NOMBRE: **GRUPO:**

Problema 1 (3 puntos)

Sean A=11001 y B=01110 dos números codificados en binario natural sin signo en 5 bits.

1. Codifique A en:

   A_{10}(decimal): 25

   A_{16}(hexadecimal): 19

   A_{BCD}(BCD): 0010 0101

2. Codifique los siguientes números en complemento a dos, siendo A y B números positivos:

   A_{c2}: 011001

   B_{c2}: 001110

   $-A_{c2}$: 100111

3. Calcule:

   $A_{c2}+B_{c2}$: 100111

   ¿Hay overflow? Justifique la respuesta. Hay overflow ya que teniendo A y B igual signo (0, positivo) el resultado de la operación tiene un signo distinto a estos (1, negativo).

4. Calcule:

   $B_{c2}-A_{c2}$: 110101

   ¿Hay overflow? Justifique la respuesta. No hay overflow ya que con 6 bits en C2 podemos representar valores en el rango [-32,+31], siendo el resultado de la operación -11.

5. El siguiente código VHDL implementa un circuito sumador de dos operandos de 8 bits con codificación en complemento a dos y detección de overflow. Completa las líneas perdidas:

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
ENTITY parcial_1_arit IS
        PORT(
                a,b: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
                c: OUT STD_LOGIC_VECTOR(7 DOWNTO 0); --resultado de la suma de a y b
                overflow: OUT STD_LOGIC -- overflow=1 si hubo overflow en la operación de suma
        );
END parcial_1_arit;

ARCHITECTURE arch OF parcial_1_arit IS
        SIGNAL c_aux: SIGNED(7 DOWNTO 0);
BEGIN
        c_aux <= SIGNED(a) + SIGNED(b);
        c <= STD_LOGIC_VECTOR(c_aux);
        overflow <= '1' WHEN a(7)=b(7) AND a(7)/=c_aux(7)                ELSE
                        '0';
END arch;
```

NOMBRE: **GRUPO:**

RÚBRICA

1. (15%)	5% cada cuestión (Bien, Mal)
2. (30%)	10% cada cuestión (Bien, Mal)
3. (15%)	5% la operación aritmética (Bien, Mal) 10% explicación overflow (Bien, Regular, Mal)
4. (15%)	5% la operación aritmética (Bien, Mal) 10% explicación overflow (Bien, Regular, Mal)
5. (35%)	5% primera sentencia (Bien, Mal) 15% segunda sentencia (Bien, Mal) 15% tercera sentencia (Bien, Mal)

Problema 2 (30 minutos – 3,5 puntos)

Dado el siguiente código VHDL que representa un circuito digital combinacional:

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

ENTITY Parcial1_2 IS
PORT (
A, B:   IN  STD_LOGIC;
C:      IN STD_LOGIC_VECTOR  ( 1 downto 0) ;
F0,F1: OUT STD_LOGIC
 );
END Parcial1_2;

ARCHITECTURE behavioral OF Parcial1_2 IS
Signal X: STD_LOGIC;
Signal Y: STD_LOGIC_VECTOR (3 downto 0);
BEGIN
        Proceso1: PROCESS (A,B)
        BEGIN
        X<= A XOR B;
                IF X='1' THEN
                        IF B='0' THEN
                        F0<= C(0);
                        ELSE F0<= NOT (A);
                        END IF;
                ELSE F0<=A OR B OR C(0);
                END IF;
        END PROCESS;

        Proceso2: PROCESS (Y,C)
        BEGIN
                CASE C IS
                WHEN "00" => Y<= "0001";
                WHEN "01" => Y<= "0010";
                WHEN "10" => Y<= "0100";
                WHEN "11" => Y<= "1000";
                WHEN OTHERS => Y<="0000";
                END CASE;
        F1<= Y(1) OR Y(3);
        END PROCESS;

END behavioral;
```

