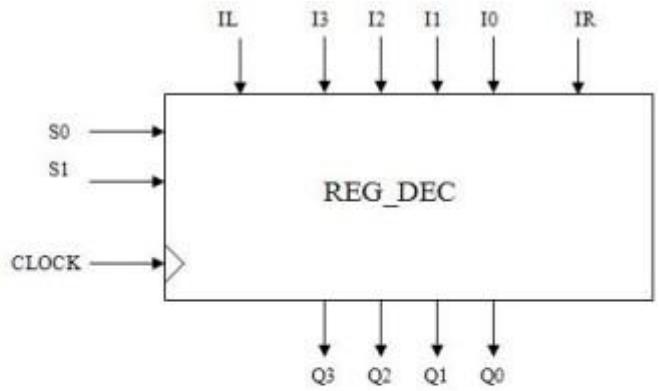
Examen VHDL 2eme session 2008

Exercice 1 : (7 points)

Le circuit d'un registre à décalage synchrone (front montant) est donné sur la figure suivante :



- 4 entrées/4 sorties
- 1 entrée horloge
- 2 entrées de sélection qui permettent de choisir le mode :

S0	S1	Fonction réalisée après le top horloge	Opération
0	0	Q[3..0] <=Q[3..0]	Pas de changement
0	1	Q[3..0] <=I[3..0]	Chargement parallèle
1	0	Q[3..1]<=Q[2..0], Q[0]<=IR	Décalage à gauche avec IR
1	1	Q[3]<=IL, Q[2..0]<=Q[3..1]	Décalage à droite avec IL

1) Ecrire l'entité qui décrit la vue externe de ce registre.
2) Donner une architecture comportementale (tableau ci-dessus) en utilisant un process. Dans le process, vous pouvez utiliser l'instruction if…. Then…..elsif et l'instruction for…..loop.

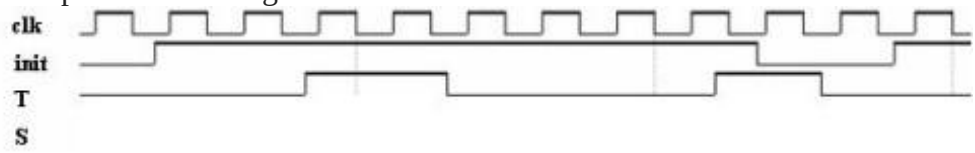
Exercice 2 : (7 points)

On considère le programme VHDL suivant qui décrit le fonctionnement d'une bascule :

```
entity basc is
Port ( T, clk, init : in bit;
                S: out bit);
end basc;

architecture primitive of basc is
signal etat : bit;

begin
S <= etat;
```

```
process (clk)
begin
if (clk'event and clk='1') then
    if ( init ='0') then
        etat <= '1';
     elsif (T='0') then
         etat <= not etat;
      end if;
end if;
end process;
end primitive;
```

1. A quoi reconnaît-on qu'il s'agit d'un circuit synchrone ?
2. La commande « init » est elle synchrone ou asynchrone ? (justifier)
3. Compléter le chronogramme ci-dessous.



4. Modifier le programme précédent pour qu'il rajoute à la bascule une commande raz, de remise à zèro, asynchrone.
5.Transformer le programme pour qu'il utilise le type st-logic. Rajouter une commande oe de contrôle qui fasse passer la sortie en haute impédence si oe='0'.

Exercice 3 :(6 points)

Il s'agit de réaliser un signal d'horloge (CLOCK) à un fréquence de 1Hz à partir d'un quartz (OSC) qui génère un signal d'horloge à 10 MHz, qu'il va donc falloir diviser par 10.10^6afin d'obtenir la fréquence voulue. Cette opération peut se faire en 2 étapes tout d'abord avec un diviseur par 5.10^6 puis un diviseur par 2.

1. Donner la description en langage VHDL (Entity et Architecture) d'un diviseur par 5.10^6 à partir de 3 compteurs 8 bits. Utiliser pour cela 3 variables (COUNT_0, COUNT_1, COUNT_2) de type integer, représentant chacune la valeur d'un compteur.

