INF3500 : Conception et réalisation de systèmes numériques

Examen intra

Mercredi 20 juin 2012

Durée: 2h.
Pondération: 20%.
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 question durant l'examen. En cas de doute sur le sens d’une question, énoncez clairement toute supposition que vous faites.

Question 1. (4 points)

Considérez le code VHDL suivant décrivant un circuit combinatoire:

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

entity question_1 is
port(
   in_port : in signed(9 downto 0);
   out_port : out signed(9 downto 0)
);
end question_1;

architecture arch of question_1 is


   -- déclaration de constantes
   constant cst_1 : integer := 2 ** 4;
   constant cst_2 : integer := -4;

   -- déclaration de fonctions
   function process_val(val : integer) return signed is
   begin
      return to_signed( cst_1 * val / cst_2 + 24, 10);
   end function process_val;

   -- déclaration des signaux
   signal a : signed(9 downto 0);
   signal b : signed(9 downto 0);

begin

   process( in_port, a, b ) is
   begin

      a <= process_val( to_integer(in_port) );
      b <= process_val( to_integer(a) );
      out_port <= a + b;

   end process;

end arch;
```

a) Quelle est la valeur prise par le signal `out_port` lorsque `in_port` vaut `"0000010000"`.

b) Ce code VHDL est-il synthétisable ? Justifiez votre réponse.

Réponse:

a) La réponse à la question est obtenue en utilisant le banc d'essai suivant. L'entrée à considérer correspond à une valeur de +16. La fonction `process_val` donne `a` = (16×16)/(-4)+24 = -40. Dès lors, on obtient pour `b` = ((-40×16)/(-4)+24) = 184. La sortie `out_port` vaut par conséquent -40 + 184 = 144, soit `"0010010000"`.

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

```
entity question_1_tb is
end question_1_tb;

architecture arch of question_1_tb is

   component question_1 is
   port(
      in_port : in signed(9 downto 0);
      out_port : out signed(9 downto 0)
   );
   end component;

   signal in_port : signed(9 downto 0);
   signal out_port : signed(9 downto 0);

begin

   u: question_1
   port map ( in_port => in_port, out_port => out_port );

   process
   begin

      in_port <= "0000010000";
      wait for 10 ns;
      assert false report integer'image( to_integer(out_port) ) & LF severity note;
      assert false report "Simulation ended" & LF severity failure;

   end process;

