INF3500 : Conception et réalisation de systèmes numériques

Examen final

Avril 2010

Durée: 2h30.

Pondération: 40%.

Documentation: Une feuille recto verso 8.5”×11” ou A4 permise.

Calculatrice: Programmable permise.

Directives particulières:

- Ordinateurs interdits.
- Répondre à toutes les questions, la valeur de chaque question est indiquée.
- Répondre dans le cahier fourni.
- Ne pas remettre le questionnaire.
- Ne posez pas de questions. En cas de doute sur le sens d’une question, énoncez clairement toute supposition que vous faites.

Question 1. (10 points)

Considérez le problème de la conception d'un multiplicateur complexe.

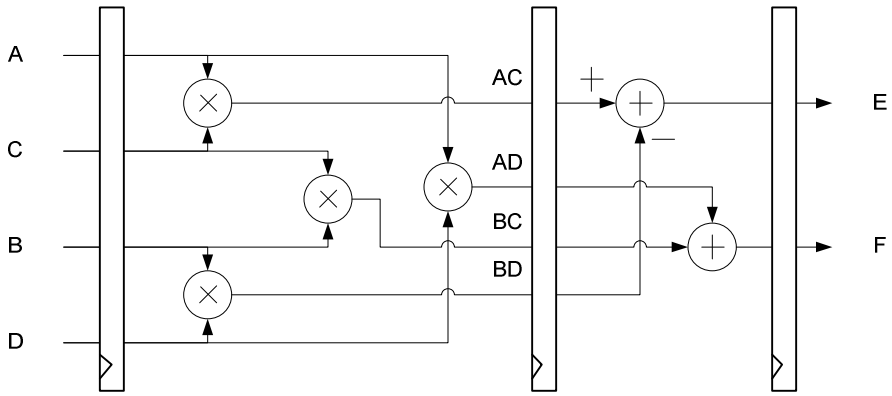
Les entrées du circuit sont A, B, C, D, représentant les nombres $Z_1 = A + jB$ et $Z_2 = C + jD$. Les sorties du circuit sont E et F représentant $Z_3 = E + jF$, avec $Z_3 = Z_1 \times Z_2$. Les entrées sont des nombres entiers signés exprimés avec 16 bits. Le multiplicateur doit être implémenté sur un FPGA.

a. Faites la conception d'un multiplicateur complexe qui maximise le débit des calculs. Donnez un diagramme du chemin des données de votre circuit, et, si nécessaire, le diagramme d'états de l'unité de contrôle. Ne donnez pas de code VHDL.

b. Faites la conception d'un multiplicateur complexe qui minimise la surface utilisée. Donnez un diagramme du chemin des données de votre circuit, et, si nécessaire, le diagramme d'états de l'unité de contrôle. Ne donnez pas de code VHDL.

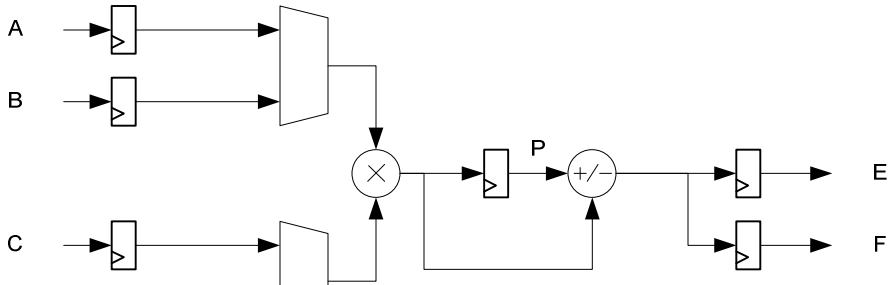
Solution :

a.

