INF3500 : Conception et réalisation de systèmes numériques

Examen intra

17 Juin 2009

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.
Ne posez pas de questions. En cas de doute sur le sens d'une question, énoncez clairement toute supposition que vous faites.

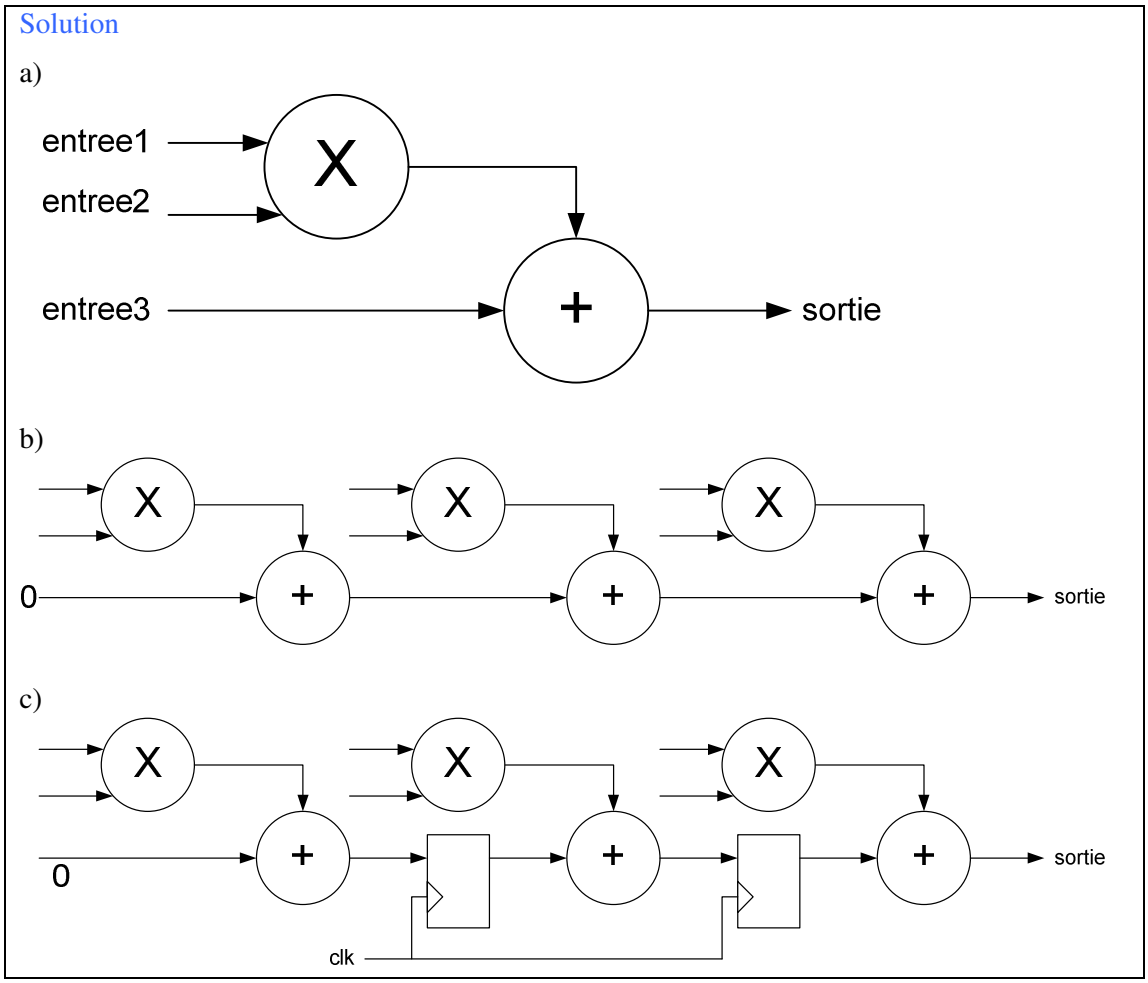
Question 1. (5 points)-exercice sur la synthèse

Soit le code VHDL suivant.

