Question 1. (2.5 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 extraits du code VHDL donnés ci-après rappellent le fonctionnement de l'ALU et de la mémoire.

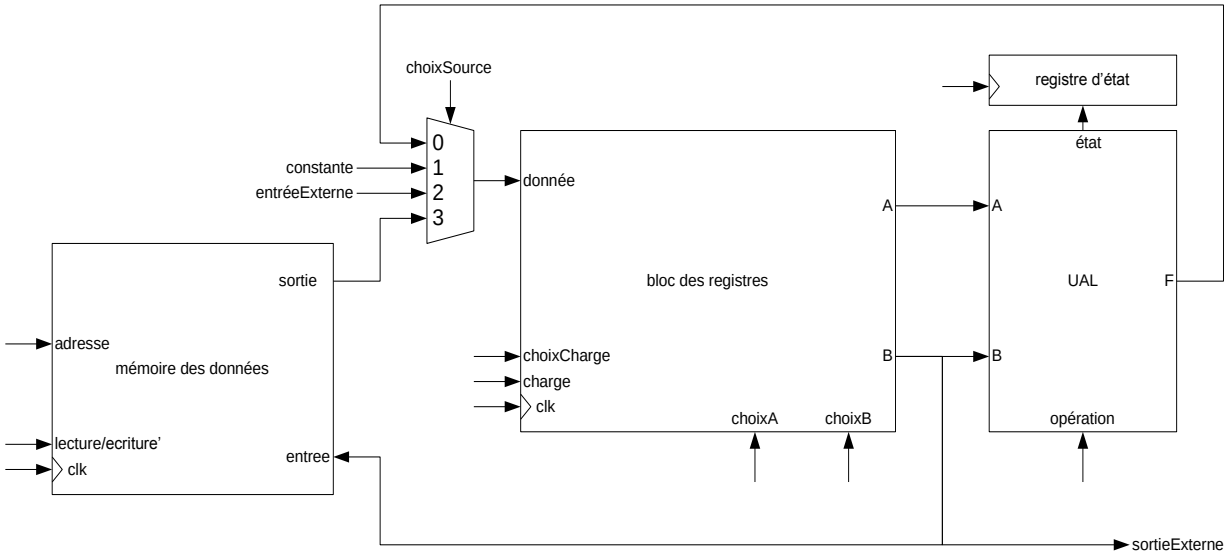


Figure 1

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

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
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;


process (CLK)
begin
    if rising_edge(CLK) then
        if lectureEcritureN = '0' then
            memoireDonnees(adresseMemoireDonnees) <= B;
        end if;
    end if;
end process;


