Nom : _____ Matricule : _____

INF3500 : Conception et réalisation de systèmes numériques
Examen final – Automne 2017

Durée: 2h30
Pondération: 50%

Documentation: Une feuille de notes recto verso 8.5"×11" ou A4 permise.
Calculatrice: Programmable permise.

Directives particulières:

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

Question 1. (8 points)

Votre stage de l'été 2018 porte sur l'implémentation de réseaux de neurones profonds pour la reconnaissance automatique de visages par des caméras de sécurité dans les métros et les autobus de la Société de Transport de Montréal (STM). La STM possède environ 1000 autobus et 800 voitures de métro. On prévoit installer trois caméras par véhicule. Le directeur de l'ingénierie a entendu parler des FPGA et il voudrait votre avis. Son équipe a déjà éliminé l'option d'utiliser un microprocesseur embarqué à cause de son débit insuffisant, mais il pense à une solution en logique fixe. Pour la solution FPGA, vous estimez le temps de conception à 6 mois pour 3 ingénieurs à 100 k$/an et les licences d'outils coûtent 5 k$/an/poste. Pour la logique fixe, le temps de conception est 50% plus grand que pour le FPGA et les licences d'outils sont quatre fois plus chères. En plus, les frais de la fonderie s'élèvent à 400 k$ pour les 1000 premières puces, puis les puces coûtent 1 $ chacune par la ensuite. Les délais de la fonderie sont de 28 semaines avant la livraison. Le FPGA le moins cher qui accommoderait le design coûte 75 $ l'unité et peut être livré dans 6 semaines.

Que dites-vous au directeur de l'ingénierie ? Justifiez complètement votre argumentation. Énoncez clairement toutes vos suppositions et montrez tous vos calculs.

(utilisez le verso si nécessaire)

Question 2. (20 points)

L'algorithme CORDIC permet entre autres de calculer le sinus et le cosinus d'un angle. L'algorithme peut être décrit par les micro-opérations suivantes. Init est un signal de contrôle en entrée qu'on active pour démarrer les calculs et Fini est un signal de sortie indiquant que les calculs sont terminés. Z0 est l'angle dont il faut calculer le sinus et le cosinus. X0, Y0 et Cmax sont des constantes : en pratique $X0 \approx 9949 / 2^{16}$, Y0 = 0 et Cmax = 12 itérations. L'opérateur >> est le décalage et il a préséance sur l'addition et la soustraction. Q >> R signifie *décaler Q vers la droite de R positions*. La mémoire ROM est un tableau de Cmax valeurs constantes utilisées dans les calculs. ROM[C] retourne la valeur entreposée à l'adresse C.

Init : C ← 0; Init' ET C ≠ Cmax : C ← C + 1;

Init : X ← X0; Init' ET C ≠ Cmax ET Z ≥ 0 : X ← X – Y >> C; Init' ET C ≠ Cmax ET Z < 0 : X ← X + Y >> C;

Init : Y ← Y0; Init' ET C ≠ Cmax ET Z ≥ 0 : Y ← Y + X >> C; Init' ET C ≠ Cmax ET Z < 0 : Y ← Y – X >> C;

Init : Z ← Z0; Init' ET Z ≥ 0: Z ← Z – ROM[C]; Init' ET Z < 0 : Z ← Z + ROM[C];

Init : Fini ← 0; Init' ET C = Cmax : Fini ← 1;

a. (7 points) Donnez le diagramme d'un chemin des données correspondant à ces micro-opérations.

(utilisez le verso si nécessaire)

b. (6 points) Donnez le code VHDL synthétisable complet d'un module implémentant l'algorithme CORDIC tel que décrit ici. Écrivez du code le plus clair et le plus général possible, et portez bien soin à vos choix de types pour les signaux d'entrée et de sortie ainsi que pour les valeurs internes.

(*Utilisez le verso si nécessaire*)

c. (5 points) En supposant que X, Y et Z sont exprimés sur 16 bits, et que Cmax = 12, estimez les ressources nécessaires pour implémenter votre chemin des données dans un FPGA de la série-7 de Xilinx, en termes de bascules, de LUT et de tranches DSP48E. Énoncez clairement vos suppositions et justifiez toutes vos estimations.

