Unité de calcul modulaire
ALU "Arithmetic Logic Unit"

Mandat

Le but de ce projet est de concevoir, réaliser et tester une description VHDL synthétisable d'une unité modulaire pour des opérations arithmétiques et logiques. Un tel circuit est communément nommé ALU pour *Arithmetic Logic Unit*. Ce circuit sera de type combinatoire. Il devra répondre aux spécifications données ci-après (N= 4).

Port	Direction	Taille	Description
Opcode_i	Entrée	2	Entrée indiquant quelle opération le circuit doit exécuter, ce signal est sur 2 bits. Il y a 4 opérations possibles
A_i	Entrée	N	Opérande des opérations (N bits, en notation "complément à 2")
B_i	Entrée	N	Opérande utilisée dans certaines opérations (N bits, en notation "complément à 2")
Carry_i	Entrée	1	Propagation de retenue en entrée
Result_o	Sortie	N	Résultat de l'opération effectuée (N bits, en notation "complément à 2")
Carry_o	Sortie	1	Propagation de retenue en sortie
à définir	Entrées et sorties		Différentes entrées et sorties nécessaires pour le chainage de plusieurs unités. Après identification des besoins, choisir les signaux à propager dans chaque opération en essayant d'optimiser des ressources. Démontrer l'utilité de vos choix.
à définir	Sorties		Sorties nécessaires pour le chainage de plusieurs unités et pour la génération des fanions de sortie demandés pour la conception de l'ALU 12-bits. Identifier ces signaux et justifier leur utilité.

Spécification du fonctionnement souhaité

L'unité ALU est capable d'effectuer des opérations d'addition, soustraction, multiplication par 2 et division par 2 pour des nombres non-signés et signé sur 4 bits. Le tableua ci-dessous vous donne la correspondance entre les valeurs de `Opcode_i` et l'opération à réaliser:

Opcode_i	Opération	Valeur sur Result_o
00	Addition	A_i + B_i
01	Soustraction	A_i – B_i
10	Multiplication par 2	A_i * 2
11	Division par 2	A_i / 2

Le circuit réalisé devra nous permettre d'intégrer plusieurs ALUs 4-bits pour construire des ALUs d'une taille plus importante. On doit pouvoir intégrer 3 ALUs 4-bits pour créer une ALU -bits, ou 8 ALUs 4-bits pour créer une ALU 32-bits. Pour ceci, il faut générer les sorties de propagation nécessaires pour chaque opération, afin de pouvoir récupérer la valeur sur les entrées de propagation des autres ALUs.

Lors de la division, le résultat entier sera toujours arrondi vers le bas (5/2 = 2 et -5/2 = -3).

Contraintes

Afin de minimiser le nombre de portes logiques lors de l'implémentation des systèmes numériques, il faudra éviter d'utiliser un bloc soustracteur en parallèle à un bloc additionneur. En effet, l'expression A – B peut être réécrite A + (–B). La soustraction sera donc réalisée en réutilisant le même additionneur utilisé pour l'addition.

Dans la vue RTL de votre ALU synthétisée, il ne devra y avoir qu'un seul composant additionneur 4-bits (ou un 3-bits plus un autre à 1-bit), mais aucun soustracteur.

Utilisations et Test

ALU 12-bits pour nombres signés

Pour vérifier si votre ALU 4-bits modulaire a été correctement décrite on vous demandera de la tester en implémentant une ALU 12-bits à partir de 3 ALUs 4-bits. L'ALU 12-bits aura les entrées/sorties suivantes (N = 12) :

Port	Direction	Taille	Description
Opcode_i	Entrée	2	Entrée indiquant quelle opération le circuit doit exécuter, ce signal est sur 2 bits. Il y a 4 opérations possibles
A_i	Entrée	N	Opérande des opérations (N bits, en notation "complément à 2")
B_i	Entrée	N	Opérande utilisée dans certaines opérations (N bits, en notation "complément à 2")
Result_o	Sortie	N	Résultat de l'opération effectuée (N bits, en notation "complément à 2")
Zero_o	Sortie	1	Fanion signalant un résultat égal à 0
Ovr_o	Sortie	1	Fanion signalant un résultat dépassant la capacité de représentation sur N bits.

