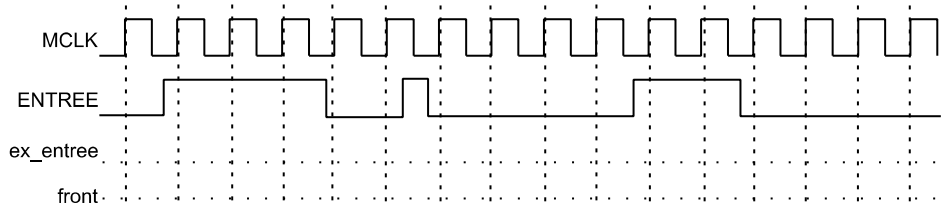
Examen langage V H D L

Durée 1H30

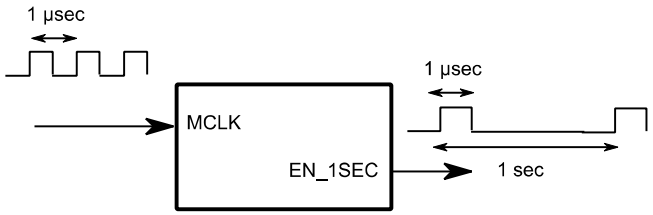
1- a) Quelle est la fonction du programme VHDL ci-dessous.

```
process(MCLK)
begin
      if MCLK'event and MCLK='1' then
            front <= '0' ;
            ex_entree <= ENTREE ;
            if ex_entree = '0' and ENTREE = '1' then
                  front <= '1' ;
            end if ;
      end if ;
end process ;
```

b) Compléter le diagramme de temps pour des signaux « ex_entree » et « front ».

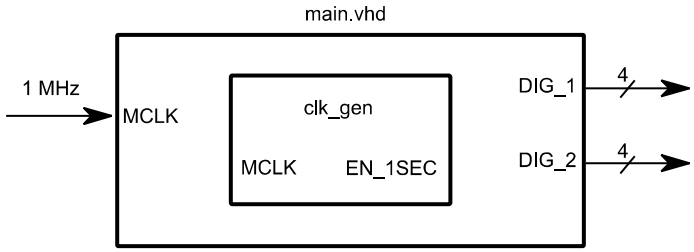


2- Ecrire un programme VHDL pour réaliser le circuit présenté (juste l'architecture).




```
entity clk_gen is
port (MCLK: in std_logic, EN_1SEC : out std_logic);
end clk_gen;
```

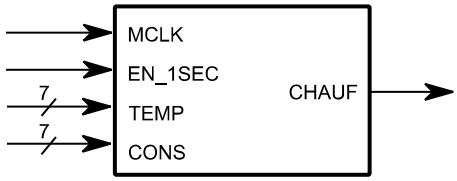
3- On suppose que le circuit de la question 2 a été déjà développé et que vous l'utiliserez en tant que component dans votre programme. Ecrire un programme VHDL pour crée un compteur décimal à deux digits allant de 00 à 99 (la valeur après 99 est 00). Le comptage se fait toutes les secondes.



4- Nous avons un capteur de température qui nous fournit la température ambiante sur 7 bits sous format signé complément à deux (l'intervalle de variation est de -64 à 63 degré). Le but est de contrôler la température ambiante et de la maintenir à une consigne donnée. Le contrôle se fait par un seul bit (tout ou rien) qui sera appliqué à un circuit de chauffage.

L'algorithme à suivre est le suivant :

1- On mesure la température et on la compare avec la consigne.
2- Dans le cas où une différence positive est constatée (consigne plus chaude que la pièce), on envoie '1' en sortie "CHAUF" pendant la durée suivante : la durée en minute = 4 fois le montant de la différence de température. Par exemple si $T_{consigne}$=25° et $T_{ambiante}$=18°, on envoie '1' en sortie pendant (25-18)*4=28 minutes.
3- A la fin de cette durée, on se met dans l'état attente pendant 15 minutes où la sortie reste à '0' et ensuite on repart à l'état initial où une nouvelle lecture de température sera faite.



a) Ecrire l'entité.
b) Avant de programmer, dessiner la machine à état de votre système.
c) Donner le programme VHDL correspondant au circuit.

