INF3500 : Conception et réalisation de systèmes numériques

Examen final

15 août 2007, 09h30-12h00

salle M-2110

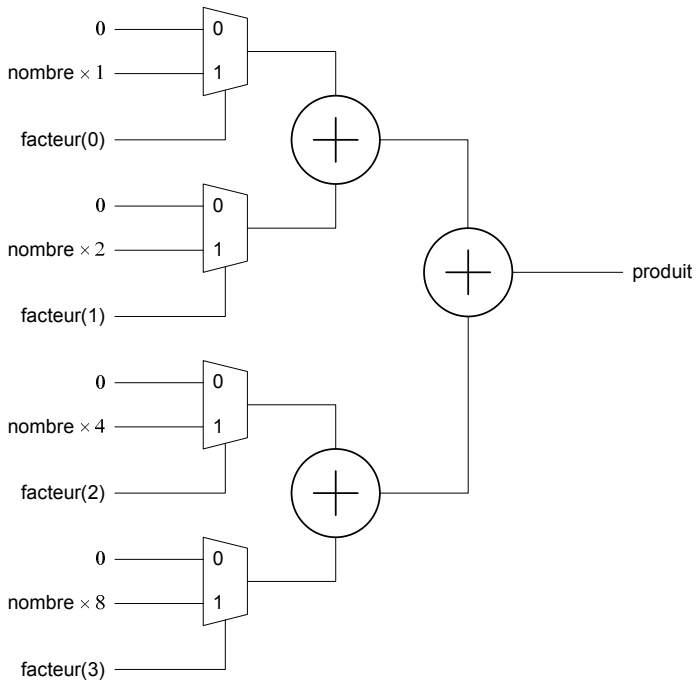
Durée: 2h30.
Pondération: 40%.
Documentation: Toute permise.
Calculatrice: Programmable permise.
Directives particulières: Ordinateurs interdits, répondre à toutes les questions, la valeur de chaque question est indiquée.
Ne posez pas de questions. En cas de doute sur le sens d'une question, énoncez clairement toute supposition que vous faites.

Question 1. (8 points)

Faites la conception d'un circuit numérique combinatoire qui accepte en entrée un nombre non-signé exprimé avec 8 bits ainsi qu'un facteur exprimé avec 4 bits. La sortie doit être un produit de 12 bits du nombre et de son facteur. Utilisez uniquement les opérations d'addition, soustraction et décalage. Votre circuit doit être purement combinatoire, il ne doit pas inclure d'éléments à mémoire.

a. Donnez un diagramme de votre circuit.

Solution :



b. Donnez sa description en VHDL.

Solution :

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

entity finalMultiplicateur is
    port (
        nombre : in unsigned(7 downto 0);
        facteur : in unsigned(3 downto 0);
        produit : out unsigned(11 downto 0)
    );
end finalMultiplicateur;

