Nom : _____ Matricule : _____

Question 1. (2 points)

Donnez un diagramme d'un chemin des données implémentant les micro-opérations suivantes.

a. K1K2 : S ← (S * sll A, B) + C; K1K2' : S ← (S * slr A, B) + C
(où sll X, Y veut dire : faire un décalage logique vers la gauche du registre X du nombre de bits spécifié par le registre Y et slr X, Y veut dire : faire un décalage logique vers la droite du registre X du nombre de bits spécifié par le registre Y)

b. K1'K0': S ← S – 5; K1'K0: S ← S – 10 ; K1K0': S ← S – 25; K1K0: S ← S – 100;

Question 2. (4 points)

a. (2 points) Soit l'entité module1 ainsi que le banc d'essai module1_tb permettant de le vérifier. Les codes VHDL de ces modules sont comme suit:

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

entity module1 is port (
      A, B : in std_logic;
      F : out std_logic );
end module1;
architecture arch of module1 is
      signal S1, S2: std_logic;
begin
      S1 <= A and B;
      S2 <= not(B);
      process (S1, S2)
            begin
                  F <= S1 xor S2;
      end process;
end arch;
```

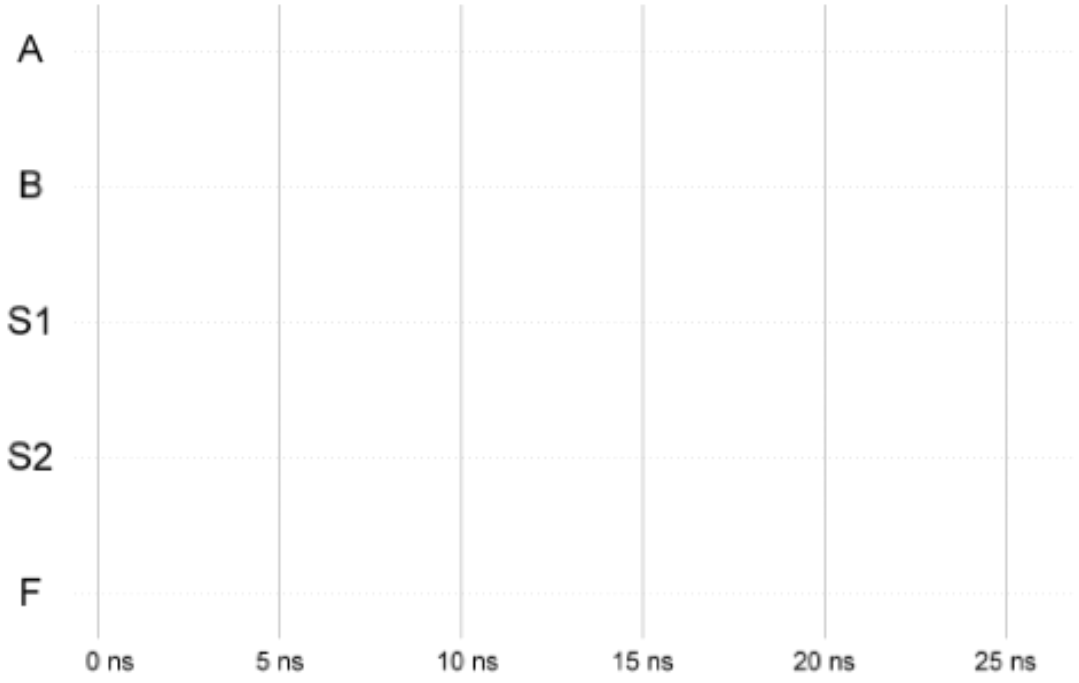
```
library IEEE;
use IEEE.std_logic_1164.all;

entity module1TB is
end module1TB;