end arch;
```

b) Ce code VHDL et synthétisable puisque, pour chaque entrée valide, il est possible de trouver la réponse en sortie du circuit en employant des opérateurs arithmétiques synthétisables (+,-,* et / par une puissance de 2).

Question 2. (4 points)

En justifiant clairement mais brièvement vos réponses, indiquez pour chacun des énoncés VHDL suivants si l'opération est synthétisable ou non. On supposera dans tous les cas que les signaux `sig_1`, `sig_2` et `sig_3` sont de type `signed` et qu'ils sont correctement dimensionnés.

a) `sig_3 <= sig_1 - sig_2;`

b) `sig_3 <= ( sig_1 + sig_2 ) / sig_3;`

c) `sig_3 <= sig_1 / ( +16 ) + sig_2 / ( -8 );`

d) `sig_3 <= 16 mod sig_1;`

e) `sig_3 <= sig_1 * sig_2;`

f) `sig_3 <= sig_2 rem 4;`

g) `sig_3 <= 10*sig_1 + 5*sig_2;`

h) `sig_3 <= 6*(sig_1 - sig_2/16);`

i) `sig_3 <= 1.0*sig_2 + 4.0*sig_1;`

j) `sig_3 <= shift_left(sig_2, 10);`

k) `sig_3 <= shift_right(sig_1, to_integer(sig_2));`

l) `sig_3 <= sig_1 ** to_integer(sig_2);`

m) `sig_3 <= sig_1 ** 2;`

n) `sig_3 <= 16*(sig_1 - sig_2/6);`

o) `sig_3 <= to_signed( integer( sin( 3.14 ) * 2.0**8 ), 8 );`

p) `sig_3 <= to_signed( integer( sin( real( to_integer( sig_1 ) ) ) * 2.0**8 ), 8 );`

Réponse :

a) `sig_3 <= sig_1 - sig_2;`

Addition de deux signaux de type signed. Synthétisable.

b) `sig_3 <= ( sig_1 + sig_2 ) / sig_3;`

Division par un signal de type signed. Non synthétisable.

c) `sig_3 <= sig_1 / ( +16 ) + sig_2 / ( -8 );`

Addition de deux signaux de type signed divisés par des entiers, puissances de 2. Synthétisable.

d) `sig_3 <= 16 mod sig_1;`

Opération modulaire avec à sa gauche un signal de type signed. Non synthétisable.

e) `sig_3 <= sig_1 * sig_2;`

Produit de deux signaux de type signed. Synthétisable.

f) `sig_3 <= sig_2 rem 4;`

Opération modulaire avec à sa gauche un entier, puissance de 2. Synthétisable.

g) `sig_3 <= 10*sig_1 + 5*sig_2;`

Addition de deux signaux de type signed multipliés par des entiers. Synthétisable.

h) `sig_3 <= 6*(sig_1 - sig_2/16);`

Produit d'un entier par un signal de type signed, obtenu en faisant la soustraction de deux signaux de types signed, dont un est divisé par une puissance de 2. Synthétisable.

i) `sig_3 <= 1.0*sig_2 + 4.0*sig_1;`

Addition de deux signaux de type signed multipliés par des réels. Non synthétisable.

j) `sig_3 <= shift_left(sig_2, 10);`

Décalage à gauche constant sur un signal de type signed. Synthétisable.

k) `sig_3 <= shift_right(sig_1, to_integer(sig_2));`

Décalage à gauche variable sur un signal de type signed. Synthétisable.

l) `sig_3 <= sig_1 ** to_integer(sig_2);`

Puissance par un signal de type signed sur un signal de type signed. Non synthétisable.

m) `sig_3 <= sig_1 ** 2;`

Puissance de 2 sur un signal de type signed. Non synthétisable.

n) `sig_3 <= 16*(sig_1 - sig_2/6);`

Produit d'un entier par un signal de type signed, obtenu en faisant la soustraction de deux signaux de types signed, dont un est divisé par un entier qui n'est pas une puissance de 2. Non synthétisable.

o) `sig_3 <= to_signed( integer( sin( 3.14 ) * 2.0**8 ), 8 );`

Assignation d'une valeur constante à un signal de type signed. Synthétisable.

p) `sig_3 <= to_signed( integer( sin( real( to_integer( sig_1 ) ) ) * 2.0**8 ), 8 );`

Sinus sur un signal de type signed. Non synthétisable.

Question 3. (4 points)

Considérez le code VHDL suivant décrivant un circuit séquentiel:

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

entity question_3 is
port (
   reset : in std_logic;
   clk : in std_logic;
   i : in std_logic;
   o : out std_logic
);
end question_3;

architecture arch of question_3 is
   type states is (x, y, z, w);
   signal etat : states := x;
begin
   process(clk, reset) is
   begin
      if reset = '1' then
         etat <= x;
         o <= '0';
      elsif rising_edge( clk ) then
         case etat is
            when x =>
               if ( i = '1' ) then
                  etat <= y;
               else
                  etat <= z;
               end if;
               o <= '1';
            when y =>
               etat <= w;
               o <= '0';
            when z =>
               etat <= x;
               o <= '0';
            when others => -- w
               if ( i = '1' ) then
                  etat <= z;
               else
                  etat <= y;
               end if;
               o <= '1';
         end case;
      end if;
   end process;
end arch;
```

a) S'agit-il d'une machine de Moore ou d'une machine de Mealy ? Justifiez.

b) Dessinez le diagramme d'états de la machine à états.

c) Partant de l'état `x`, quel sera la sortie `o` de la machine après 4 fronts montant de l'horloge si on maintient l'entrée `i` à `'1'`. Quel sera l'état à ce moment là?

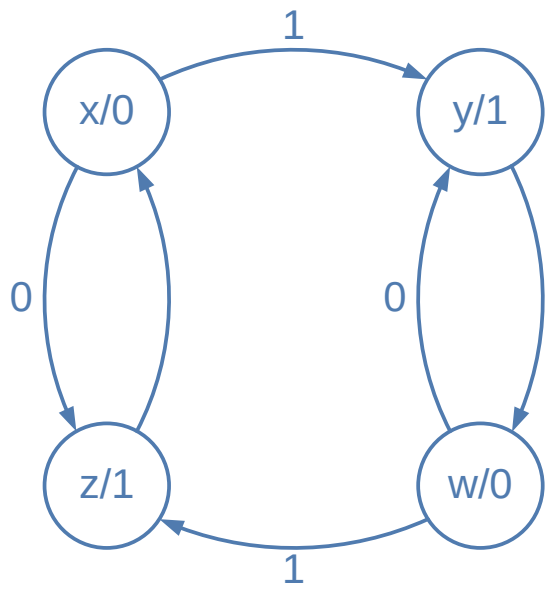
d) En admettant l'encodage suivant `x = '00'`, `y = '01'`, `z = '11'`, `w = '10'`, donnez l'expression booléenne de la sortie `o`. Notez les bits d'états q_1q_0.

