INF3500 : Conception et réalisation de systèmes numériques

Examen intra #1 – 10 février 2022

Q1	
Q2	
Q3	
Q4	
Total	

Durée : 1 heure. Documentation : Une feuille recto verso 8.5"×11" ou A4 permise.
Pondération: 10%. Calculatrice: Programmable permise.
Directives particulières :

- Ordinateurs interdits. Appareils mobiles interdits.
- Répondre à toutes les questions, la valeur de chaque question est indiquée.
- Répondre sur le questionnaire et le remettre.
- Ne posez pas de questions. En cas de doute sur le sens d'une question, énoncez clairement vos suppositions.

Question 1. (2 points) (12 minutes)

Faites la conception d'un circuit CMOS qui implémente le code VHDL suivant. Utilisez le moins de transistors possible.

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

entity combinatoire2b is
    port (
        A, B, C, D: in std_logic;
        F : out std_logic
    );
end combinatoire2b;
```

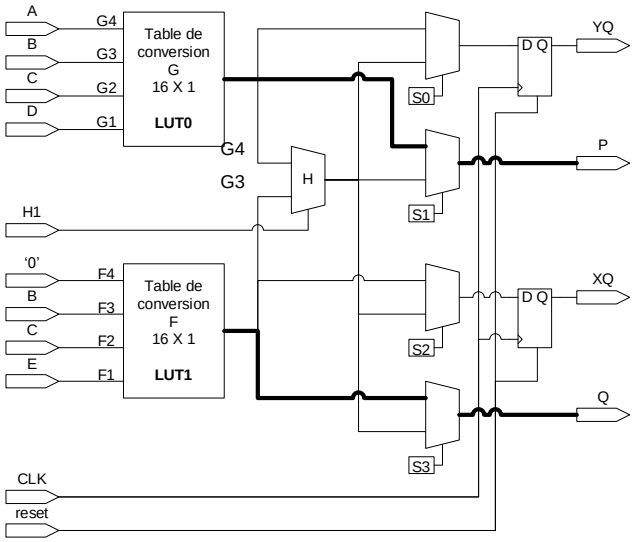
```
architecture arch of combinatoire2b is

signal M, N : std_logic;

begin
    M <= C and not(D);
    N <= not(B) or M;
    F <= not(A) and N;
end arch;
```

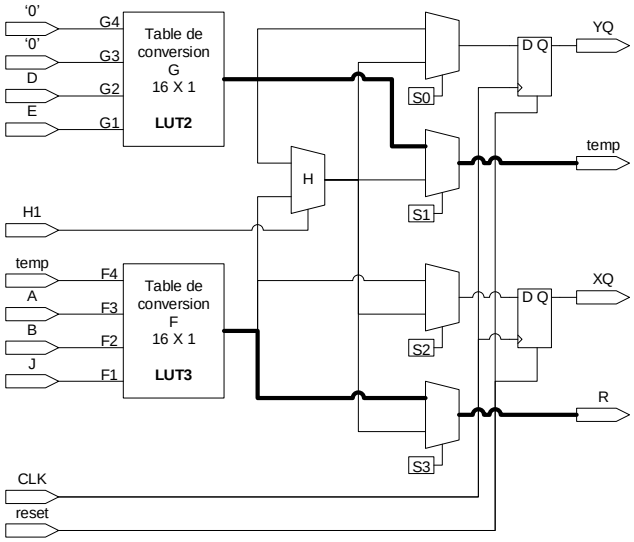
Question 2. (2 points) (12 minutes)

Considérez le code VHDL suivant et le diagramme annoté d'un FPGA dans lequel ce code a été implémenté. Donnez le contenu des tables de vérité du FPGA.

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

entity combinatoire6a is
  port (
        A, B, C, D, E, J : in std_logic;
        P, Q, R : out std_logic
    );
end combinatoire6a;

architecture arch of combinatoire6a is
begin

P <= '1' when A and B and C and not(D)
else '1' when not(B) and D and
((not(A) and not(C)) or (A and C)) else '0';

R <= (A and not(B) and J) xor (D or not(E));

process(all)
variable somme : natural range 0 to 15;
begin
    somme := 0;
    if B = '1' then
        somme := somme + 3;
    end if;
    if C = '1' then
        somme := somme + 5;
    end if;
    if E = '1' then
        somme := somme + 7;
    end if;

    if somme > 5 then
        Q <= '1';
    else
        Q <= '0';
    end if;
end process;

end arch;
```