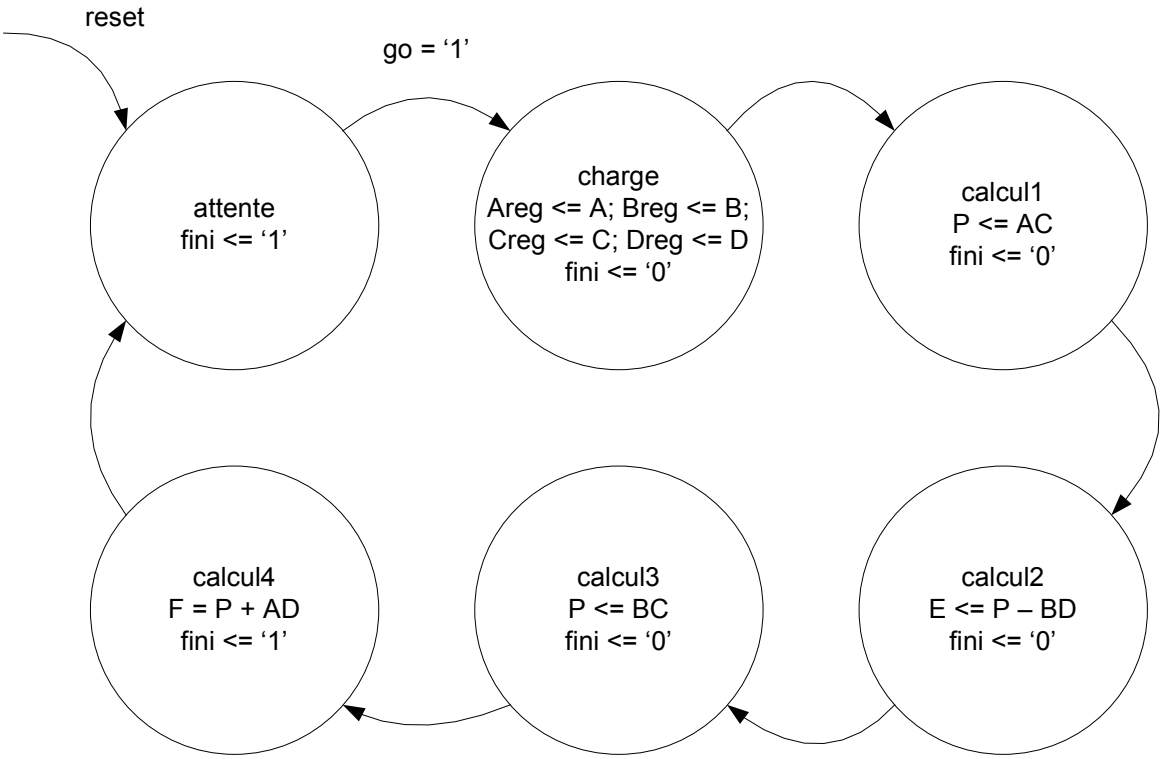


Les rectangles indiquent des registres de pipeline.
Les sorties des multiplicateurs sont sur 32 bits.
Les sorties des additionneurs sont sur 33 bits.
Il n'y a pas d'unité de contrôle.

b.

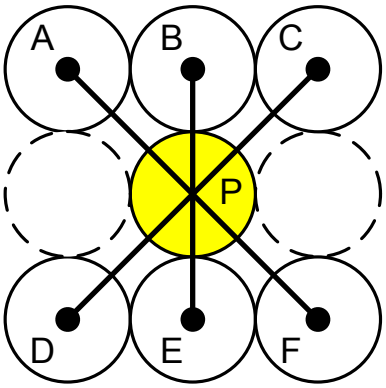


Les rectangles indiquent des registres.
Chaque registre a un signal de chargement et un signal de réinitialisation provenant de l'unité de contrôle.
Chaque multiplexeur a un signal de sélection provenant de l'unité de contrôle.
On pourrait éliminer le registre P et réutiliser les registres E et F au prix de quelques structures de contrôle supplémentaires.
On suppose ici 4 ports d'entrée et 2 ports de sortie. Il faudrait sérialiser les entrées et sorties s'il y en avait moins.



