EJERCICIOS DE SISTEMAS ELECTRÓNICOS DIGITALES: HOJA 1
2º CURSO DE INGENIERÍA TÉCNICA INDUSTRIAL. ESPECIALIDAD EN ELECTRÓNICA INDUSTRIAL

LENGUAJES DE ALTO NIVEL

1) Implementa en VHDL
   a) Un Latch D

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY latchd IS
        PORT
        (
                d,c                     : IN        STD_LOGIC;
                q                       : OUT       STD_LOGIC
        );

END latchd;

ARCHITECTURE a OF latchd IS
SIGNAL  qs: STD_LOGIC;
BEGIN
        PROCESS (d,c)
        BEGIN
                IF (c='1') THEN
                        qs <= d;
                ELSE
                        qs<=qs;
                END IF;
        END PROCESS;
        q<= qs;
END a;
```

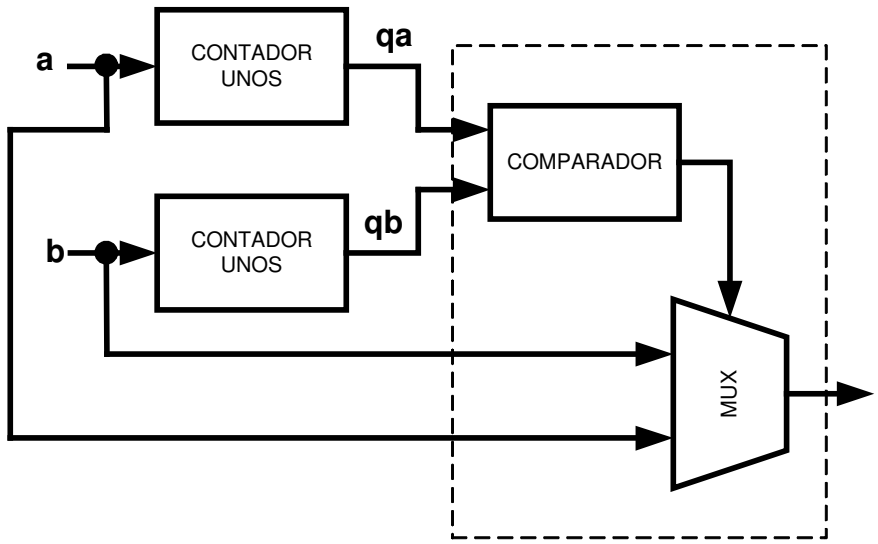
   b) Un Flip-Flop JK con clear y preset asíncronos

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY ffjk IS
        PORT
        (
                j: IN       STD_LOGIC;
                k: IN       STD_LOGIC;
                clk: IN     STD_LOGIC;
                clr: IN     STD_LOGIC;
                pre: IN     STD_LOGIC;
                q: OUT      STD_LOGIC
        );
END ffjk;

ARCHITECTURE a OF ffjk IS
        SIGNAL  qs          : STD_LOGIC;
BEGIN
        PROCESS (clk, clr,pre)
        BEGIN
                IF clr = '0' THEN
                        qs <= '0';
                ELSIF pre = '0' THEN
                        qs <= '1';
                ELSIF (clk'EVENT AND clk= '1') THEN
                        IF (j='1' AND k='1' ) THEN
                                qs<= NOT qs;
                        ELSIF (j='1' AND k='0' ) THEN
                                qs<= '1';
                        ELSIF (j='0' AND k='1' ) THEN
                                qs<= '0';
                        ELSIF (j='0' AND k='0' ) THEN
                                qs<= qs;
                        END IF;
                END IF;
        END PROCESS;
        q <= qs;
END a;
```

2) Implementa en VHDL un sistema que proporcione el valor absoluto de la resta de dos números $|a-b|$. Realízalo en modo RTL y en algorítmico.

```
ENTITY fabs IS
	PORT
	(
		a: IN	INTEGER RANGE 0 TO 1000;
		b: IN	INTEGER RANGE 0 TO 1000;
		c: OUT	INTEGER RANGE 0 TO 1000
	);
END fabs;

--RTL
ARCHITECTURE a OF fabs IS

BEGIN
c<= a-b WHEN a>b ELSE b-a;
END a;

--Algorítmic
ARCHITECTURE a OF fabs IS

BEGIN
	PROCESS (a,b)
	BEGIN

		IF a>b THEN
			c <= a-b;
		ELSE
			c<= b-a;
		END IF;
	END PROCESS;
END a;
```

