

Laboratoire de systèmes logiques semestre automne 2022 - 2023

Laboratoire ALU

Informations générales

Le rendu pour ce laboratoire se fera **par groupe de deux**, chaque groupe devra rendre son travail.

Le laboratoire sera évalué sous la forme d'un quiz à la fin de la dernière séance du laboratoire. La note sera ainsi pondérée par rapport à la qualité de votre circuit et surtout sur votre compréhension sur ce dernier qui sera évalué grâce au quiz, ciblé sur ce laboratoire.

 **N'oubliez pas de sauvegarder et d'archiver votre projet à chaque séance de laboratoire**

NOTE 1 : Afin de ne pas avoir de pénalité pensez à respecter les points suivants

- Toutes les entrées d'un composant doivent être connectées. (-0.1 sur la note par entrée non-connectée)
- Lors de l'ouverture de Logisim, bien préciser votre nom en tant que User
- Ne pas modifier (enlever/ajouter/renommer) les entrées/sorties déjà placées
- Ne pas modifier le nom des composants déjà présents

NOTE 2 : Lors de la création de votre circuit, tenez compte des points suivants afin d'éviter des erreurs pendant la programmation de la carte FPGA :

- Nom d'un circuit \neq Label d'un circuit
- Nom d'un signal (Pin) \neq Label et/ou Nom d'un circuit, toutes les entrées/sorties doivent être nommées
- Les composants doivent avoir des labels différents

NOTE 3 : Nous vous rappelons que si vous utilisez les machines de laboratoire situées au niveau A, il ne faut pas considérer les données qui sont dessus comme sauvegardées. Si les machines ont un problème, nous les remettons dans leur état d'origine et toutes les données présentes sont effacées.

Objectifs du laboratoire

L'objectif principal de ce laboratoire est la réalisation d'une unité arithmétique et logique (ALU) 4 bits. Pour cela, vous devrez d'abord créer un certain nombre de composants puis les utiliser afin de concevoir l'ALU. L'idée est de développer un système de A à Z afin que vous puissiez faire chaque étape vous-même et ainsi bien comprendre les concepts vus dans la théorie du cours afin de les appliquer dans un cas pratique.

Ce laboratoire est noté. Vous devez rendre le projet Logisim et répondre à un quiz. Le quiz sera orienté de façon à pouvoir évaluer votre compréhension du laboratoire.

Outils

Pour ce labo, vous devez utiliser les outils disponibles sur les machines de laboratoire (A07/A09) ou votre ordinateur personnel avec Logisim installé.

⚠ La partie programmation d'une FPGA ne peut se faire que sur les ordinateurs présents dans les salles (A07/A09).

Fichiers

Vous devez télécharger à partir du site Cyberlearn le projet Logisim dédié à ce laboratoire.

Logisim fourni

Vous allez recevoir un projet Logisim qui contient la plupart des entités que vous allez réaliser dans le cadre de ce laboratoire. Vous devrez compléter ces entités afin de réaliser les fonctions demandées. De plus ne modifiez surtout pas les noms des entrées/sorties déjà placées dans ces entités et n'ajoutez pas d'entrée/sortie supplémentaires.

1 ALU

Travail à effectuer

Dans le cadre de ce laboratoire, vous allez implémenter une ALU 4-bits capable de réaliser les opérations arithmétiques et logiques listées ci-dessous.

Opération	Fonctionnalité	Opcode
A + B (signé)	Add/Sub	0000 $\phi\phi$
A - B (signé)		0010 $\phi\phi$
A + B (non-signé)		0001 $\phi\phi$
A - B (non-signé)		0011 $\phi\phi$
A \geq B (signé)	Comparateur	011001
A < B (signé)		011010
A \neq B		011011
A = B		011100
A \geq B (non-signé)		011101
A < B (non-signé)		011110
A and B	Logique	10 $\phi\phi$ 00
A or B		10 $\phi\phi$ 01
A nand B (non-signé)		10 $\phi\phi$ 10
A xor B (non-signé)		10 $\phi\phi$ 11
Comptage du nombre de 1 dans A	Custom	11 $\phi\phi\phi\phi$

Architecture de l'ALU

Cette ALU est composée de quatre sous-blocs (voir figure ci-dessous), chacun réalisant une fonctionnalité particulière. Ces quatre blocs contiennent :

- Un circuit combinatoire capable d'additionner et de soustraire,
- Un circuit combinatoire exploitant le résultat du soustracteur pour déterminer si deux nombres sont égaux, ou pas,
- Un circuit combinatoire réalisant les fonctions logiques indiquées dans la liste d'opérations de l'ALU,
- Un circuit combinatoire "custom" réalisant une opération spécifique. Dans notre cas, le bloc custom indique le nombre de 1 dans l'entrée A.