sortieMemoireDonnees <= memoireDonnees(adresseMemoireDonnees);
```

a. Complétez la Table 1 suivante, en donnant les instructions selon les valeurs des signaux de contrôle indiquées.

Instruction	choixSource	choixCharge	charge	choixA	choixB	opération	Lecture/Ecriture
	0	2	1	4	-	7	1
	-	-	0	-	2	-	0
	0	3	1	1	4	0	1
	3	4	1	-	-	-	1

Table 1

Ajoutez une instruction de la forme Rk _ #WXYZ, permettant de charger le registre *k* avec une valeur immédiate de 16 bits de large (WXYZ représente 4 chiffres hexadécimaux). Cette instruction nécessite deux mots de 16 bits dans la séquence des instructions : le premier représente l'instruction (chargement de valeur immédiate de 16 bits) et le deuxième contient la valeur WXYZ.

b. Modifiez le diagramme d'états de l'unité de contrôle donné à l'**Annexe I** pour ajouter cette nouvelle instruction.

c. Donnez l'encodage de votre nouvelle instruction. Au besoin, inspirez-vous de la Table 2.

instruction	bits 15-12	bits 11-8	bits 7-4	bits 3-0
$R_{dest} \leftarrow R_{source1} \Diamond R_{source2}$	$\{0 \mid op_2 \mid op_1 \mid op_0\}$	destination	source1	source2
$R_{dest} \leftarrow$ MD[adresse]	1000	destination	adresse[7:4]	adresse[3:0]
MD[adresse] $\leftarrow R_{source}$	1001	source	adresse[7:4]	adresse[3:0]
JUMP adresse	1100	condition	adresse[7:4]	adresse[3:0]
STOP	1111	-	-	-

Table 2

Question 2. (3 points)

Un système embarqué équipé d'un processeur à but spécifique contrôle l'entrée dans un garage. L'entrée est permise si deux jetons sont déposés par les conducteurs. Donnez le schéma du chemin de données et la machine à états de l'unité de contrôle de ce système. Le système a une entrée qui indique la présence d'un nouveau jeton et un signal de sortie pour accorder l'accès.

Question 3. (4 points)

Soit le circuit représenté à l'**Annexe II**. 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. Le temps de propagation dans les portes ET, OU, XOR et INV de 3 ns, 3 ns, 4 ns, 1 ns respectivement. Négligez les délais des interconnexions.

a. Déterminez la fréquence maximale d'horloge.

b. Déterminez la marge libre de préparation de chaque chemin.

c. Quelles sont les bornes minimales et maximales du déphasage d'horloge acceptables tout en garantissant cette fréquence maximale?

d. En considérant un déphasage d'horloge nul, modifiez le circuit donné à l'**Annexe II** de sorte à ajouter un étage de pipeline permettant d'augmenter la fréquence maximale. Déterminez la nouvelle fréquence maximale.

Question 4. (2.5 points)

a. Considérez le code suivant pour un module combinatoire et son banc de test.

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

entity module is
     port (
          A, B, C : in std_logic;
          F, G : out std_logic
     );
end module;

architecture arch of module is
signal S, T, U : std_logic;
begin
     S <= A or B;
     T <= A xor B;
     U <= A and C;

     process (S, T, C)
     begin
          F <= S and T and C;
     end process;

     process (B)
     begin
          G <= B or U;
     end process;
end arch;

library ieee;
use ieee.std_logic_1164.all;

entity module_TB is
end module_TB;

architecture arch of module_TB is
     signal A, B, C, F, G : std_logic;
begin
     UUT : entity module(arch) port map (A, B, C, F, G);
     A <= '1' after 0 ns;
     B <= '0' after 0 ns;
     C <= '0' after 0 ns, '1' after 10 ns;
end arch;
```

Donnez la liste des événements dressée par un simulateur qui exécute le banc d'essai.

b. Discutez l'utilisation des boucles et conditions dans un code VHDL synthétisable.

Question 5. (4 points)

Au laboratoire, vous avez eu à concevoir un processeur à usage général dont les instructions tiennent sur 28 bits et dont la déclaration d'entité suit. Le jeu d'instructions de ce processeur est donné à l'**Annexe III**.

```
entity processeurv20 is
    generic (
        Nreg        : integer   := 32; -- nombre de registres
        Wd          : integer   := 16; -- largeur du chemin des données en bits
        Wi          : integer   := 28; -- largeur des instructions en bits
        Mi          : integer   := 12; -- nombre de bits d'adresse de la mémoire d'instructions
        Md          : integer   :=  8; -- nombre de bits d'adresse de la mémoire des données
        resetvalue  : std_logic := '1'
    );
    port(
        reset               : in  std_logic;
        clk                 : in  std_logic;
        entreeExterne       : in  signed(Wd - 1 downto 0);
        fifoInNotReady      : in  std_logic;
        reqEntreeExterne    : out std_logic;
        sortieExterne       : out signed(Wd - 1 downto 0);
        fifoOutNotReady     : in  std_logic;
        ecrireSortieExterne : out std_logic
    );
end processeurv20;
```

