```
BEGIN
```

```
    w <= not spar & D;
```

```
    ppar <= not spar;
```

```
    pimp <= spar;
```

```
    with w select
```

```
        BP <=  '1' when "00000" | "00011" | "00101" | "00110" | "01001" | "01010" |
                "01100" | "01111" | "10001" | "10010" | "10100" | "10111" |
                "11000" | "11011" | "11101" | "11110",
```

```
        '0' when others;
```

```
END sol;
```

EJERCICIO 5 : SOLUCION b

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

ENTITY eje5b IS
    PORT( D           : IN    STD_LOGIC_VECTOR(3 DOWNT0 0);
          spar        : IN    STD_LOGIC;
          BP, ppar, pimp : OUT  STD_LOGIC);
END eje5b;

ARCHITECTURE sol OF eje5b IS

    SIGNAL w : STD_LOGIC_VECTOR(2 DOWNT0 0);
    SIGNAL bi: std_logic;

BEGIN
    w<="000"+D(3)+D(2)+D(1)+D(0);
    ppar <= not spar;
    pimp <= spar;
    bi <= '0' when w="000" or w="010" or w="100" else '1';
    BP <= bi when spar='0' else not bi;
END sol;

```