TP Architecture des Ordinateurs

Objectif : réaliser un additionneur 4 bits en VHDL

Exercice 1 : réaliser un ½ additionneur 1 bit :

- Entrées : a et b les 2 bits à additionner
- Sorties : s la somme et r la retenue

Rappel : s = a xor b et r = a.b

On utilisera de préférence le type std_logic au type bit.

On simulera le temps de traversée de chaque porte par un temps d'attente d'1 ns. Par exemple, avec l'instruction : s <= a xor b after 1 ns.

Quel est le temps de stabilisation maximal pour le ½ additionneur ?

Exercice 2 : réaliser un additionneur complet 1 bit à partir de deux ½ additionneurs 1 bit :

- Entrées : a et b les 2 bits à additionner, r_in la retenue entrante
- Sorties : s la somme et r_out la retenue sortante

Quel est le temps de stabilisation maximal pour l'additionneur complet ?

Exercice 3 : réaliser un additionneur 4 bits à partir de 4 additionneurs complets 1 bit :

- Entrées : a et b deux mots de 4 bits et r_in la retenue entrante
- Sorties : s la somme sur 4 bits et r_out la retenue sortante

Quel est le temps de stabilisation maximal pour l'additionneur 4 bits ?

Exercice 4 : optimisation du calcul de la retenue.

Le temps de traversée de l'additionneur 4 bits est dû en grande partie à la propagation de la retenue entre les différents additionneurs complets (exemple 0001 + 1111). Nous allons étudier une solution qui permet d'optimiser ce calcul.

On définit les fonctions suivantes pour i = 0 à 3 (0 poids faible, 3 poids fort).

- génération de retenue : $g_i = a_i.b_i$
- propagation de retenue : $p_i = a_i$ xor b_i
- calcul de la retenue : $c_i = g_i + p_i.c_{i-1}$
- calcul de la somme : $s_i = p_i$ xor c_{i-1}

Remarque : (g_i,p_i) constituent la sortie d'un ½ additionneur appliqué à a_i et b_i.

Questions :

1. expliquer ces formules
2. donner les formules de s_0, c_0, s_1, c_1, s_2, c_2, s_3, c_3 en fonction de g_0, p_0, g_1, p_1, g_2, p_2, g_3, p_3 et c_e (la retenue entrante).
3. en déduire le schéma du nouvel additionneur 4 bits à partir de 4 ½ additionneurs
4. réaliser le circuit en VHDL
5. vérifier le gain dans le temps de traversée de l'additionneur 4 bits en affectant 1 pénalité de 1 ns à chaque porte utilisée. On pourra utiliser des portes OU et ET à un nombre quelconque d'entrées.