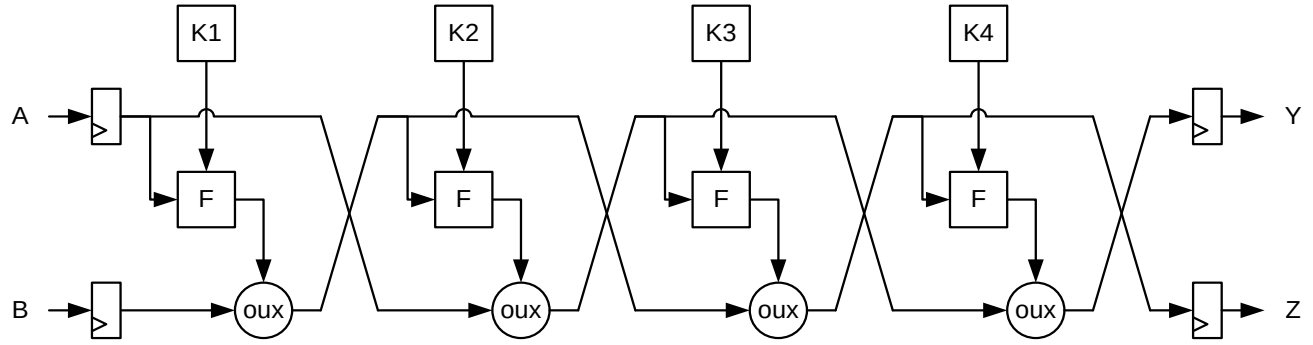
	Nombre	**Justifications**
#LUT		
#FF		
#DSP48E		

d. (2 points) En supposant que votre circuit fonctionne avec une horloge de 100 MHz et que Cmax = 12, donnez le débit du système en résultats par seconde et la latence en secondes. Justifiez votre réponse.

(utilisez le verso si nécessaire)

Question 3. (10 points)

Le réseau de Feistel, dont un exemple simple à quatre étages est illustré ici, est utilisé dans les algorithmes de chiffrement par bloc. Le message à chiffrer est décomposé en un flux de nombres appliqués aux entrées A et B du réseau. Les quatre clés secrètes K1, K2, K3 et K4 restent constantes pour le chiffrement du message. Les sorties Y et Z sont un flux de nombres représentant le message chiffré. On constate qu'à chaque étage le signal du haut est combiné à la clé par la fonction combinatoire F. On effectue ensuite un ou-exclusif bit à bit avec le signal du bas. À la fin de l'étage, les signaux du haut et du bas sont interchangés pour le prochain étage.



Le réseau est implémenté sur un FPGA. Après implémentation, on a caractérisé les différentes parties du réseau comme suit. Les bascules ont un délai de 1 ns, un temps de préparation t_{su} de 0.25 ns et un temps de maintien t_h de 0.1 ns. Les blocs combinatoires F ont un délai de 5 ns, et les ou-exclusif ont un délai de 3 ns. Les fils d'interconnexion ont des délais de 0.1 ns chacun. Les blocs des clés K1 à K4 n'ont pas de délai, ce sont des constantes.

a. (5 points) Identifiez le chemin critique du circuit sur le diagramme et donnez la fréquence maximale d'opération.

b. (5 points) Modifiez le circuit pour atteindre un débit de 50×10^6 résultats par seconde, où un résultat est une paire (Y, Z). Minimisez la latence.

1. Montrez sur le diagramme où vous insérez des registres de pipeline; 2. Indiquez le nouveau chemin critique du circuit et calculez la fréquence maximale d'opération; 3. Donnez la nouvelle latence.

(utilisez le verso si nécessaire)

Question 4. (6 points)

Votre collègue a conçu une minuterie à très grande précision qui doit être implémentée dans un FPGA et utilisée dans un accélérateur de particules. La minuterie reçoit une horloge à une fréquence de 1 MHz. Un des modules de cette minuterie est responsable de calculer la valeur du prochain instant en fonction du temps présent quand un coup d'horloge est reçu. Ce module reçoit en entrée une mesure du moment présent sur cinq entrées : le jour de la semaine (un entier de 0 à 6), l'heure (un entier de 0 à 23), la minute (un entier de 0 à 59), la seconde (un entier de 0 à 59), et une portion de seconde sur 1 million (un entier entre 0 et 999 999). Vous n'avez pas accès au code du module. Une particularité du système est qu'il doit être vérifié plusieurs fois par heure pendant son fonctionnement à cause des conditions d'opération extrêmes dans l'accélérateur de particules.