Question 2. (6 points)

L'algorithme ELA (Edge-based Line Average) a pour but de doubler la résolution verticale d'une image. La figure montre les pixels impliqués dans les calculs. L'algorithme accepte en entrée les six pixels A, B, C, D, E et F, et produit en sortie le pixel P. Il calcule trois différences absolues : |A – F|, |B – E| et |C – D|. La plus petite de ces trois valeurs indique la direction d'interpolation. La valeur de P est alors donnée par la moyenne des deux pixels le long de la direction d'interpolation. Par exemple, si |A – F| a la plus petite valeur, alors P = (A + F) / 2. La valeur de tous les pixels est représentée en tons de gris sur 8 bits.



Donnez une architecture pour l'entité suivante en VHDL synthétisable, afin de réaliser l'algorithme ELA tout en maximisant le débit.

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

entity ELA is
   port (
      A, B, C, D, E, F: in unsigned(7 downto 0);
      P: out unsigned (7 downto 0)
   );
end ELA;
```

Solution :

```
architecture arch of ELA is

begin

process(A, B, C, D, E, F)
variable A9, B9, C9, D9, E9, F9: unsigned(8 downto 0); -- pixels expimés sur 9 bits
variable D1, D2, D3: unsigned(7 downto 0); -- trois différences
variable I1, I2, I3: unsigned(7 downto 0); -- trois interpolations
begin

   -- on exprime la valeur des pixels sur 9 bits de façon à éviter les
   -- débordements dans les calculs
   A9 := resize(A, 9); B9 := resize(B, 9); C9 := resize(C, 9);
   D9 := resize(D, 9); E9 := resize(E, 9); F9 := resize(F, 9);

   -- pour les trois différences, il faut passer au type signed parce que la
   -- différence peut être négative avant de prendre sa valeur absolue
   D1 := resize(unsigned(abs(signed(A9) - signed(F9))), 8);
   D2 := resize(unsigned(abs(signed(B9) - signed(E9))), 8);
   D3 := resize(unsigned(abs(signed(C9) - signed(D9))), 8);

   -- pour les trois interpolations, il faut:
   -- 1. faire l'addition sur 9 bits (p. ex., 128 + 128 = 256 nécessite 9 bits)
   -- 2. diviser par deux
   -- 3. exprimer le résultat sur 8 bits
   I1 := resize((A9 + F9) / 2, 8);
   I2 := resize((B9 + E9) / 2, 8);
   I3 := resize((C9 + D9) / 2, 8);

   if (D1 <= D2 and D1 <= D3) then
      P <= I1;
   elsif (D2 <= D1 and D2 <= D3) then
      P <= I2;
   else
      P <= I3;
   end if;

end process;