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


entity MAC is
  generic (
       W : positive := 4
    );

  port (
    clk : in std_logic;
    reset : in std_logic;
    entree1 : in signed(W-1 downto 0);
    entree2 : in signed(W-1 downto 0);
    entree3 : in signed(W-1 downto 0);
    sortie : out signed(2*W downto 0)
    );
end MAC;

architecture beh of MAC is
begin  -- beh
       sortie <= (entree1 * entree2) + entree3;
end beh;
```

a - Donnez le diagramme du circuit MAC correspondant au code VHDL ci-avant. Indiquez la largeur en bits des ports et des signaux intermédiaires en sachant que W est égal à 4. Vous pouvez utiliser des opérations logiques et arithmétiques, la comparaison, des multiplexeurs, décodeurs et encodeurs, éléments mémoire, etc.
b – Donnez le diagramme du circuit itératif combinatoire à 3 étages qui cascade 3 MAC en sachant que le port *entree3* du premier étage de MAC prendra la valeur zéro. En sachant que le port *entree3* du prochain étage prendra la valeur du port *sortie* du module de l'étage précédent.
c – Donnez le diagramme du circuit itératif séquentiel de 3 étages qui cascade 3 MAC. En sachant que le port *entree3* du premier étage de MAC sera prendra la valeur zéro.

Solution



Question 2. (6 points)

Soit un circuit d'un gestionnaire de réservoir d'essence de voiture *niveau_réservoir*. Deux jeux de senseurs sont positionnés aux deux niveaux du réservoir comme illustré par la Figure 1. Ces senseurs sont placés dans le réservoir. Ils donnent une valeur logique quand ils baignent dans l'essence, et une valeur 0 autrement. Ces signaux permettent de calculer le niveau d'essence du réservoir. Le niveau d'essence est calculé comme suit :

- Si les senseurs du niveau_H sont égaux et leurs valeurs sont différentes de zéro alors le niveau de sortie *niv_out* prendra la valeur de *niv_S2_A*.
- Dans le cas contraire, le niveau de sortie *niv_out* sera égal à la valeur de *niv_S1_A* si les valeurs de *niv_S1_A* et *niv_S1_B* sont égales, sinon *niv_out* prendra la valeur zéro.

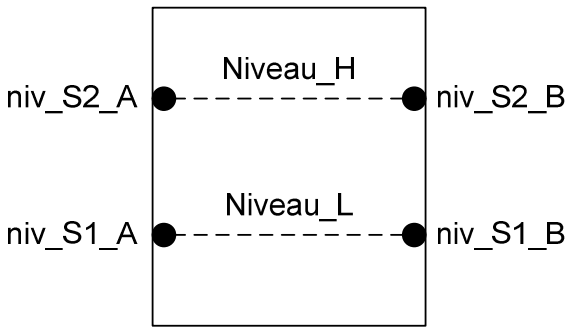


Figure1. Exemple du positionnement des senseurs dans le réservoir.

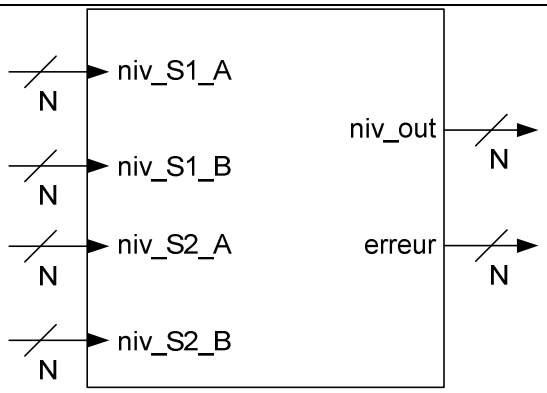


Figure 2. Interfaces du réservoir

Les ports de l'entité *niveau_réservoir* sont décrits comme suit :

- Les entrées *niv_S2_A* et *niv_S2_B* donnent la valeur du niveau supérieur.
- Les entrées *niv_S1_A* et *niv_S1_B* donnent la valeur du niveau inférieur.
- Le port de sortie *niv_out* aura la valeur *niv_S2_A* si le niveau supérieur est actif à savoir que *niv_S2_A* et *niv_S2_B* sont égaux et que leur valeur est différente de zéro. *Niv_out* prendra la valeur de *niv_S1_A* dans le cas contraire et seulement si *niv_S1_A* et *niv_S1_B* sont égaux sinon niv_out aura la valeur zéro.
- Le port de sortie *erreur* sera activé si *niv_S2_A* et *niv_S2_B* ont des valeurs différentes OU si *niv_S1_A* et *niv_S1_B* ont des valeurs différentes.

Le port erreur est un **std_logic** alors que les ports, *niv_s2_B*, *niv_S1_A*, *niv_S1_A* et *niv_out* sont des **vecteurs de N bits non-signés** (unsigned).

Donnez l'entité et l'architecture synthétisable de niveau_réservoir.

Solution

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

entity niveau_reservoir is

  generic (
    N : positive);

  port (
    niv_S1_A : in unsigned(N-1 downto 0);
    niv_S1_B : in unsigned(N-1 downto 0);
```

```
    niv_S2_A : in unsigned(N-1 downto 0);
    niv_S2_B : in unsigned(N-1 downto 0);
    niv_out : out  unsigned(N-1 downto 0);
    erreur : out std_logic
    );

end niveau_reservoir;


architecture beh of niveau_reservoir is

begin  -- beh

  process (niv_S1_A, niv_S1_B, niv_S2_A, niv_S2_B)
  begin  -- process
    if niv_S2_A /= to_unsigned(0, N) and niv_S2_A = niv_S2_B  then
      niv_out <= niv_S2_A ;
    elsif  niv_S1_A = niv_S1_B  then
      niv_out <= niv_S1_A ;
    else
      niv_out <= (others => '0');
    end if;

    erreur <= '0';
    if niv_S2_A /= niv_S2_B or niv_S1_A /= niv_S1_B then
      erreur <= '1';
    end if;

  end process;

end beh;
```

