Nom : _____ Matricule : _____

INF3500 : Conception et réalisation de systèmes numériques
Examen final – 7 mai 2021

1. Conditions de l'examen

- Cet examen est calibré pour une durée de 2 h 30. Une période additionnelle de 12 heures est allouée pour la remise de l'examen dans Moodle.
- L'examen est à livres ouverts, toutes les sources sont permises. Si vous utilisez du matériel trouvé en ligne ou ailleurs, citez bien vos sources dans votre réponse.
- L'examen doit être fait individuellement. Toute forme de communication avec d'autres personnes que le surveillant est interdite pendant l'examen. Une attestation à cet effet vous sera demandée dans la dernière question.
- La pondération de l'examen est de 50 %. La valeur de chaque question est indiquée.

2. Communication avec le surveillant pendant l'examen

- Le surveillant sera disponible de façon intermittente via le serveur Discord du cours pour aider à résoudre les problèmes techniques. Si le serveur Discord du cours ne fonctionne pas, envoyez un courriel à pierre.langlois@polymtl.ca.
- Ne posez pas de questions au surveillant concernant les questions de l'examen. Aucune réponse ne sera donnée. En cas de doute sur le sens d'une question, faites une supposition raisonnable, énoncez-la clairement dans votre réponse et poursuivez.

3. À propos de la remise

- Remettez vos réponses dans l'activité devoir de Moodle au plus tard à 23 h 59 (heure de Montréal).
- Les réponses en retard sont permises mais seront pénalisées de 1 point par minute après minuit.
- Les réponses doivent être remises dans un fichier .vhd (pour la question 2) et dans un document .pdf (pour les autres questions). Dans le document .pdf, en plus des réponses tapées et des tableaux remplis, il est acceptable d'inclure des captures d'écran, des diagrammes faits à la main, du texte écrit à la main etc.
- Sauvegardez souvent votre travail. Vous pouvez faire plusieurs remises dans Moodle.
- Pour les images incluses, assurez-vous :
  - qu'elles soient orientées correctement, bien cadrées, bien rognées et bien alignées;
  - qu'elles soient nettes partout, que l'éclairage soit uniforme et que le contraste soit suffisamment élevé;
  - d'en limiter la taille – le document au complet doit faire moins de 5 Mo.
- Commencez la réponse à chaque question sur une nouvelle page.

À l'usage des correcteurs :

Q1	Q2	Q3	Q4	Q5	Q6	Q7 déclaration	Total sur 50

Question 1. (8 points : 24 minutes)

C'est aujourd'hui le lundi 10 mai, et vous débutez votre stage d'été chez Montréal-SN, une petite entreprise fondée par Charles et Rana, deux diplômés récents de Polytechnique. Ils ont développé un système numérique pour sécuriser les transactions dans les guichets automatiques. La Banque Nationale leur a demandé de livrer 25 prototypes pour faire des tests, mais ils ont l'œil sur un marché potentiel de plus de 500 k guichets automatiques en Amérique du Nord. Charles privilégie une solution sur FPGA pour sa flexibilité, mais Rana pense qu'il vaut mieux investir en logique fixe tout de suite à cause de son potentiel supérieur de performance. Vous discutez avec Charles et vous estimez qu'il reste environ 2 mois de travail à 3 ingénieur/es pour terminer la conception pour FPGA. Vous estimez aussi que l'achat de cartes de développement et des outils de synthèse et d'implémentation pour l'équipe coûterait autour de 5 k$. Ensuite, les cartes avec FPGA qui pourraient servir pour le produit final coûteraient 235 $ l'unité, et elles pourraient être livrées dans 4 semaines, soit avant que la conception soit terminée. Rana vous explique que, pour la logique fixe, la conception prendrait 6 mois à 4 ingénieur/es, parce qu'il faudrait aussi concevoir une carte sur laquelle monter les puces et prévoir de l'alimentation et des interfaces. Les licences de logique fixe coûteraient au total 25 k$. Malgré la pénurie mondiale actuelle de puces à semi-conducteurs, Rana a obtenu une soumission pour la fabrication de puces dans une technologie éprouvée et les frais fixes de la fonderie seraient limités à 95 k$, plus 2 $ par puce fabriquée, mais la livraison serait faite 4 mois après la fin de la conception. Pour faire fabriquer les cartes et y monter les puces en logique fixe, les coûts seraient de 35 $ par carte. Pour les deux technologies, les salaires et avantages sociaux annuels des ingénieur/es s'élèvent à 120 k$ par personne.

a. (6 points) Analyse des coûts

Critère	FPGA	Logique fixe

À partir de combien d'unités la solution en logique fixe serait-elle plus rentable que la solution FPGA ? Montrez tous vos calculs.