end arch;
```

Question 3. (6 points)

Faites la conception d'un processeur pour un télémètre laser. Le télémètre a un bouton pour déclencher la prise de mesure. Quand le bouton est pressé, une impulsion lumineuse est générée et un chronomètre est activé. L'impulsion lumineuse se propage dans l'air, frappe la cible et revient vers un détecteur. Quand l'écho de l'impulsion lumineuse est perçu par le détecteur, le chronomètre est arrêté et la distance peut être calculée en tenant compte de la vitesse de propagation de la lumière dans l'air. Soit T le temps mesuré par le chronomètre, en secondes, alors la distance D est donnée par $D = T \times c / 2$, où $c = 3 \times 10^8$ m/s.

Le processeur a deux entrées : le bouton et un signal provenant du détecteur indiquant qu'une impulsion lumineuse a été reçue. Il a deux sorties : un signal vers le laser pour déclencher une impulsion lumineuse et un autre signal indiquant la distance mesurée.

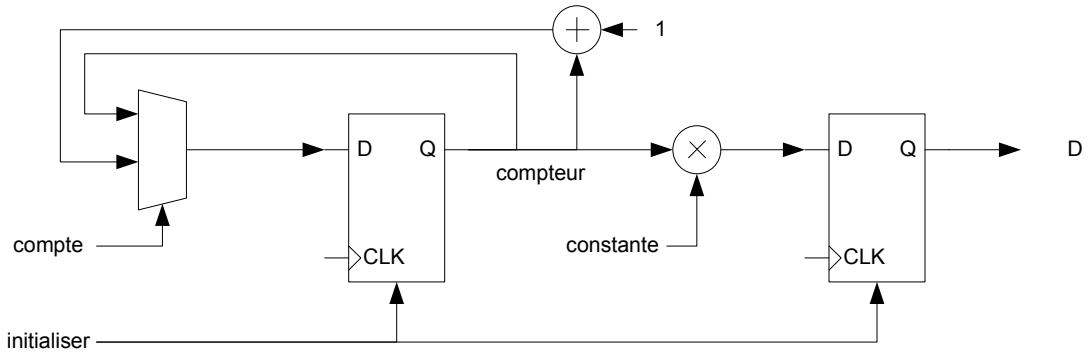
a. Donnez un diagramme représentant le chemin des données du processeur.

b. Donnez le diagramme d'états de l'unité de contrôle du processeur.

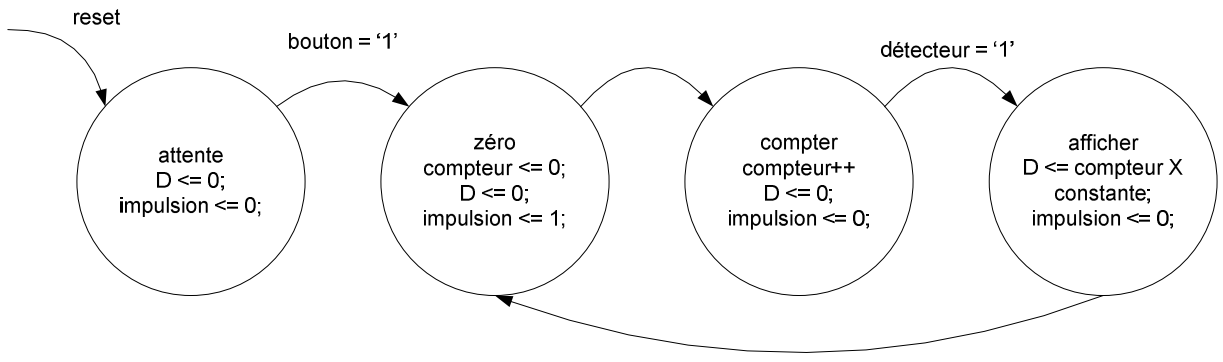
c. Donnez la précision du télémètre laser en fonction de la fréquence d'horloge du circuit et sa portée maximale en fonction de la taille du chronomètre.

Solution :

a.



b.



Il faudrait ajouter une valeur maximale au compteur. Quand cette valeur serait atteinte, indiquant qu'un écho lumineux n'a pas été reçu, le système afficherait un message d'erreur.

c.

$D = \frac{T \times 3 \times 10^8 \, m/s}{2}$ $T = compteur \times période\_CLK = \frac{compteur}{f\_CLK}$ $D = \frac{compteur \times 3 \times 10^8 \, m/s}{2 \times f\_CLK}$ $\text{constante} = \frac{3 \times 10^8 \, m/s}{2 \times f\_CLK}$	Par exemple, pour f_CLK = 150 MHz, on a constante = 1 m et donc chaque incrément du compteur correspond à un mètre de distance de la cible. La constante est donc la résolution du système. La valeur maximale du compteur détermine la distance maximale. Par exemple, pour un compteur de 10 bits et f_CLK = 150 MHz, la distance maximale serait 1023 m.

Question 4. (6 points)

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

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

entity module3 is
   port (
      A, B, C : in std_logic;
      F, G : out std_logic
   );
end module3;

architecture arch of module3 is
signal S, T, U : std_logic;
begin

   S <= A or B;
   T <= A xor B;
   U <= A and C;

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

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

end arch;
```