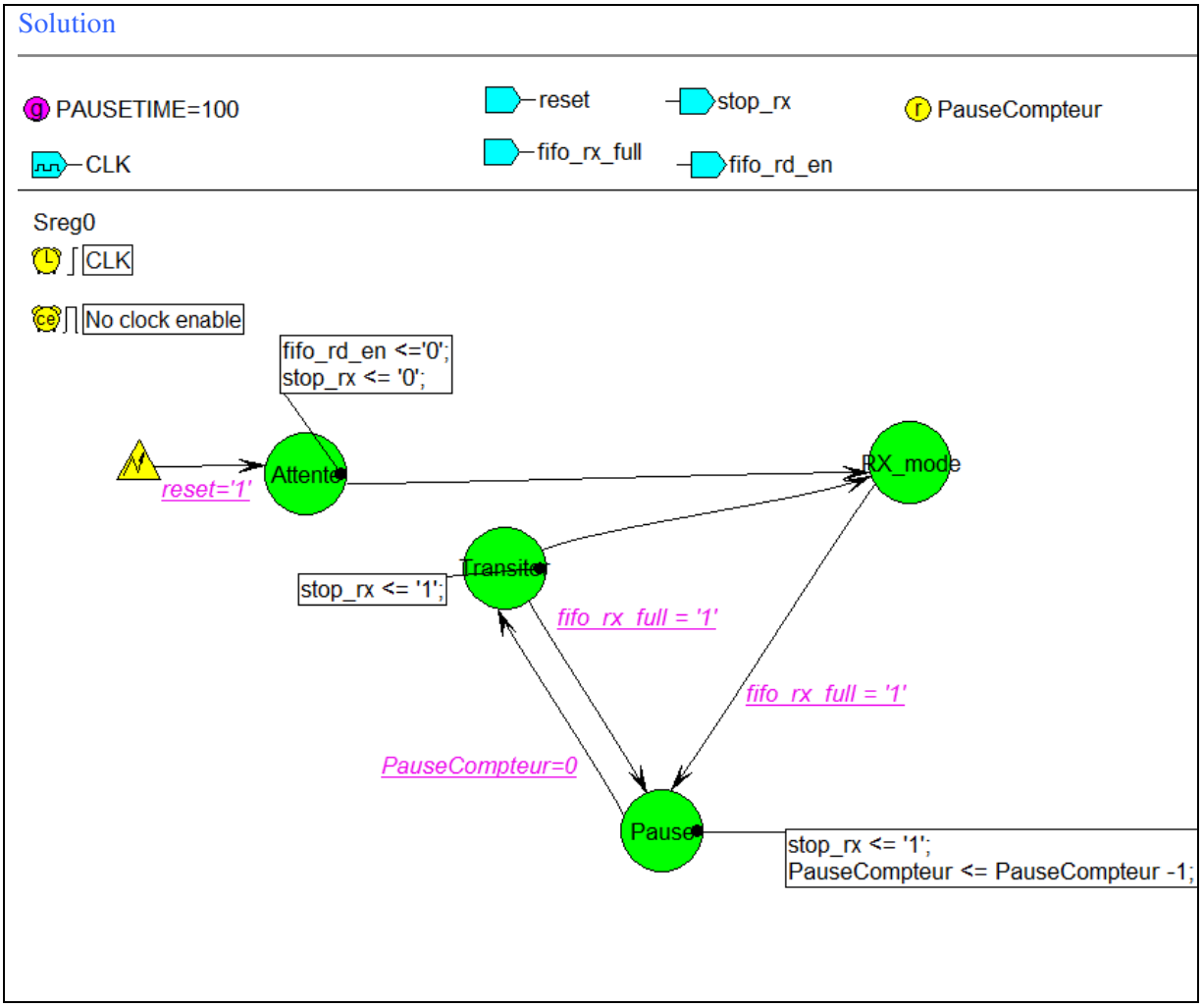
Question 3. (3 points)

