SUJET DU TP N°1

Pour l'ensemble du TP le circuit à spécifier pour les propriétés des projets est le :

VIRTEX-4 XCL4VSX35-FF668-10

OBJECTIFS DU TP

Par des exemples simples (multiplieurs, additionneurs, etc.) ce TP permet de mettre en œuvre plusieurs solutions pour réaliser un même circuit arithmétique. En jouant sur les options de synthèse ce TP permet d'observer l'impact sur les performances d'un circuit de l'utilisation de logique distribuée ou de logique câblée.

COMPTE RENDU

un compte rendu détaillé doit être remis en fin de séance. Celui-ci doit contenir les fichiers VHDL des différents circuits (un seul par largeur de mot, donc 3 fichiers VHDL). Les résultats de synthèse obtenus dans les différents cas seront largement comparés et analysés, il est conseillé d'utiliser tous les outils disponibles dans le panel ISE pour s'aider dans l'analyse (en particulier FPGA Editor). Il est possible de réaliser un ou plusieurs circuits supplémentaires si les résultats qu'ils apportent aident à la compréhension des résultats globaux obtenus.

Remarque : *Il est vivement conseillé de faire un projet pour chaque circuit !*

MULTIPLICATION D'UN NOMBRE PAR UNE CONSTANTE CODEE SUR 10 BITS

Dans les exemples suivants la constante vaut le nombre décimal 100 (mais vous pouvez tester une autre constante)

1. Réalisez un premier multiplieur simple d'un mot codé sur N bits avec une constante codée sur 10 bits en utilisant une description comportementale (pensez à utiliser la librairie ieee.numeric_std.all), simulez le design avec ModelSim (en prenant N=8 par exemple).

   RECUPERER LE CODE VHDL SUR LE SITE INTERNET DE VOTRE ENSEIGNANT !

2. Dans les options de synthèse décocher la case « Add IO buffers » afin de supprimer l'insertion des pattes d'E/S du calcul du chemin critique. Positionnez le paramètre « DSP48 usage » à la valeur « off ». Comment les multiplieurs vont-ils être synthétisés ? Vous le vérifierez en regardants les schémas RTL et technologique du circuit ainsi synthétisé.

Pour N=8, 16, 26, 32 et 64, notez les résultats de synthèse obtenus :

- ressources du composant utilisées (nombre de blocs DSP et/ou de slices),
- fréquence maximale (à partir de l'estimation du chemin critique),
- Ces informations sont disponibles dans le rapport de synthèse fourni par ISE

3. Synthétisez une nouvelle fois le projet en forçant l'utilisation de Xtreme DSP Slice lors de l'implémentation du multiplieur (DSP usage = yes). Pour N=8, 16, 26, 32 et 64. Vous relèverez les mêmes informations que précédemment.

4. Quels sont les avantages de l'utilisation des DSP Block ? Dans quelles limites ?

5. Faire une synthèse de l'ensemble des résultats obtenus (entre autre en utilisant des courbes taille de la donnée en entrée/fréquence max, ressources utilisées).

MULTIPLICATION ET ADDITION DE DEUX NOMBRES SIGNES

6. Maintenant vous allez utiliser multiplieur asynchrone manipulant des nombres signés en description comportementale (fourni par votre enseignant). Faites varier la largeur des mots d'entrée avec les valeurs suivantes : 8 bits, 16 bits, 26 bits, 32 bits et 64 bits. Pour chaque largeur de mots d'entrée, forcez le synthétiseur à utiliser des éléments logiques tout d'abord puis des Xtreme DSP Slice.

7. Notez dans un tableau les résultats de synthèse obtenus pour chaque largeur de mots et comparez les résultats dans les deux cas. Vous pouvez représenter les résultats sous forme de courbes.

8. Réalisez un additionneur synchrone de nombres signés en description comportementale. Faire varier la largeur des mots d'entrée avec les valeurs suivantes : 8 bits, 16 bits, 26 bits, 32 bits et 64 bits. Dans cette expérience, l'utilisation de bloc DSP n'a aucune incidence sur l'implantation matérielle car les additionneurs sont réalisés à l'aide de Look Up Table uniquement.

9. Superposez les résultats obtenus avec ceux des multiplieurs afin de comparer les performances.

DIVISION DE 2 NOMBRES ENTIERS

10. Ecrivez un diviseur entier non signé sur 8 bits (pour les 2 entrées) en VHDL. Vérifiez sont fonctionnement sous *ModelSim* et synthétisez le à l'aide d'ISE pour évaluer ses performances.

11. Augmenter la taille du diviseur afin de le faire traiter des données sur 32 bits et évaluez à nouveau ses performances.

12. Concluez sur le cout des opérations de division vis a vis des opérateurs d'addition et de multiplication

MULTIPLICATION DE DEUX NOMBRES COMPLEXES

13. En utilisant l'outil *Core Generator*, réaliser un multiplieur complexe de mots de 16 bits. Vous étudierez les différentes implémentations proposées. Une fois le multiplieur généré vous devez l'instancier, l'utilisation du VHDL template rend bien service ;-)

14. Faire une synthèse et notez les résultats obtenus

CONCLUSION ET SYNTHESE