architecture arch of finalMultiplicateur is
signal add0, add1, add2, add3 : unsigned(11 downto 0);
begin
    add0 <= "0000" & nombre when facteur(0) = '1' else (others => '0');
    add1 <= "000" & nombre & "0" when facteur(1) = '1' else (others => '0');
    add2 <= "00" & nombre & "00" when facteur(2) = '1' else (others => '0');
    add3 <= "0" & nombre & "000" when facteur(3) = '1' else (others => '0');
    produit <= (add3 + add2) + (add1 + add0);
end arch;
```

Question 2. (8 points)

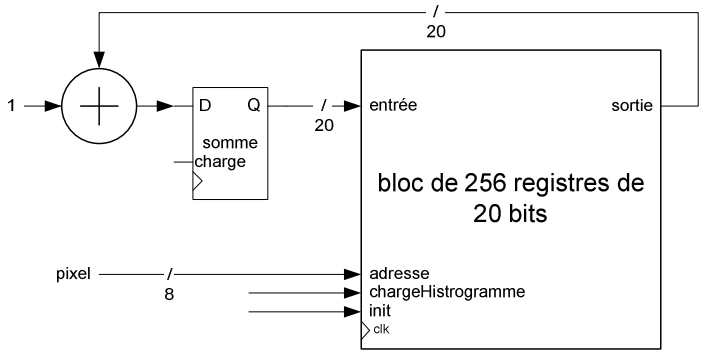
L'histogramme d'une image représente la distribution des intensités de ses pixels. Pour une image encodée en tons de gris sur 8 bits, l'histogramme comporte 256 valeurs correspondant aux intensités 0 (noir) à 255 (blanc). Chaque valeur indique combien de pixels dans l'image ont cette intensité.

Faites la conception d'un processeur qui calcule l'histogramme des intensités d'une image. L'image est composée d'une matrice carrée de 1024 × 1024 pixels de 8 bits chacun.

Le processeur a en entrée une horloge de 100 MHz, et les pixels arrivent à une fréquence fixe de 25 MHz, soit un pixel tous les quatre cycles. Le processeur a en entrée un signal de synchronisation actif un cycle d'horloge avant l'arrivée du premier pixel de l'image. Quand tous les pixels de l'image ont été reçus et traités, le processeur doit activer un signal pour indiquer qu'il a terminé. (Un autre processeur serait responsable de lire les valeurs de l'histogramme, ce qui ne fait pas partie de la question).

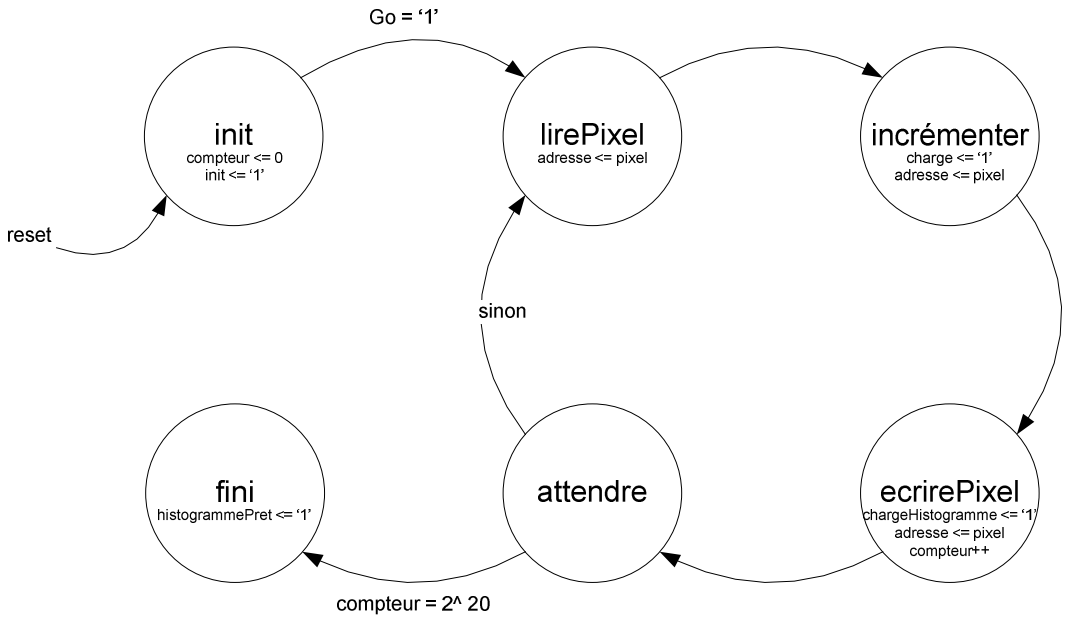
a. Donnez un diagramme montrant le chemin des données du processeur. Indiquez clairement toutes les ressources requises, leur nombre et leur largeur en bits. Indiquez clairement les signaux de contrôle.

Solution :



b. Donnez la machine à états du processeur.

Solution :



c. Donnez une description VHDL du processeur.

Solution :

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
entity histogramme is
    generic (
        dimImage : integer := 2 ** 20 -- dimension de l'image en pixels
    );
    port (
        reset, CLK, GO : in STD_LOGIC;
        pixel : in unsigned(7 downto 0);
        histogrammePret : out STD_LOGIC
    );
end histogramme;
architecture arch of histogramme is
type type_etat is (init, lirePixel, incrementer, ecrirePixel, attendre, fini);
signal etat : type_etat;
signal compteur : integer range 0 to dimImage;
type lesRegistres_type is array(0 to 255) of integer range 0 to dimImage;
signal lesRegistres : lesRegistres_type;
signal somme : integer range 0 to dimImage;
begin
    process(CLK, reset) is
    begin
        if (reset = '0') then
           etat <= init;
        elsif (rising_edge(CLK)) then
           case etat is
              when init =>
                 compteur <= 0;
                 lesRegistres <= (others => 0);
                 if GO = '1' then
                    etat <= lirePixel;
                 else
                    etat <= init;
                 end if;
              when lirePixel =>
                 etat <= incrementer;
              when incrementer =>
                 somme <= lesRegistres(to_integer(pixel)) + 1;
                 etat <= ecrirePixel;
              when ecrirePixel =>
                 etat <= attendre;
                 compteur <= compteur + 1;
              when attendre =>
                 if (compteur = dimImage) then -- a-t-on traité tous les pixels?
                    etat <= fini;
                 else
                    etat <= lirePixel;
                 end if;
              when fini =>
                 etat <= fini;
           end case;
        end if;
    end process;
    histogrammePret <= '1' when etat = fini else '0';
end arch;
```