Schéma bloc de l'ALU

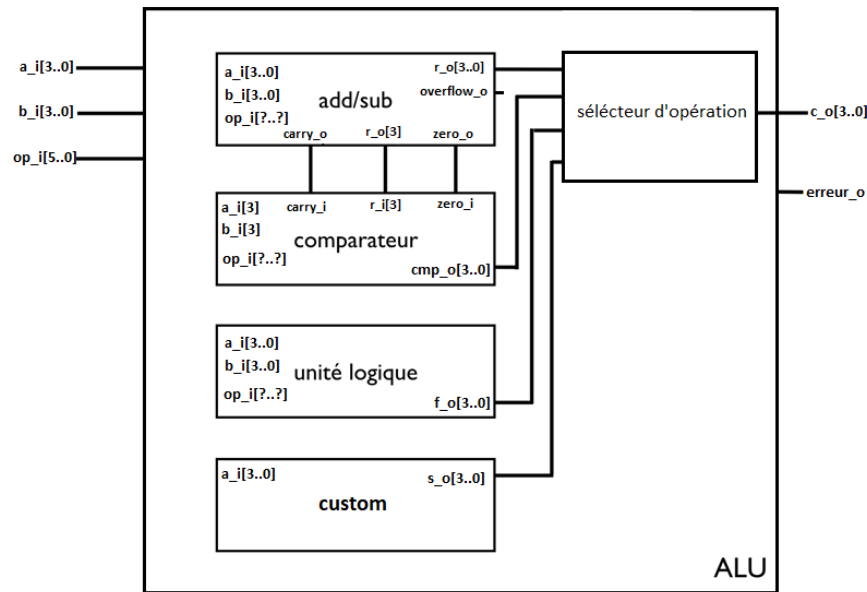


FIGURE 1 – Schéma bloc de l'ALU

Nom I/O	Description
a_i	Données A
b_i	Données B
op_i	Opcode
c_o	Résultat de l'ALU
erreur_o	Signal d'erreur

Travail à effectuer

Etape 1-a : Opcode

Identifier les bits de l'opcode (op_i) déterminant la fonctionnalité à exécuter. Ces bits seront utilisés par un multiplexeur (voir schéma bloc de l'ALU).

Dans le cas où l'opcode est invalide, l'ALU doit retourner une erreur.

Etape 1-b : Add/Sub 4 bits

Concevoir un additionneur/soustracteur 4-bits. Il doit être constitué **d'un seul** additionneur 4 bits.

L'entrée op[? : ?] sélectionne l'opération que réalise le bloc Add/Sub.

Le bloc Add/Sub génère une sortie R[3 : 0] à 4 bits avec la somme ou la différence entre A et B, ainsi que quatre sorties à 1 bit. Ces sorties à 1 bit sont :

- Overflow : pour un dépassement de capacité.
- Carry : représente une retenue.
- Zero : si le résultat de l'opération Add/Sub vaut zéro.
- R[3] : le bit de poids fort du résultat de l'additionneur/soustracteur.

Ces trois dernières sorties seront utilisées par le bloc comparateur.

Le bit overflow est différent pour les opérations arithmétiques avec des nombres signés et non-signés. Il peut être calculé avec l'étape suivante.

Etape 1-c : Overflow

Concevoir le bloc (add/sub) de l'ALU en intégrant l'additionneur-soustracteur 4-bits et le calcul de dépassement de capacité (overflow). Prévoir les sorties indiquées sur le schéma de l'ALU afin de le connecter au bloc comparateur.

NOTE : Le bit « overflow » se calcule de manière différente lorsque l'on fait des opérations avec des nombres signés et non-signés. Cas signé : Voir tableau ci-dessous Cas non-signé : le bit carry (ou emprunt) sera considéré. Une seule sortie « overflow » sortira du bloc add/sub.

Opération	A	B	Résultat indiquant un overflow
A + B	≥ 0	≥ 0	< 0
A + B	< 0	< 0	≥ 0
A - B	≥ 0	< 0	< 0
A - B	< 0	≥ 0	≥ 0

Etape 1-d : Comparateur

L'entrée op[?] du bloc comparateur sélectionne l'opération à réaliser.

Toutes ces opérations peuvent être réalisées par des fonctions combinatoires exploitant les entrées A[3], B[3], R[3], carry et zero.