_____

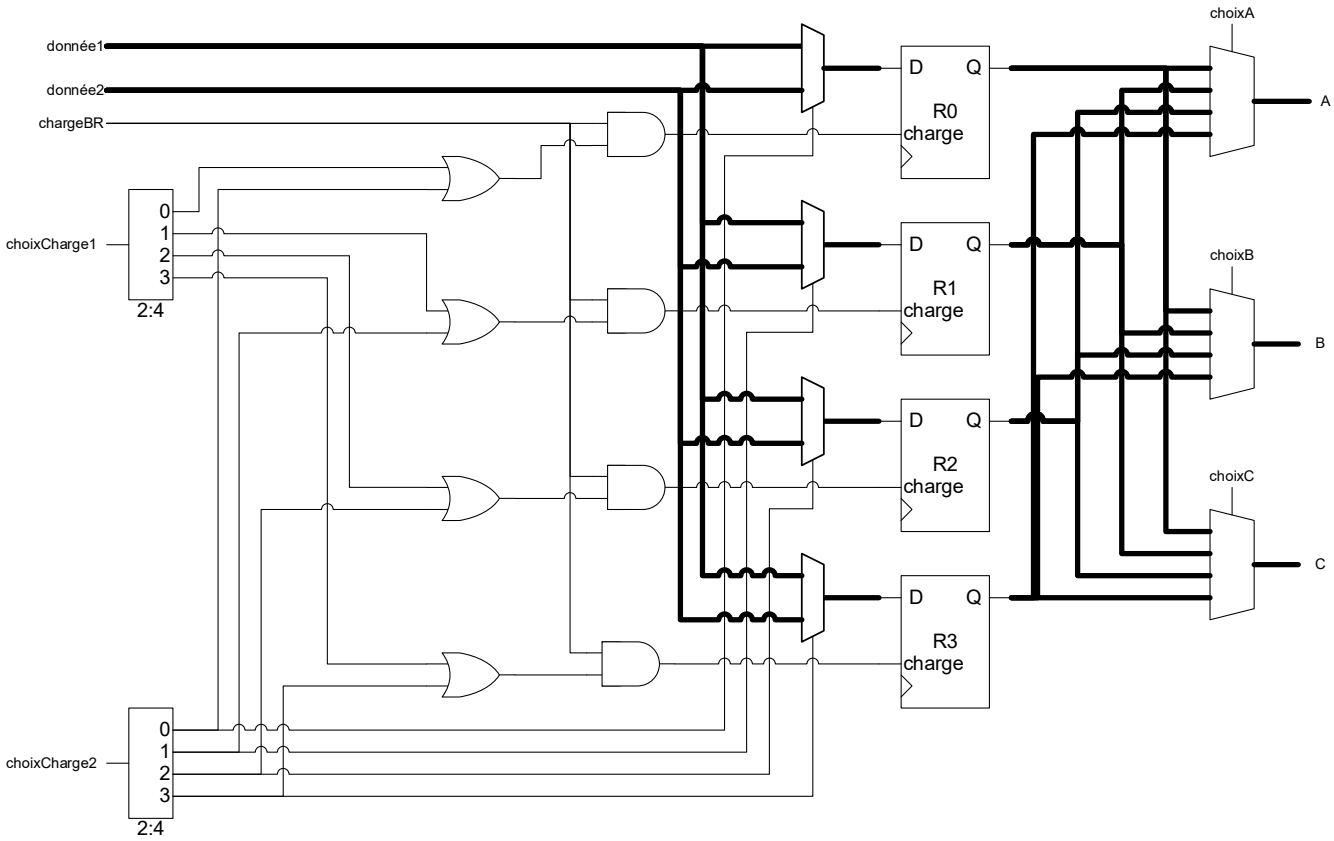
b. (2 points) Quelle stratégie recommandez-vous à ces entrepreneurs ? Pourquoi ?

_____
_____
_____
_____
_____
_____
_____

Question 2. (10 points : 30 minutes)

Considérez le diagramme suivant représentant un bloc des registres à deux ports d'entrée et trois ports de sortie. Ce bloc des registres pourrait remplacer le bloc des registres du processeur PolyRISC vu en classe, de façon à étendre son jeu d'instructions et en augmenter la performance.

Les signaux de contrôle choixA, choixB et choixC indiquent respectivement le numéro du registre à aiguiller sur chacun des ports de sortie A, B, C. Les signaux choixCharge1 et choixCharge2 indiquent respectivement dans quel registre écrire les valeurs sur les ports donnée1 et donnée2. Le signal chargeBR indique si ces valeurs doivent être chargées ou non. Dans le cas problématique où choixCharge1 = choixCharge2, alors c'est le registre indiqué par choixCharge2 qui sera chargé et la valeur placée sur donnée1 ne sera pas considérée. [Un circuit plus robuste détecterait cette erreur et lancerait une exception, mais ce n'est pas fait ici.]



