Conception d'un Additionneur-soustracteur & implantation sur la maquette DE2

1. PRÉSENTATION.

Nous allons à travers ces séances de TP mener à bien un mini projet consistant à réaliser un additionneur-soustracteur qui sera implanté sur la platine de développement Altera DE2. Cette étude nous permettra de mettre en œuvre les différentes fonctions de logique combinatoire décrites sous forme de logigramme ou bien en vhdl.

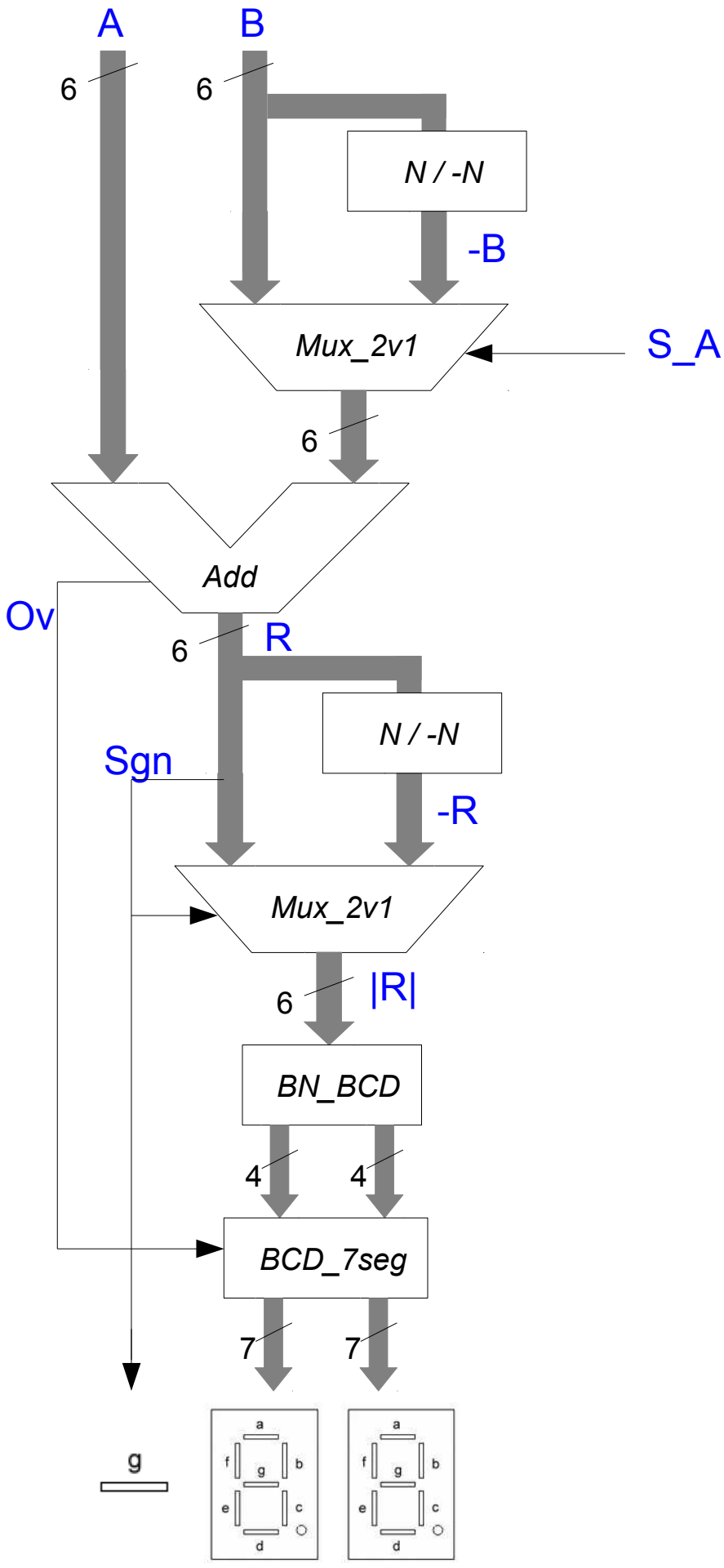
Cahier des charges :

Le système à réaliser devra, à partir de deux valeurs binaires 6 bits codées sur des swicths de la platine, afficher le résultat sur deux afficheurs à 7 segments, correspondant soit à la somme, soit à la différence des deux valeurs binaires. Le choix entre addition ou soustraction se fera par un 13° switch.