architecture arch of module1TB is
      component module1
            port (
                  A, B : in std_logic;
                  F : out std_logic );
      end component;
      signal A,B,F : std_logic;
begin
      UUT : module1
      port map (A, B, F);
      A <= '0' after 0 ns, '1' after 20 ns;
      B <= '1' after 0 ns, '0' after 10
      ns,'1' after 15 ns;
end arch;
```

Donnez le chronogramme des ports A, B, F et des signaux internes S1 et S2 durant une simulation comprise entre les temps 0 ns jusqu'à 25 ns. **Remplissez le gabarit de la page suivante.**

b. (1 point) Expliquez le principe de la vérification exhaustive et la vérification pseudo aléatoire. Discutez ces types de vérification dans le contexte de l'utilisation du langage VHDL.

c. (1 point) Donnez les quatre tâches d'un banc d'essai permettant la vérification d'un circuit.

Feuille de réponse question 2



Question 3. (3 points)

Considérez le code VHDL suivant. Donnez un schéma du circuit correspondant à ce code.

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity Entity1 is
      generic (
            W : positive := 16
      );
      port (
            clk, reset : in std_logic;
            A, B, C, D : in signed(W - 1 downto 0);
            E, F : out signed(2 * W downto 0)
      );
end Entity1;



architecture arch of Entity1 is
signal At, Bt, Ct, Dt : signed(W - 1 downto 0);
signal AC, AD, BC, BD : signed(2 * W - 1 downto 0);
begin
      process(clk, reset)
      begin
            if rising_edge(clk) then
                  if reset = '1' then
                        At <= (others => '0'); Bt <= (others => '0');
                        Ct <= (others => '0'); Dt <= (others => '0');
                        AC <= (others => '0'); BC <= (others => '0');
                        AD <= (others => '0'); BD <= (others => '0');
                        E <= (others => '0'); F <= (others => '0');


                  else
                        At <= A; Bt <= B; Ct <= C; Dt <= D;
                        AC <= At * Ct; AD <= At * Dt;
                        BC <= Bt * Ct; BD <= Bt * Dt;
                        E <= resize(AC, E'length) - resize(BD, E'length);
                        F <= resize(AD, F'length) + resize(BC, F'length);
                  end if;
            end if;
        end process;
end arch;
```

Question 4. (3 points)

a. Donnez une courte définition pour les notions suivantes :

i. liste de sensitivité pour un processus VHDL

ii. CPLD

iii. GAL

b. Donnez le circuit spécifié dans le code suivant

i.

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity entity1 is
        generic (
                W : integer := 8
        );
        port(
                reset : in STD_LOGIC;
                CLK : in STD_LOGIC;
                mode : in STD_LOGIC_VECTOR(1 downto 0);
                entreeSerie : in STD_LOGIC;
                D : in STD_LOGIC_VECTOR(W - 1 downto 0);
                Q : out STD_LOGIC_VECTOR(W - 1 downto 0)
        );
end entity1;
architecture arch of entity1 is
begin
        process (CLK, reset)
        variable Qinterne : STD_LOGIC_VECTOR(W - 1 downto 0);
        begin
                if reset='0' then
                        Qinterne := (others => '0');
                        Q <= (others => '0');
            elsif CLK='1' and CLK'event then
                 case mode is
                        when "00" =>
                                Qinterne := Qinterne;
                        when "01" =>
                                Qinterne := D;
                        when "10" =>
                                Qinterne := Qinterne(W - 2 downto 0) & entreeSerie;
                        when "11" =>
                                Qinterne := entreeSerie & Qinterne(W - 1 downto 1);
                        when others =>
                                Qinterne := Qinterne;
                        end case;
                        Q <= Qinterne;
                end if;
        end process;
end arch;
```

ii.

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

entity entity2 is
     generic (
            n : positive := 3
     );
     port (
            D : in std_logic_vector(2 ** n - 1 downto 0);
            S: in unsigned(n - 1 downto 0);
            F : out std_logic
     );
end entity2;

architecture arch2 of entity2 is
begin
     process (D, S)
     begin
            F <= D(to_integer(S));
     end process;
end arch2;
```