Question 3. (8 points)

On peut déterminer le plus grand facteur commun (PGFC) entre deux nombres A et B par un algorithme assez simple, donné ici en pseudocode :

```
tant que (A != B) {
    si (A > B) {
        A := A - B;
    } sinon {
        B := B - A;
    }
}
retourner A;
```

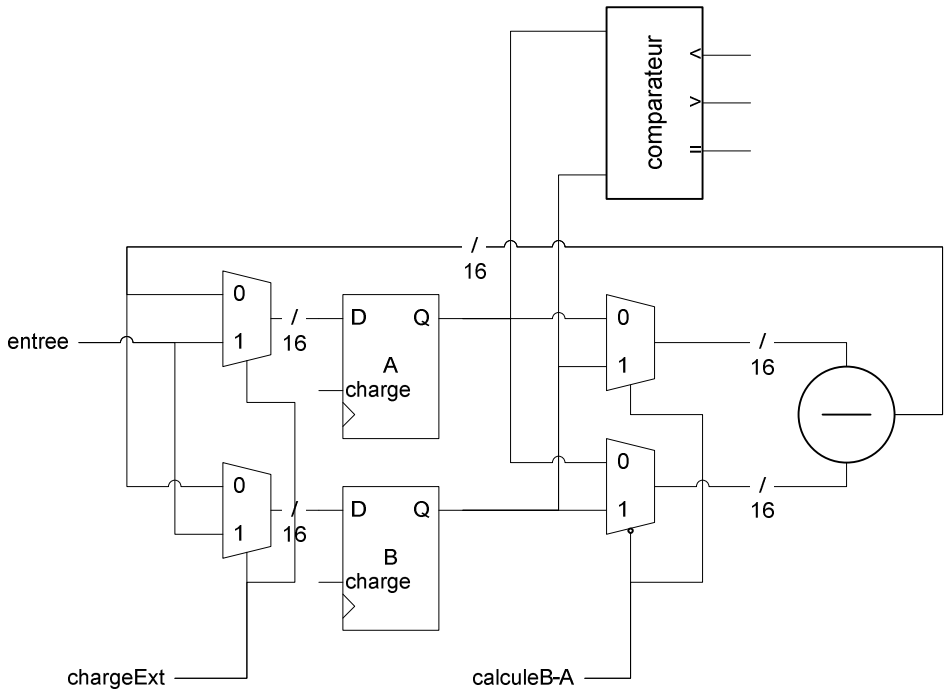
Par exemple, pour les nombres 42 et 56, pour lesquels le PGFC est 14, on aurait la séquence suivante :