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

entity module3_TB is
end module3_TB;

architecture arch of module3_TB is
signal A, B, C, F, G : std_logic;
begin

   UUT : entity module3(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;
```

a. Donnez la liste des événements, telle qu'elle pourrait être dressée par un simulateur qui exécuterait le banc d'essai.

b. En vous basant sur votre liste des événements, donnez la valeur de tous les ports et signaux internes du module combinatoire en fonction du temps, en tenant compte des délais deltas.

Solution :

a. Liste des événements

À t = 0 + 0 Δ, initialisation de la simulation (tout à 'U')

À t = 0 + 1 Δ, assignation de 100 à (A, B, C)

À t = 0 + 2 Δ, évaluation des signaux S, T, U et G (provoquées par les changements sur A, B, C)

À t = 0 + 3 Δ, évaluation de F et G (provoquées par les changements sur S, T, U)

Aucun nouvel événement n'est ajouté à la liste d'événements.

À t = 10 ns + 0 Δ, assignation de 1 à C

À t = 10 ns + 1 Δ, évaluation de U (provoquée par le changement sur C), ** mais pas d'évaluation de F parce que C n'est pas dans la liste de sensitivité du processus **

À t = 10 ns + 2 Δ, évaluation de G (provoquée par le changement sur U)

Aucun nouvel événement n'est ajouté à la liste d'événements.

b.

Time	Delta	UUT/A	UUT/B	UUT/C	UUT/S	UUT/T	UUT/U	UUT/F	UUT/G
0 ps	0	U	U	U	U	U	U	U	U
0 ps	1	1	0	0	U	U	U	U	U
0 ps	2	1	0	0	1	1	0	U	U
0 ps	3	1	0	0	1	1	0	0	0
10000 ps	0	1	0	1	1	1	0	0	0
10000 ps	1	1	0	1	1	1	1	0	0
10000 ps	2	1	0	1	1	1	1	0	1

Question 5. (6 points)

Considérez le code VHDL suivant.

a. Donnez un diagramme correspondant composé de modules combinatoires (multiplexeurs, décodeurs, etc.), d'unités fonctionnelles (additionneurs, multiplicateurs, comparateurs, etc.) et d'éléments à mémoire tel qu'il pourrait être généré par un processus de synthèse.

b. Estimez le nombre de ressources nécessaires, en termes de LUTs et de bascules, pour implémenter ce circuit sur le FPGA utilisé dans les laboratoires du cours. Justifiez complètement votre réponse.

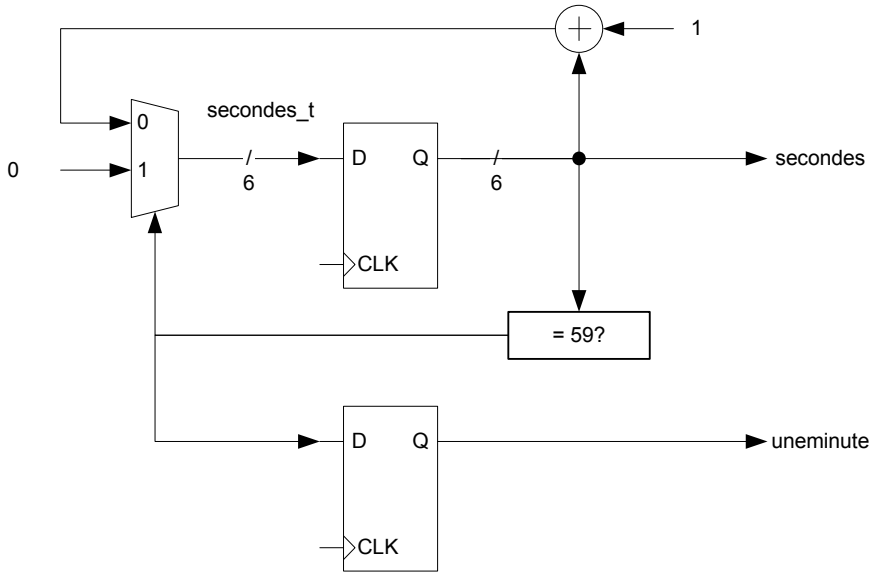
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity compteurSecondes is
    port (
        reset_n, clk1Hz : in std_logic;
        secondes : out unsigned(5 downto 0);
        uneminute : out std_logic
    );
end compteurSecondes;

architecture arch of compteurSecondes is
signal secondes_t : natural range 0 to 59;
begin
    secondes <= to_unsigned(secondes_t, secondes'length);
    process(clk1Hz) is
    begin
        if rising_edge(clk1Hz) then
            if reset_n = '0' then
                secondes_t <= 0;
                uneminute <= '0';
            else
                if secondes_t = 59 then
                    secondes_t <= 0;
                    uneminute <= '1';
                else
                    secondes_t <= secondes_t + 1;
                    uneminute <= '0';
                end if;
            end if;
        end if;
    end process;
end arch;
```

Solution :

a.



b.