G4/F4	G3/F3	G2/F2	G1/F1	LUT3	LUT2	LUT1	LUT0
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				





Question 3. (3 points) (18 minutes)

C'est l'été 2023 et vous êtes en stage chez CodecMTL, une petite entreprise montréalaise qui a conçu un processeur pour faire la compression de flux vidéo. Les clients de CodecMTL intègrent le processeur dans différents appareils : téléphones mobiles, tablettes, voitures autonomes, caméras intelligentes, lunettes de réalité virtuelle, etc. Le processeur peut être implémenté dans trois technologies, ASIC (logique fixe), FPGA-HG (haute gamme) et FPGA-MG (moyenne gamme), dont les paramètres sont résumés dans le tableau suivant. Les ingénieur/es coûtent 160 k$/an à l'entreprise, incluant leur salaire, les avantages sociaux, les frais de soutien administratif, etc.

Technologie	Temps de développement (mois)	# d'ingénieur/es	Coût des outils par ingénieur/e (k$ / projet)	Frais de fonderie (k$)	Coût par puce (k$)	Fréquence d'horloge maximale (MHz)	Ressources de calcul (blocs)
ASIC	12	10	15	1 250	0.05	2500	100 000
FPGA-HG	3	3	5	-	4.95	500	7 500
FPGA-MG	3	2	5	-	0.85	250	2 000

Chaque processeur occupe 500 blocs logiques et prend 10 coups d'horloge pour traiter un pixel. Le processeur est modulaire : il est facile d'en instancier plusieurs en parallèle pour multiplier le débit, sans coûts additionnels. En revanche, la solution finale doit tenir dans une seule puce.

Vous devez conseiller un client dont l'application traite des vidéos 4K de taille 3840 × 2160 pixels à un taux de 60 images par seconde. Quelles technologies rencontrent les besoins du client ? À partir de combien d'unités produites devient-il rentable de passer à une technologie plus performante ? Justifiez complètement votre réponse.

utiliser le verso si nécessaire

Question 4. (3 points) (18 minutes)

Donnez le code VHDL synthétisable d'un circuit combinatoire qui accepte en entrée un vecteur de bits et qui indique si ce vecteur contient au moins deux bits à '1' et au moins deux bits à '0'. Complétez la définition d'architecture suivante. Donnez du code de la meilleure qualité possible.

Barème de correction : si l'architecture donnée fonctionne uniquement pour W = 4, la note maximale sera 2 / 3. Si l'architecture donnée fonctionne pour 4 <= W < 256, le maximum de points sera 3 / 3.