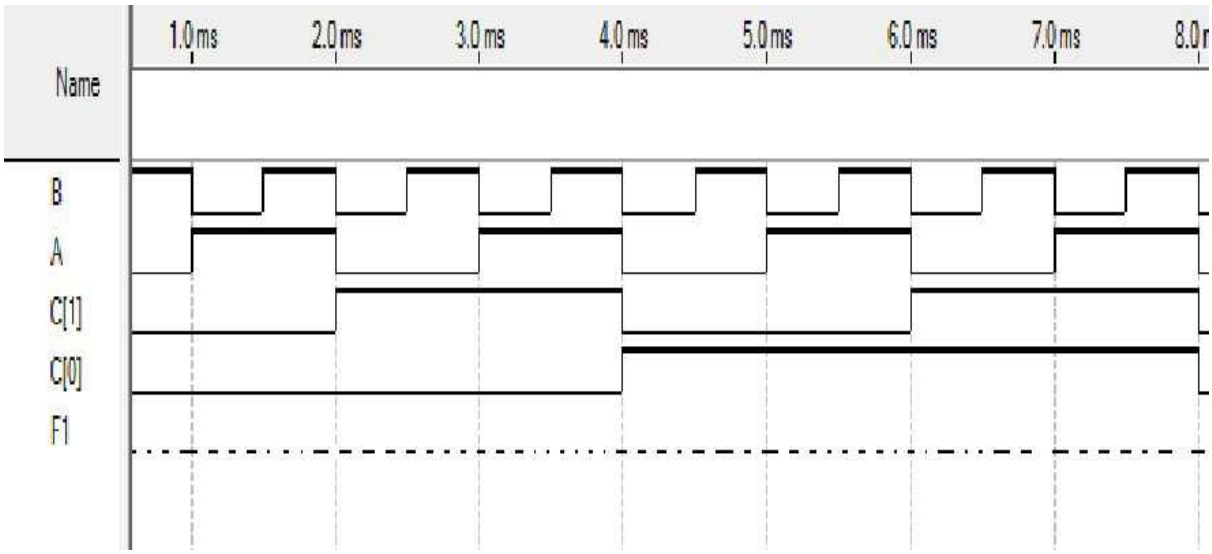
Se pide:

1. (**10%**) Defina todas las Entradas y Salidas del circuito, indicando el número de bits de cada una de las señales.

2. (**30%**) Complete la Tabla de verdad del Proceso 1.

C(0)	A	B	X	F0

3. (**30%**) Complete el siguiente cronograma de la señal F1 del proceso 2



4. (**30%**)Dibuje el esquema del circuito, **utilizando los siguientes Bloques Combinacionales**:

   Dos Multiplexores 2:1, Un Decodificador 2:4 y Puertas Lógicas.

Solución

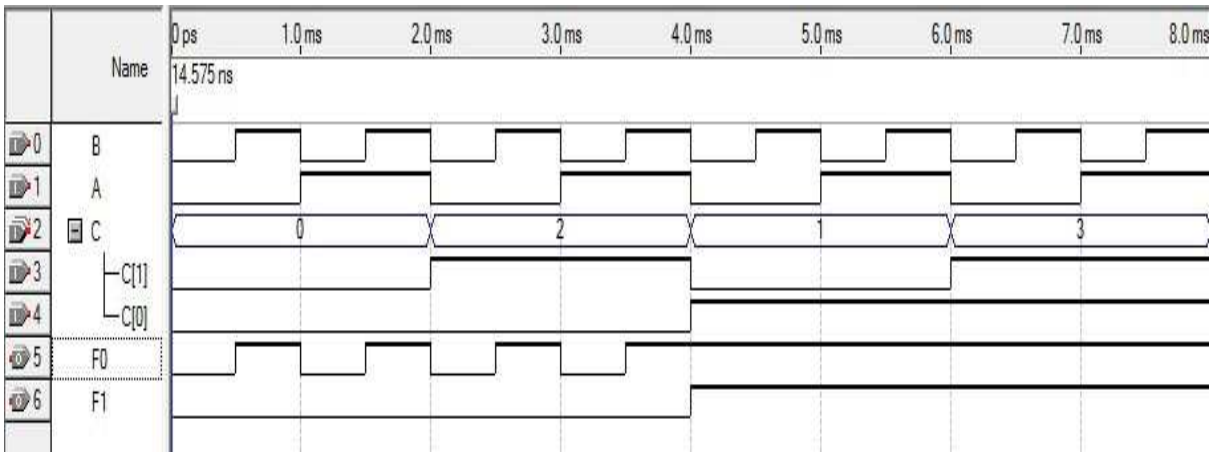
1)

```
1. ENTITY Parcial1_2 IS
2. PORT (
3. A, B:   IN  STD_LOGIC;                    → 2 Entradas de 1 bit
4. C:      IN STD_LOGIC_VECTOR  ( 1 downto 0) ;   → 1 Entrada de 2 Bits
5. F0,F1: OUT STD_LOGIC                      → 2 Salidas de 1 Bit
6.  );
7. END Parcial1_2;
8.
```