itération :	0	1	2	3
A :	42	42	28	14
B :	56	14	14	14

Faites la conception d'un processeur pour calculer le PGFC de deux nombres de 16 bits. Le processeur doit avoir un port d'entrée unique, deux signaux de contrôle `chargeA` et `chargeB` pour charger chacun des nombres tout à tour via le port d'entrée, un signal de contrôle `GO` pour démarrer les calculs, un port de sortie de 16 bits pour le résultat, et un signal de contrôle de sortie `PGFCpret` qui indique que le résultat est valide.

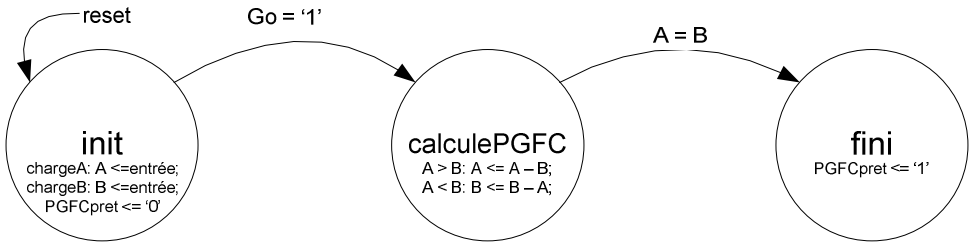
a. Donnez un diagramme montrant le chemin des données du processeur. Indiquez clairement toutes les ressources requises, leur nombre et leur largeur en bits. Indiquez clairement les signaux de contrôle.

Solution :



b. Donnez la machine à états du processeur.

Solution :



c. Donnez une description VHDL de l'architecture du processeur correspondant à l'entité suivante.

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

entity pgfc is
    generic (
        W : integer := 16
    );
    port (
        reset, CLK : in STD_LOGIC;
        entree : in unsigned(W - 1 downto 0);
        go, chargeA, chargeB : in std_logic;
        sortie : out unsigned(W - 1 downto 0);
        PGFCpret : out std_logic
    );
end pgfc;
```

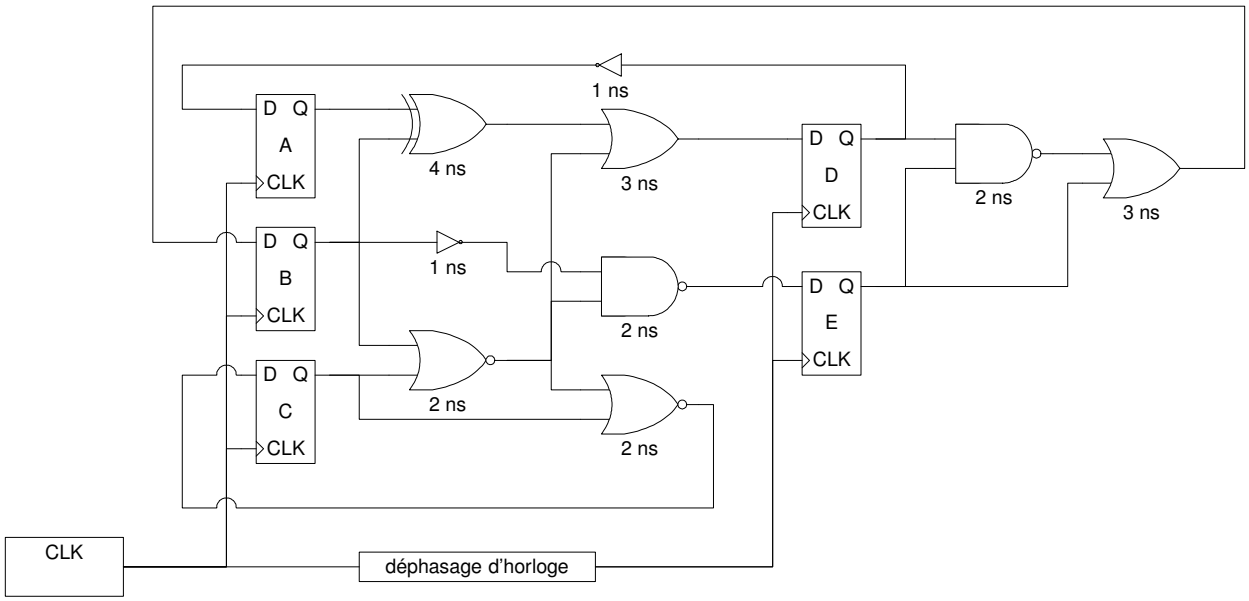
Solution:

```
architecture arch of pgfc is
type type_etat is (init, calculePGFC, fini);
signal etat : type_etat;
signal A, B : unsigned(W - 1 downto 0);
begin
    process(CLK, reset) is
    begin
        if (rising_edge(CLK)) then
            if (reset = '1') then
               etat <= init;
            else
               case etat is
                  when init =>
                     if (chargeA = '1') then
                           A <= entree;
                     end if;
                     if (chargeB = '1') then
                           B <= entree;
                     end if;
                     if (go = '1') then
                           etat <= calculePGFC;
                     else
                           etat <= init;
                     end if;
                  when calculePGFC =>
```

```
                    if (A = B) then
                        etat <= fini;
                    else
                        etat <= calculePGFC;
                        if (A > B) then
                            A <= A - B;
                        else
                            B <= B - A;
                        end if;
                    end if;
                when fini =>
                    etat <= fini;
                when others =>
                    etat <= fini;
            end case;
        end if;
    end if;
end process;
PGFCpret <= '1' when etat = fini else '0';
sortie <= A;
end arch;
```

Question 4. (8 points)

Considérez le circuit numérique suivant. Supposez que 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. Supposez qu'on peut négliger les délais des interconnexions.



a. Énumérez tous les chemins possibles entre paires de bascules et donnez le délai sur chacun. Supposez qu'il n'y a pas de déphasage d'horloge.

Solution:

A/B à D : 2 + 4 + 3 = 9 ns

B/C à D : 2 + 2 + 3 = 7 ns

B à E : 2 + 1 + 2 = 5 ns

C à E : 2 + 2 + 2 = 6 ns

C à C : 2 + 2 + 2 = 6 ns

B à C : 2 + 2 + 2 = 6 ns

D à A : 2 + 1 = 3 ns

D/E à B : 2 + 2 + 3 = 7 ns

b. Identifiez le chemin critique et déterminez la fréquence maximale d'horloge. Supposez qu'il n'y a pas de déphasage d'horloge.

Solution:

A/B à D : 2 + 4 + 3 = 9 ns

La fréquence maximale d'horloge est $(9 \text{ ns} + 1 \text{ ns})^{-1} = 100 \text{ MHz}$

