Examen partiel

Département de génie électrique et de génie informatique
Systèmes VLSI - GIF19264

le 26 octobre 2005

Vous avez droit à tous les documents. SVP, *pas* d'ordinateurs portables.
Durée de l'examen: 2 heures (13h30-15h30).

---

1. *Synthèse combinatoire* (10 points)

(a) Quel type de synthèse effectue-t-on lorsqu'on utilise les tables de Karnaugh? Soyez précis.

Synthèse combinatoire à deux niveaux

(b) Quel algorithme permet d'automatiser ce type de synthèse?

Algorithme de Quine-McClusky

(c) Identifiez les avenues permettant d'optimiser le nombre de portes logiques au-delà de ce que donnent les tables de Karnaugh.

1. Élimination d'inverseurs via le théorème de deMorgan
2. Mise en commun de portions de circuits dans la génération de plusieurs sorties
3. Synthèse "multi-niveaux"

(d) Quels peuvent être les désavantages d'une optimisation plus poussée comme suggérée en (c)?

1. Délai de propagation potentiellement plus grand pour les options 2 et 3
2. Effort de conception plus grand

(e) Quelles conditions doit-on respecter pour qu'un processus en VHDL mène à la synthèse d'un circuit combinatoire?

Toutes les entrées doivent être dans la liste de sensitivité et aucune sortie du processus ne doit être lue en entrée.

2. *syntaxe, sémantique, types, attributs* (20 points)

(a) (6 points) Soit la description VHDL suivante:

```
entity mux2a1
   port( A, B: in std_ulogic;
         Sel: in std_ulogic;
         Y: out std_ulogic));
end entity mux2a1;

architecture flot of mux2a1 is
begin
   P1: process (A, B, Sel)
   begin
      if Sel='0' then
         Y <= A;
      else
         Y <= 'Z';
      end if;
   end process P1;

   P1: process (A, B, Sel)
   begin
      if Sel='1' then
         Y <= B;
      else
         Y <= 'Z';
      end if;
   end process P2;

   Y <= 'Z' when Sel = 'Z';
end architecture flot;
```

Cette description est-elle erronée? Si oui, quelles sont les erreurs? Peut-elle être synthétisée avec ISE? Justifiez.

Plusieurs pilotes (les processus P1 et P2, l'assignation parallèle) alimentent le signal Y qui est pourtant d'un type non-résolu (std_ulogic). Un tel type ne permet pas plusieurs pilotes. La description ne peut être synthétisée car l'assignation teste que la ligne Sel = 'Z'; or, un circuit ne peut tester l'état haute-impédance, qui se manifeste comme une tension flottante. Erreurs mineures: les deux processus ont la même étiquette (P1); il manque un mot-clé **is** *aux lignes 1, 8 et 16; il y a une parenthèse de trop à la ligne 4.*

(b) (2 points) Étant donné la description en (a), quelle est la valeur de l'attribut flot'Structure?

false

(c) (2 points) Étant donné la description en (a), quelle est la valeur de l'attribut flot'Behavior?

true

(d) (10 points) Soit la déclaration suivante:

subtype periodes **is time range** 10 ns **to** 10 ms;

Quelle est la valeur des attributs suivants:

(i) 'right 10^{13} *fs*

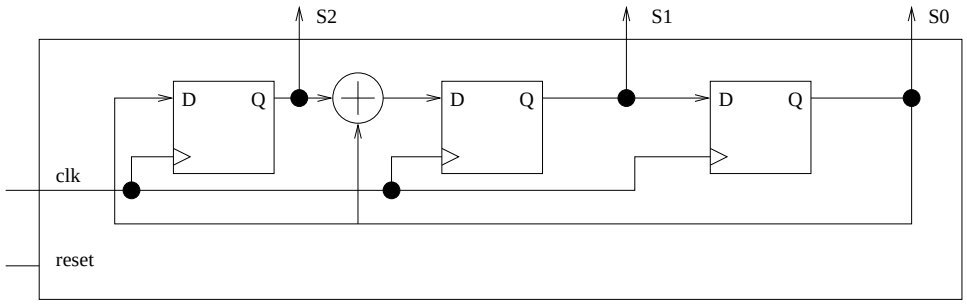
(ii) 'low 10^{7} *fs*

(iii) 'ascending *true*

(iv) 'succ(10 ns) *10000001 fs*

(v) 'pred(10 ns) *génère une erreur - pas d'élément précédent 10 ns.*

3. *Registres à décalage à rétroaction linéaire* (30 points) On souhaite utiliser le registre à décalage à rétroaction linéaire (normalement employé pour générer des bits aléatoires) suivant pour réaliser un compteur:



(a) (20 points) Écrivez une description comportementale en VHDL (entité et architecture) qui réalise ce circuit.

