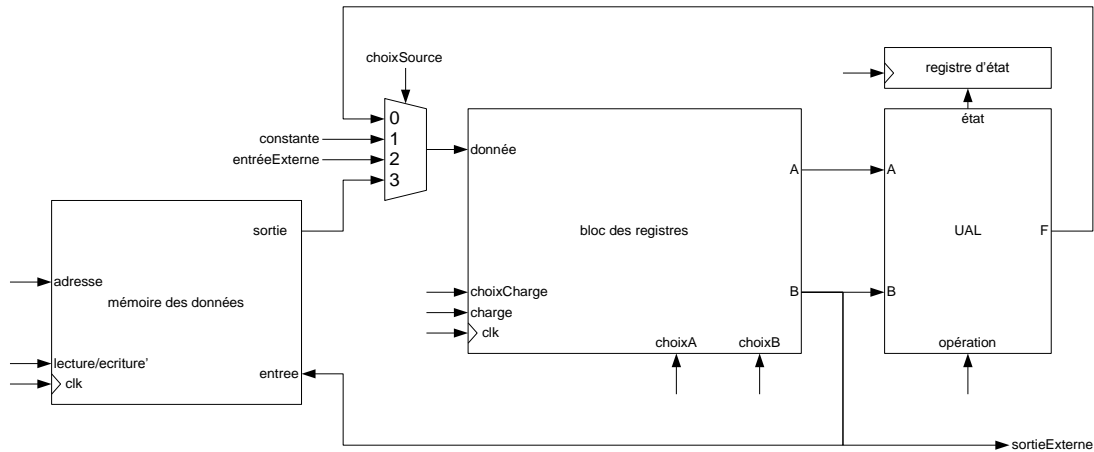


Question 1. (3 points)

Considérez le processeur d'usage général présenté dans les notes de cours. L'architecture de son chemin de données est illustrée à la Figure 1. Les codes VHDL donnés à la Figure 2 rappellent le fonctionnement de l'ALU et de la mémoire.

**Figure 1**

- dans la partie déclarative de l'architecture

```
signal F : signed(Wd - 1 downto 0);
signal Z : std_logic;
signal N : std_logic;
signal op : integer range 0 to 7;
```

-- dans le corps de l'architecture

```
process(A, B, op)
begin
  case op is
    when 0 => F <= A + B;
    when 1 => F <= A - B;
    when 2 => F <= shift_right(A, 1);
    when 3 => F <= shift_left(A, 1);
    when 4 => F <= not(A);
    when 5 => F <= A and B;
    when 6 => F <= A or B;
    when 7 => F <= A;
    when others => F <= (others => 'X');
  end case;
end process;
```

-- dans la partie déclarative de l'architecture

```
type memoireDonnees_type is array(0 to 2 ** Md - 1) of signed(Wd - 1 downto 0);
signal memoireDonnees : memoireDonnees_type;
signal sortieMemoireDonnees : signed(Wd - 1 downto 0);
signal adresseMemoireDonnees : integer range 0 to 2 ** Md - 1;
signal lectureEcritureN : std_logic;
```

-- dans le corps de l'architecture

```
-- mémoire des données
process (CLK)
begin
  if rising_edge(CLK) then
    if lectureEcritureN = '0' then
      memoireDonnees(adresseMemoireDonnees) <= Entrée;
    end if;
  end if;
end process;

sortieMemoireDonnees <= memoireDonnees(adresseMemoireDonnees);
```

Figure 2

a. Complétez la table suivante, en donnant les valeurs des signaux de contrôle pour l'ensemble d'instructions indiquées dans la première colonne.

Instruction	choixSource	choixCharge	charge	choixA	choixB	opération	Lecture/Ecriture
$R0 \leftarrow R2$							
$R2 \leftarrow R1 + R4$							
$M[25] \leftarrow R1$							
$R3 \leftarrow M[50]$							
$R2 \leftarrow R2 \text{ ou } R1$							

b. Complétez la Figure 1 en ajoutant l'architecture de l'unité de contrôle du processeur considéré. Considérez une architecture de type Harvard.

Question 2. (6 points)

Considérez le circuit représenté à la Figure 3. Les bascules ont un temps de préparation de 1 ns, un temps de maintien de 0.5 ns et un délai de propagation de 2 ns. Négligez les délais des interconnexions.