Donnez le diagramme d'états correspondant au code VHDL suivant. Identifiez clairement les états, les transitions entre états et les conditions correspondantes, et la valeur des signaux de sortie.

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

entity exo4 is
    generic (
        PAUSETIME: INTEGER := 100 );
    port (
        CLK: in STD_LOGIC;
        fifo_rx_full: in STD_LOGIC;
        reset: in STD_LOGIC;
        fifo_rd_en: out STD_LOGIC;
        stop_rx: out STD_LOGIC);
end exo4;

architecture fsm of exo4 is
```

```
-- diagram signals declarations
signal PauseCompteur: INTEGER range PAUSETIME-1 downto 0;

-- SYMBOLIC ENCODED state machine: Sreg0
type Sreg0_type is (
    Attente, RX_mode, Pause, Transiter
);
-- attribute enum_encoding of Sreg0_type: type is ... -- enum_encoding attrib-
ute is not supported for symbolic encoding

signal Sreg0: Sreg0_type;

begin


-------------------------------------------------------------------
-- Machine: Sreg0
-------------------------------------------------------------------
Sreg0_machine: process (CLK, reset)
begin
   if reset='1' then
      Sreg0 <= Attente;
      -- Set default values for outputs, signals and variables
      PauseCompteur <= PAUSETIME;
   elsif CLK'event and CLK = '1' then
      -- Set default values for outputs, signals and variables
      PauseCompteur <= PAUSETIME;
      case Sreg0 is
         when Attente =>               -- transition d'un cycle apres le re-
set
            Sreg0 <= RX_mode;
         when RX_mode =>               -- mode de reception normal
            -- -- rentrer en mode pause lorsque la fifo de reception est
pleine
            if fifo_rx_full = '1' then
               Sreg0 <= Pause;
            end if;
         when Pause =>                 -- attendre la fin de la pause
            PauseCompteur <= PauseCompteur -1;
            if PauseCompteur=0 then
               Sreg0 <= Transiter;
            end if;
         when Transiter =>                     -- -- si la fifode reception
est toujours pleine, lancer une autre pause
            if fifo_rx_full = '1' then
               Sreg0 <= Pause;
            else
               Sreg0 <= RX_mode;
            end if;
--vhdl_cover_off
         when others =>
            null;
--vhdl_cover_on
      end case;
   end if;
end process;

-- signal assignment statements for combinatorial outputs
fifo_rd_en_assignment:
fifo_rd_en <= '0' when (Sreg0 = Attente) else
              '1';
```