Aide Syntaxe

Déclarations

```
constant <name>: <type> := <value>;
type <type_name> is (<string1>, <string2>, ...);
```

```
entity nom_de_l_entite is
            {generic(liste_des_parametres)}
            {port(liste_des_port_avec_leutr_mode)}
      end {nom_de_l_entite}
```

Déclaration de component

```
component DEMI_ADD
    port(A,B  : in std_logic;
         SUM,C: out std_logic);
end component;
```

Instantiation du component :

```
DEMI_ADD_INST: DEMI_ADD port map
    (A => SIG_A, B => SIG_B, SUM => SOMME,
    C => RETENUE);
```

Commandes séquentielles

```
if <condition> then
      <statement>
elsif <condition> then
      <statement>
else
      <statement>
end if;

for <name> in <lower_limit> to <upper_limit> loop
   <statement>;
   <statement>;
end loop ;

   case (<2-bit select>) is
      when "00" =>
         <statement>;
      when "01" =>
         <statement>;
      when "10" =>
         <statement>;
      when "11" =>
         <statement>;
      when others =>
         <statement>;
   end case
```

Commandes concurrentes

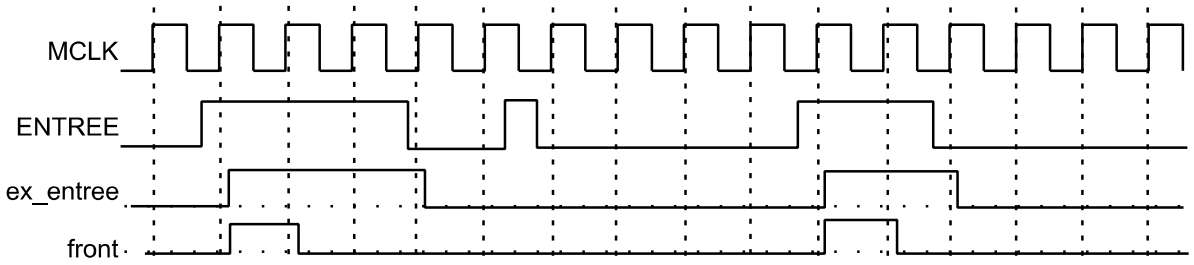
```
with <choice_expression> select
   <name> <= <expression> when <choices>,
             <expression> when <choices>,
             <expression> when others;

<name> <= <expression> when <condition> else
          <expression> when <condition> else
          <expression>;
```

Corrigé

1- A) Ce programme permet de détecter un front montant sur un signal qui n'est pas l'horloge principale. Ainsi, on peut faire exécuter un process quand ce front montant a été détecté. Par exemple le process ci-dessous est

```
process(MCLK)
begin
    if MCLK'event and MCLK='1' then
        if front = '1' then
            le traitement
        end if ;
    end if ;
end process ;
```

B)