2. Donner la description d'un diviseur par 2 en utilisant la bascule D avec remise à zèro asynchrone (RST).

Correction

Exercice 1 :

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

entity register1 is port (
H, IR,IL,S0,S1 : in std_logic;
I: in std_logic_vector (3 downto 0);
Q : buffer std_logic_vector (3 downto 0));
```

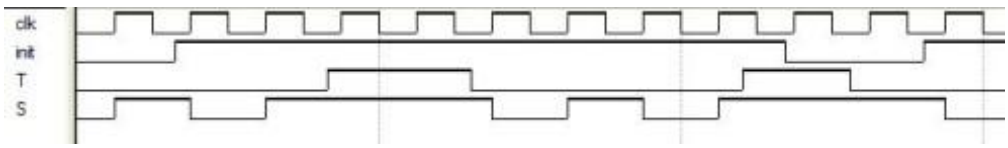
```
end register1;

architecture arch of register1 is
Begin
process (S0, S1, H)
variable Z : integer;
variable N: Integer:=0;
begin
if (H'event and H ='1') then
if (S0='0' and S1= '0') then Q<=Q;
elsif (S0='0' and S1= '1') then Q<=I;
elsif (S0 ='1' and S1='0') then
Boucle1: For Z IN 3 downto 1 Loop
N:=Z-1;
Q(Z)<=Q(N);
End  loop Boucle1;
Q(N)<=IR;
elsif (S0 ='1' and S1='1') then
Boucle2 : For Z IN 0 to 2 Loop
N:=Z+1;
Q(Z)<=Q(N);
End  loop Boucle2;
Q(N)<=IL;
end if;
end if;
end process;
end arch;
```

Exercice 2 :

1. Seul le signal d'horloge clk fait partie de la liste de sensibilité de process. Seul un changement d'état du signal clk va déclencher le process et par conséquent évaluer les instruction de celui-ci.

1. La commande « init » est synchrone de l'horloge clk, et elle n'est pas mentionnés dans la liste de sensiblité du process, par conséquence le process ne sera déclenché que par le signal d'horloge clk. La commande ne fera que si un front montant sur clk aura lieu.

1. chronogramme



4.

```
entity basc is
Port ( T, clk, init,raz : in bit;
          S: out bit);
end basc;
 architecture primitive of basc is
signal etat : bit;
begin
S <= etat;
process (clk,raz)
begin
if (raz='1') then etat<= '0';
elsif (clk'event and clk='1') then
```

```
if ( init ='0') then
 etat <= '1';
 elsif (T='0')then
etat <= not etat;
end if;
end if;
end process;
end primitive;
```

5.

```
Library ieee;
use ieee.std_logic_1164.all;
entity basc is
Port ( T, clk, init,raz,oe : in std_logic;
          S: out std_logic);
end basc;
architecture primitive of basc is
signal etat : std_logic;
begin
S <= etat;
process (clk,raz,oe)
begin
--wait until (clk = '1');
if (oe='0') then etat<= 'Z';
elsif (raz='1') then etat<= '0';
elsif (clk'event and clk='1') then
if ( init ='0') then
 etat <= '1';
 elsif (T='0')then
etat <= not etat;
end if;
end if;
end process;
end primitive;
```

Exercice 3 :

1.

```
Library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity div_freq is port
(OSC : in std_logic;
CLOCK : inout std_logic);
end div_freq;
 architecture arch_div_freq of div_freq is
begin
process (OSC)
variable COUNT_0, COUNT_1, COUNT_2 :  integer range 0 to 250;
begin
if (OSC'event and OSC ='1') then
COUNT_0:= COUNT_0+1;
if COUNT_0=250 then
  COUNT_1:= COUNT_1+1;
   COUNT_0:=0;
   if COUNT_1=250 then
```

```
        COUNT_2:= COUNT_2+1;
        COUNT_1:=0;
        if COUNT_2=80 then
           COUNT_2:=0;
           if CLOCK='1' then CLOCK<='0';
              else CLOCK <='1';
              end if;
          end if;
        end if;
      end if;
    end if;
end process;
end arch_div_freq;
```

2.

```
Library ieee;
use ieee.std_logic_1164.all;
entity div_2 is
Port ( CLOCK : inout std_logic; RST : in std_logic; Q: out std_logic);
end div_2;
 architecture arch_div_2  of div_2  is
signal D,QN : std_logic;
begin
process (CLOCK,RST)
begin
if (RST='1') then Q<= '0'; QN<= '1';
elsif (CLOCK'event and CLOCK ='1') then  Q<=D; QN<=not D;
end if;
end process;
D <= QN;
end arch_div_2  ;
```