Aide Memoire pour le VHDL

Reference rapide :

- Assignation : <=
- Arithmetique : +, - et * (souvent, pas de division)
- Logique : AND, OR, NOT, XOR et parfois XNOR.
- Concatenation (coller des termes ensembles) : &
- Pour representer un bit on utilise des apostrophes
- Pour representer plusieurs bits, on utilise des guillemets
- Pour dire que TOUS les bits sont 0, on utilise ceci : (OTHERS => '0') . Dans cet exemple, on met tous les bits de s_signal a 0 : `s_signal <= (OTHERS => '0') ;`
- On ne peut pas LIRE un sortie. Pour eviter ce probleme, on peut se créer un signal intermediaire qu'on assigne a la sortie.
- Un process est un bloc qui est execute quand au moins un des signaux dans sa liste de sensibilite change.
- Tous les process peuvent s'executer en parallele
- Les process s'executent en meme temps que les autres commandes qui ne sont PAS dans un process.
- Un signal (ou une sortie) ne peut PAS etre modifie par plus que 1 process.

Format d'un code VHDL:

1) Declaration des librairies : On recopie 3 lignes.
2) Definition de l'entite : On definit les entrees et les sorties.
3) Definition de l'architecture : On retrouve le vrai code ici.

Pour declarer les librairies on recopie ces 3 lignes:

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
```

L'entite se definit de la facon suivante:

```
ENTITY nom_entite IS
    PORT (
          entree_1bit  : IN  STD_LOGIC ;
          sortie_2bits : OUT STD_LOGIC_VECTOR(1 DOWNTO 0)
         ) ;
END nom_entite ;
```

Remarquez que la derniere definition (sortie_2bits dans ce cas-ci) ne finit pas avec un point-virgule.

L'architecture est decrite de la facon suivante :

```
ARCHITECTURE nomarchitecture OF nom_entite IS

SIGNAL un_bus_de_5bits_dans_le_design : STD_LOGIC_VECTOR(4 DOWNTO 0);

BEGIN
...
END;
```

La logique combinatoire (sans mémoire) peut s'ecrire de 2 facons :

- Dans un process
- A l'exterieur d'un process

Les deux font EXACTEMENT la meme chose si on decrivait bien les choses.

Exemple a l'interieur d'un process :

```
PROCESS (entreea, entreeb)
BEGIN
    sortie <= entreea + entreeb ;
END PROCESS ;
```

Exemple a l'exterieur d'un process :

sortie <= entreea + entreeb ;

Certaines commandes ne se trouvent QUE dans un process. On parle notamment de IF/ELSIF/ELSE et les CASE.

Exemple d'un multiplexeur a l'interieur d'un process :

```
PROCESS (entree0, entree1, sel, sortie)
BEGIN
    IF sel = '0' THEN
        sortie <= entree0;
    ELSE
        sortie <= entree1;
    END IF;
END PROCESS;
```

Regle a respecter pour un process combinatoire :

- Tous les signaux lus DOIVENT etre dans la liste de sensibilite (ici, on a entree0, entree1 et sel)
- Toutes les situations doivent etre tenu en compte (si sel=0, on met entree0 a la sortie et si sel=1, on met entree1… il y avait 2 options, on a couvert les 2).

Exemple d'un multiplexeur a l'exterieur d'un process.

```
sortie <= entree0 WHEN sel = '0'
                  ELSE
          entree1;
```

Exemple d'un multiplexeur a l'inteiru d'un process avec CASE :

```
PROCESS (entree0, entree1, entree2, entree3, sel)
BEGIN
    CASE sel IS
        WHEN "00" =>
            sortie <= entree0;
        WHEN "01" =>
            sortie <= entree1;
        WHEN "10" =>
            sortie <= entree2;
        WHEN OTHERS =>
            sortie <= entree3;
    END CASE;
END PROCESS;
```

Pour la logique sequentielle, il faut obligatoirement utiliser un process. Les regles dans ce process sont differentes.

- La liste de sensibilite devrait contenir : une horloge et peut-etre un reset. RIEN DE PLUS.
- Chaque assignation correspond a une (ou plusieurs) flip flops
- On a le droit de ne pas tout specifier : quand ce n'est pas specifie, ca garde la valeur precedente

La structure generale devrait TOUJOURS etre du type 1 ou du type 2

Type 1 :

```
PROCESS (clk)
BEGIN
    IF clk'EVENT AND clk = '1' THEN
        ...
    END IF;
END PROCESS ;
```

Type 2 :

```
PROCESS (clk, rst_an)
BEGIN
    IF rst_an = '0' THEN
        ...
    ELSIF clk'EVENT AND clk = '1' THEN
        ...
    END IF;
END PROCESS ;
```

Il ne devrait PAS Y AVOIR D'AUTRES STRUCTURES. Dans les sections avec les points de suspension, on fait ce qu'on veut. On peut mettre d'autres IFs, des case, etc. Cependant, la structure « externe » devrait seulement etre une des deux presentees precedemment.