Donnez une architecture pour l'entité suivante en VHDL correspondant à ces spécifications. Remettez votre code dans un fichier blocregistres2in3out.vhd.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity blocregistres2in3out is
  generic (
    Nreg : positive := 4;     -- nombre de registres
    Wd : positive := 8      -- largeur du chemin des données en bits
  );
  port(
    reset, CLK : in std_logic;
    donnee1, donnee2 : in signed(Wd - 1 downto 0);
    chargeBR : in std_logic;
    choixCharge1, choixCharge2, choixA, choixB, choixC : in natural range 0 to Nreg - 1;
    A, B, C : out signed(Wd - 1 downto 0)
  );
end blocregistres2in3out;
architecture arch of blocregistres2in3out is
-- votre code ici
begin
-- votre code ici
end arch;
```

Question 3. (10 points : 30 minutes)

On peut calculer la racine carrée X = sqrt(A) d'un nombre A par la méthode itérative de Newton (https://fr.wikipedia.org/wiki/Méthode_de_Newton#Racine_carrée).

À chaque itération k, on calcule : $X_{k+1} = (X_k + A / X_k) / 2$

et la valeur de Xk converge vers sqrt(A) après quelques itérations.

Si A et Xk sont exprimés sur 16 et 12 bits, respectivement, on peut débuter les calculs avec l'approximation $X_0 = A / 32$.

Pour A = 42871, on obtiendrait la séquence suivante en arrondissant les Xk à l'entier le plus proche à chaque étape :

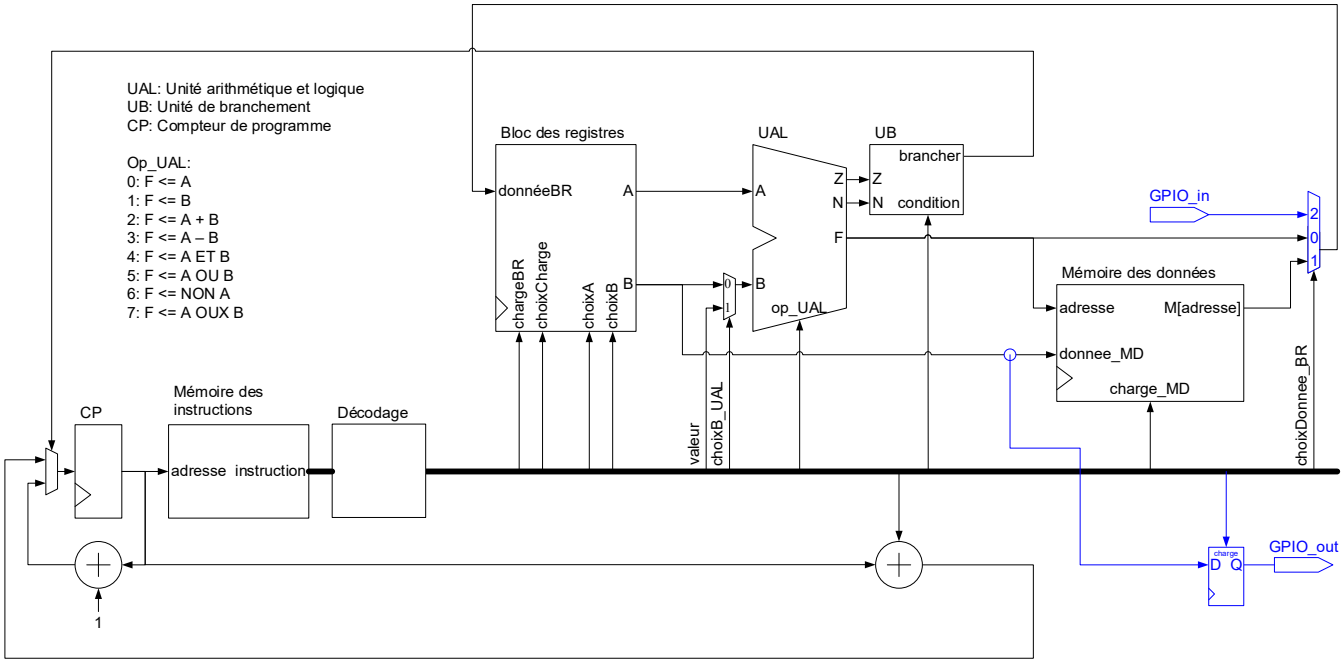
k	Xk
0	1340
1	686
2	374
3	244
4	210
5	207

a. (5 points) Faites la conception d'un chemin des données correspondant à cet algorithme et donnez son diagramme. En plus des blocs fonctionnels habituels, vous pouvez supposer que vous avez accès à la fonction de division d'un nombre de 16 bits par un nombre de 12 bits. Votre chemin des données doit faire 5 itérations comme montré dans l'exemple du calcul de la racine carrée de 42871.

b. (5 points) Estimez le nombre de ressources nécessaires (LUT, FF et DSP48E) pour implémenter votre chemin des données de la partie a. dans un FPGA de la famille Artix-7, **à l'exclusion de la fonction de division**.

Question 4. (6 points : 18 minutes)