c. Supposez que la fréquence d'opération choisie est de 50 MHz. Considérez uniquement le chemin de la bascule D vers la bascule A via l'inverseur. Quelle sont les valeurs positives acceptables du déphasage d'horloge pour ce chemin?

Solution:

La période correspondante à 50 MHz est 20 ns.

Après un front montant de l'horloge, la sortie de la bascule D devient stable 2 ns plus tard. L'inverseur retarde ce signal de 1 ns.

Si on considère le temps de préparation, alors il faut T > td + tcomb + tsu – tcs, donc tcs > 2 + 1 + 1 – 20, donc tcs > -16 ns ou > 4 ns.

Si on considère le temps de maintien, alors il faut td + tcomb – tcs > th, donc tcs < td + tcomb – th, donc tcs < 2 + 1 – 0.5 ns ou 2.5 ns.

Donc pour ce chemin le déphasage d'horloge doit être entre 0 et 2.5 ns, ou bien entre 4 ns et 20 ns. Il ne peut pas être entre 2.5 et 4 ns.

Question 5. (8 points)

a. Considérez le système d'équations logiques suivant :

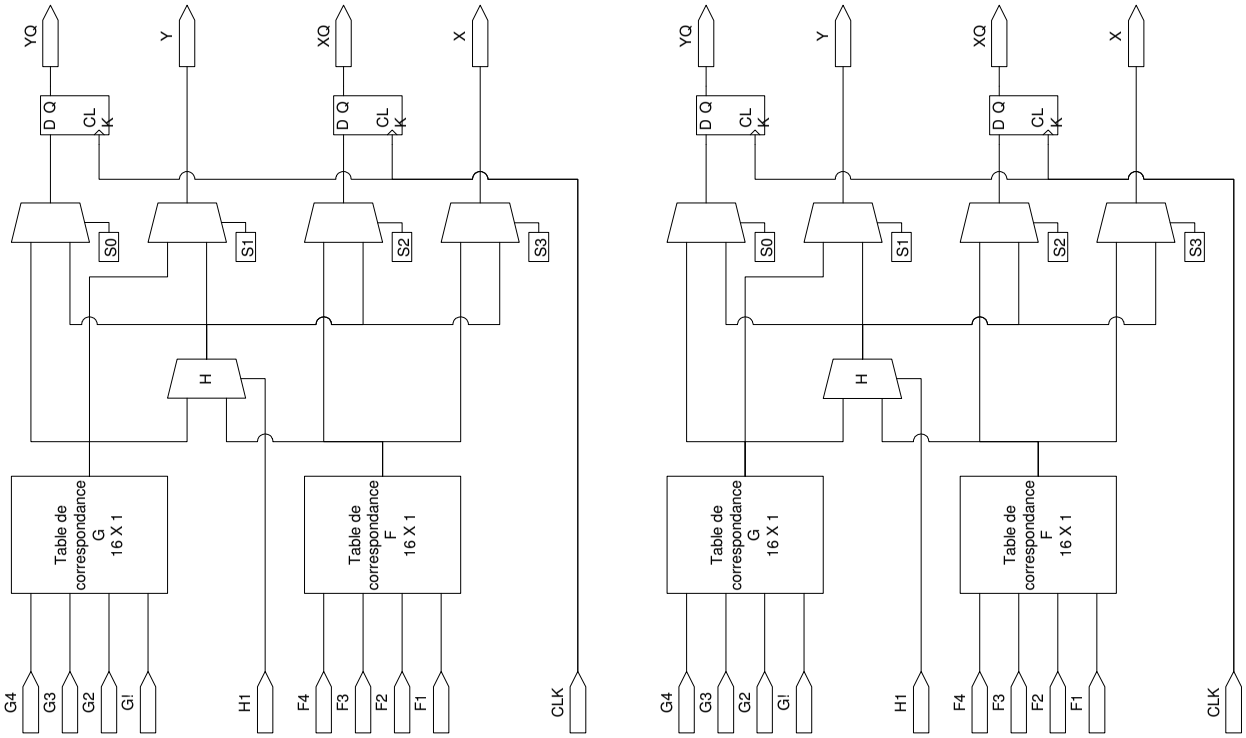
$$P = ABCD + B'D' + \overline{A}\,\overline{C}D + EF'G$$

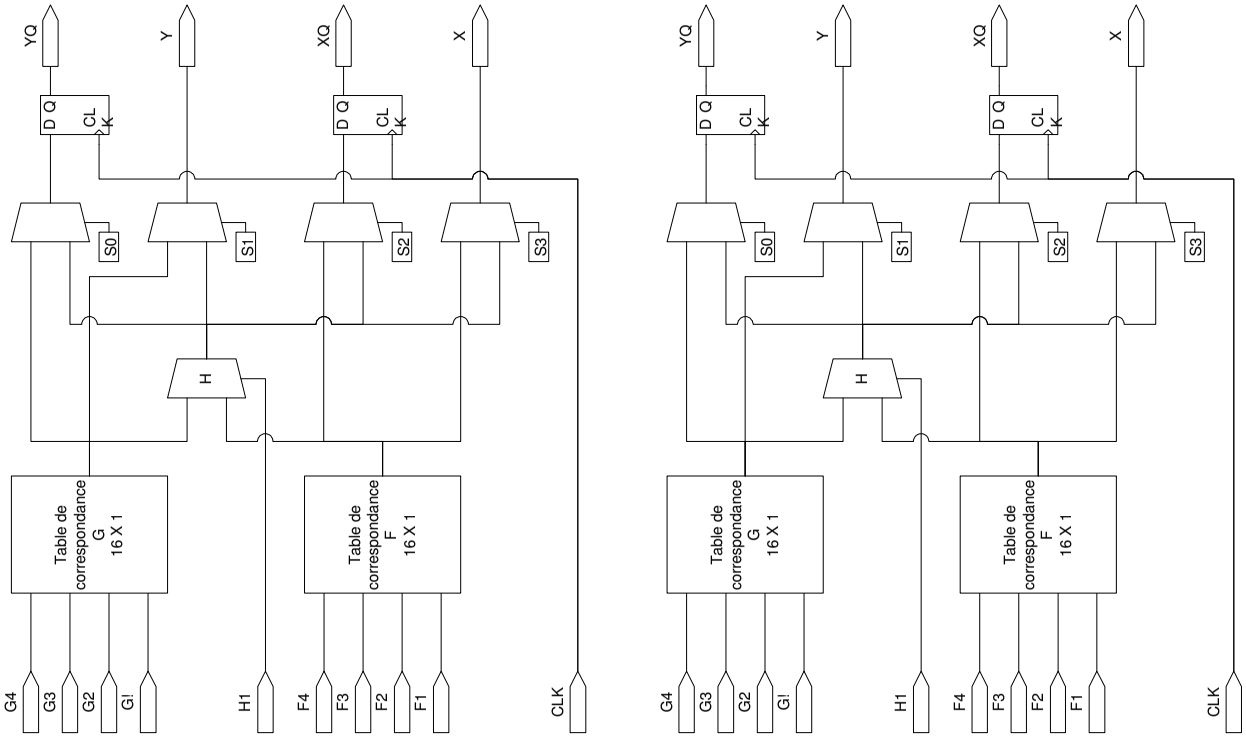
$$Q = BC + AB' + C'D$$