Réponse :

a) S'agit-il d'une machine de Moore ou d'une machine de Mealy ? Justifiez.

La sortie dépend de l'état. Il s'agit donc d'une machine de Moore.

b) Dessinez le diagramme d'états de la machine à états.



c) Partant de l'état `x`, quel sera la sortie `o` de la machine après 4 fronts montant de l'horloge si on maintient l'entrée `i` à `'1'`. Quel sera l'état à ce moment là?

La sortie sera 0. L'état sera x.

d) En admettant l'encodage suivant `x = '00'`, `y = '01'`, `z = '11'`, `w = '10'`, donnez l'expression booléenne de la sortie `o`. Notez les bits d'états q_1q_0.

$o = q_0$.

Question 4. (2 points)

En utilisant des registres, des multiplexeurs, décodeurs et autres circuits combinatoires vus en classe, dessinez les chemins de données permettant de réaliser chacune des micro-opérations suivantes, où R_0, R_1 et R_2 sont des registres et K_1, K_2 sont des signaux de contrôle :

a) K_1: $R_0 \leftarrow R_1+R_0$

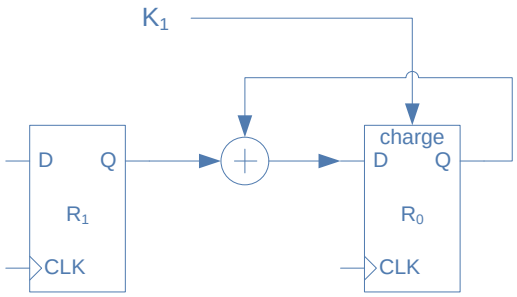
b) K_2K_1: $R_2 \leftarrow R_1-R_0$, $K_2'K_1$: $R_2 \leftarrow R_1+R_0/2$

c) K_2': $R_2 \leftarrow R_0$, K_2: $R_2 \leftarrow R_1$

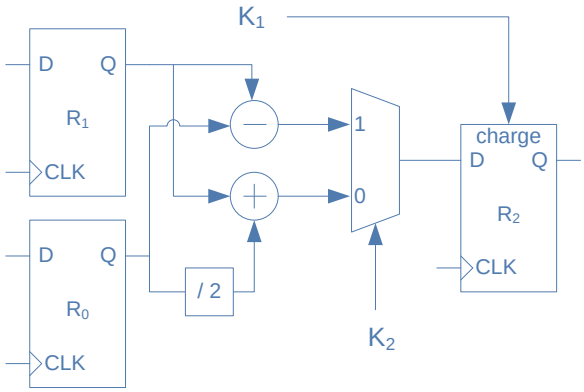
d) $R_2 \leftarrow 4R_1+R_0/2$

Réponse:

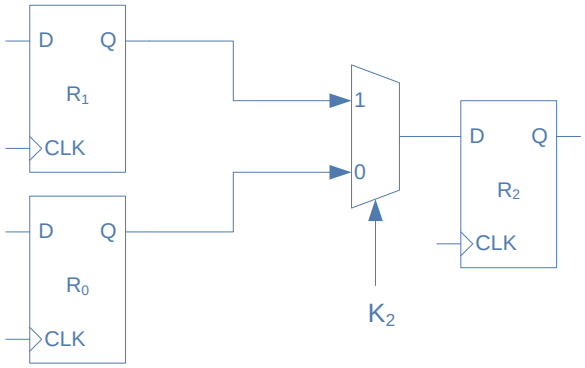
a) K_1: $R_0 \leftarrow R_1+R_0$



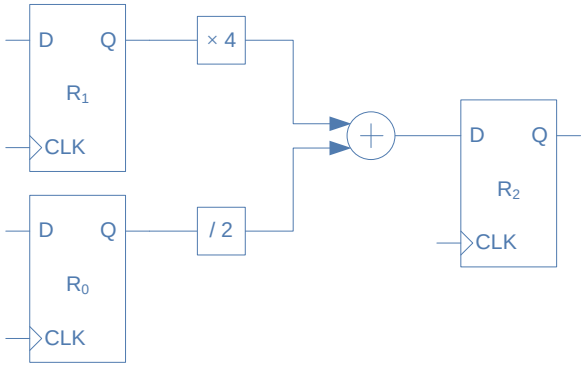
b) K_2K_1: $R_2 \leftarrow R_1-R_0$, $K_2'K_1$: $R_2 \leftarrow R_1+R_0/2$



