Examen du 04/03/2012

Réponses

1. Entiers Réels Physiques Et Enumérés. 0.5 PTS
   Exemple : **Type bit is ('0', '1'); Type boolean is (false,true);etc.** 0.25PTS
   S'transaction : retourne un signal. 0.25PTS
   C'est un signal de type bit qui change d'états pour chaque transaction du signal. 0.5PTS

EXERCICE 1:

```
library IEEE;                                   0,25PTS
use IEEE.std_logic_1164.all;                    0,25PTS

entity exo1 is                                  0,25PTS
 port (
A, B: in std_logic_vector(0 to 3);              0,25PTS
C: in std_logic;                                0,25PTS
X: out std_logic_vector(0 to 4));               0,25PTS
end exo1;

architecture bev of exo1 is                     0,25 PTS
signal U: std_logic_vector(0 to 4);             0,25 PTS
begin
process(A, B, U)                                0,25 PTS
begin
U(0) <= C;                                      0,5 PTS
for i in 0 to 3 loop                            0,25 PTS
X(i) <= A(i) xor B(i) and (not U(i));           1 PT
U(i+1) <= (A(i) and B(i)) or (B(i) and U(i)); 1 PT
end loop;
X(4) <= U(4) xor A(0);                          0,5 PTS
end process;
end;
```

Exercice 2:

DECLARATION DU PACKAGE ET ECRITURE DU TYPE 1.5 PTS

```
library IEEE;                                   0.25 PTS
use IEEE.STD_LOGIC_1164.ALL;                    0.25 PTS
PACKAGE mon_pack IS                             1 PT
TYPE input_data IS array (3 DOWNTO 0) OF std_logic_vector(7 DOWNTO 0);
END PACKAGE mon_pack;
```

```
-------------------------------------------------------------------------------------------
library IEEE;                                          0.25 PTS
use IEEE.STD_LOGIC_1164.ALL;                           0.25 PTS
use work.my_package.ALL;                               0.5  PTS
-------------------------------------------------------------------------------------------
ENTITY mux is                                          0.25 PTS
Port (s: IN std_logic_vector(1 downto 0);              0.25 PTS
      E: IN input_data;                                0.25 PTS
      D: OUT std_logic_vector(7 DOWNTO 0)); 0.25 PTS
end ENTITY mux;

ARCHITECTURE behavioral of mux IS                      0.25 PTS
begin

WITH s SELECT                                          0.25 PTS
   D<=  E(0) WHEN "00",                                0.5 PTS
        E(1) WHEN  "01",                               0.5 PTS
        E(2) WHEN "10",                                0.5 PTS
        E(3) WHEN "11",                                0.5 PTS
       "--------" WHEN OTHERS;                         0.5 PTS
end ARCHITECTURE behavioral;
```

Exercice 3:

SOLUTION: 8 ERREURS (0,5X8=4PTS)+CODE CORRIGÉ(0,5PTS)+FONCTION(1PT)= 6.5PTS

1ERE ERREUR : LE VECTEUR STD_LOGIC EST MAL ECRIT.

```
entity xxx is
generic(n: natural);
port(X: in std_logic_vector(n-1 downto 0);
    f: out std_logic_vector(2**n-1 downto 0));
end xxx;
```

2EME ERREUR : L'ENTITE XXX N'EST PAS REPRESENTEE DANS L'ARCHITECTURE :

```
architecture behav of xxxx is
```

3ÈME ERREUR: ETIQUETTE BODY

Il y a une erreur dans l'étiquette, elle ne doit pas être un mot réservé => BODY

4EME ERREUR : Il N'YA PAS DE LISTE DE SENSIBILITE

Puisque le process décrit n'a pas de wait à l'intérieur donc il doit avoir une liste de sensibilité.

```
Body_1 :process(x) is
```

5EME ERREUR : LA FONCTION DE CONVERSION N'EST PAS RECONNUE 'CONV_TO_INTEGER'.

La corriger par « conv_integer(x) ».

6EME ERREUR : ASSIGNATION DE SIGNAL ET NON PAS DE CONDITION BOOLEENNE SUR F(I) = '1' ET F(I) = '0'.

Correction: f (i) <= '1' ET f (i) <= '0'

7EME ERREUR: LA FIN DE L'INSTRUCTION IF "ENDIF";

Correction => end if ;

8EME ERREUR : LA FIN DE L'ARCHITECTURE NE S'ECRIT PAS SANS LE « E » COMME : END ARCHITECTUR BEHAV;

Correction => end architecture behav;

LE CODE CORRIGE EST DONC :

```
entity xxx is
generic(n: natural:=3);
port(X: in std_logic_vector(n-1 downto 0);
    f: out std_logic_vector(2**n-1 downto 0));
end xxx;
architecture behav of xxx is
begin
body_1: process(x) is
variable inp : integer;
begin
inp := conv_integer(X);
for i in 0 to 2**n -1 loop
if i=inp then f(i)<= '1';
else f(i) <='0';
end if;
end loop;
end process body_1;
end architecture behav;
```

CE CODE REPRESENTE UN DECODEUR 3 :8 SANS LE SIGNAL D'HABILITATION.