Considérez le circuit montré à la page suivante, qui représente une portion d'un réseau prédiffusé programmable (FPGA) général. En utilisant ce circuit, montrez comment implémenter ce système d'équations. Indiquez clairement la position de chaque port, le contenu des tables de correspondance (Look-Up Table – LUT) et le routage des différents signaux.

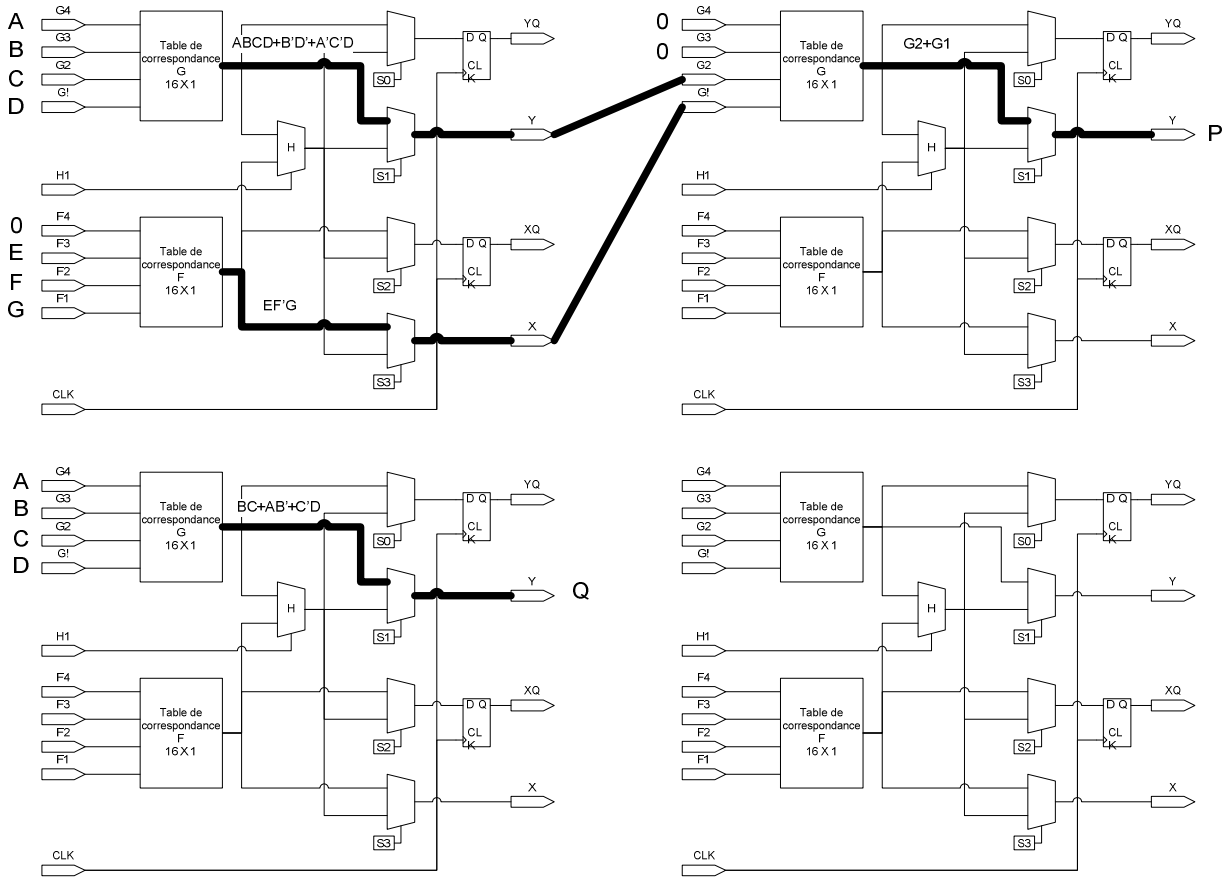
b. Considérez l'extrait de code VHDL suivant. Donnez son équivalent en matériel à l'aide d'un schéma de portes logiques.

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

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

architecture arch of monModule is
begin
   F <= not(A and (B xor not(C)));
   G <= '0' when (A = '1' or B /= C) else '1';
   with A select H <= B when '0', C when others;
end arch;
```


YQ
Y
XQ
X
D Q
CL
K
S0
S1
S2
S3
H
Table de correspondance G 16 X 1
Table de correspondance F 16 X 1
G4
G3
G2
GI
H1
F4
F3
F2
F1
CLK


YQ
Y
XQ
X
D Q
CL
K
S0
S1
S2
S3
H
Table de correspondance G 16 X 1
Table de correspondance F 16 X 1
G4
G3
G2
GI
H1
F4
F3
F2
F1
CLK

Solution:


A
B
C
D
G4
G3
G2
G'
Table de correspondance G 16 X 1
ABCD+B'D'+A'C'D
H
H1
0
E
F
G
F4
F3
F2
F1
Table de correspondance F 16 X 1
EF'G
CLK
S0
S1
S2
S3
D Q
CL
LK
YQ
Y
XQ
X
0
0
G2+G1
P
BC+AB'+C'D
Q