- a. Identifiez le chemin critique et déterminez la fréquence maximale d'horloge.
- b. Ajoutez un niveau de pipeline et donnez la fréquence maximale d'horloge résultante.
- c. Déterminez la marge libre de préparation de chaque chemin.
- d. Quel est l'effet d'avoir un déphasage d'horloge égal à 1 ns sur le circuit sans pipeline?

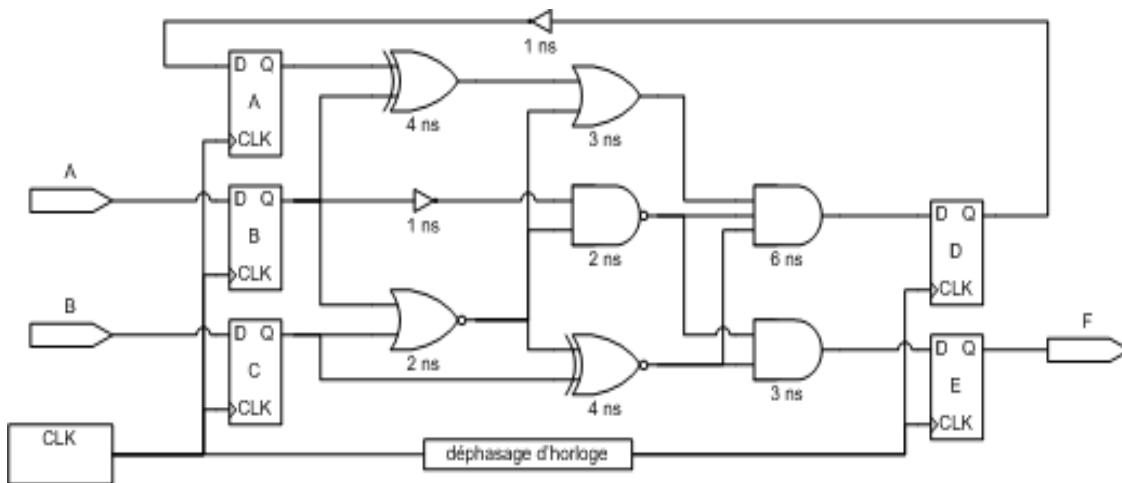


Figure 3

Question 3. (3.5 points)

Considérez le code suivant pour réaliser un circuit effectuant un tri sur quatre nombres en entrées : a, b, c et d. Ainsi, si $(a, b, c, d) = (6, 4, 2, 8)$, les sorties devraient être affectées après 5 cycles d'horloge, de sorte que $(x, y, z, w) = (2, 4, 6, 8)$.

```

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

entity trieuse is
    generic(
        width: positive :=18);
    port(
        clk : in std_logic;
        a, b, c, d : in unsigned(width-1 downto 0);
        x, y, z, w : out unsigned(width-1 downto 0)
    ); end trieuse;

architecture arch of trieuse is

    type aunsigned is array (0 to 3) of unsigned(width-1 downto 0);
    type a_unsigned is array(0 to 4) of aunsigned;
    signal aus : a_unsigned;

begin
    process( clk ) is
    begin
        if ( rising_edge( clk ) ) then
            for i in 0 to 1 loop
                for j in 0 to 1 loop
                    if( aus( 2*i )( 2*j ) > aus( 2*i )( 2*j+1 ) ) then
                        aus( 2*i+1 )( 2*j ) <= aus( 2*i )( 2*j+1 );
                        aus( 2*i+1 )( 2*j+1 ) <= aus( 2*i )( 2*j );
                    else
                        aus( 2*i+1 )( 2*j ) <= aus( 2*i )( 2*j );
                        aus( 2*i+1 )( 2*j+1 ) <= aus( 2*i )( 2*j+1 );
                    end if;
                end loop;
            end loop;

            aus(2*i+2)(0) <=aus(2*i+1)(0);
            aus(2*i+2)(3) <=aus(2*i+1)(3);

            if( aus( 2*i+1 )( 1 ) > aus( 2*i+1 )( 2 ) ) then
                aus(2*i+2)(1) <=aus(2*i+1)(2);
                aus(2*i+2)(2) <=aus(2*i+1)(1);
            else
                aus(2*i+2)(1) <=aus(2*i+1)(1);
                aus(2*i+2)(2) <=aus(2*i+1)(2);
            end if;
        end if;
    end process;
end architecture arch;

```

```

    end loop;

    aus( 0 ) <= (a, b, c, d);
end if;
end process;
(x, y, z, w) <= aus( 4 );
end arch;