Considérez le diagramme suivant du processeur PolyRISC. Le bloc des registres a 16 registres de 32 bits et le compteur de programme a 16 bits.



a. (4 points) Complétez le tableau suivant des signaux de contrôle du chemin des données pour chaque opération. Si une opération est impossible à réaliser avec ce chemin des données, placez des 'F' dans les cases de la ligne correspondante.

opération	chargeBR	choixCharge	choixA	choixB	valeur	ChoixB_UAL	Op_UAL	Charge_MD	choixDonnee_BR	charge (GPIO_out)
M[R0 + 33] := GPIO_in;										
R3 := M[R13 + 55];										
M[R5 xor 77] := R15;										
R7 := R17 OU 99;										

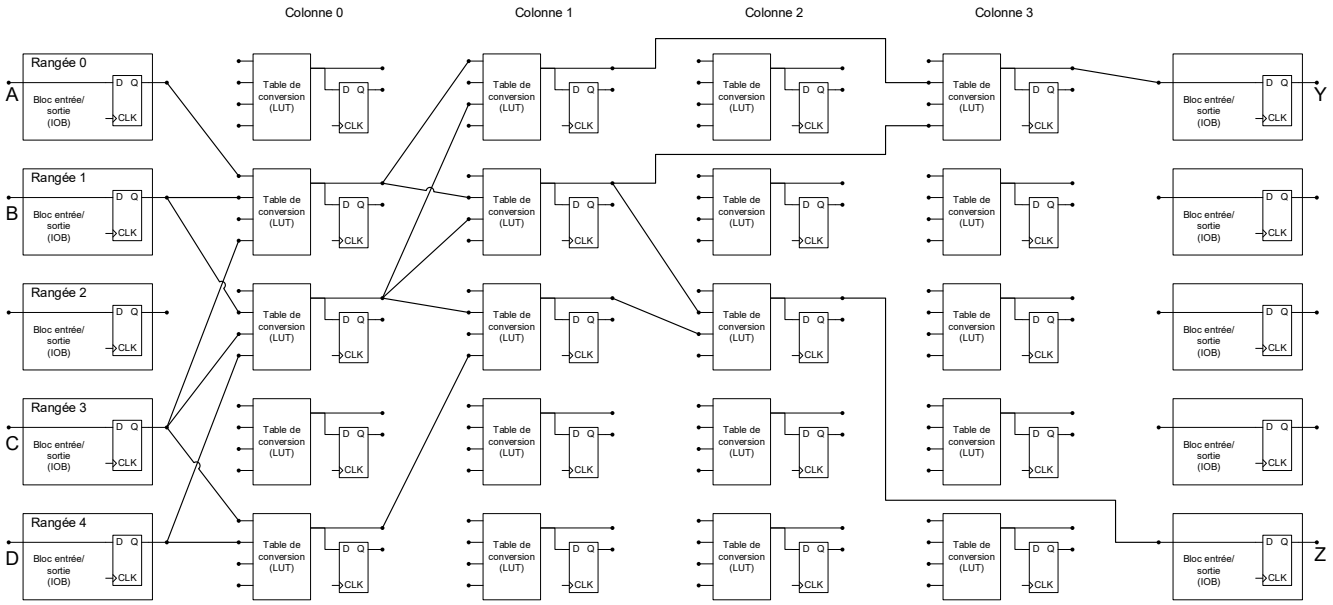
b. (2 points) L'unité de branchement contribue à réaliser les instructions du type suivant : `si (RA condition RB) goto (CP + valeur)`. Elle a trois entrées et une sortie. Sa sortie doit être vraie quand il faut effectuer un branchement. Pour cette instruction, l'unité arithmétique et logique du processeur calcule la différence RA – RB et fournit les signaux `Z` et `N` à l'unité de branchement. L'entrée `Z` indique si la différence est 0 et l'entrée `N` indique si la différence est négative. L'entrée `condition`, exprimée avec 3 bits (un entier de 0 à 7) spécifie quand il faut brancher : 0: si RA = RB, 1: si RA != RB, 2: si RA < RB, 3: si RA > RB, 4: si RA <= RB, 5: si RA >= RB 6: brancher, indépendamment de Z et N, 7: ne pas brancher, indépendamment de Z et N

Estimez les ressources nécessaires pour implémenter l'unité de branchement de PolyRISC dans un FPGA de la famille Artix-7. Énoncez clairement toutes vos suppositions.

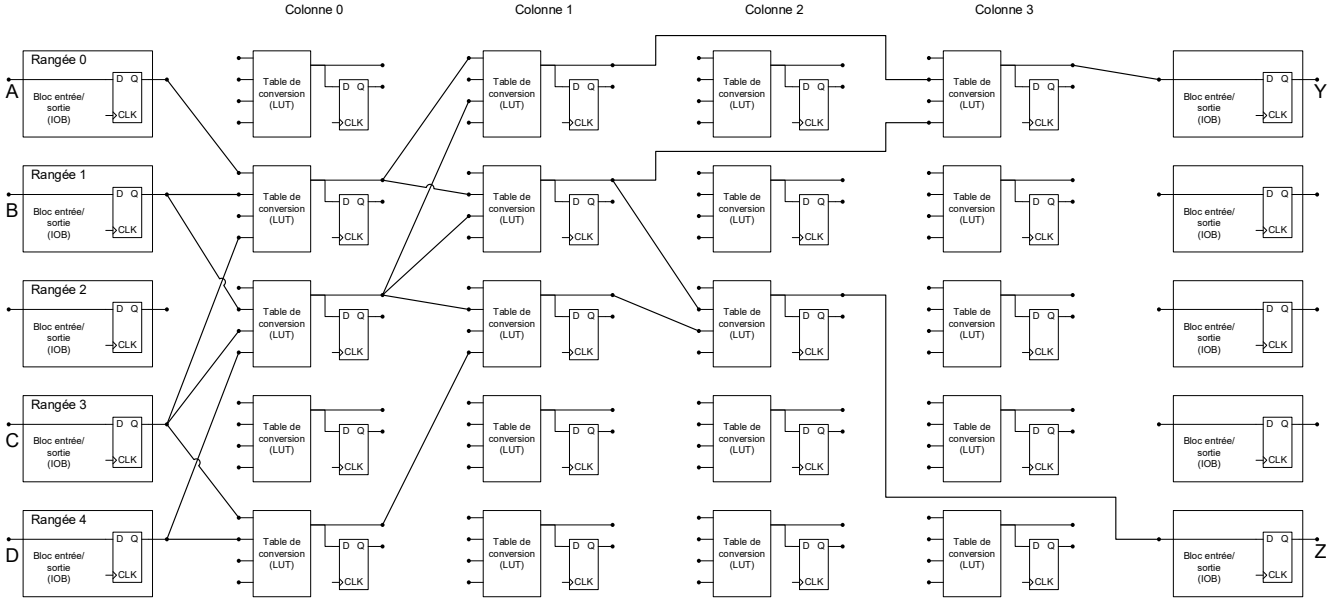
Question 5. (10 points : 30 minutes)

Le diagramme suivant montre un circuit implémenté sur un FPGA simplifié. Les entrées du circuit sont A, B, C et D et les sorties sont Y et Z. Le FPGA est composé de tranches ayant chacune une table de conversion (LUT) et une bascule pouvant être utilisée ou non, et de blocs d'entrée/sortie (IOB) dans lesquels les bascules sont toujours utilisées. Les LUTs ont un délai de 3.5 ns. Les bascules ont un délai de 1.0 ns, un temps de préparation t_{su} de 0. 5 ns, et un temps de maintien t_h de 0.65 ns. Les interconnexions ajoutent un délai de 0.1 ns pour chaque rangée et chaque colonne de distance.