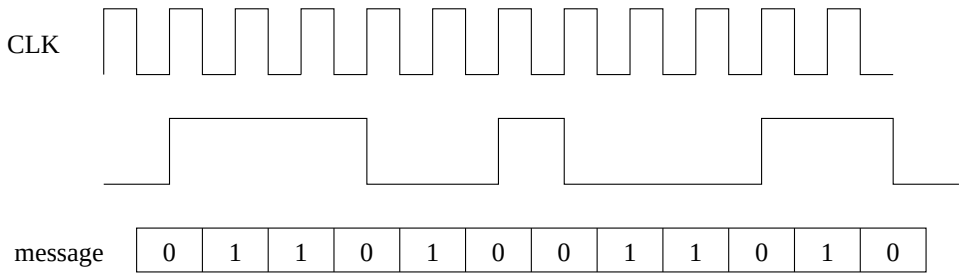
```
entity compteur
   port( clk, reset: in std_logic;
         S0, S1, S2: out std_logic));
end entity compteur;

architecture comport of compteur is
begin
   P1: process(clk, reset)
   begin
      if reset='1' then
         S0 <= '1';
         S1 <= '0';
         S2 <= '0';
      elsif clk='1' and clk'event then
         S2 <= S0;
         S1 <= S2 xor S0;
         S0 <= S1;
      end if;
   end process P1;
end architecture comport;
```

(b) (10 points) Selon vous, quels seraient les avantages et inconvénients d'un tel compteur par rapport à un compteur conventionnel?

Avantages: Il est très simple à implanter et plus rapide qu'un compteur conventionnel, car il n'y a pas de propagation de retenue. Il passe par tous les états une et une seule fois avant de boucler — il est donc efficace au niveau de l'utilisation des bascules. Désavantages: la séquence produite n'est pas une séquence croissante naturelle. Ne permet d'état "000".

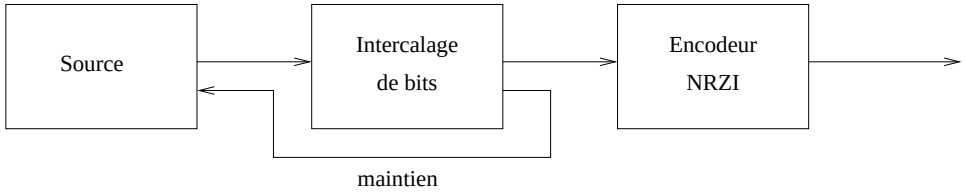
4. *Design* (40 points) On utilise en communications numériques sur ligne câblée (e.g. paire torsadée, câble coaxial) divers "codes de ligne" pour communiquer en bande passante, c'est-à-dire sans moduler sur une porteuse. Ces codes constituent tout simplement une façon de représenter les bits à transmettre qui procure des caractéristiques spectrales intéressantes et / ou des qui propriétés faciliteront la synchronisation au récepteur.

Un tel code de ligne est le code "NRZI" (Non Return to Zero Inverted) qui est utilisé dans le standard USB (Universal Serial Bus). Dans ce code, seuls les '0' génèrent une transition. Ainsi, on a:



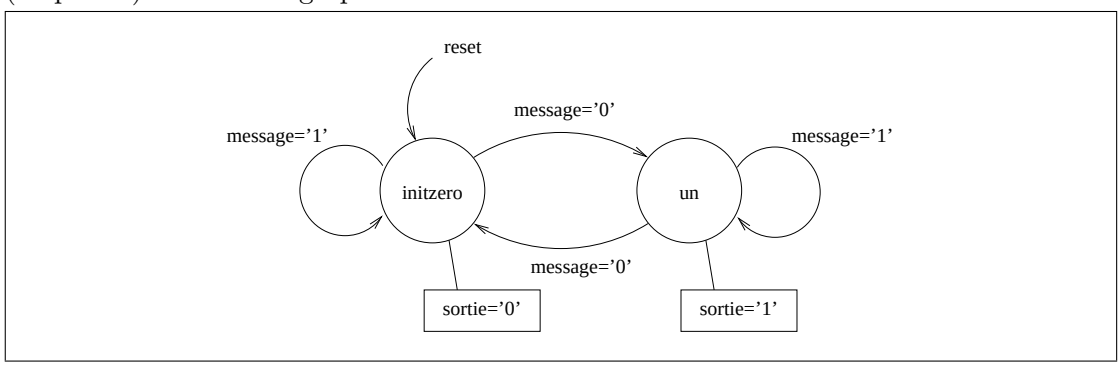
Cet encodeur possède un désavantage majeur: si une trop longue séquence de bits '1' successifs est transmise, il n'y a aucune transition sur la ligne ce qui rend la synchronisation entre le transmetteur et le récepteur impossible. Pour remédier à cela, le standard USB prévoit une opération d'intercalage de bits ("bit stuffing"): dès qu'une chaîne de 6 '1' est transmise, on intercale un '0' pour forcer une transition. Il est facile d'enlever ces '0' au récepteur pour recomposer le message original.