Le bloc comparateur génère une sortie cmp[3:0] 4-bits indiquant si le résultat de la comparaison est vrai (cmp[3:0] = '0001') ou faux (cmp[3:0] = '0000').

Concevoir le bloc comparateur en synthétisant les fonctions logiques indiquées dans le tableau ci-dessous.

Opération	Fonction logique
A \geq B (signé)	(not(A[3]) and B[3]) or (not(R[3]) and (A[3] xnor B[3]))
A < B (signé)	(A[3] and not(B[3])) or (R[3] and (A[3] xnor B[3]))
A \neq B	not(zero)
A = B	zero
A \geq B (non-signé)	carry
A < B (non-signé)	not(carry)

NOTE : Ce comparateur se base sur l'utilisation du bloc Add/Sub. Par exemple, pour déterminer si deux nombres sont égaux, il fait une soustraction et détermine une différence lorsque le résultat n'est pas zéro.

Etape 1-e : Unité Logique

Concevoir le bloc « unité logique » réalisant les opérations logiques de l'ALU. Prévoir un bus de sortie F à 4 bits comme indiqué ci-dessous.

L'entrée op[?] de l'unité logique sélectionne l'opération logique bit à bit des entrées A et B à réaliser. L'unité logique génère une sortie à 4 bits F[3:0].

Par exemple, si $op = 00$, l'unité logique doit fournir en sortie $F[3] = A[3]$ and $B[3]$, $F[2] = A[2]$ and $B[2]$, $F[1] = A[1]$ and $B[1]$, $F[0] = A[0]$ and $B[0]$.

Etape 1-f : Unité custom

Ce bloc combinatoire compte le nombre de bit à 1 sur l'entrée $A[3:0]$ et génère une sortie $S[3:0]$ sur 4 bits correspondant aux nombres de bits de A qui sont à 1.

Par exemples :

Si $A = '0010'$, alors $S = 1$.

Si $A = '1110'$, alors $S = 3$.

Pour votre implémentation vous pouvez utiliser des additionneurs 1-bit et un additionneur 4-bits.

Etape 1-g : ALU

Concevoir le bloc « ALU_Top » en intégrant les quatre précédents blocs comme indiqué sur le schéma-bloc. Il doit être constitué des 3 entrées a_i , b_i , $opcode_i$ et des 2 sorties c_o , $erreur_o$.

Le signal $erreur_o$ doit être à '1' si l'opcode utilisé ne correspond pas à une opération valide **OU** si le résultat de l'opération demandé n'est pas correct. Par exemple, si l'utilisateur demande l'exécution d'une addition non-signée et qu'un overflow est détecté, le signal $erreur_o$ doit être mis à '1'.

Etape 1-h : Simulation Simulez manuellement votre ALU pour vérifier son bon fonctionnement.

Vous disposez également d'un testbench en ligne (<http://reds-calculator/logisim-validator/>) qui vous permet de valider le fonctionnement de votre ALU. Référez-vous au labo d'introduction pour son utilisation.

Le simulateur qui est mis à disposition n'est actuellement pas capable de vous retourner une erreur si vous ne respectez pas les points recommandés pour créer un circuit (NOTE 1 et 2). Donc si la simulation de votre circuit sur la page web ne s'arrête pas (dépasse 40s), vérifiez et respectez bien les points suivants :

- Nom du fichier `labo_alu_etu.circ`
- Nom du composant `ALU_Top`
- Respecter bien les informations données dans les NOTE 1 et 2 au début de la donnée.

Etape 1-i : Intégration / Validation

Pour l'intégration sur carte, utilisez le composant `ALU_MAXV`.

Intégrez le projet « `ALU_MAXV` » avec le but de programmer un circuit programmable et tester votre ALU. Utilisez la console d'interrupteurs pour donner les opérandes (A et B), les « dip-switch » de la carte Max V pour indiquer l'opération à réaliser et les LEDs sur la console d'interrupteurs pour afficher les sorties du système. Lors de la programmation, dans le menu « FPGA commander », sélectionner la carte « `MAX_V_CONSOLE` » (Choose target board).

Tester le fonctionnement sur la carte.

Faites valider le fonctionnement par l'assistant ou le professeur.

Rendu

Pour ce laboratoire, vous devez rendre :

— votre fichier *.circ*

Vous devez déposer les rendus sur Cyberlearn jusqu'à la date indiquée dans l'espace de rendu consacré à votre classe. Ainsi, vous recevrez un feedback dans le courant de semaine suivante.

CONSEIL : Faire une petite documentation sur cette partie vous préparerait directement pour le quiz et vous fera directement un résumé pour l'examen.
--