c) K_2': $R_2 \leftarrow R_0$, K_2: $R_2 \leftarrow R_1$



d) $R_2 \leftarrow 4R_1+R_0/2$



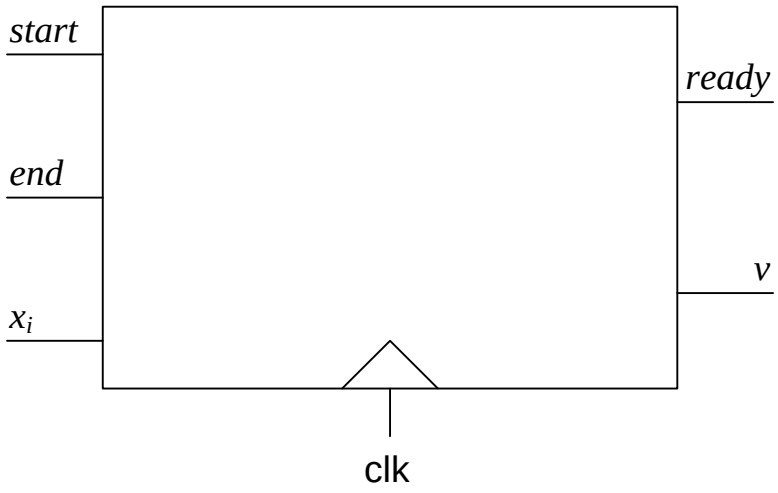
Question 5. (6 points)

On désire concevoir un circuit séquentiel qui prend N entrées binaires x_1, x_2, x_3, ..., x_N et qui calcule la fonction logique $v = x_N (x_1 \oplus x_2 \oplus x_3 ... \oplus x_{N-1})$, qu'il convient de lire comme v est la conjonction (ET logique) de x_N et du résultat de l'opération XOR sur les entrées x_1 à x_{N-1}.

Les entrées x_i arrivent de manière sérielle. Au début de la séquence, qui coïncide avec l'arrivée de x_1, le signal d'entrée *start* prend la valeur 1. À la fin de la séquence, qui coïncide avec l'arrivée de l'entrée x_N, le signal *end* prend la valeur 1. Les signaux *start* et *end* prennent la valeur 1 durant un cycle d'horloge seulement durant une séquence d'entrée x_i. Vous êtes assurés d'avoir au moins 3 éléments durant une séquence d'entrée ($N \geq 3$).

Le résultat de l'opération v doit apparaître à la sortie en même temps que l'on met la sortie *ready* à 1. Le signal *ready* demeure à 1 tant qu'une nouvelle séquence de traitement n'a pas débuté. Il vaut 0 quand le circuit traite une nouvelle séquence.

Le schéma suivant devrait vous aider à saisir les ports d'entrées/sorties de l'entité que nous considérons.

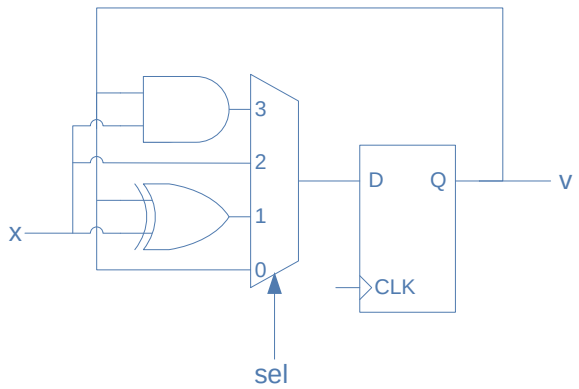


a) Proposez un chemin de données permettant de réaliser ce circuit.

b) Proposez le diagramme d'états de la machine à états qui permet de contrôler ce circuit.

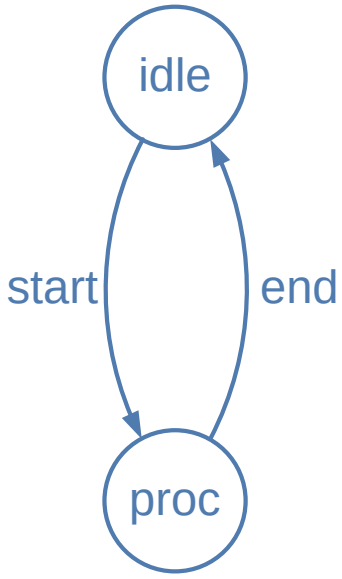
Réponse :

a) Proposez un chemin de données permettant de réaliser ce circuit.



b) Proposez le diagramme d'états de la machine à états qui permet de contrôler ce circuit.

On utilisera une machine de Mealy à deux états et à deux sorties. On suppose impossible d'avoir en entrée (start, end) = (1, 1).



	sel end start 00	01	10	11
idle	0	2	0	-
proc	1	1	3	-

	ready end start 00	01	10	11
idle	1	1	1	-
proc	0	0	0	-