Une portion de la chaîne de communication correspondante est illustrée ici:



La source génère les bits du message à transmettre sur le câble USB. Le bloc d'intercalage doit être capable de détecter une séquence de 6 '1' successifs et de transmettre alors un '0' à l'encodeur. À ce moment, la ligne "maintien" est placée à '1' pour un coup d'horloge, afin de signaler à la source de maintenir son bit courant qui ne pourra être transmise qu'au prochain coup (après le '0' intercalé).

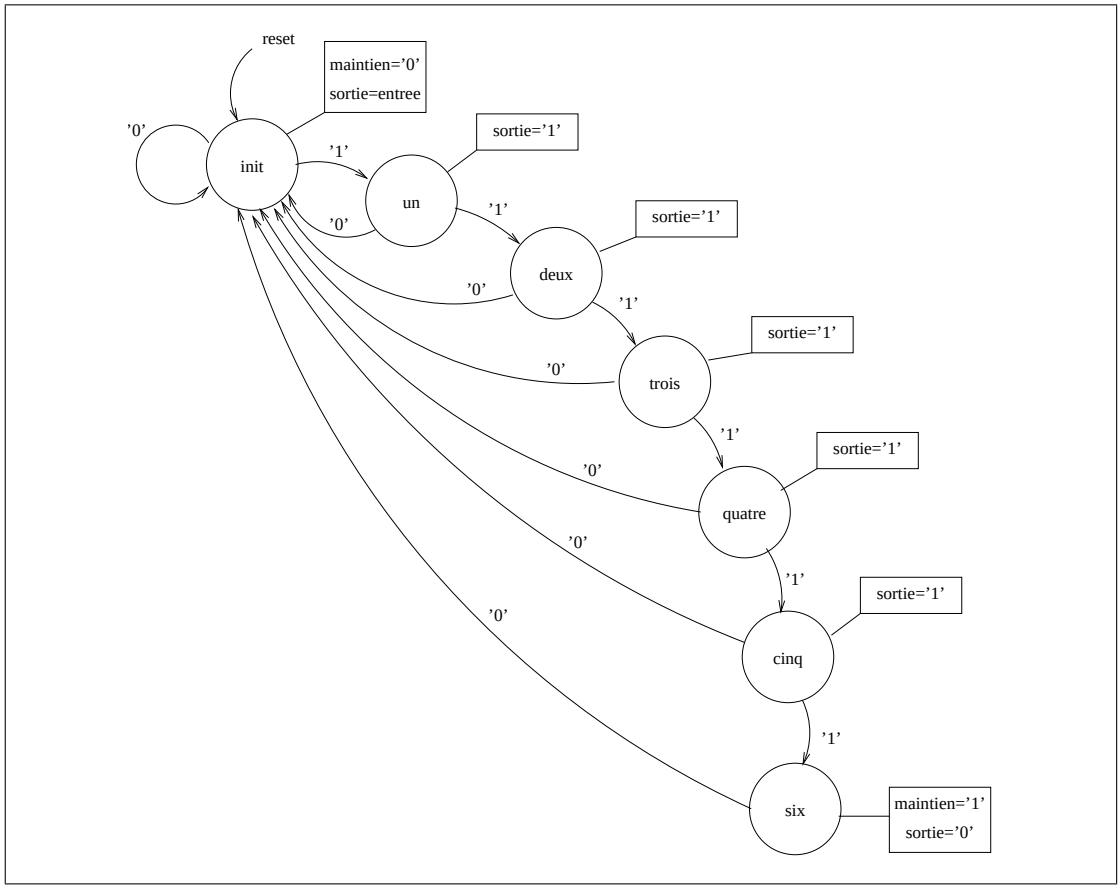
(a) (15 points) Dessinez le graphe détaillé d'une machine à états réalisant l'encodeur NRZI.



(b) (10 points) Écrivez une description VHDL (entité et architecture) correspondante en séparant la partie synchrone et la partie combinatoire en des processus distincts.

```
entity nrzi
   port( clk, reset: in std_logic;
         message: in std_logic;
         sortie: out std_logic));
end entity nrzi;

architecture comport of nrzi is
   type type_etat is (initzero, un);
   signal etat_courant, nouvel_etat: type_etat;
begin
   P1: process( clk, reset )
   begin
      if reset='1' then
         etat_courant <= init;
         nouvel_etat <= init;
      elsif clk='1' and clk'event then
         etat_courant <= nouvel_etat;
      end if;
   end process P1;

   P2: process( message ) is
   begin
      case etat_courant is
         when initzero =>
            sortie <= '0';
            if message='0' then
               nouvel_etat <=un;
            else
               nouvel_etat <= initzero;
            end if;
         when un =>
            sortie <= '1';
            if message='0' then
               nouvel_etat <=initzero;
            else
               nouvel_etat <= un;
            end if;
      end case;
   end process P2;
end architecture comport;
```

(c) (15 points) Dessinez le graphe détaillé d'une machine à états réalisant le bloc d'intercalage de bits.



Bonne chance et bonne semaine de lecture!