Question 5. (3 points)

Lors du dernier laboratoire, nous avons vu de quelle manière il est possible d'implémenter la division en matériel à l'aide d'une *look-up table*. En effet, pour effectuer l'opération a/b, nous mettions en mémoire morte (ROM) les résultats de 1/b afin de faire la multiplication a*(1/b).

a) Quels sont les nécessités en terme de ressources FPGA (slices, multiplieurs, mémoire) afin d'implémenter la table comportant les résultats de 1/b? (b est encodé sur 8 bits, est le résultat de 1/b comporte 8 bits de partie entière et 2 bits de partie fractionnaire, en virgule fixe)

b) Discutez de la plus grande limitation de cette approche. Donnez une autre méthode pour faire le calcul et discutez ses avantages et inconvénients.

c) En utilisant le code VHDL suivant :

```
    ......
component frac is
    generic(
           Went : integer := 8;
           Wfrac : integer := 4
    );
    port (
           a : in  unsigned(Went - 1 downto 0);
           b : in  unsigned(Went - 1 downto 0);
           f : out unsigned(Went + Wfrac - 1 downto 0)
    );
end component frac;

signal a,b : unsigned(7 downto 0);

begin

frac1:   frac
       GENERIC MAP(Went  => 8, --nombre bit de la partie entière
                   Wfrac => 2) --nombre de bit de la partie fractionnaire
           PORT MAP(a =>num,    --numérateur
                    b =>den,    --dénominateur
                    f =>res);   --résultat de la division

a <= ''00001000'';
b <= ''00000010'';
sortie <= res*to_unsigned(3,10);
```

Que vaut le signal *sortie*?

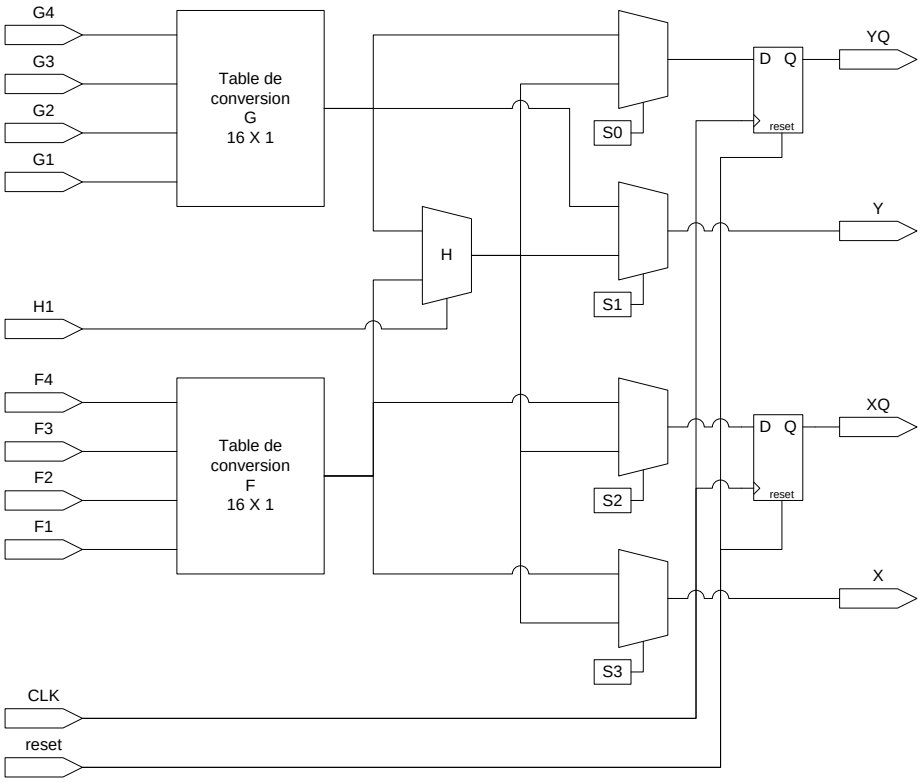
Question 6. (5 points)