Proposez une stratégie pour vérifier adéquatement et efficacement ce module. Justifiez et détaillez complètement votre réponse. Estimez le temps nécessaire pour effectuer la vérification selon la stratégie que vous avez choisie.

(utilisez le verso si nécessaire)

Question 5. (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 arch3 of module3 is
signal S1, S2, S3 : std_logic;
begin
    S1 <= A xor C;
    S2 <= not(B or C);
    S3 <= A and C;
    process (S1, S2, S3)
    begin
        F <= S1 or S2;
        G <= S2 and S3;
    end process;
end arch3;
```

```
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. Toutes les lignes pourraient ne pas être requises.

À t = 0 + 0 Δ : _____

À t = 0 + 1 Δ : _____

À t = 0 + 2 Δ : _____

À t = 0 + 3 Δ : _____

À t = 0 + 4 Δ : _____

À t = 10 + 0 Δ : _____

À t = 10 + 1 Δ : _____

À t = 10 + 2 Δ : _____

À t = 10 + 3 Δ : _____

À t = 10 + 4 Δ : _____

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.

Temps	delta	A	B	C	S1	S2	S3	F	G
0 ns	0	U	U	U	U	U	U	U	U
0 ns	1								
0 ns	2								
0 ns	3								
0 ns	4								
10 ns	0								
10 ns	1								
10 ns	2								
10 ns	3								
10 ns	4								