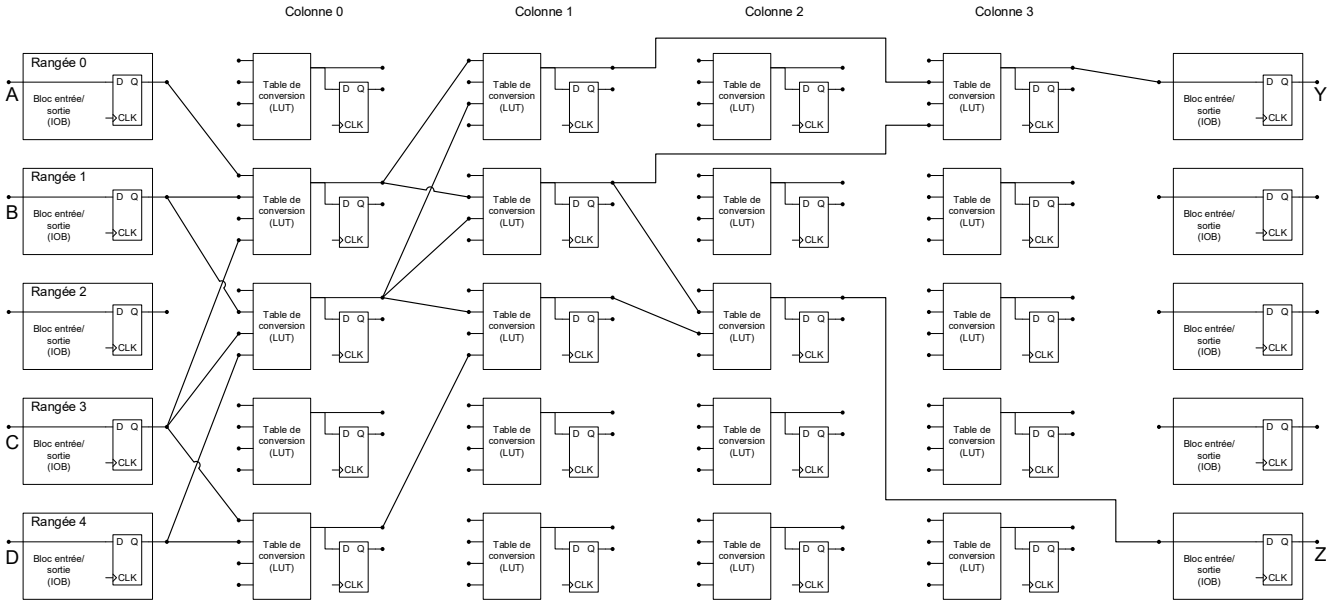
a. (4 points) Annotez le diagramme pour identifier le chemin critique du circuit. Calculez la fréquence maximale d'horloge. Montrez tous vos calculs.



b. (3 points) On veut pipeliner le circuit pour atteindre un débit de 185×10^6 résultats par seconde, où un résultat est une paire {Y,Z}, tout en ayant la latence la plus courte possible, et sans changer la placement indiqué sur le diagramme. Indiquez clairement sur le diagramme suivant où et comment ajouter des registres de pipeline. Identifiez le nouveau chemin critique et donnez la nouvelle fréquence maximale d'horloge. Montrez tous vos calculs.



c. (2 points) Par rapport à votre réponse en b., est-il possible d'augmenter encore le débit sans augmenter la latence et sans changer le placement des ports d'entrée et de sortie, qui sont déterminés par la position des pattes de la puce sur la carte ? Expliquez complètement votre réponse en annotant le diagramme suivant, et, le cas échéant, donnez la fréquence d'horloge atteinte.



d. (1 point) On veut atteindre une fréquence d'horloge de 250 MHz pour ce circuit sur ce FPGA, en ajoutant autant de registres de pipeline que nécessaire en déplaçant si nécessaire les ressources, les entrées et les sorties. Est-ce possible ? Justifiez complètement votre réponse.

Question 6. (6 points : 18 minutes)

Le laboratoire #3 ce de trimestre portait sur la conception d'un contrôleur de feux de circulation. Cette question porte sur vos travaux pendant ce laboratoire, et en particulier sur le problème de la vérification du circuit séquentiel que vous avez conçu. *Comme pour le reste de l'examen, et bien que vous ayez travaillé en équipe sur ce laboratoire, votre réponse à cette question doit nécessairement être individuelle.*