a. Quelle sera la valeur du registre `r0` à la fin de l'exécution de ce programme ?

```
constant memoireInstructions : memoireInstructions_type :=
(x"A080018", -- r8    <- +24               / addr x00
 x"A090001", -- r9    <- +1                / addr x01
 x"A000000", -- r0    <- +0                / addr x02
 x"9000000", -- M[r0] <- r0                / addr x03
 x"0000009", -- r0    <- r0 + r9           / addr x04
 x"9000000", -- M[r0] <- r0                / addr x05
 x"0000009", -- r0    <- r0 + r9           / addr x06
 x"10A0008", -- rA    <- r0 - r8           / addr x07
 x"C040012", -- jump to addr x12 si N = '0' / addr x08
 x"1000009", -- r0    <- r0 - r9           / addr x09
 x"8020000", -- r2    <- M[r0]             / addr x0A
 x"1000009", -- r0    <- r0 - r9           / addr x0B
 x"8010000", -- r1    <- M[r0]             / addr x0C
 x"0000009", -- r0    <- r0 + r9           / addr x0D
 x"0000009", -- r0    <- r0 + r9           / addr x0E
 x"0030102", -- r3    <- r1 + r2           / addr x0F
 x"9000003", -- M[r0] <- r3                / addr x10
 x"C000006", -- jump to addr x06           / addr x11
 x"1000009", -- r0    <- r0 - r9           / addr x12
 x"8000000", -- r0    <- M[r0]             / addr x13
others => (others => '1'));
```

b. Quel est le nombre de cycles d'horloge nécessaire à l'exécution du programme ? Justifiez clairement votre réponse en énonçant vos hypothèses de travail.

c. Décrivez dans un pseudo-code le plus lisible possible l'algorithme précédent. On supposera que l'algorithme retourne la valeur du registre `r0`.

d. Il n'est pas nécessaire pour exécuter ce programme de recourir à la mémoire de données. Proposez un programme qui exécute le même algorithme sans utiliser la mémoire de données. Vous êtes libre d'utiliser tous les registres à votre disposition mais vous devez absolument vous assurer que le résultat de votre programme se retrouve au registre `r0` à la fin de son exécution.

Question 6. (4 points)

On vous demande de réaliser un circuit permettant de trouver la valeur médiane de trois nombres signés donnés en entrée : a, b et c. Ainsi, si (a, b, c) = (+3, +8, +5), la sortie devrait être z = +5. Si (a, b, c) = (+3, +8, +3), la sortie devrait être z = +3. Si (a, b, c) = (+8, +8, +3), la sortie devrait être z = +8.