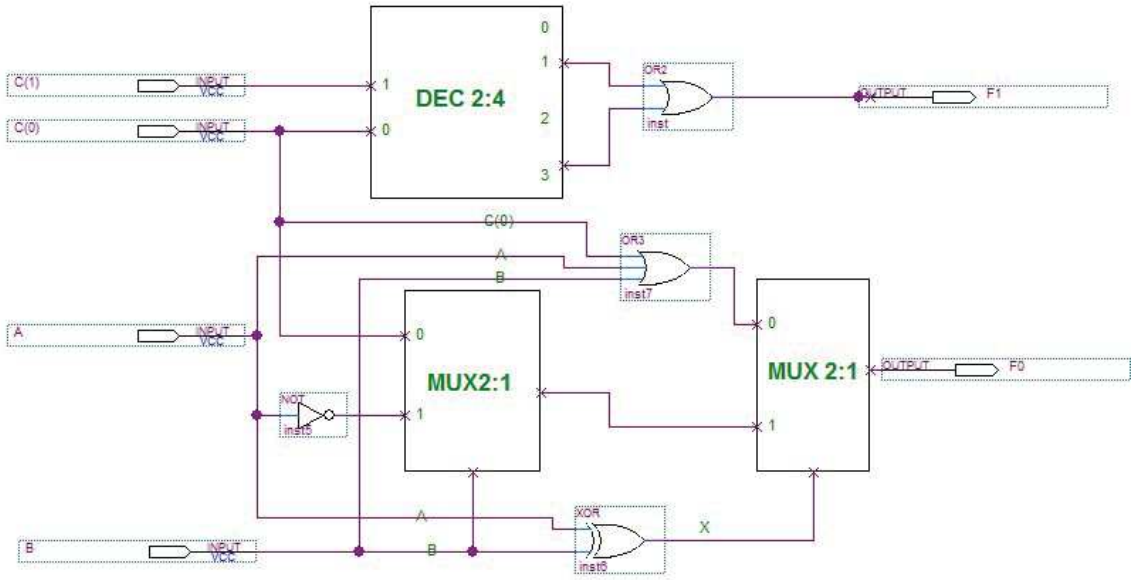
2)

C(0)	A	B	X	F0
0	0	0	**0**	**0**
0	0	1	1	**1**
0	1	0	1	**0**
0	1	1	**0**	**1**
1	0	0	**0**	**1**
1	0	1	1	**1**
1	1	0	1	**1**
1	1	1	**0**	**1**