[L'évaluation de cette question est globale, et portera sur votre compréhension des principes de vérification et des techniques de test vus à la semaine #7.]

a. Décrivez votre approche pour effectuer la vérification de votre circuit, soit la spécification de base (partie 2 du laboratoire), ou bien la spécification avancée (partie 3b du laboratoire). Soyez honnête ! Vous pouvez écrire "nous n'avons pas complété ces parties" ou "c'est ma co-équipière qui a complété cette partie."

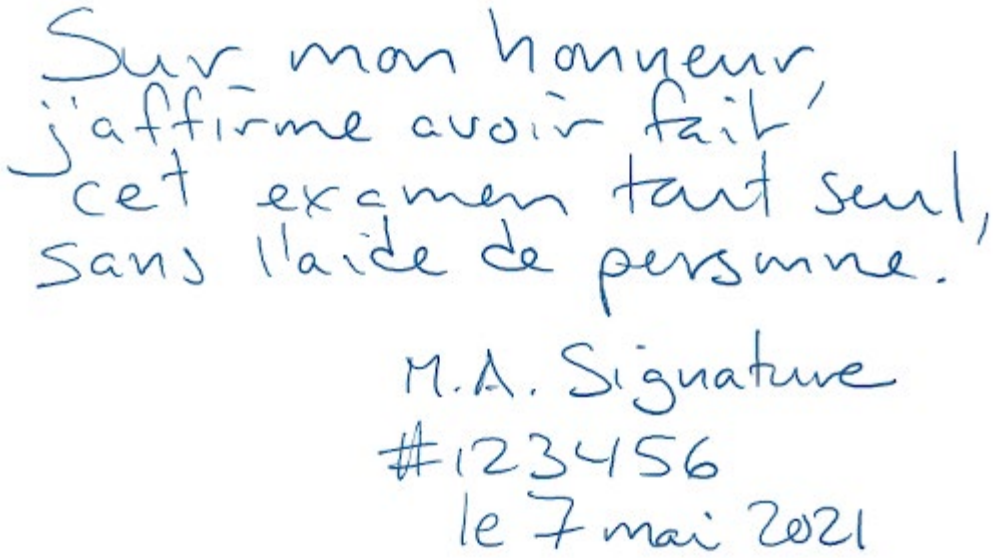
b. En vous basant sur les principes de la vérification et sur les techniques de tests de systèmes numériques vus dans le cours ce trimestre, faites une évaluation critique de votre approche de vérification telle que décrite en a., ou bien proposez une approche pour vérifier un circuit séquentiel numérique pour contrôler des feux de circulation.

Question 7. (0 points)

Déclaration sur l'honneur (obligatoire)

Produisez une affirmation solennelle sur votre honneur, écrite à la main, signée et datée, à l'effet que vous avez complété cet examen seul/e et sans l'aide de personne.

Joignez une image de votre affirmation en guise de réponse. Voir l'exemple ci-dessous.



Temps pour compléter l'examen (facultatif)

Il m'a fallu _____ heures et _____ minutes pour répondre à cet examen, à l'exclusion du temps pour numériser mes réponses et soumettre mes fichiers dans Moodle.

Il m'a fallu _____ heures et _____ minutes pour numériser mes réponses et soumettre mes fichiers dans Moodle.

Solutions

#1

salaires	120000	par an	
critère	fpga	fixe	
salaires	60000	240000	
frais fixes	5000	120000	
total	65000	360000	
frais unitaires	235	37	
équilibre	1490		

a. On pose 65 k$ + 235 $ × n = 360 k$ + 37 $ × n, on trouve n = 1490.

La solution en logique fixe devient plus rentable à partir de 1500 unités environ.

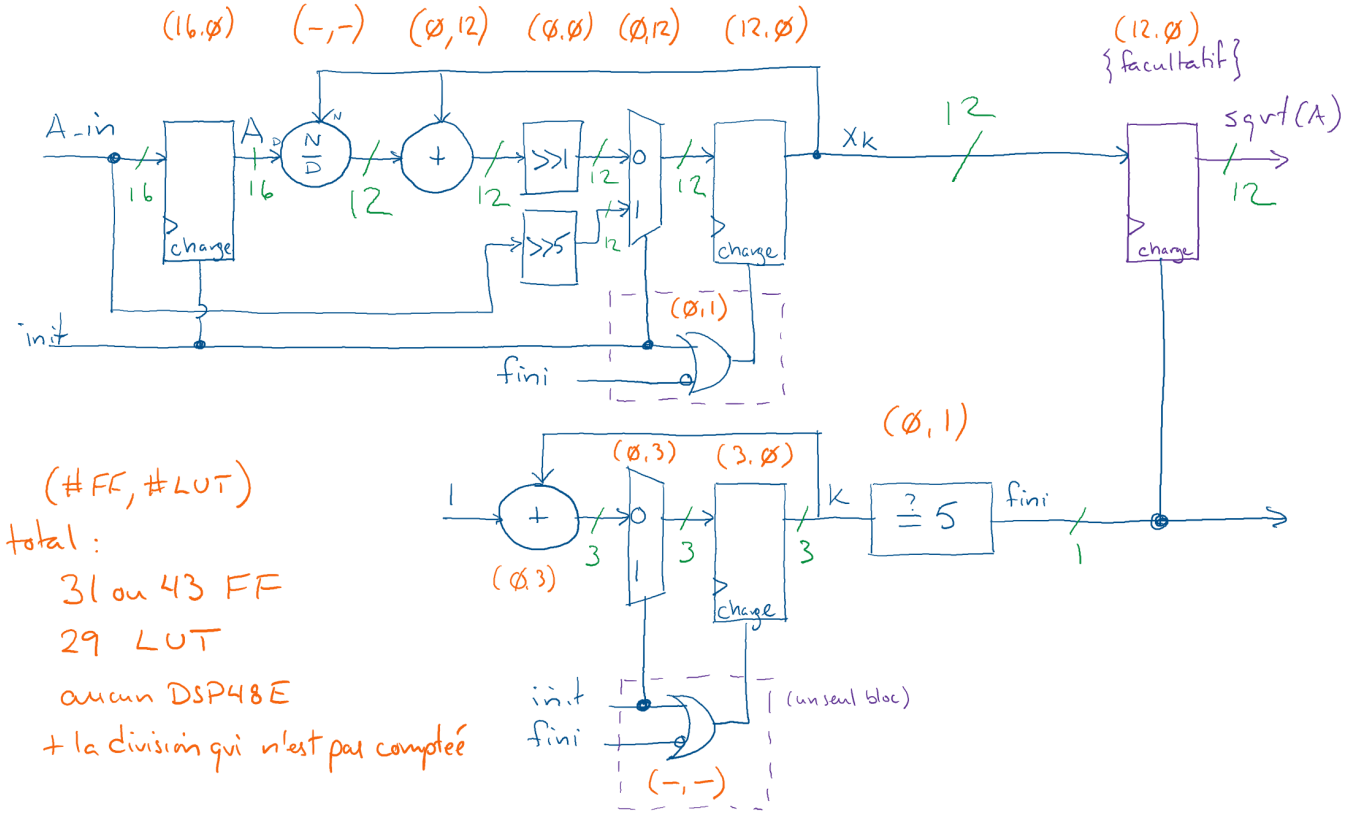
b. Pour le prototype, on devrait nécessairement y aller avec des FPGA, en plus du fait que la livraison est très rapide. On pourrait effectivement installer les prototypes dans deux mois.