```
stop_rx_assignment:
stop_rx <= '0' when (Sreg0 = Attente) else
           '1' when (Sreg0 = Pause) else
           '1' when (Sreg0 = Transiter) else
           '0';
end fsm;
```



Question 4. (4 points)

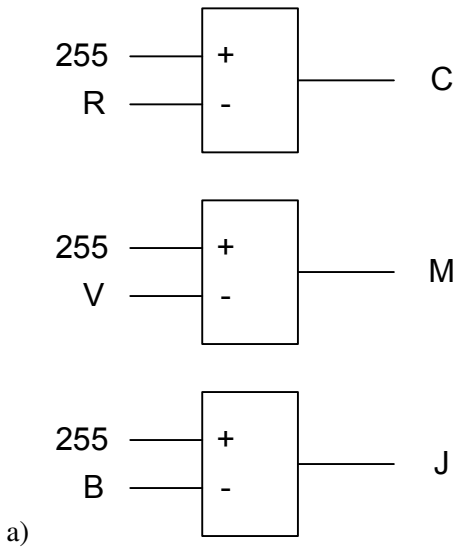
Une imprimante couleur reçoit, pour chaque point de la feuille à imprimer, un triplet (R, V, B) pour l'intensité de couleur en rouge, vert et bleu, respectivement. Les intensités sont exprimées en nombres non signés sur 8 bits, de 0 à 255. L'imprimante utilise trois cartouches: une cyan, une magenta et une jaune. Les équations de conversion de RVB à CMJ sont données par C = 255- R; M = 255-V; Y = 255- B.

a. Donnez le diagramme d'un circuit numérique combinatoire qui convertit des coordonnées RVB à CMJ. Vous pouvez utiliser des opérations logiques et arithmétiques, la comparaison, des multiplexeurs, décodeurs et encodeurs, éléments mémoire, etc.

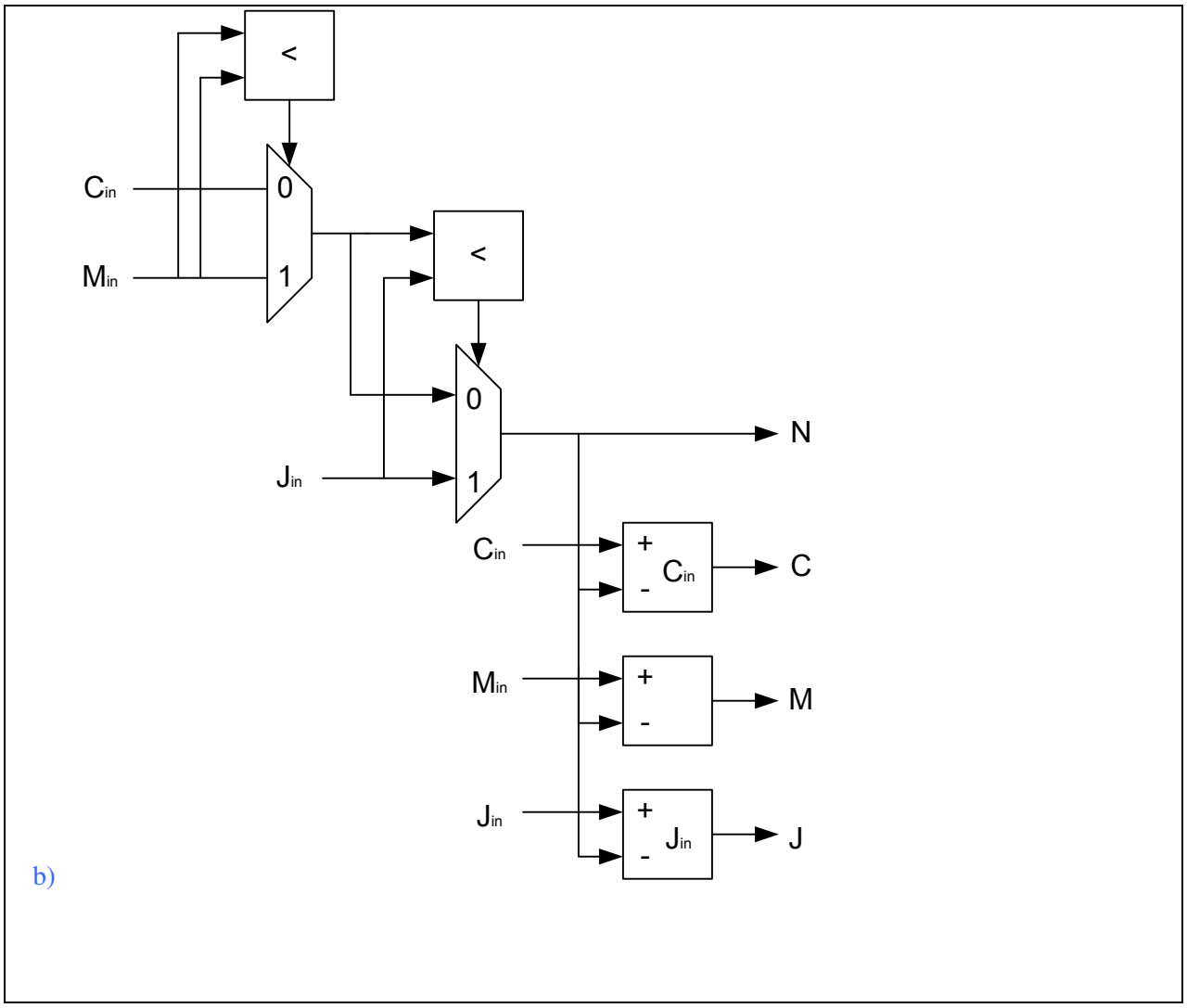
Pour économiser l'encre de couleur, les imprimantes utilisent habituellement une quatrième cartouche, noire celle-là. On enlève des coordonnées CMJ la valeur la plus petite d'entre elles, et cette valeur donne l'intensité du noir à utiliser. Par exemple, des valeurs RVB de (5, 55, 55) correspondent à des valeurs CMJ de (250, 200, 200). Ces coordonnées correspondent à (200, 200, 200) + (50, 0, 0). L'imprimante peut donc utiliser une intensité de 200 pour le noir et 50 pour le cyan, sans utiliser ni le magenta ni le jaune.

b. Donnez le diagramme d'un circuit numérique combinatoire qui convertit des coordonnées CMJ à NCMJ (où N correspond au noir). Vous pouvez utiliser des opérations logiques et arithmétiques, la comparaison, des multiplexeurs, décodeurs et encodeurs, éléments mémoire, etc.

Solution



a)



Question 5. (2 points)

Soit l'entité module1 ainsi que le banc d'essai module1_tb permettant de le vérifier. Les codes VHDL de ces modules sont comme suit:

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

entity circuit1 is
  port (
    clk : in std_logic;
    x1 : in std_logic;
    x2 : in std_logic;
    x3 : in std_logic;
    y1 : out std_logic;
    y2 : out std_logic
    );
end circuit1;

architecture beh of circuit1 is
  signal p1 : std_logic;
```

