HEIG VD | REDS Institut Reconfigurable and Embedded Digital Systems

Prof. Romuald Mosqueron

Prof. Olivier Auberson

Laboratoire de systèmes logiques
semestre automne 2024 - 2025

Laboratoire Add/Sub

Informations générales

Le rendu pour ce laboratoire se fera **par groupe de deux**, chaque groupe devra rendre son travail avant la date mentionnée sur Cyberlearn.

Ce laboratoire se déroule sur **une séance** et sera évalué de la façon suivante :

- Evaluation du circuit rendu
- Evaluation des réponses aux questions

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 vos noms en tant que User
- Ne pas modifier (enlever/ajouter/renommer) les entrées/sorties déjà placées
- Contrairement à ce que vous avez pu voir en cours, merci de ne pas utiliser des portes XOR sur plus d'un bit.

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. Pensez à sauvegarder votre travail sur un autre support.

Outils

Pour ce laboratoire, 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 Logisim fourni

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

Le projet contient certaines des entités que vous allez réaliser dans le cadre de ce laboratoire. Vous devrez compléter ces entités et en créer de nouvelles 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.

Add / Sub : contexte

Lors du tutoriel d'utilisation de Logisim, vous avez eu l'occasion de réaliser un additionneur 1 bit, puis de l'instancier plusieurs fois pour réaliser un additionneur 4 bits.

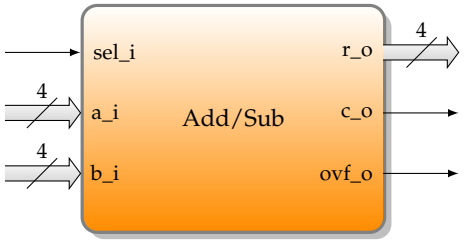
Dans ce laboratoire, vous allez explorer plus en profondeur l'utilisation de l'additionneur pour gérer des soustractions et des nombres signés.

Vous pourrez également étudier le fonctionnement du bit de carry et de l'overflow et d'en comprendre l'importance dans les calculs binaires.

1 Add/Sub 4 bits

Travail à effectuer

Entité du bloc Add/Sub 4bits



Nom I/O	Description
sel_i	Permet de sélectionner l'addition ou la soustraction
a_i	Données A
b_i	Données B
r_o	Résultat de l'addition/soustraction
c_o	Flag indiquant un carry
ovf_o	Flag indiquant un overflow

Etape 1-a : Implémenter un additionneur 4 bits

Créer un composant additionneur 4 bits à l'aide d'additionneurs 1 bit dans Logisim et tester son fonctionnement.

Etape 1-b : Implémenter un soustracteur 4 bits

À l'aide d'un seul additionneur 4 bits et de portes logiques, créer un soustracteur 4 bits dans Logisim et tester son fonctionnement.

QUESTION 1 : Que pouvez vous constater quant au comportement du bit de carry lors d'une soustraction ? (Il ne vous est pas demandé de modifier votre circuit, répondez simplement à la question.)

Etape 1-c : Implémenter un additionneur/soustracteur 4 bits

À l'aide d'un seul additionneur 4 bits et de portes logiques, créer un additionneur/soustracteur 4 bits dans Logisim et tester son fonctionnement.

QUESTION 2 : En prenant comme paramètres d'entrée les valeurs S_A, S_B et S_R (respectivement le signe du nombre A, le signe du nombre B et le signe du résultat), établissez la table de vérité de la sortie overflow (ovf_o)

QUESTION 3 : A partir de votre table de vérité, déterminez l'équation simplifiée du bit d'overflow et implémentez sa gestion dans votre composant add/sub 4 bits.

NOTE : Pour les questions 2 et 3, merci de **ne pas utiliser** la propriété vue en cours théorique qui permet de calculer l'overflow en fonction de carry_n XOR carry_n-1.

Etape 1-d : Correction du carry en soustraction

Dans le composant `main_4bits`, instanciez votre add/sub 4 bits et trouvez un moyen de corriger le bit de carry en fonction de l'opération (addition ou soustraction).

Etape 1-e : Intégration sur carte Intégrez votre composant `main_4bits` sur la carte MAXV en suivant le mappage suivant :

Nom I/O	**Description**
Nom I/O	Position sur carte
sel_i	SW1
a_i[3 :0]	S7 :S4
b_i[3 :0]	S3 :S0
r_o[3 :0]	L3 :L0
c_o	L7
ovf_o	L6

QUESTION 4 : Une fois votre implémentation intégrée sur la carte, testez cette dernière et remplissez le tableau suivant :

Signé/Non signé	Sel	Nombre A	Nombre B	Résultat	Carry	Overflow
Non signé	0	0010	0011			
Non signé	0	0111	0010			
Non signé	0	1010	0111			
Non signé	1	0100	0011			
Non signé	1	0001	0101			
Non signé	1	1000	0111			
Signé	0	0010	0011			
Signé	0	0111	0010			
Signé	0	1010	0111			
Signé	1	0100	0011			
Signé	1	0001	0101			
Signé	1	1000	0111			

Rendu

Pour ce laboratoire, chaque binôme devra rendre :

— votre fichier *.circ*
— un fichier au format *.pdf* contenant les réponses aux différentes questions théoriques.

Vous devez déposer les rendus sur Cyberlearn jusqu'à la date indiquée dans l'espace de rendu consacré à votre classe.