3) Implementa en VHDL un multiplexor que, a partir de dos buses de entrada de cuatro bits *a* y *b,* proporcione a su salida el bus que tenga mayor número de *unos*.




```
ENTITY ej3 IS
	PORT (a,b: IN BIT_VECTOR (3 DOWNTO 0);
		c: OUT BIT_VECTOR (3 DOWNTO 0));

END ej3;

ARCHITECTURE arch1 OF ej3 IS
SIGNAL  qa : INTEGER RANGE 0 TO 4;
SIGNAL  qb : INTEGER RANGE 0 TO 4;

BEGIN
```

```
	PROCESS (a)
	VARIABLE n: INTEGER;
	BEGIN
		n:=0;
		FOR i IN a'RANGE LOOP
			IF a(i)='1' THEN
				n:=n+1;
			END IF;
		END LOOP;
		qa<= n;
	END PROCESS;
	PROCESS (b)
	VARIABLE nb: INTEGER;
	BEGIN
		nb:=0;
		FOR i IN b'RANGE LOOP
			IF b(i)='1' THEN
				nb:=nb+1;
			END IF;
		END LOOP;
		qb<= nb;
	END PROCESS;
	c<= a WHEN qa>qb ELSE b;
END arch1;
```

4) Determina la funcionalidad del siguiente código VHDL.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux IS
	PORT (	a,b,c,d: IN STD_LOGIC_VECTOR (2 DOWNTO 0);
		seleccio:	IN INTEGER RANGE 0 TO 3;
		enable: IN STD_LOGIC;
		e: OUT STD_LOGIC_VECTOR (2 DOWNTO 0));
END mux;

ARCHITECTURE comportament OF mux IS
BEGIN
	PROCESS (a,b,c,d,seleccio,enable)
	BEGIN
		IF enable='1' THEN
			CASE seleccio IS
				WHEN 0 =>
					e <= a;
				WHEN 1=>
					e <= b;
				WHEN 2=>
					e <= c;
				WHEN 3=>
					e <= d;
			END CASE;
		ELSE
			e <= "ZZZ";
		END IF;
	END PROCESS;
END comportament;
```

Solución:

Se trata de un multiplexor de cuatro buses de tres bits cada uno (de los cuatro se selecciona uno sólo). El multiplexor está habilitado por una señal de “enable”, si enable=1 entonces funciona correctamente, si enable=0 entonces la salida está en alta impedancia.

5) Describe el anterior código funcional en forma concurrente

Solución:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
```

```
ENTITY mux2 IS
        PORT (  a,b,c,d: IN STD_LOGIC_VECTOR (2 DOWNTO 0);
                seleccio:   IN INTEGER RANGE 0 TO 3;
                enable: IN STD_LOGIC;
                e: OUT STD_LOGIC_VECTOR (2 DOWNTO 0));
END mux2;

ARCHITECTURE concurrente OF mux2 IS
SIGNAL mig: STD_LOGIC_VECTOR (2 DOWNTO 0);
BEGIN

WITH seleccio SELECT
mig<=   a WHEN 0,
                b WHEN 1,
                c WHEN 2,
                d WHEN 3;

e<=     "ZZZ" WHEN enable='0' ELSE mig;
```

6) Implementa en VHDL una unidad aritmética de forma que sobre dos palabras '*a*' y '*b*' de 4 bits realice las funciones suma, resta, incremento de '*a*' en una unidad, y decremento de '*a*' una unidad en función de una variable de selección. Una vez implementada haz la simulación con MAX+PLUS II

7) Implementa en VHDL una unidad lógica de forma que sobre dos palabras '*a*' y '*b*' de 4 bits realice las funciones AND, OR, XOR y NOT '*a*' en función de una variable de selección. Una vez implementada haz la simulación con MAX+PLUS II
8) Implementa una ALU de cuatro bits que realice las ocho funciones anteriores (*a+b, a-b, a+1, a-1, a AND b, a OR b, a XOR b, NOT a*) en función de una variable de selección. Realiza su simulación con el MAX+PLUSII

```
                                                -- ALU concurrent
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

ENTITY alu IS
        PORT (  a,b: IN STD_LOGIC_VECTOR(3 DOWNTO 0) ;
                seleccio:   IN INTEGER RANGE 0 TO 7;
                c: OUT STD_LOGIC_VECTOR(4 DOWNTO 0)  );

END alu;

ARCHITECTURE comportament OF alu IS
BEGIN

WITH seleccio SELECT
c<=     a+b WHEN 0,
                a-b WHEN 1,
                a+'1' WHEN 2,
                a-'1' WHEN 3,
                a AND b WHEN 4,
                a OR b WHEN 5,
                a XOR b WHEN 6,
                NOT a WHEN 7;

END comportament;
```