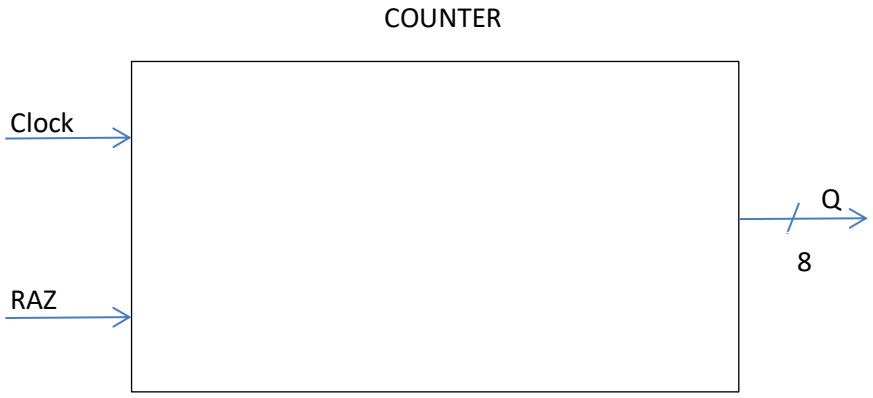
Correction du TD Modélisation VHDL

Exercice 1 :

➔ Boîte noire correspondant à l'entité :



Exercice 2 :

➔ Entité correspondant à la boîte noire de la Figure 2 :

```
Entity DFF is
        PORT(   clock : IN STD_LOGIC ;
                D : IN STD_LOGIC;
                Q : OUT STD_LOGIC
);
```

➔ Entité correspondant à la boîte noire de la Figure 3 :

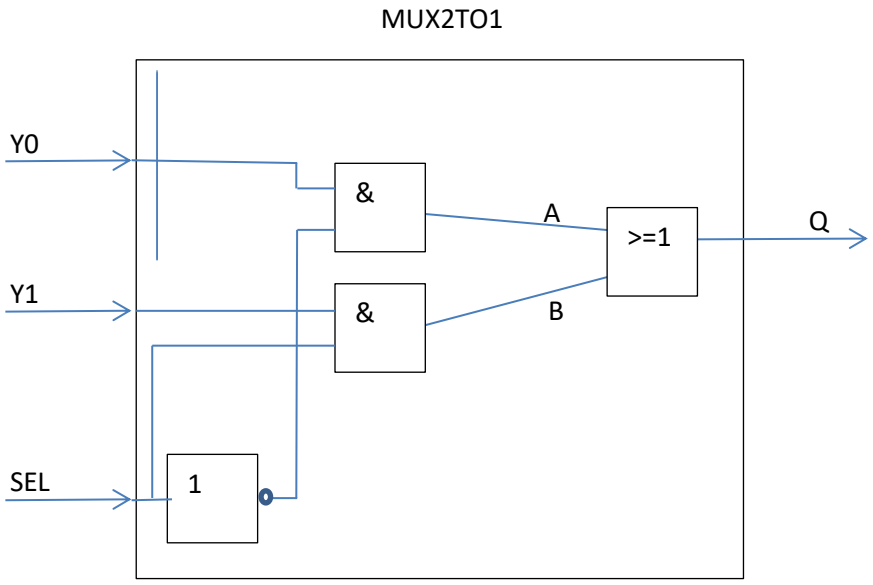
```
Entity REG8 is
        PORT(   clock : IN STD_LOGIC ;
                IN_ : IN STD_LOGIC_Vector(7 downto 0);
                OUT_ : OUT STD_LOGIC_Vector(7 downto 0)
        );
```

➔ Entité correspondant à la boîte noire de la Figure 4 :

```
Entity RAM1K is
        PORT(   CS_Bar : IN STD_LOGIC ;
                OE_Bar : IN STD_LOGIC;
                WE_Bar : IN STD_LOGIC;
                Adress : IN unsigned (9 downto 0);
                I_O : INOUT unsigned (7 downto 0)
        );
```

Exercice 3 :

➔ Boîte noire et schéma électronique en en portes logiques



Exercice 4 :

Pour chaque question, la déclaration des librairies et l'entité reste identique.