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


entity circuit1_tb is

end circuit1_tb;

architecture beh of circuit1_tb is
  component circuit1
    port (
      clk : in std_logic;
      x1 : in std_logic;
      x2 : in std_logic;
      x3 : in std_logic;
```

```
  signal p2 : std_logic;
  signal p3 : std_logic;
begin  -- beh

  p1 <= x1 and x2 and not x3;
  p2 <= not x1 and x2 and x3;
  p3 <= not x1 and not x2;

  process (clk)
  begin  -- process
    if clk'event and clk = '1' then
      y1 <= p1 or p2;
    end if;
  end process;

  y2 <= p3;
end beh;
```

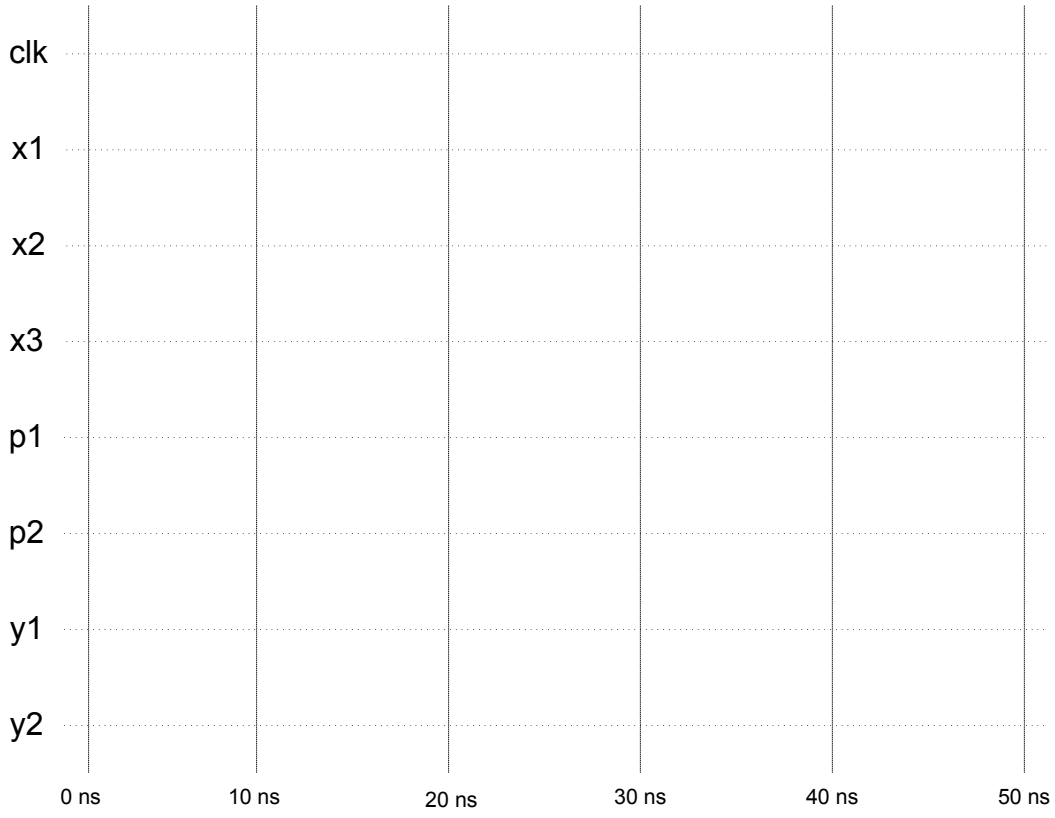
```
      y1 : out std_logic;
      y2 : out std_logic
      );
  end component;

