UNIVERSITE M'Hamed BOUGARA – BOUMERDES
FACULTE DES SCIENCES
DEPARTEMENT D'INFORMATIQUE

Module : Architecture des Ordinateurs
Filière : Licence Informatique
Semestre : 3

Responsable de la matière : H. Mokrani
Responsables des TD/TP : M. Bennai
K. Ould Babaali

TP1 : Réalisation d'un circuit combinatoire
(Additionneur/soustracteur 4 bits)

Objectifs

Le but de ce premier **TP** est de familiariser avec l'outil **Logisim**, et de réaliser un circuit combinatoire d'une manière hiérarchique. Le circuit que vous devez construire est un additionneur/soustracteur **4 bits**.

Pour réaliser un tel circuit vous devez passer par la réalisation d'un demi-additionneur **1 bit**, un additionneur complet **1 bit**, un additionneur **4 bits**, un additionneur **4 bits** avec des indicateurs, et en terminant par une extension de l'additionneur **4 bits** pour la prise en compte de la soustraction.

Nous supposons dans ce **TP** que les nombres représentés sur **4 bits** sont codés en complément à deux (**Cà2**). Nous vous demandons dans ce **TP** de n'utiliser que des portes **NAND** dans la réalisation des circuits.

I. Réalisation d'un demi-additionneur

Ce premier circuit représente la brique de base de notre additionneur **4 bits**, il prend en entrée deux bits **A** et **B** et produit une sortie **S** et un retenue **C**. La figure (Figure 1) montre le schéma des entrées/sorties d'un additionneur complet 1 bit.

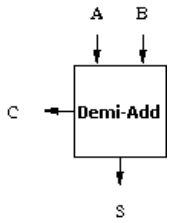

Figure1 : Schéma des entrées sorties d'un demi-additionneur 1 bit.

Dans cette partie, on vous demande de :

a) Calculer la table de vérité d'un demi-additionneur.
b) Réaliser un demi-additionneur (**1 bit**) en se basant sur sa table de vérité. Lors de génération de circuit, cocher l'option en utilisant que des portes **NAND** uniquement (Use **NAND** Gates Only). Nommer le circuit « **Demi_Add_1b** ».
c) Réaliser quelques tests (avec le mode de simulation de Logisim) pour vérifier le bon fonctionnement de votre circuit.

II. Réalisation d'un additionneur complet

Un additionneur complet prend en entrée deux bits **A** et **B** ainsi que la retenue précédente $\mathbf{C_{in}}$. Il calcule la somme **S** de ces trois valeurs binaires ainsi que la retenue $\mathbf{C_{out}}$. La figure (Figure 2) montre les entrées sorties du circuit.

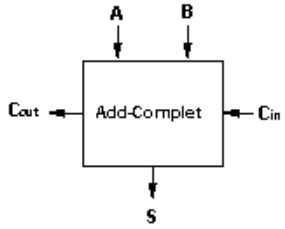

Figure2 : Schéma des entrées/sorties d'un additionneur complet 1 bit.

Dans cette partie, on vous demande de :

a) Réaliser un additionneur complet (**1 bit**) en utilisant le demi-additionneur que vous venez de construire. Nommer le circuit « **add_1b** ». Utiliser que des portes **NAND**.
b) Réaliser quelques tests pour vérifier le bon fonctionnement de votre circuit.

III. Réalisation d'un additionneur 4 bits

La méthode la plus simple pour construire un additionneur **4 bits** et d'utiliser la notion de propagation de retenue. L'idée vient de la méthode d'addition manuelle. Les paires de bits sont additionnés colonne par colonne et les retenues sont propagées vers la gauche. La figure (Figure 3) montre la composition d'un additionneur 4 bits avec le principe de propagation de retenue.

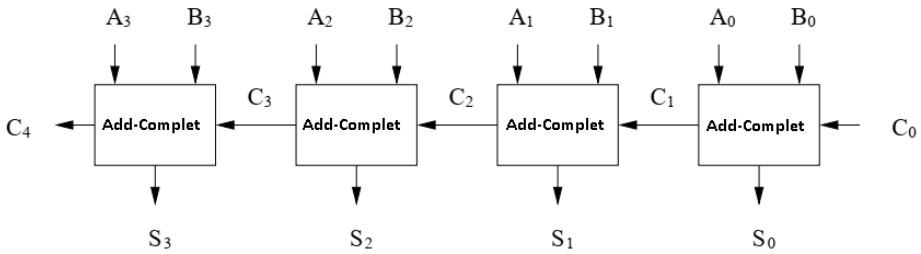

Figure 3 : Schéma de réalisation d'un additionneur 4 bits en se basant sur 4 additionneurs complet 1 bit.

Dans cette partie, on vous demande de :

a) Réaliser un additionneur **4 bits** en utilisant **4** additionneurs complets **1 bit**. Nommer votre circuit « Add_4b ». On met la valeur de $\mathbf{C_0}$ toujours à **0**, vu qu'on n'a pas une retenue à calculer dans la première colonne (le poids faible).
b) Si on additionne **0111** et **0100**, qu'affiche le circuit (simuler le) ? Expliquer le résultat obtenu.

IV. Première extension (Calcul des indicateurs)

Dans une UAL, il existe des indicateurs (Flags). Ces indicateurs permettent de signaler : si le résultat obtenu est nul, si le résultat est négatif, ou si le résultat est erroné (dépassement de capacité). Elle montre même la dernière retenue calculée. Pour réaliser ce fonctionnement, il suffit d'étendre l'additionneur par des circuits, chaque circuit réalise une fonction parmi les quatre.