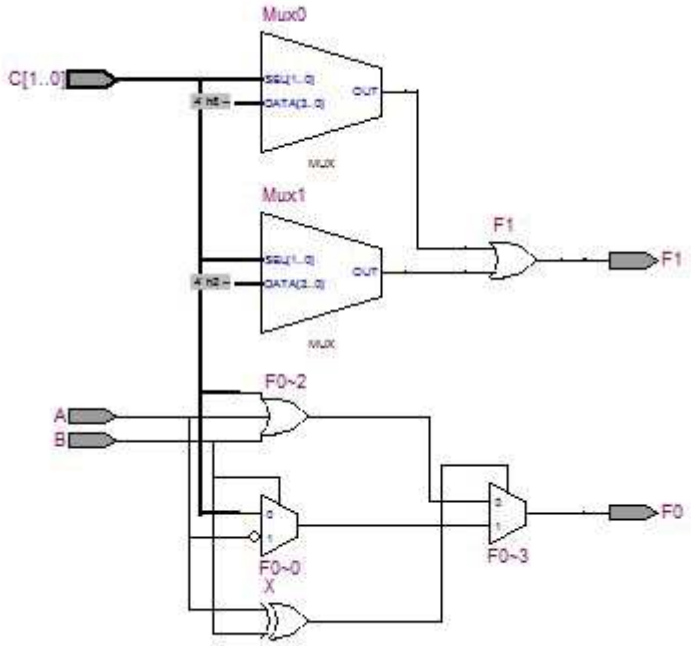
3)



4) El esquema quedaría:



Circuito Sintetizado en Quartus II :



NOMBRE Y APELLIDOS: GRUPO:

Problema 3 (3,5 puntos) 35 minutos SOLUCIÓN

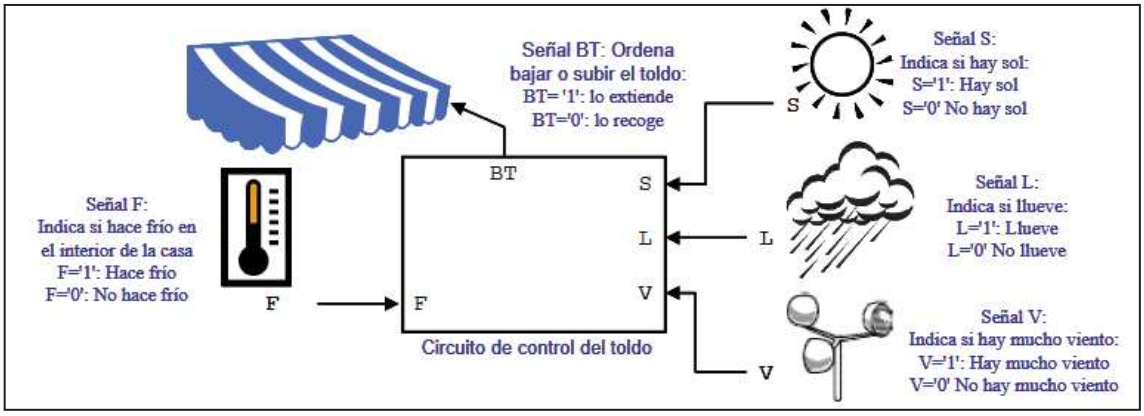
Se desea realizar un circuito de control para el toldo de una terraza de una vivienda. El toldo tiene la función tanto de dar sombra como de proteger del viento y de la lluvia. Así que es un toldo resistente al viento y a la lluvia, manteniendo la terraza seca en los días de lluvia.

Para el circuito de control tenemos las siguientes entradas:

- Señal S: Indica si hay sol
- Señal L: Indica si llueve
- Señal V: Indica si hay mucho viento
- Señal F: Indica si hace frío en el interior de la casa.

Según los valores de estas entradas se bajará o subirá el toldo. Esto se realizará mediante la señal de salida **BT** (Bajar Toldo). Si **BT='1'** indica que el toldo debe estar extendido (bajado) y si **BT='0'** indica que el toldo debe estar recogido (subido).

El sistema se muestra en la figura.



El circuito que acciona el toldo que debe funcionar según las siguientes características:

- Independientemente del resto de señales de entrada, siempre que llueva se debe de extender el toldo para evitar que se moje la terraza. No se considerará posible que simultáneamente llueva y haga sol.
- Si hace viento se debe extender el toldo para evitar que el viento moleste. Sin embargo, hay una excepción: aunque haya viento, si el día está soleado y hace frío en la casa, se recogerá el toldo para que el sol caliente la casa.
- Por último, si no hace viento ni llueve, sólo se bajará el toldo en los días de sol y cuando haga calor en el interior, para evitar que se caliente mucho la casa.