  signal vecteur_test : std_logic_vector(2
downto 0);
  signal y1 : std_logic;
  signal y2 : std_logic;

begin  -- beh
  UUT : circuit1
    PORT MAP(vecteur_test(0), vecteur_test(2),
vecteur_test(1), y1, y2);

  process
  begin  -- process
    for i in 3 to 7 loop
      vecteur_test <=
std_logic_vector(to_unsigned(i, vec-
teur_test'length));
      wait for 10 ns;
      assert (y1='1' and y1 /= y2)
        report "Erreur y1 and y2 ne peuvent pas
actif en meme temps" severity error;
    end loop;
    report "Fin simulation" severity failure;
  end process;

end beh;
```

a- Indiquez quelle technique de vérification a été appliquée.
b- Donnez le chronogramme des ports x1, x2, x3, y1 et y2 ainsi que ceux des signaux internes p1, p2 et p3 durant la simulation.
c - Précisez tous les textes qui pourraient être affichés durant la simulation.
d – Indiquez à quel temps *t* la simulation se terminera.

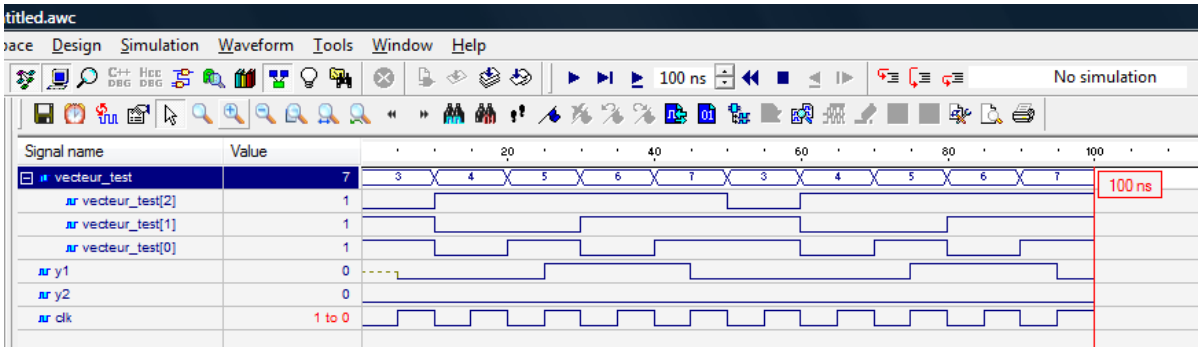
Remplissez le gabarit de la page suivante. N'oubliez pas de remettre la page dans votre cahier d'examen.

Nom :

Matricule :



Solution


titled.awc
Design
Simulation
Waveform
Tools
Window
Help
100 ns
No simulation
Signal name
Value
vecteur_test
7
vecteur_test[2]
1
vecteur_test[1]
1
vecteur_test[0]
1
y1
0
y2
0
clk
1 to 0
20
40
60
80
100
100 ns