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

entity deux_uns_deux_zeros is
generic (
    W : positive := 4
);
port (
    I : in std_logic_vector(W - 1 downto 0);
    F : out std_logic
);
end deux_uns_deux_zeros;
```

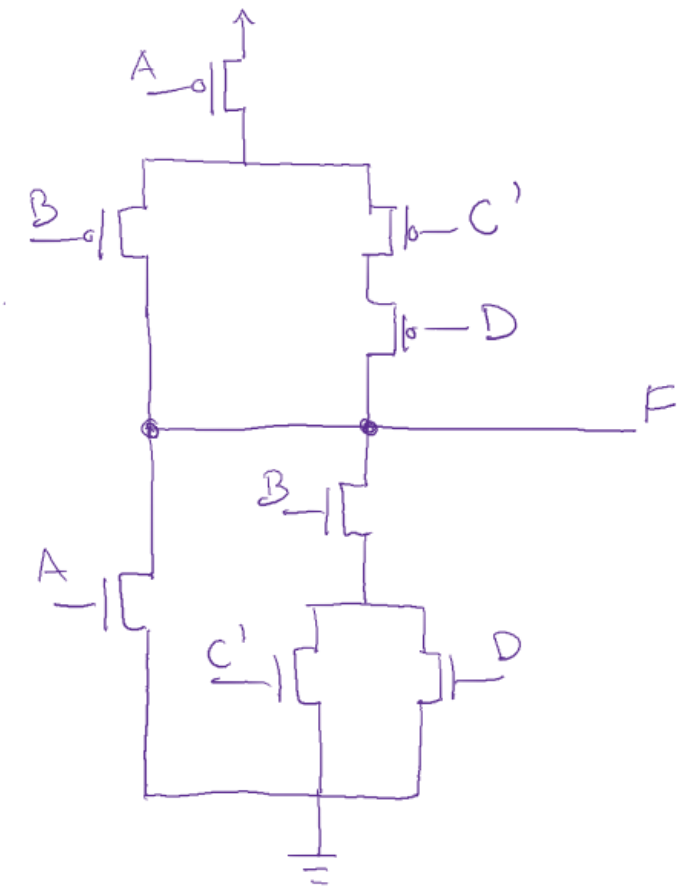
utilisez le verso si nécessaire

Solutions

#1 CMOS

$F = A' \cdot (B' + CD')$

$F' = A + [B \cdot (C' + D)]$



#2 Solution

G4/F4	G3/F3	G2/F2	G1/F1	LUT3	LUT2	LUT1	LUT0
0	0	0	0	0	1	0	0
0	0	0	1	0	0	1	1
0	0	1	0	0	1	0	0
0	0	1	1	0	1	1	0
0	1	0	0	0	-	0	0
0	1	0	1	1	-	1	0
0	1	1	0	0	-	1	0
0	1	1	1	0	-	1	0
1	0	0	0	1	-	-	0
1	0	0	1	1	-	-	0
1	0	1	0	1	-	-	0
1	0	1	1	1	-	-	1
1	1	0	0	1	-	-	0
1	1	0	1	0	-	-	0
1	1	1	0	1	-	-	1
1	1	1	1	1	-	-	0

#3 Analyse des coûts

Le débit requis est 3840 × 2160 pixels × 60 Hz = 498 Mpixels/s.
Le processeur prend 10 coups d'horloge pour traiter un pixel.

Technologie	Fréquence d'horloge maximale	Débit pour un processeur	# processeurs en parallèle requis pour >= 498 Mpixels/s	# blocs
A. ASIC	2500 MHz	250 Mpixels/s	2	1000 < 100 000 OK
B. FPGA-HG	500 MHz	50 Mpixels/s	10	5000 < 7500 OK
C. FPGA-MG	250 MHz	25 Mpixels/s	20	10 000 > 2000 NON

Il reste donc à choisir entre le ASIC et le FPGA-HG.
Coûts ASIC : 10 ingénieur/es × (160 k$ / an × 12 mois × 1 an / 12 mois + 15 k$) + 1250 k$ + n × 0.050 k$
Coûts FPGA-HG : 3 ingénieur/es × (160 k$ / an × 3 mois × 1 an / 12 mois + 5 k$) + 0 k$ + n × 4.950 k$
On pose Coûts ASIC = Coûts FPGA-HG et on trouve n ~= 585 unités.

#4. Code VHDL, il y a plusieurs réponses possibles.

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

entity deux_uns_deux_zeros is
    generic (
        W : positive := 5 -- le nombre de bits d'entrée
    );
    port (
        I : in std_logic_vector(W - 1 downto 0);
        F : out std_logic
    );
end deux_uns_deux_zeros;

-------------------------------------------------------------------------
-- pour le cas W = 4
-------------------------------------------------------------------------
architecture W_egale_4 of deux_uns_deux_zeros is

begin

    assert W = 4 report "Ne fonctionne que pour W == 4." severity failure;

    with I select
    F <= '1' when "0011" | "0110" | "1100" | "1001" | "0101" | "1010",
    '0' when others;

end W_egale_4 ;


-------------------------------------------------------------------------
-- pour les cas W >= 4 (fonctionnerait aussi pour W < 4, mais pas vérifié)
-------------------------------------------------------------------------
architecture comportementale of deux_uns_deux_zeros is

    -- retourne le nombre de fois où la valeur v de type std_logic
    -- est présente dans le vecteur s de type std_logic_vector
    --
    function compte_valeurs(s : std_logic_vector; v : std_logic) return natural is
    variable compte : natural;
    begin
        compte := 0;
        for k in s'range loop
            if s(k) = v then
                compte := compte + 1;
            end if;
        end loop;
        return compte;
    end;

begin

    assert W >= 4 report "Ne fonctionne que pour W >= 4." severity failure;

    process (all)
    begin
       if compte_valeurs(I, '1') >= 2 and compte_valeurs(I, '0') >= 2 then
          F <= '1';
       else
          F <= '0';
       end if;
    end process;

end comportementale;
```