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

entity mediane3 is
generic(
    w : positive := 8
);
port(
    a : in signed(w-1 downto 0);
    b : in signed(w-1 downto 0);
    c : in signed(w-1 downto 0);
    z : out signed(w-1 downto 0)
);
end mediane3;
```

a. Écrivez le code VHDL synthétisable de l'entité **mediane3**.

b. Quelle est la plus petite taille de l'ensemble de vecteurs de test permettant d'effectuer un test exhaustif de l'entité **mediane3** lorsque w=8 ? Justifiez votre réponse par un calcul.

On vous propose trois classes pour chacune des entrées du circuit **mediane3** (on suppose $w \geq 2$) :

$\{ \{-2^{w-1}, -2^{w-1} + 1, \ldots -2^{w-2} - 1\},$
$\{-2^{w-2}, \ldots +2^{w-2} - 1\},$
$\{+2^{w-2}, +2^{w-2}+1, \ldots +2^{w-1} - 1\} \}$

c. Quelle est la taille minimale de l'ensemble de vecteurs de test permettant d'effectuer un test faible sur l'entité mediane3 et ce, en recourant aux classes identifiées précédemment. Justifiez votre réponse et proposez un exemplaire d'un tel ensemble pour w = 8.

d. Quelle est la taille minimale de l'ensemble de vecteurs de test permettant d'effectuer un test fort sur l'entité mediane3 et ce, en recourant aux classes identifiées précédemment. Justifiez votre réponse et proposez un exemplaire d'un tel ensemble pour w = 8.

e. Quelle est la couverture de branchement de l'implémentation proposée à la question (a) en utilisant l'ensemble de vecteurs de test proposé telle qu'à la question (d). Justifiez adéquatement votre réponse.

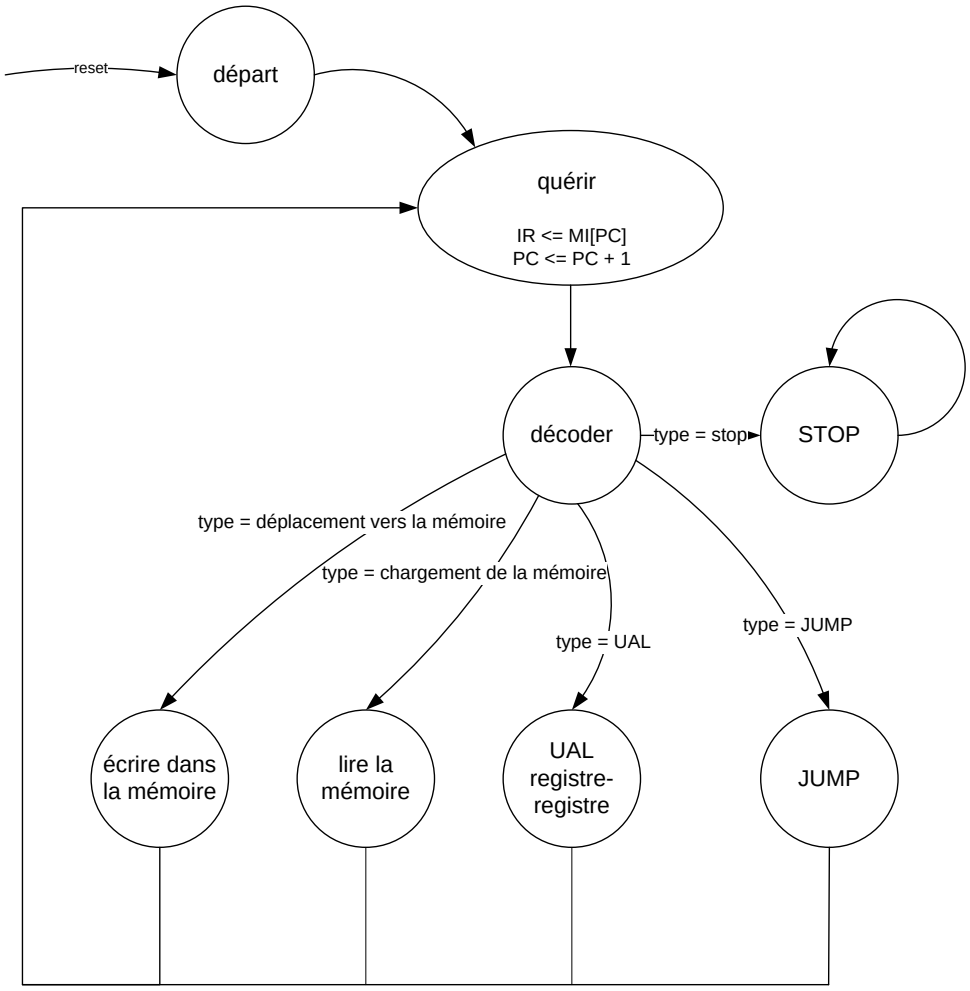
Annexe I – Question 1.b.

Joindre cette feuille à votre cahier d'examen.