Solutions

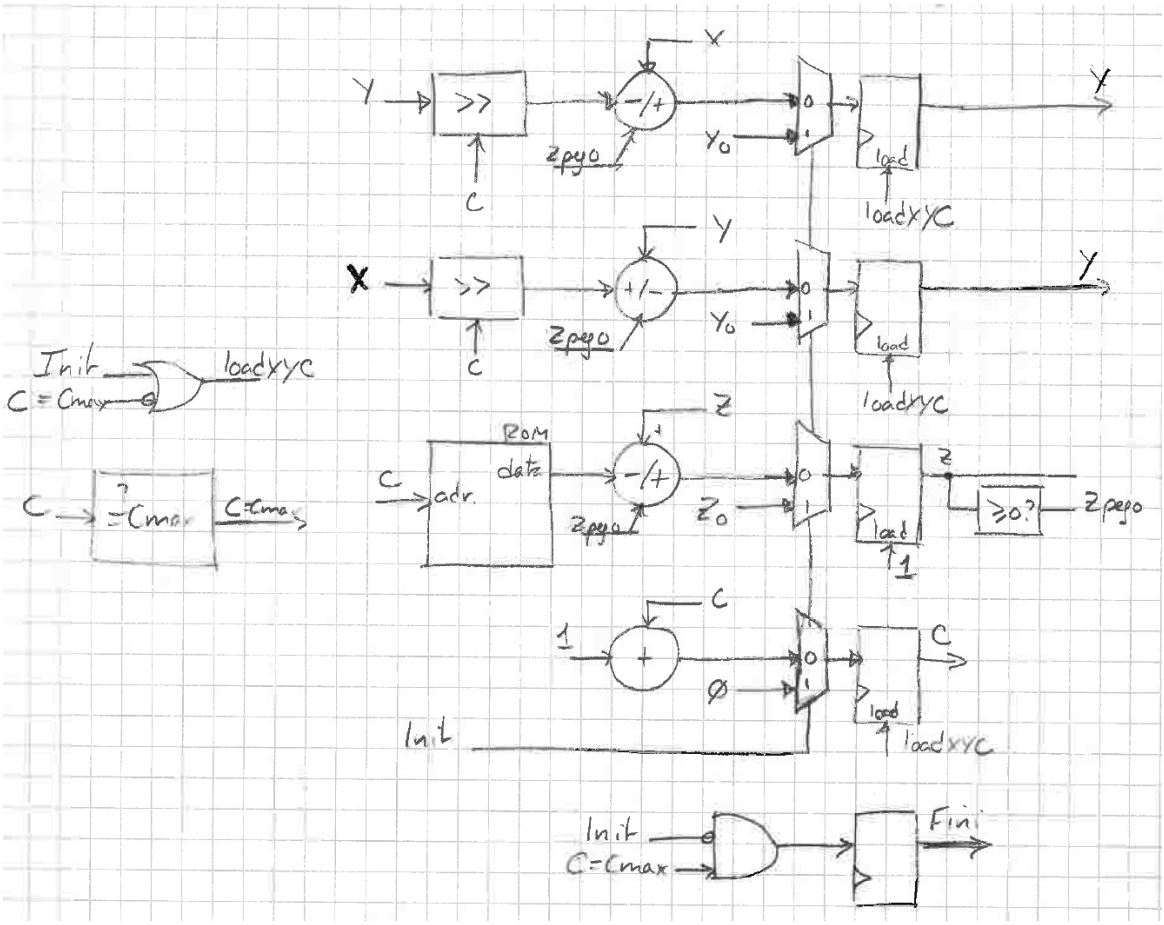
#1 options d'implémentation

	FPGA	Logique fixe
Ingénieurs	3 × 6 / 12 × 100 k$ = 150 k$	150 k$ × 1.5 = 225 k$
Licences	3 × 6 / 12 × 5 k$ = 7.5 k$	7.5 k$ × 4× 1.5 = 45 k$
Fonderie	-	400 k$ (inclut 1000 puces)
Total frais fixes	~157.5 k$	~670 k$
Total des frais unitaires	75 $	1000 puces incluses 1 $ ensuite
Total des frais pour 5400 installations	157500 + 75 × 5400 = 562500 $	670000 + 1 × (5400 – 1000) = 674400 $

Même avec une commande importante de plus de 5000 unités, les frais fixes énoncés font que la solution FPGA reste la plus économique. Il faudrait aussi tenir compte des délais de livraisons, dont les impacts financiers ne sont pas chiffrés dans la question. Il faudrait aussi tenir compte du risque associé à la logique fixe en cas de bogue.

#2 CORDIC

a. chemin des données



b. code

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

entity cordicfinal2017 is
   generic (
      W : positive := 16 -- largeur des registres
   );
   port(
   clk, init : in std_logic;
   Z0 : in signed(W - 1 downto 0); -- l'angle dont on veut le cos et le sin
   fini : out std_logic; -- le calcul est terminé
   X, Y : out signed(W - 1 downto 0) -- le cosinus et le sinus de l'angle
   );
end cordicfinal2017;

architecture arch of cordicfinal2017 is

constant X0 : signed(W - 1 downto 0) := to_signed(9949, W); -- sur 16 bits
constant Y0 : signed(W - 1 downto 0) := to_signed(0, W);
constant Cmax : integer := 12;

type ROM_type is array (natural range <>) of integer range 0 to 2 ** (W - 1);
constant ROM : ROM_type := (-- les valeurs ne sont pas données dans la question
16384, 9672, 5110, 2594, 1302, 652, 326, 163, 81, 41, 20, 10, 5);

signal C : integer range 0 to Cmax + 1;
signal Z : signed(W - 1 downto 0);

begin

   process(clk)
   begin
      if rising_edge(clk) then
         if init = '1' then
            C <= 0;
            fini <= '0';
            Z <= Z0;
            Y <= Y0;
            X <= X0;
         elsif C < Cmax then
            C <= C + 1;
            fini <= '1';
            if Z >= 0 then
               X <= X - shift_right(Y, C);
               Y <= Y + shift_right(X, C);
               Z <= Z - ROM(C);
            else
               X <= X + shift_right(Y, C);
               Y <= Y - shift_right(X, C);
               Z <= Z + ROM(C);
            end if;
         end if;

      end if;

   end process;