Dans cette partie, on vous demande de :

a) Réaliser une extension de l'additionneur **4 bits** par un circuit qui vérifier si la sortie de l'additionneur est égale à **0**. Donc, le circuit répond par **1** si la valeur est zéro, **0** sinon. Nommer la sortie de ce circuit « **z** ».
b) Ajouter aussi un circuit qui répond par **1** si un nombre est négatif, **0** si le nombre est positif. Nommer la sortie de ce circuit « **N** ». On suppose que le nombre et écrit en complément à deux (**Cà2**).
c) Ajouter un troisième circuit qui répond par **1** s'il existe un dépassement de capacité. Pour vous aider, pour détecter un déplacement de capacité lorsque les nombres sont codés en (**Cà2**). Il suffit de comparer la valeur de sortie des deux dernières retenues. Si les deux retenues sont différentes alors on a un dépassement de capacité. Nommer la sortie de ce circuit « **O** ».

d) Ajouter un dernier circuit qui affiche la dernière retenue. Nommer la sortie de ce circuit « **C** ».
e) Additionner **0111** et **0100**, qu'affiche le circuit maintenant ?
f) Nommer cette extension de l'additionneur **4 bits** avec les **4** indicateurs « **Add_Ind_4b** ».

V. Deuxième extension (Réalisation d'un additionneur/soustraction)

On peut réaliser des opérations d'addition et de soustraction en utilisant une petite extension sur l'additionneur !!! L'idée est très simple. Pour calculer la différence entre deux valeurs binaires **A – B**, il suffit de réaliser l'opération d'addition de **A** et **(-B)**, donc **A + (-B)**. Autrement dit, on additionne **A** avec le complément à deux (**Cà2**) de **B**.

Rappel : On calcul le complément à deux d'une valeur binaire en inversant les bits et en additionnant 1 à la fin.

Alors, comment peut-on réaliser ce circuit ?

Heureusement que la réalisation de ce circuit est aussi simple que l'idée. Il suffit d'inverser chaque bit et de mettre **1**, pour rajouter le **1**) sur la retenu C_0 à la place de **0**. Autrement dit, C_0 est utiliser pour augmenter la valeur de l'inverse de **B** par **1** et aussi un bit pour détecter si on prend **B** ou **–B**. La figure (Figure 4) montre l'idée de la réalisation de ce circuit.

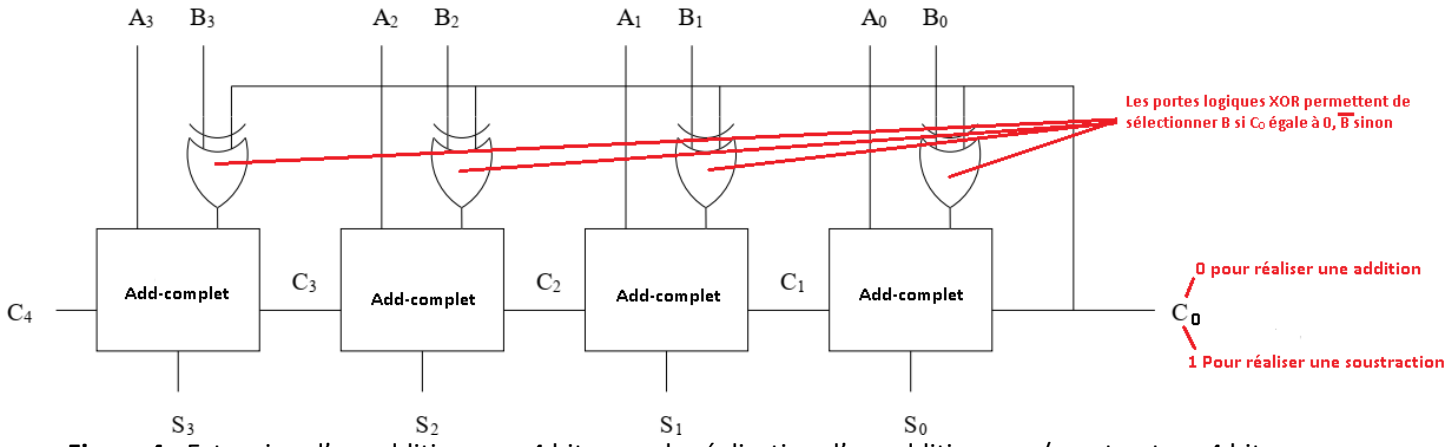

Figure 4 : Extension d'un additionneur 4 bits pour la réalisation d'un additionneur/soustracteur 4 bits.

Dans cette partie, on vous demande de :

a) Réaliser le circuit additionneur/soustracteur. Nommer ce circuit « **Add_Sous_4b** ».
b) Réaliser les opérations suivantes 0011 + 1011, 1011-1100, 1111 + 0001. Qu'affiche le circuit ?

Notes d'information aux étudiants :

1- L'organisation des étudiants est en trinôme.
2- Le travail doit être réalisé en une semaine.
3- Chaque trinôme doit déposer un rapport de réalisation du TP, qui ne dépasse pas les 10 pages.
4- La taille des caractères sur le rapport est (11 pts pour le texte, et 12 pts sur les titres).
5- Le style utilisé doit être : Calibri ou Arial.
6- Le rapport de contenir une : Introduction, Réalisation des étapes de TP et une conclusion.
7- Le copier/coller dans une section engendre une note de 0 dans cette section.
8- Des séances de présentation de votre travail sur le simulateur seront réalisées dans les séances de TP.