Exercici 4: ALU

NOTA: Per aquest exercici necessitareu GHDL i GTKWave.

NOTA: Aquest exercici serà objecte de lliurament per a la seva avaluació. La data de lliurament sortirà a la tasca corresponent al Moodle de l'assignatura.

En aquest exercici practicareu amb operacions aritmètiques sobre vectors de bits, tot experimentant els efectes de compartició de recursos.

Dissenyeu i escriviu el codi VHDL d'un circuit digital que implementi una ALU de 8 bits amb la declaració següent (trobareu templates a la carpeta de l'exercici):

```
library IEEE;
use IEEE.Std_logic_1164.all;
entity ALU is
    port(A, B : in Std_logic_vector(7 downto 0);
        Op : in Std_logic_vector(3 downto 0);
        F : out Std_logic_vector(7 downto 0);
        Cout : out Std_logic;
        Equal: out Std_logic;
end ALU;
```

Totes les sortides són combinacionals. No es tracta d'un circuit seqüencial. El funcionament de la ALU ha de ser el següent:

- La sortida Equal val '1' si i només si A = B, i '0' en qualsevol altre cas. Això és independent de l'operació que faci la ALU.
- A i B són de tipus Std_logic_vector, però totes les operacions s'avaluen utilitzant el tipus signed.
- L'entrada Op és de tipus Std_logic_vector de 4 bits, que configura 13 operacions diferents i 3 que no s'utilitzen (dummy). La Taula 1 mostra un recull de les operacions.
- Les operacions de desplaçament (shift) són lògiques, és a dir, que els bits que deixa *buits* una operació shift a dreta o esquerra, s'omplen amb '0'.
- Un 'shift left' desplaça el bit 7 a Cout.
- Un 'rotate left' desplaça el bit 7 a Cout i també al bit 0.
- Per totes les operacions tret de 'shift' i 'rotate' les entrades A i B s'interpreten com a nombres binaris de 8 bits de tipus **signed** (notació complement a 2), i el resultat és un nombre **signed** de 9 bits els 8 bits de menys pes s'assignen a F, i el de més pes a Cout, que és en realitat el bit de signe. Si Cout és '1' indica que el resultat de l'operació és negatiu.

Simuleu el vostre disseny tot utilitzant el testbench donat. El testbench té tres senyals **OK**, un per F, un per Cout i el darrer per Equal. Aquests senyals **OK** són 'true' si el resultat és correcte.

També hi ha tres senyals, F_Error_Count, Cout_Error_Count i Equal_Error_Count, que compten el nombre d'errors detectats per cadascuna de les tres sortides. Evidentment, si aquests senyals no són 0 és que el disseny no funciona correctament.

No us amoïneu si veieu que obteniu molts d'errors. El testbench prova moltes combinacions dels valors d'entrada. Aneu resolent els problemes d'un en un.

Taula 1: Recull d'operacions de la ALU segons 'Op'

Nom operand	Operació	Ор	Cout	F ₇	F_6	F ₅	F ₄	F ₃	F ₂	F ₁	F ₀
AplusB	A + B	0000									
AminusB	A - B	0001									
BminusA	B - A	0010									
OnlyA	Α	0100	A_7	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
OnlyB	В	0101	B ₇	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀
MinusA	-A	0110									
MinusB	-В	0111									
ShiftleftA	desp. esq. A	1000	A ₇	A ₆	A ₅	A_4	A ₃	A ₂	A ₁	A ₀	0
SiftrightA	desp. dret A	1001	0	0	A ₇	A_6	A_5	A_4	A_3	A ₂	A_1
RotateleftA	rot. esq. A	1010	A ₇	A ₆	A_5	A_4	A_3	A ₂	A ₁	A ₀	A ₇
RotaterightA	rot. dreta A	1011	0	A_0	A ₇	A ₆	A_5	A_4	A ₃	A ₂	A ₁
AllZeros	F = 0	1110	0	0	0	0	0	0	0	0	0
AllOnes	F = -1	1111	1	1	1	1	1	1	1	1	1