end arch;
```

c. ressources

FF : 3 registres de 16 bits + compteur de 4 bits + Fini = 53 FF

LUT : 3 add-sub de 16 bits = 48 LUT, 1 additionneur de 4 bits = 4 LUT, 2 décaleurs de 16 bits = 32 LUT, 3 mux de 16 bits = 48 LUT, 4 mux en parallèle de 1 bits = 4 LUT, 1 comparateur pour y = 0 : 5 LUT, 1 comparateur pour Cmax = 4 LUT, 2 portes logiques = 2 LUT, ROM de 12 mots de 16 bits = 12 LUT, total environ 160 LUT

Pas de DSP48E

d. Le débit est de 1 résultat par 12 cycles. La latence est de 12 cycles.

#3 Feistel

a. Le chemin critique va de la bascule pour le signal A à la bascule pour le signal Y, en passant par les quatre boîtes 'F' et les quatre ou-exclusifs. Le délai total est donné par :

délai bascule + 4 × délai 'F' + 4 × délai 'oux' + 9 × délai fil + tsu = 1 + 4 × 5 + 4 × 3 + 9 × 0.1 + 0.25 = 34.15 ns.

La fréquence maximale est donc 29.3 MHz

b. On insère un registre de pipeline à la sortie du 2e étage.

Le chemin critique est alors délai bascule + 2 × délai 'F' + 2 × délai 'oux' + 5 × délai fil + tsu = 1 + 2 × 5 + 2 × 3 + 5 × 0.1 + 0.25 = 17.75 ns. La fréquence maximale d'horloge est 56.3 MHz. On peut alors opérer à une fréquence d'horloge de 50 MHz et atteindre un débit de 50 × 106 résultats par seconde.

La latence est maintenant de 2 cycles, incluant le registre de pipeline et le registre des ports de sortie.

#4 Vérification

Test exhaustif est possible mais pas efficace, environ 605 G tests à faire.

Le partitionnement en classe est une option intéressante. Un choix de classes possible serait de considérer que pour la plupart des valeurs le but du module est de faire une incrémentation simple, sauf pour la dernière valeur qui provoque un changement dans les autres valeurs. Par exemple, pour 10 secondes, la prochaine valeur est 11 secondes. Pour 59 secondes, la prochaine valeur est 0 et il y a une incrémentation des minutes. On aurait donc:
Jour : {0, 1, 2, 3, 4, 5} et {6}
Heures : {0, 1, 2, … 22} et {23}
Minutes : {0, 1, 2, … 58} et {59}
Secondes : {0, 1, 2, … 58} et {59}
Millionièmes : {0, 1, 2, … 999998} et {999999}

Pour un test fort, il faudrait 2 × 2 × 2 × 2 × 2 = 32 vecteurs de test, ce qui prendrai 32 / 1E6 = 32 microsecondes.

L'approche des valeurs limites est probablement la meilleure option. On aurait :
jour : {-1, 0, 1, 3, 5, 6, 7}
heures : {-1, 0, 1, 13, 22, 23, 24}
secondes : {-1, 0, 1, 31, 58, 59, 60}
minutes : {-1, 0, 1, 31, 58, 59, 60}
millionièmes : {-1, 0, 1, 500023, 999998, 999999, 1000000}

En pratique on ne pourrait pas appliquer les valeurs interdites, donc on n'aurait que 5 valeurs par entrée; p. ex. pour le jour on aurait : {0, 1, 3, 4, 5}. Au final on aurait 5^5 = 3125 vecteurs de tests, ce qui ne prendrait 3.125 ms à faire.

#5 simulation

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 S1, S2, S3 (provoquées par les changements sur A, B, C)

À t = 0 + 3 Δ, évaluation de F et G (provoquées par les changements sur S1, S2, S3)

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 S1, S2, S3 (provoquée par le changement sur C)

À t = 10 ns + 2 Δ, évaluation de F et G (provoquée par les changements sur S1, S2, S3)

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

b.

temps	delta	A	B	C	S1	S2	S3	F	G
0 ns	0	U	U	U	U	U	U	U	U
0 ns	1	1	0	0	U	U	U	U	U
0 ns	2	1	0	0	1	1	0	U	U
0 ns	3	1	0	0	1	1	0	1	0
10 ns	0	1	0	1	1	1	0	1	0
10 ns	1	1	0	1	0	0	1	1	0
10 ns	2	1	0	1	0	0	1	0	0