La logique fixe serait considérablement plus rentable si une grosse commande (5 k, 10 k, 100 k + unités) devait être passée, mais on n'aurait les systèmes que dans 10 mois. Il faudrait être vraiment certains d'avoir la commande avant de se lancer dans la logique fixe, mais alors les coûts seraient beaucoup plus bas. Les risques seraient cependant plus grands. On ne pourrait pas facilement mettre le système à jour en cas d'erreur ni si on devait apporter des modifications après le déploiement.

#2

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

entity blocregistres2in3out is
  generic (
    Nreg : positive := 4;     -- nombre de registres
    Wd : positive := 8      -- largeur du chemin des données en bits
  );
  port(
    reset, CLK : in std_logic;
    donnee1, donnee2 : in signed(Wd - 1 downto 0);
    chargeBR : in std_logic;
    choixCharge1, choixCharge2, choixA, choixB, choixC : in natural range 0 to Nreg - 1;
    A, B, C : out signed(Wd - 1 downto 0)
  );
end blocregistres2in3out;

architecture arch of blocregistres2in3out is

type lesRegistres_type is array(0 to Nreg - 1) of signed(Wd - 1 downto 0);
signal lesRegistres : lesRegistres_type;

begin

  process (CLK, reset)
  begin
    if rising_edge(CLK) then
      if reset = '1' then
        lesRegistres <= (others => (others => '0'));
      else
        if chargeBR = '1' then

          -- option A: super simple
          lesRegistres(choixCharge1) <= donnee1;
          lesRegistres(choixCharge2) <= donnee2;

          -- option B: itération sur les registres avec une boucle
--          for k in 0 to Nreg - 1 loop
--            if choixCharge2 = k then
--              lesRegistres(k) <= donnee2;
--            elsif choixCharge1 = k then
--              lesRegistres(k) <= donnee1;
--            end if;
--          end loop;
--
--          -- option C: à la dure ...
--          -- décrire les composantes une à une ... il y a des options intermédiaires

        end if;
      end if;
    end if;
  end process;

  A <= lesRegistres(choixA);
  B <= lesRegistres(choixB);
  C <= lesRegistres(choixC);

end arch;
```