NOM:_____

PRÉNOM:_____

MATRICULE:_____



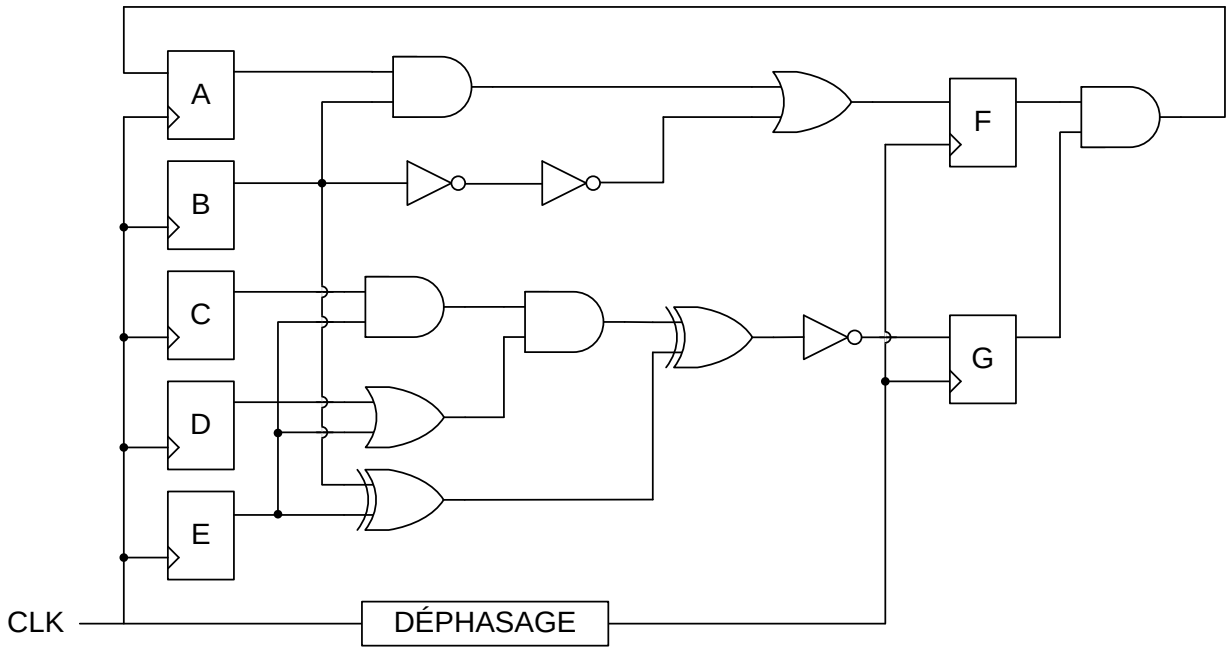
Annexe II – Question 3. a. b. c. d.

Joindre cette feuille à votre cahier d'examen. Indiquez les registres de pipeline par des barres verticales.

NOM:_____

PRÉNOM:_____

MATRICULE:_____



Annexe III – Question 5

Jeu d'instructions du processeurv20

Instructions arithmétiques: $r_{dest} \leftarrow A$ op B:

opération	choixCharge	choixA	choixB
IR[27:24] Valeurs permises: x"0" : F ← A + B x"1" : F ← A – B x"2" : F ← A >> 1 x"3" : F ← A << 1 x"4" : F ← not(A) x"5" : F ← A and B x"6" : F ← A or B x"7" : F ← A	IR[20:16] Adresse du registre de destination	IR[12:8] Adresse du registre de l'opérande A	IR[4:0] Adresse du registre de l'opérande B

Instructions pour l'interfaçage de la mémoire de données:

Instructions	choixCharge	choixA	choixB
IR[27:24] Valeurs permises: x"8": *lire* x"9": *écrire*	IR[20:16] Adresse du registre de destination (s'applique à *lire*)	IR[12:8] Adresse du registre de l'opérande A, servant à l'adressage de la mémoire (s'applique autant à *lire* qu'à *écrire*)	IR[4:0] Adresse du registre de l'opérande B qui sert à contenir la donnée à *écrire*

Instruction de chargement immédiat:

Instruction	Valeur à charger
IR[27:24] Valeurs attendue: x"A"	IR[15:0] Représentation signée

Instructions de branchement

Instruction	Type de branchement	Adresse de branchement
IR[27:24] Valeur attendue: x"C"	IR[19:16] Valeurs permises: x"0" (sans condition) x"1" si = 0 x"2" si ≠ 0 x"3" si < 0 x"4" si ≥ 0	IR[11:0] Adresse de l'instruction de destination

Instruction d'arrêt

Instruction	
IR[27:24] Valeur attendue: x"F"	