Description fonctionnelle :

Le synoptique suivant présente les différentes fonction à mettre en œuvre pour réaliser cet additionneur/soustracteur :

- **A, B :** Valeurs 6 bits à additionner (ou soustraire)
- **S_A :** Choix de l'opération
- **Blocs "N/-N"** : Calcul de l'opposé de la valeur entrante.
- **Bloc "Add"** : Addition des deux valeurs sur 6 bits
- **Blocs "Mux2v1" :** Multiplexeurs de bus (6 bits) 2 vers 1
- **Bloc "BN/BCD" :** Décodeur binaire naturel vers BCD
- **Bloc "BN/7 seg" :** Décodeur BCD vers 7 segments
- **Signal R :** Résultat (6 bits) du bloc additionneur
- **Signal Ov :** Retenue de sortie du bloc additionneur
- **Signal Sgn :** Signe du résultat (+ ou -) à afficher



2. Bloc "N/-N".

Cellule 1 bit

On considère de façon isolée un bit du nombre dont on souhaite calculer l'opposé :

N =

N_5	N_4	N_3	N_2	N_1	N_0

O =

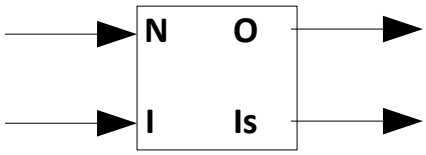
O_5	O_4	O_3	O_2	O_1	O_0

Une méthode pour calculer le l'opposé O du nombre N consiste à dire : "En partant de la droite, je complémente tous les bits du nombre initial N au premier '1' rencontré".

Autrement dit : Pour chaque bit Oi, il prendre le complément du bit Ni s'il y a au moins un bit à '1'de N_0 à N_{i-1}.

La cellule de calcul de l'opposé sur 1 bit comportera donc 2 entrées et 2 sorties :

N : Bit du nombre dont on calcule l'opposé
O : Bit du résultat du calcul
I : Bit indiquant si au moins un des bits précédents est à '1'
Is : Bit de sortie indiquant au rang suivant si au moins un des bits du nombre est à '1' (y compris le rang en cours)



Q1)Complétez les tables de vérités suivantes :

N	I	Is

N	I	O

Q2)En déduire les équations simplifiée des sorties O et Is.

Q3)Proposez un logigramme de réalisation.

Q4)Créez un projet quartus nommé "Calcul_Oppose" dans un répertoire du même nom, ainsi qu'un fichier schématique principal du même nom.

Q5)Créez un second fichier schématique nommé "Cellule" dans lequel vous implanterez les logigramme précédents. Placez les broches d'E/S.

Q6)Créez un symbole à partir de ce fichier (→ "File" → "Create Symbol from current file")

Le symbole correspondant est maintenant créé, et peut être placé sur un schéma. Il apparaît dans le répertoire de votre projet ("project") lorsque vous placerez un composant  .

Q7)Sur le fichier schématique principal, placez 6 symboles "Cellule" correctement reliés entre eux de façon à réaliser un calcul de l'opposé sur 6 bits.

Q8)Affectez les switchs SW5 à SW0 au nombre N, et les leds LEDR5 à LEDR0 au nombre O.

Q9)Compilez votre projet puis créez un fichier de simulation afin de valider son fonctionnement (groupez les entrées et sorties en bus au format décimal signé).

Q10)Chargez votre projet dans le circuit FPGA puis testez son fonctionnement.