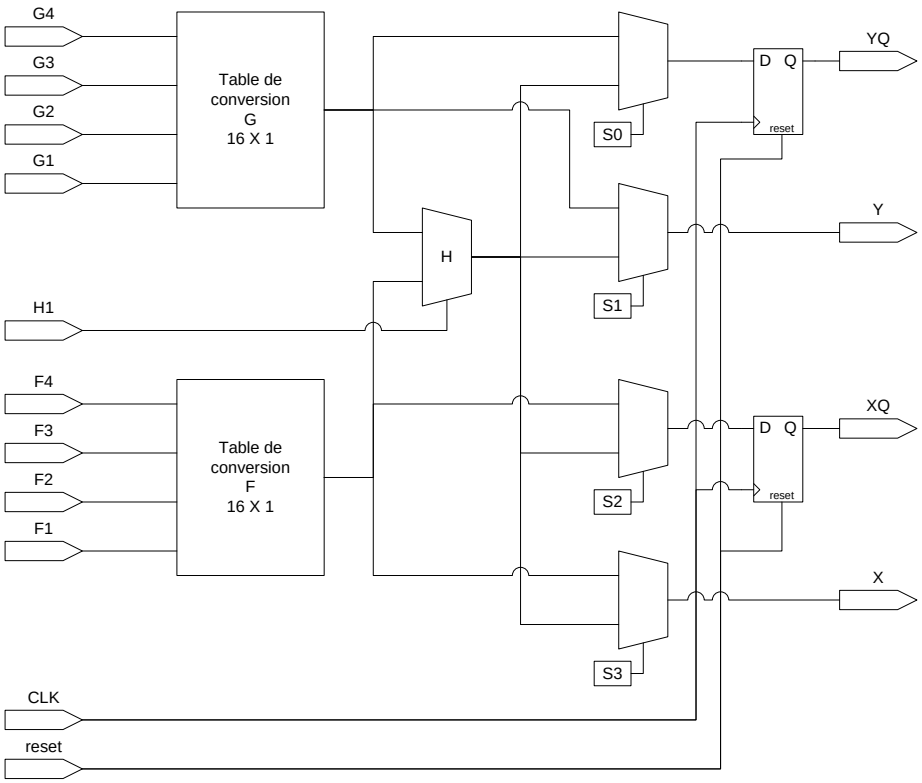
Considérez le code VHDL suivant. Montrez, sur le modèle de FPGA fourni **aux pages suivantes**, un résultat possible de la synthèse et de l'implémentation de ce code.

a. Indiquez directement sur le dessin où chaque signal se situe ainsi que les interconnections entre les blocs.

b. Indiquez dans les tables de vérité fournies le contenu de chacune des tables de conversion que vous utilisez.

```
library ieee;
use ieee.std_logic_1164.all;
entity monModule6 is
        port (
                reset, clk: in std_logic;
                A, B, C, D, E, F : in std_logic;
                Y, Z : out std_logic
        );
end monModule6;
architecture arch of monModule6 is
signal T1, T2 : std_logic;
begin
        T1 <= A or B;
        Z <= (T1 or T2) and (D or C);
        process(CLK, reset) is
        begin
            if (reset = '1') then
                        T2 <= '0';
                        Y <= '0';
                elsif (rising_edge(CLK)) then
                        T2 <= (A and D) xor F;
                        Y <= (T2 and B) xor (E and F);
                end if;
        end process;
end arch;
```





Feuille de réponse pour la question 6 a.

G4	G3	G2	G1	G
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

F4	F3	F2	F1	F
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

G4	G3	G2	G1	G
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

F4	F3	F2	F1	F
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Feuille de réponse question 6.b