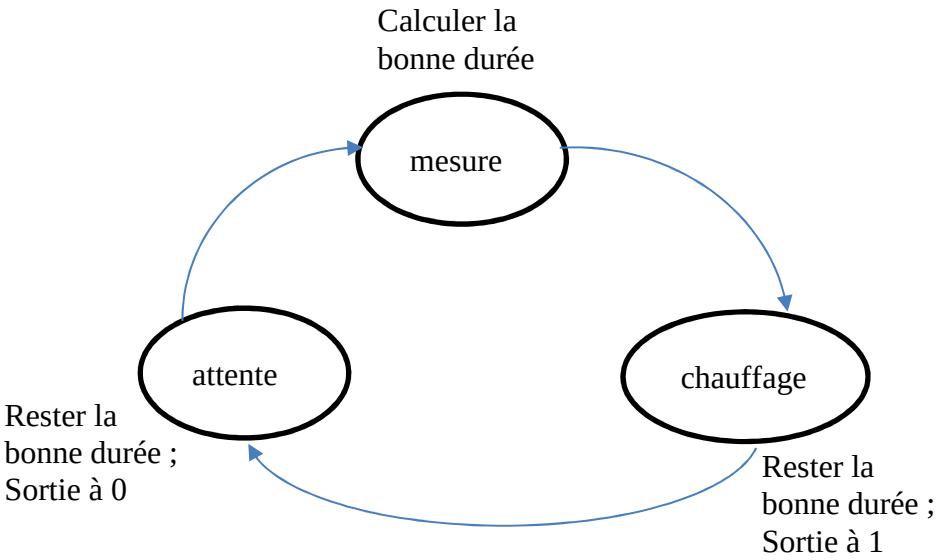
2-

```
architecture FPGA of clk_gen is
    signal cmp : integer range 0 to 1000000 ;
begin
    process(MCLK)
    begin
        if MCLK'event and MCLK='1' then
            cmp <= cmp + 1 ;
            EN_1SEC <= '0' ;
            if cmp = 999999 then
              cmp <= 0 ;
              EN_1SEC <= '1' ;
            end if ;
        end if ;
    end process ;
end FPGA ;
```

3-

```
entity compt_dec is
    port (MCLK : in std_logic ;
          DIG1, DIG2 : out std_logic_vector(3 downto 0) ;
end compt_dec ;

archtecture FPGA of compt_dec is
    component clk_gen
        port (MCLK : in std_logic ;
          EN_1SEC : out std_logic) ;
    end component ;
    signal dig_1_i, dig_2_i : std_logic_vector(3 downto 0) ;
begin
    CLK_GEN_INST : clk_gen port map
        (MCLK => MCLK, EN_1SEC => en_1sec) ;

    process(MCLK)
    begin
        if MCLK'event and MCLK = '1' then
            if en_1sec = '1' then
                dig_1_i <= dig_1_i + 1 ;
                if dig_1_i = "1001" then
                    dig_1_i <= "0000" ;
                    dig_2_i <= dig_2_i + 1 ;
                    if dig_2_i = "1001" then
                        dig_2_i <= "0000" ;
                    end if ;
                end if ;
            end if ;
        end if ;
    end process ;

    end FPGA ;
```

4-



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity temp_cntrl is
	port(MCLK, EN_1SEC : in std_logic ;
		TEMP, CONS : in std_logic_vector(6 downto 0) ;
		CHAUF : out std_logic ) ;
end temp_cntrl ;


architecture FPGA of temp_cntrl is
	type T_etat is (mesure, chauffage, attente);
	signal etat : T_etat;
	signal duree, cmpt : std_logic_vector(14 downto 0) ;
begin
	process(MCLK)
		variable diff : std_logic_vector(6 downto 0) ;
	begin
		if MCLK'event and MCLK = '1' then
			if EN_1SEC = '1' then
				case etat is
				when mesure =>
					diff := CONS - TEMP ;
					if diff(6) ='0' then
						duree <= diff * x"F0" ; -- = 240 = 60*4
						CHAUF <= '1' ;
						etat <= chauffage ;
					end if ;
				when chauffage =>
					cmpt <= cmpt + 1 ;
					if cmpt = duree then
						cmpt <= (others => '0') ;
						etat <= attente ;
						CHAUF <= '0' ;
					end if ;
				when attente =>
					cmpt <= cmpt + 1 ;
					if cmpt = "000001110000011" then  --   = 60*15 -1
						cmpt <= (others => '0') ;
						etat <= mesure ;
					end if ;
				end case;
			end if ;
		end if ;
	end process ;
end FPGA ;
```