```

- Quel est le nombre de vecteurs de test nécessaire pour effectuer un test exhaustif de l'entité `trieuse` ? Justifiez votre réponse par un calcul.
- Votre collègue vous fournit le banc d'essai suivant qu'il affirme avoir utilisé pour s'assurer du bon fonctionnement de son circuit `trieuse`.

```

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

entity trieuse_tb is
    generic(width : positive := 4);
end trieuse_tb;

architecture tb of trieuse_tb is
    component trieuse is
        generic(
            width : positive := 18
        );
        port(
            clk : in std_logic;
            a, b, c, d : in unsigned(width-1 downto 0);
            x, y, z, w : out unsigned(width-1 downto 0);
        );
    end component;

    signal clk : std_logic := '1';
    signal rst : std_logic := '1';

    type aunsigned is array (0 to 3) of unsigned(width-1 downto 0);
    type tb_a_unsigned is array(0 to 7) of aunsigned;
    constant aus : tb_a_unsigned :=
        ((x"2", x"4", x"6", x"8"), (x"2", x"4", x"8", x"6"),
         (x"2", x"6", x"4", x"8"), (x"2", x"6", x"8", x"4"),
         (x"2", x"8", x"4", x"6"), (x"2", x"8", x"6", x"4"),
         (x"4", x"2", x"6", x"8"), (x"4", x"2", x"8", x"6"));

    signal entrees, sorties : aunsigned;
    signal a, b, c, d, x, y, z, w : unsigned(width-1 downto 0);

```

```
begin

    dut : trieuse generic map(width) port map(clk, a, b, c, d, x, y, z, w);

    rst <= '1' after 0 ns, '0' after 15 ns;
    clk <= not( clk ) after 5 ns;

    (a, b, c, d) <= entrees;
    sorties <= (x, y, z, w);

    process( clk ) is
        variable i : integer := 0;
    begin
        if( rst = '1' ) then
            entrees <= (x"0", x"0", x"0", x"0");
            i := 0;
        elsif( falling_edge( clk ) ) then
            if ( i >= 5 ) then
                assert ( sorties = (x"2", x"4", x"6", x"8") )
                report "Erreur sur les entrees" severity error;
            end if;

            if ( i < 8 ) then
                entrees <= aus( i );
                i := i + 1;
            elsif( i = 13 ) then
                report "Fin de simulation simulation" severity failure;
            else
                i := i + 1;
            end if;
        end if;
    end process;
end tb;
```

Ce banc d'essai vous assure-t-il hors de tout doute du bon fonctionnement du circuit ? Justifiez clairement votre réponse.

- c. Donnez en pourcentage la couverture de branchements obtenue en appliquant ce banc d'essai. Justifiez votre réponse par un calcul.

Question 4. (4 points)

- Donnez trois (3) exemples pour lequel un code VHDL n'est pas synthétisable (l'écriture des codes n'est pas obligatoire – expliquez surtout les caractéristiques des codes).
- Quels sont les objectifs du processus de placement ?
- Quels sont les objectifs du processus de routage ?

- d. Donnez la liste des événements, telle quelle pourrait être dressée par un simulateur qui exécuterait le code suivant:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity entity1 is
    port (
        X : in std_logic;
        Y : in std_logic;
        Z : in std_logic;
        A : out std_logic;
        B : out std_logic
    );
end entity1;

architecture arch1 of entity1 is
    signal T1 : std_logic;
    signal T2 : std_logic;

begin
    A <= T1 and X;
    B <= T1 or T2;
    T1 <= X xor Y;
    T2 <= X or Y;

end arch1;

end flotdonnees2;
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity entity1TBsimple is
end entity1TBsimple;

architecture archTB of entity1TBsimple is
    signal Cin, X, Y, Cout, S : std_logic;
begin
    UUT : entity entity1(arch1)
        port map (X, Y, Z, A, B);
    X <= '0' after 0 ns;
    Y <= '0' after 0 ns, '1' after 10 ns;
    Z <= '0' after 0 ns;
end archTB;
```

Question 5. (3.5 points)

1. Donnez le chemin du flot de conception des circuits numériques discutés en classe.
2. Expliquez en 2-3 lignes la démarche pour le changement de la tâche d'un processeur à usage spécifique. Donnez également cette démarche dans le cas d'un processeur à usage général.
3. Discutez l'exploitation de l'architecture pipeline dans le cas des FPGA.