Il y a un total de 7 bascules.

Pour les LUTs, une analyse de base procède comme suit. L'additionneur à 6 bits nécessite 6 LUTs (un par bit). Le bloc de comparaison d'égalité nécessite 6 LUTs (un par bit). Le multiplexeur à deux entrées de 6 bits nécessite 6 LUTs.

Le total serait donc de 7 bascules et 18 LUTs.

{Dans les faits, une analyse des résultats de synthèse révèle que le bloc de comparaison et le multiplexeur peuvent être combinés en 5 bascules seulement. Ceci s'explique entre autres parce que les LUTs peuvent avoir jusqu'à 4 entrées. Avec les 6 LUTs de l'additionneur, au total 7 bascules et 11 LUTs sont nécessaires.}

Question 6. (6 points)

Réponses brèves.

a. On vous propose de participer au développement d'un nouveau langage de description matérielle, PolyHDL. Ce langage devra servir à modéliser des circuits numériques qui seront implémentés sur FPGA. Donnez quatre caractéristiques que vous considérez essentielles pour ce langage.

Solution : pouvoir représenter la concurrence, pouvoir modéliser des composantes matérielles, permettre la simulation du modèle, permettre la synthèse du modèle, inclure des opérateurs correspondant à des structures disponibles sur FPGA, permettre la description de bancs d'essais, favoriser l'abstraction et la hiérarchie, etc.

b. Énumérez deux stratégies à suivre pour choisir des vecteurs de test afin de vérifier un circuit numérique.

Solutions possibles : Analyse des valeurs limites, partitionnement en classes, cas spéciaux choisis manuellement, valeurs pseudo-aléatoires. Pour la moitié des points : tests de boîte noire, tests de boîte blanche, couverture de code, couverture de branchements, etc.

c. Expliquez comment un double tampon peut permettre de synchroniser l'échange d'informations entre deux domaines d'horloge différents.

Solution : Un double tampon consiste de deux bascules en série. L'entrée asynchrone est appliquée à la première bascule. Il est possible que celle-ci entre dans un état métastable. Elle finira en pratique par se stabiliser sur un 1 ou un 0. La deuxième bascule permet d'accorder le plus long temps possible à la première bascule pour se stabiliser, soit une période d'horloge.

d. Expliquez pourquoi la division générale n'est pas synthétisable par la plupart des synthétiseurs de code HDL ciblant des FPGA.

Solutions : La division générale n'est pas synthétisable par choix des concepteurs de synthétiseurs. Puisque les FPGA ne possèdent pas de bloc implémentant la division, il faudrait choisir une architecture de diviseur en particulier et la réaliser avec un arrangement de LUT, FF, etc. Les concepteurs de synthétiseurs préfèrent laisser ce choix aux concepteurs de systèmes numériques qui écrivent du code HDL.

e. Énumérez deux principes à suivre pour la conception et l'implémentation de circuits numériques sur FPGAs.

Quelques solutions possibles : conception synchrone, ne pas utiliser de loquets, signal d'initialisation global, routage spécial du signal d'horloge, exploitation de toutes les ressources de la puce, gestion manuelle de la disposition, arithmétique en virgule fixe plutôt que flottante.

f. En VHDL, les attributs permettent d'obtenir de l'information à propos de types, signaux, variables et constantes. Donnez deux exemples d'attributs en VHDL.

Solution possibles: 'left, 'right, 'low, 'high, 'range, 'length, 'image, 'value, 'event, etc.