Nota: Para combinaciones de entrada imposibles, considere que la salida BT toma el valor '0'.

NOMBRE Y APELLIDOS: GRUPO:

Se pide:

1. Complete en la tabla de verdad el valor de la señal que controla el toldo **BT** a partir de las señales **S, L, V y F.(20%)**

L	V	S	F	BT
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

2. Indique ¿cuál sería la acción llevada por la lógica de control si los valores de las entradas son **L=1, V=1, S=0 y F=1?(15%)**

Debido a que llueve y como no hay sol, no hay incongruencia, prevalece el BT=1 para no mojar la terraza.

NOMBRE Y APELLIDOS: GRUPO:

3. Complete la entidad creada para implementar el circuito digital que se quiere diseñar. **(15%)**

```
-----LIBRERIA---------------------
library IEEE;
use ieee.std_logic_1164.all;


-----ENTITY-------------------

entity parcial1 is
  port (
    L,V,S,F: in  std_logic;
    BT:   out std_logic
  );
end parcial1;
```

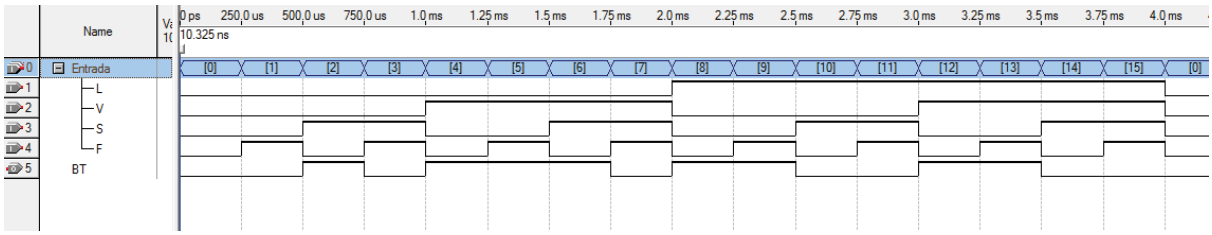
4. Implemente la salida del sistema **BT**, haciendo uso de sentencias concurrentes.**(25%)**

```
-----Arquitectura-------------------

architecture toldo of parcial1 is

signal entrada: std_logic_vector (3 downto 0);


begin

entrada<=L&V&S&F;

WITH entrada SELECT
BT<=    '1' WHEN "0010",
        '1' WHEN "0100",
        '1' WHEN "0101",
        '1' WHEN "0110",
        '1' WHEN "1000",
        '1' WHEN "1001",
        '1' WHEN "1100",
        '1' WHEN "1101",
        '0' WHEN OTHERS;



end toldo;
```

NOMBRE Y APELLIDOS: GRUPO:

Otra solución posible, implementada a partir de la TV, muy tediosa y puede conllevar algunos errores al editar

```
-----Arquitectura--------------------

architecture toldo of parcial1 is

signal entrada: std_logic_vector (3 downto 0);


begin

entrada<=L&V&S&F;


BT<=    '1' WHEN entrada="0010" or entrada="0100" or entrada="0101" or entrada="0110" or
        entrada="1000" or entrada="1001" or entrada="1100" or entrada="1101"
        else '0';



end toldo;
```



5. Implemente la salida la salida del sistema **BT**, haciendo uso de sentencias secuenciales. **(25%)**

NOMBRE Y APELLIDOS: GRUPO:

```
-----Arquitectura-------------------

architecture toldo of parcial1 is
signal entrada: std_logic_vector (3 downto 0);
begin

entrada<=L&V&S&F;

process (entrada)

begin

    case entrada is
    when "0010" => BT<='1';
    when "0100" => BT<='1';
    when "0101" => BT<='1';
    when "0110" => BT<='1';
    when "1000" => BT<='1';
    when "1001" => BT<='1';
    when "1100" => BT<='1';
    when "1101" => BT<='1';
    when others => BT<='0';
    end case;
end process;
end toldo;
```