1. D'après la figure 4 et l'encadré de la dernière page du TD (Instruction concurrente puis Affectation sélective) :

```
ARCHITECTURE ARCHI2 OF MUX2TO1 IS
BEGIN
        WITH SEL SELECT
        Q <=    Y0 WHEN '0',
                Y1 WHEN '1',
                '0' WHEN OTHERS;
END ARCHI2;
```

2. D'après la figure 5 et l'encadré de la dernière page du TD (Instruction concurrente puis Affectation conditionnelle) :

```
ARCHITECTURE ARCHI3 OF MUX2TO1 IS
BEGIN
        Q <=    Y0 WHEN (SEL = '0'),
        ELSE Y1;
END ARCHI3;
```

3. D'après la figure 6 et l'encadré de la dernière page du TD (Processus puis Instruction séquentielle IF) :

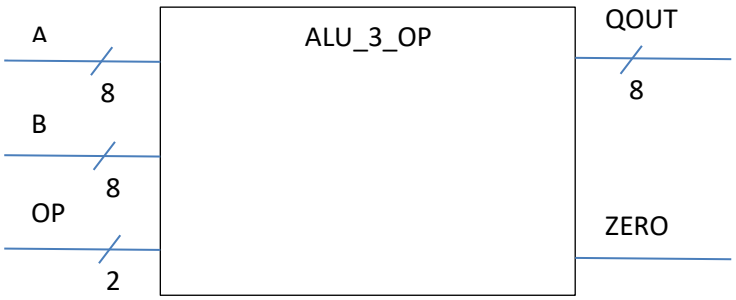
```
ARCHITECTURE ARCHI4 OF MUX2TO1 IS
BEGIN
        PROCESS(SEL, Y0, Y1)
        BEGIN
                IF SEL = '0' THEN
                Q <= Y0;
                ELSE
                Q <= Y1;
                END IF;
        END PROCESS;
END ARCHI4;
```

4. D'après la figure 7 et l'encadré de la dernière page du TD (Processus puis Instruction séquentielle CASE):

```
ARCHITECTURE ARCHI4 OF MUX2TO1 IS
BEGIN
        PROCESS(SEL, Y0, Y1)
        BEGIN
                CASE SEL IS
                        WHEN '0' => Q <= Y0;
                        WHEN '1' => Q <= Y1;
                        WHEN OTHERS => Q <= '0';
                END CASE;
        END PROCESS;
END ARCHI5;
```

Exercice 5 :

1. Boîte noire correspondante



2.

Si OP = "00" alors RESULT = 0&A + 0&B = ''0 0001 0010''
donc QOUT = ''0001 0010'' et ZERO = '0'
➔ Addition entre A et B

Si OP = "01" alors RESULT = 0&A - 0&B = ''0 0000 1100''
donc QOUT = ''0000 1100'' et ZERO = '0'
➔ Soustraction entre A et B

Si OP = "10" alors RESULT = 0&A AND 0&B = ''0 0000 0011''
donc QOUT = ''0000 0011'' et ZERO = '0'
➔ AND entre A et B

Si OP = "11" alors RESULT = ''0 0000 0000''
donc QOUT = ''0000 0000'' et ZERO = '1'
➔ Aucune opération

Le programme réalise donc une UAL pouvant effectuer le « + » et le « - » arithmétique et le « AND » logique ainsi qu'une détection de résultat nul.

3.

Afin de réaliser l'opération (30 – 20), il faut mettre 30 dans A (c'est à dire : A = ''0001 1110'') et mettre 20 dans B (c'est-à-dire : B = ''0001 0100'') et pour faire l'opérande « - » mettre OP = '01''.

4.

Pour que l'UAL puisse générer un signal de retenue C, il suffit d'ajouter les 2 lignes de code en rouge comme indiqué dans le programme ci-dessous.

Remarque : Le MSB correspondant au bit de poids le plus faible d'un mot binaire à l'inverse du LSB

```
--ALU.VHD
LIBRARY IEEE;
use ieee.std_logic_1164.all;   -- pour le type STD_LOGIC
use ieee.numeric_std.all;      -- pour le type UNSIGNED

ENTITY ALU_3_OP IS
        PORT(
                A : IN UNSIGNED (7 DOWNTO 0);
                B : IN UNSIGNED (7 DOWNTO 0);
                OP : IN UNSIGNED (1 DOWNTO 0);
                QOUT : OUT UNSIGNED (7 DOWNTO 0);
                ZERO : OUT STD_LOGIC);
END ALU_3_OP;

ARCHITECTURE a1_ALU OF ALU_3_OP IS
        SIGNAL RESULT : UNSIGNED (8 DOWNTO 0);
BEGIN
        ZERO <= '1' WHEN RESULT(7 DOWNTO 0) = "00000000" ELSE '0';
        QOUT <= RESULT (7 DOWNTO 0);
        WITH OP SELECT RESULT <=
                ('0'& A ) + ('0'& B) WHEN "00",
                ('0'& A ) - ('0'& B)WHEN "01",
                ('0'& A )AND('0'& B) WHEN "10",
                "000000000" WHEN       others;
END a1_ALU;
```

C : OUT STD LOGIC ;

C <= RESULT(8);