3. Bloc "Add".

Cellule 1 bit

On considère de façon isolée un bit de chacun des nombres dont on souhaite calculer la somme :

A =

A_5	A_4	A_3	A_2	A_1	A_0

B =

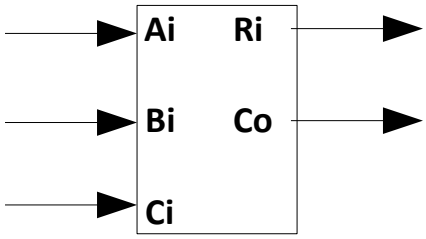
B_5	B_4	B_3	B_2	B_1	B_0

R =

R_5	R_4	R_3	R_2	R_1	R_0

On peut donc définir une cellule additionneuse élémentaire pour un bit de rang i quelconque (0<i<5) :

Ai : Bit du nombre A dont on calcule la somme
Bi : Bit du nombre B dont on calcule la somme
Ci : Bit indiquant une retenue sur l'addition des bits de rang i-1
Ri : Bit de sortie de résultat du rang i
Co : Bit de sortie de retenue du rang i



Q11)Complétez la table de vérité suivante :

Ai	Bi	Ci		Ri	Co

Q12)En déduire les équations simplifiée des sorties Ri et Co.

Q13)Proposez un logigramme de réalisation.

Q14)Créez un projet quartus nommé "Addition" dans un répertoire du même nom, ainsi qu'un fichier schématique principal du même nom.

Q15)Créez un second fichier schématique nommé "Add_1b" dans lequel vous implanterez les logigramme précédents. Placez les broches d'E/S.

Q16)Créez un symbole à partir de ce fichier (→ "File" → "Create Symbol from current file")

Q17)Sur le fichier schématique principal, placez 6 symboles "Add_1b" correctement reliés entre eux de façon à réaliser un calcul de l'opposé sur 6 bits.

Q18)Affectez les switchs SW5 à SW0 au nombre A, les switchs SW15 à SW10 au nombre B et les leds LEDR5 à LEDR0 au nombre R.

Q19)Compilez votre projet puis créez un fichier de simulation afin de valider son fonctionnement (groupez les entrées et sorties en bus au format décimal signé).

Q20)Chargez votre projet dans le circuit FPGA puis testez son fonctionnement.

4. Bloc "Mux_2v1".

Q1)Rappelez le principe de la fonction "multiplexage".

Q2)Créez un projet quartus nommé "MUX" dans un répertoire du même nom, ainsi qu'un fichier schématique principal du même nom.

Q3)En vous aidant du document de prise en main(§4"Utilisation des Megafonctions"), créez un multiplexeur ayant les caractéristiques adaptées à notre projet d'additionneur/soustracteur 6 bits.

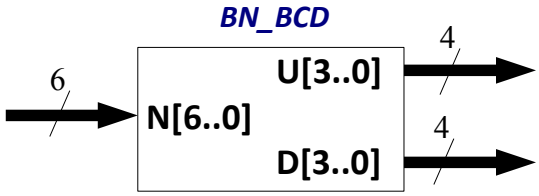
Q4)Affectez 6 switchs à chacun des bus d'entrée, 6 LED's au bus de sortie et 3 autres switchs au signal de sélection.

Q5)Compilez votre projet puis créez un fichier de simulation afin de valider son fonctionnement (groupez les entrées et sorties en bus au format décimal signé).

Q6)Chargez votre projet dans le circuit FPGA puis testez son fonctionnement.

5. Bloc "BN_BCD".

Ce bloc réalise le transcodage suivant : Binaire naturel vers BCD.



N : Nombre à décoder, sur 6 bits (valeurs positives comprises entre 0 et 32)
U : Chiffre des unités du nombre N
D : Chiffre des dizaines du nombre N

Q1)Complétez la table de vérité suivante :

N5	N4	N3	N2	N1	N0		D1	D0	U3	U2	U1	U0

Q2)En déduire les équations simplifiée des sorties U et D par la méthode des tableaux de Karnaugh.

Q3)Proposez un logigramme de réalisation.

Q4)Créez un projet quartus nommé "BN_BCD" dans un répertoire du même nom, ainsi qu'un fichier schématique principal du même nom dans lequel vous implanterez le logigramme précédent.

Q5)Affectez les switchs SW5 à SW0 au nombre N, les leds LEDR1 à LEDR0 à D et LEGG3 à LEDG0 à U.

Q6)Compilez votre projet puis créez un fichier de simulation afin de valider son fonctionnement (groupez les entrées et sorties en bus au format décimal signé).

Chargez votre projet dans le circuit FPGA puis testez son fonctionnement.

6. Bloc "BCD_7seg".

Ce bloc réalise le transcodage suivant : VERS vers "7 segments". Il permet d'afficher unités et dizaines sur deux afficheurs à anodes communes/