Soustracteur 12-bits pour nombres non-signés

Pour cette partie, il est demandé de concevoir un soustracteur 12-bits pour nombres non-signés. Pour cela, il faut utiliser trois composants ALU_4bits. Le soustracteur aura les entrées/sorties suivantes :

Port	Direction	Taille	Description
A_i	Entrée	N	Opérande des opérations (N bits, non-signé)
B_i	Entrée	N	Opérande utilisée dans certaines opérations (N bits, en notation "complément à 2")
Result_o	Sortie	N	Résultat de la soustraction (N bits, non-signé)
Zero_o	Sortie	1	Fanion signalant un résultat égal à 0
Erreur_o	Sortie	1	Fanion signalant un résultat incorrect, produit par une soustraction A-B avec A<B.

Marche à suivre

Première partie

1. Définir les signaux de propagation nécessaires pour permettre une décomposition en modules cascadés, en tenant compte des 4 opérations possibles.
2. Sur papier, réaliser le schéma d'une décomposition structurelle de l'ALU 12-bits utilisant trois circuits ALU_4bits. Ajouter à l'ALU 4-bits les entrées et sorties nécessaires pour la propagation des signaux et la génération de fanions. Au besoin, inclure la logique pour connecter ces signaux.
3. Donner le schéma bloc (sur papier) de l'ALU_4bits correspondant à votre conception. Chaque bloc sera spécifié par un symbole CEI, une TDV ou un texte.
4. Créer un symbole nommé ALU_4bits avec les E/S définies lors de votre conception.
5. Faire une description VHDL synthétisable de l'unité ALU correspondant au schéma conçu au point 3.
6. Tester la description VHDL de l'unité ALU (composant ALU_4bits) à l'aide d'une simulation interactive avec Top_Sim et la console REDS.
7. Faire la synthèse du composant ALU_4bits. Analyser la quantité de logique obtenue.
8. .
9. Utiliser le composant nommé ALU_Sgn_12bits_top et, à l'aide de l'éditeur de schémas de HDL Designer, introduire le schéma de l'unité ALU correspondant au schéma conçu au point précédent.
10. Pour tester manuellement, utiliser le composant Top_Sim_Console_ALU et exécuter la console Console_ALU_Combi.tcl depuis QuestaSim.
11. Tester le composant ALU_Sgn_12bits_top à l'aide du banc de test fourni, ALU_Sgn_12bits_top_tb.
12. Faire la synthèse et le placement-routage du composant ALU_Sgn_12bits_top pour un circuit MAX EPM7128SLC84 (CPLD).
13. Utiliser le fichier d'assignation de pins fournit.
14. Tester ce circuit sur une carte EPM 25p-25p, avec la console USB2 et le script Console_ALU_Combi.tcl.

Deuxième partie

15. Concevoir un schéma structurel utilisant trois circuits ALU_4bits afin de réaliser un soustracteur 12-bits pour nombre non-signés et générer les fanions demandés.
16. Utiliser le composant nommé SUB_Usgn_12bits_top et, à l'aide de l'éditeur de schémas de HDL Designer, introduire le schéma de l'unité SUB correspondant au schéma conçu au point précédent.
17. Pour tester manuellement, utiliser le composant Top_Sim_Console_ALU et exécuter la console Console_ALU_Combi.tcl depuis QuestaSim.
18. Faire la synthèse et le placement-routage du composant SUB_Usgn_12bits_top pour un circuit MAX EPM7128SLC84 (CPLD).
19. Utiliser le fichier d'assignation de pins fournit.
20. Tester ce circuit sur une carte EPM 25p-25p, avec la console USB2 et le script Console_ALU_Combi.tcl.