Chapitre 1 : Notions de VHDL

I Introduction

Le VHDL est un langage de description de haut niveau pour la description et les tests des composants. Il ne s'agit pas d'un langage de programmation classique de type séquentiel. Ce qui est décrit sont des éléments d'un circuit, dont tous les éléments sont concurrents…

1) *Avertissement*

Ce cours n'a pas pour objet d'étudier le VHDL de manière exhaustive mais permet de donner quelques repères. Les mots clé seront introduits au fur et à mesure des exemples.

2) *Type de description*

a. Description de composant par l'interconnexion de composants existants. (L'outil ECS de saisie de schéma)
b. Description de Tests pour tester des composants existants. (Test Bench Waveform, méthode systématique de simulation vue en TP)
c. Description de composant au niveau fonctionnel

Le logiciel utilisé en projet (Xilinx), dispose d'outils permettant de faire des descriptions avec des outils ne nécessitant pas l'apprentissage du VHDL pour les cas a et b, par contre il peut être intéressant d'utiliser une description du type c pour certains aspects, notamment pour les parties contrôle des automates PC/PO.

II Description fonctionnelle

1) *Principe*

a. Description du composant vu de l'extérieur (`ENTITY`)

i. Définir son nom
ii. Ses entrées
iii. Ses sorties
iv. Ses entrées/sorties

Exemple :

b. Description du composant au niveau fonctionnel (ARCHITECTURE)

i. Définir des signaux internes (signals)
ii. Définir les blocs fonctionnels internes (process)

iii. Définir des blocs logiques indépendants.

2) *Description d'un composant combinatoire*

a. Exemple complet simple

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

ENTITY  Exemple1 IS
    Port ( a : in std_logic;
           b : in std_logic;
           c : in std_logic;
           s : out std_logic);
end Exemple1;

architecture Behavioral of Exemple1 is
begin
     s<=(a or b) and not c;
end Behavioral;
```

b. Exemple MUX 4 vers 1 16 bits

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

entity MUX4_1_16bits is
 Port(E0,E1,E2,E3 : in std_logic_vector(15 downto 0);
      SEL : in std_logic_vector(1 downto 0);
      S : out std_logic_vector(15 downto 0));
end MUX4_1_16bits;

architecture Behavioral of MUX4_1_16bits is

begin
```

```
     end case;
 end process;
end Behavioral;
```

c. Exemple Additionneur signé.

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

entity addsub16 is
    Port ( A : in std_logic_vector(15 downto 0);
           B : in std_logic_vector(15 downto 0);
           S : out std_logic_vector(15 downto 0);
           CARRY : out std_logic;
           OVERFLOW : out std_logic);
end addsub16;

architecture Behavioral of addsub16 is
```

```
begin
```

```
     OVERFLOW <= (A(15) AND B(15) AND NOT S_interne(15)) OR
(NOT A(15) AND NOT B(15) AND S_interne(15));

end Behavioral;
```

3) Description d'un composant séquentiel : Bascule D

```
library IEEE;
entity basculeD is
    Port (     D : in std_logic_vector(15 downto 0);
               CLK,Reset : in std_logic;
               Q : out std_logic_vector(15 downto 0));
```

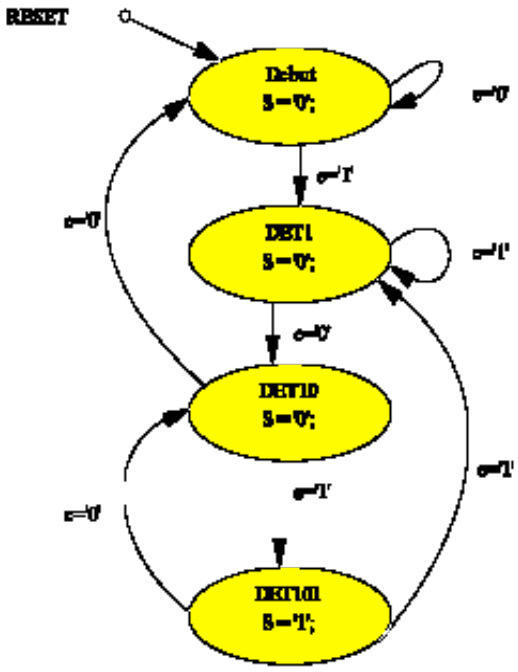
```
end basculeD;
architecture Behavioral of basculeD is
begin

    end process;
end Behavioral;
```

4) Conception d'un automate

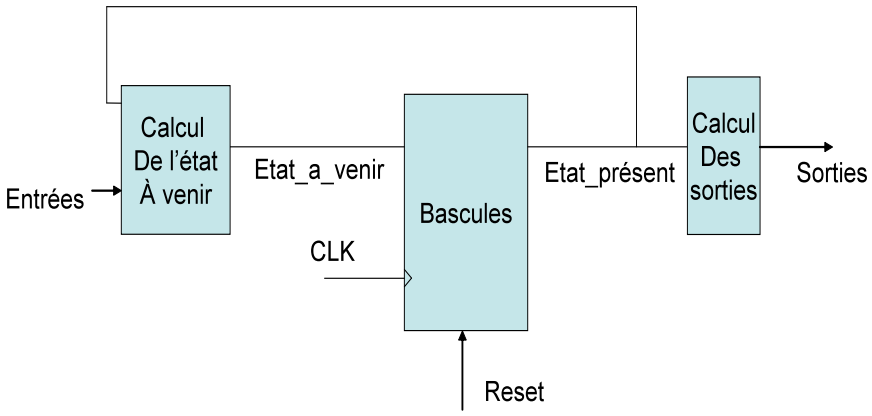
a. Rappel sur la méthode de conception

Automate, exemple sur la détection de la séquence 101 avec d'éventuels chevauchements.

Graphe d'état.



Réalisation



b. Description du composant répondant au cahier des charges

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

entity Detection101 is
    Port (
```

```
          );
end Detection101;
```

```
architecture Behavioral of Detection101 is

     type StateType is
```

```
     signal ETAT_A_VENIR,ETAT_PRESENT : StateType;

begin
BASCULES :
```

```
     process (CLK,RESET)
     begin
          if (RESET='1') then
               ETAT_PRESENT<=
          elsif (CLK'EVENT AND CLK='1') THEN
               ETAT_PRESENT<=
          end if;
     end process BASCULES;

CALCUL_SORTIES_ET_ETAT_A_VENIR :
     process
     begin
          case ETAT_PRESENT is
               when debut =>
                    end if;
               when det1 =>
                    s<='0';
                    if (e='1') then
                         ETAT_A_VENIR<=det1;
                    else
                         ETAT_A_VENIR<=det10;
                    end if;
               when det10 =>
                    s<='0';
                    if (e='1') then
                         ETAT_A_VENIR<=det101;
                    else
                         ETAT_A_VENIR<=debut;
                    end if;
               when det101 =>
                    s<='1';
                    if (e='1') then
                         ETAT_A_VENIR<=det1;
                    else
                         ETAT_A_VENIR<=det10;
                    end if;
          end case;
     end process CALCUL_SORTIES_ET_ETAT_A_VENIR;
end Behavioral;
```