#3.



(#FF, #LUT)

total :

31 ou 43 FF

29 LUT

aucun DSP48E

+ la division qui n'est pas comptée

#4.

a.

opération	chargeBR	choixCharge	choixA	choixB	valeur	ChoixB_UAL	Op_UAL	Charge_MD	choixDonnee_BR	charge (GPIO_out)
M[R0 + 33] := GPIO_in;	F	F	F	F	F	F	F	F	F	F
R3 := M[R13 + 55];	1	3	13	-	55	1	2	0	1	0
M[R5 xor 77] := R15;	0	-	5	15	77	1	7	1	-	0
R7 := R17 OU 99;	1	7	17	-	99	1	5	0	0	0

b.

La fonction réalisée par le bloc UB est combinatoire et a cinq entrées (Z, N et trois bits pour condition).
Il faut donc une seule LUT (qui a un maximum de six entrées), aucun registre, et aucune tranche DSP48E.

#5

a. Le chemin critique est montré ici. Le délai sur le chemin critique est:

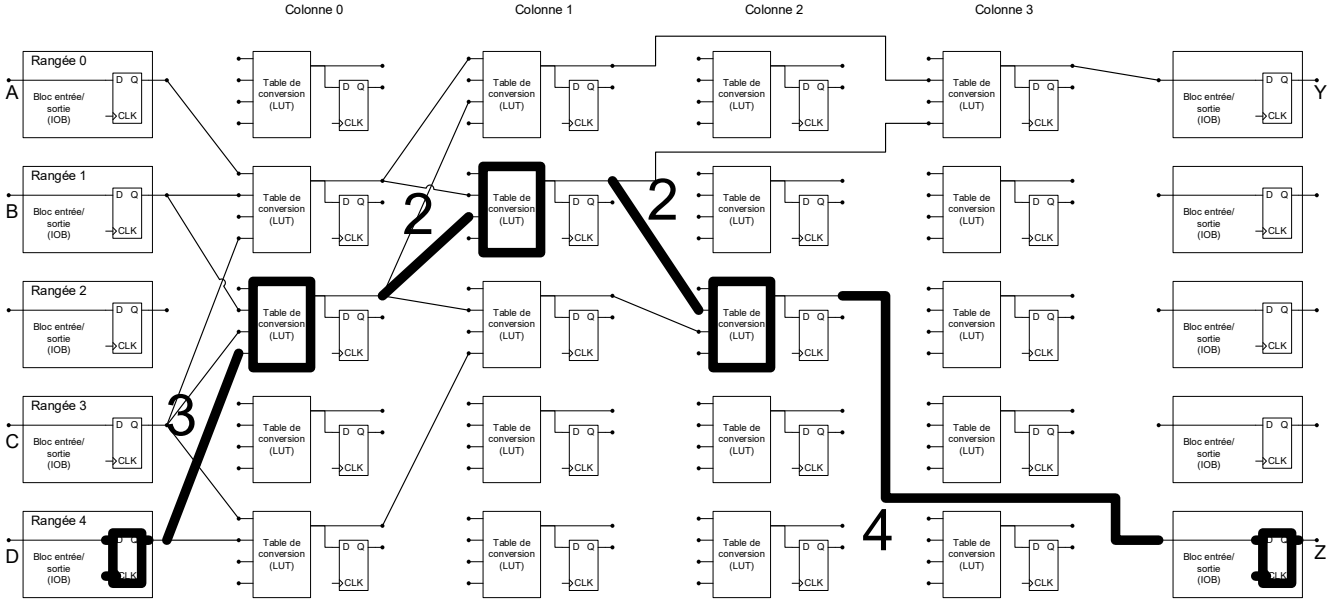
1 × td = 1.0 ns

3 × LUT = 10.5 ns

1 × tsu = 0.5 ns

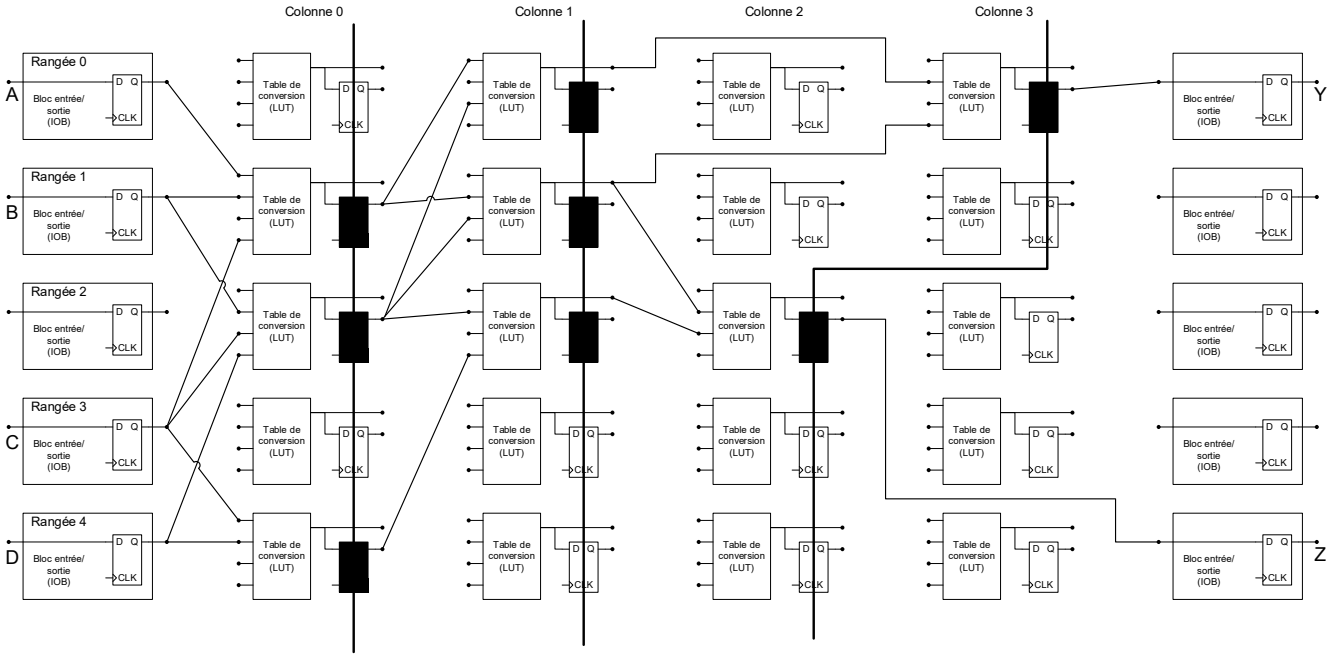
(3 + 2 + 2 + 4 =11) × fils = 1.1 ns

total 13.1 ns, pour une fréquence maximale d'horloge de 76.3 MHz



b. On ajoute trois niveaux de pipeline tel que montré ici.

Le chemin critique est td + 1 × LUT + tsu + 3 × fils = 1.0 + 3.5 + 0.5 + 0.3 = 5.3 ns, pour une fréquence d'horloge de 188.7 MHz.



c. Non ce n'est pas possible si on ne peut pas bouger les entrées B et D. En effet, la LUT en position (R2C0) est à 3 fils de distance de l'entrée D. On pourrait la déplacer en position (R3C0), mais alors elle serait à une distance de 3 fils de l'entrée B. Le chemin critique aura donc nécessairement toujours 3 fils, pour une fréquence maximale d'horloge de 188.7 MHz tel que vu en b.

d. Non ce n'est pas possible, par ce que le délai minimal pour ce FPGA serait :

1 × td = 1.0 ns

1 × LUT = 3.5 ns

1 × tsu = 0.5 ns

1 × fil = 0.1 ns

total 5.1 ns, pour une fréquence maximale d'horloge de 196.1 MHz.

#6. La discussion doit inclure les éléments suivants :

- l'importance de garantir la sécurité des usager;
- la possibilité ou non de faire un test exhaustif;
- la difficulté d'énumérer tous les cas possibles;
- la difficulté de bien cerner la spécification complète;
- la nécessité absolue de faire un test de couverture de code et atteindre 100% de couverture.

Dans le cas du système de contrôle des feux de circulation, il faut idéalement vérifier la bonne séquence des états. Ça impliquerait donc de construire un modèle du contrôleur, parallèle à celui qui a été conçu au départ. On devrait confier cette tâche à une équipe séparée.

Il est très important de vérifier complètement (c'est difficile d'en être certain, mais pas impossible) que les situations dangereuses interdites (p. ex. feux verts dans deux directions) ne se produisent pas.

Un grand défi consiste à confirmer que les délais observés pour chaque état sont bien ceux qui sont spécifiés.

Le système ayant très peu d'entrées (l'horloge et le reset ne comptent pas), les approches de partitionnement en classe ou de tests de valeurs limites s'appliquent mal.