INF1500 - laboratoire #2

Affichage alterné d'un nombre hexadécimal en signé et non signé à 4 bits

I- Objectifs

Les objectifs de ce laboratoire sont de décrire et de simuler un circuit de logique combinatoire pour une application numérique. Il permet de pratiquer les étapes de conception d'un circuit numérique. En plus, il permet aussi de mieux comprendre la représentation de nombres binaires positifs et leur représentation en notation hexadécimale. Finalement, on aborde dans ce laboratoire des principes de réutilisation de blocs ainsi que la description hiérarchique d'un design.

II- Travail préparatoire, à réaliser avant le laboratoire

II.1 Principe de l'affichage à 7 segments

On peut représenter sur 4 bits les nombres hexadécimaux non signés allant de 0 à F. Selon cet encodage le "bit 0" a un poids de 1, le "bit 1" a un poids de 2, le "bit 2" a un poids de 4, et le "bit 3" a un poids de 8, tel que décrit dans le tableau 1.

Pour afficher un chiffre, on peut se servir d'un des quatre afficheurs à 7 segments disponibles sur la planchette de développement. Le diagramme de la figure 1, tiré du manuel de l'utilisateur de la planchette de développement, montre les quatre affichages, et spécifiquement comment faire pour afficher le chiffre « 2 » sur l'affichage de gauche.

Tableau 1 Représentation sur 4 bits des nombres non signés

Nombre non Signé	Encodage sur 4 bits (b3b2b1b0)	bit3	bit2	bit1	bit0
0	0000	0	0	0	0
1	0001	0	0	0	1
2	0010	0	0	1	0
3	0011	0	0	1	1
4	0100	0	1	0	0
5	0101	0	1	0	1
6	0110	0	1	1	0
7	0111	0	1	1	1
8	1000	1	0	0	0
9	1001	1	0	0	1
A	1010	1	0	1	0
B	1011	1	0	1	1
C	1100	1	1	0	0
D	1101	1	1	0	1
E	1110	1	1	1	0
F	1111	1	1	1	1

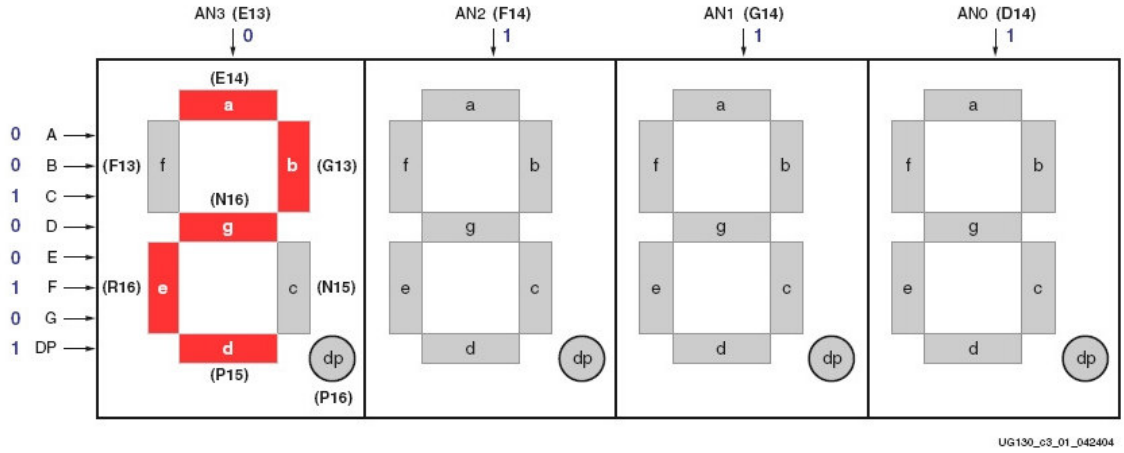


Figure 1: Afficheur à 7 segments

On constate que les quatre afficheurs sont décomposés en 7 segments plus le point (pour un total, en fait, de 8 segments contrôlables). Chacun des segments est connecté à une patte du FPGA. Pour faire allumer un segment particulier, il faut forcer un « 0 » à la patte correspondante, et aussi forcer un « 0 » correspondant à l'anode de l'afficheur désiré. L'affichage fonctionne donc en **logique négative**, puisqu'une tension de 0 V fait allumer le segment. Par exemple, pour afficher le chiffre « 2 » sur l'afficheur de gauche, il faut forcer un « 0 » aux segments a, b, d, e, et g. Les segments c, f, et le point "dp" doivent quant à eux recevoir un « 1 ».

II.2 Travail à réaliser avant le laboratoire

Vous devez tracer le circuit de décodage qui accepte 4 bits en entrée (un bus **bit(3:0)**), correspondant au nombre à afficher (**hexadécimal non signé**), et qui a un bus de 8 bits (**ssd(7:0)**) en sortie: un bit pour chacun des 7 segments et un bit pour le point.

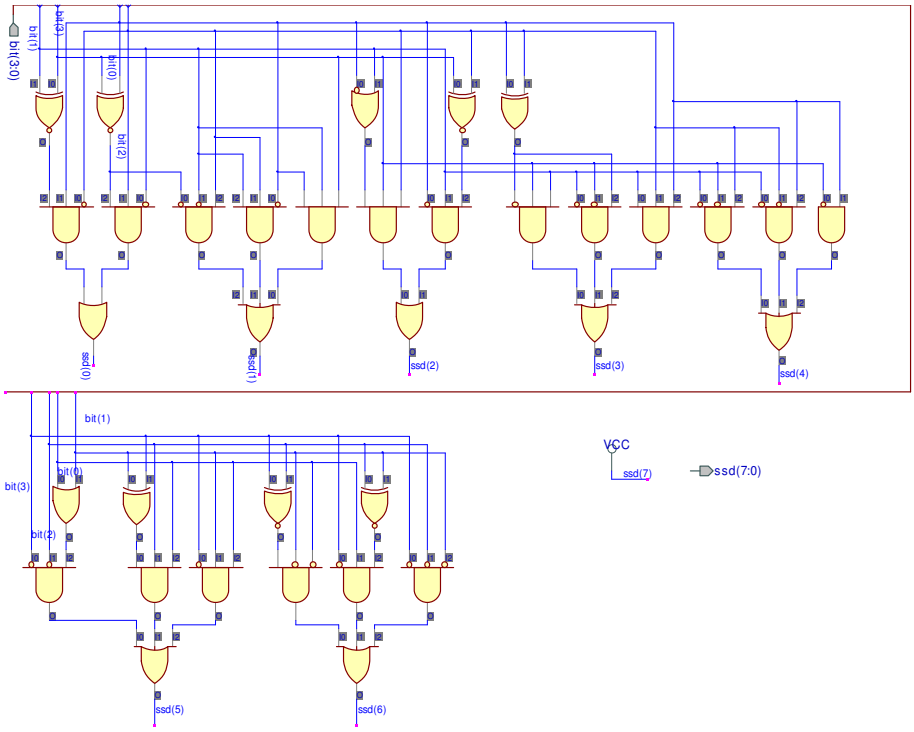
Voici les 10 premiers chiffres représentés avec l'affichage 7 segments ainsi que leur complément en hexadécimal :

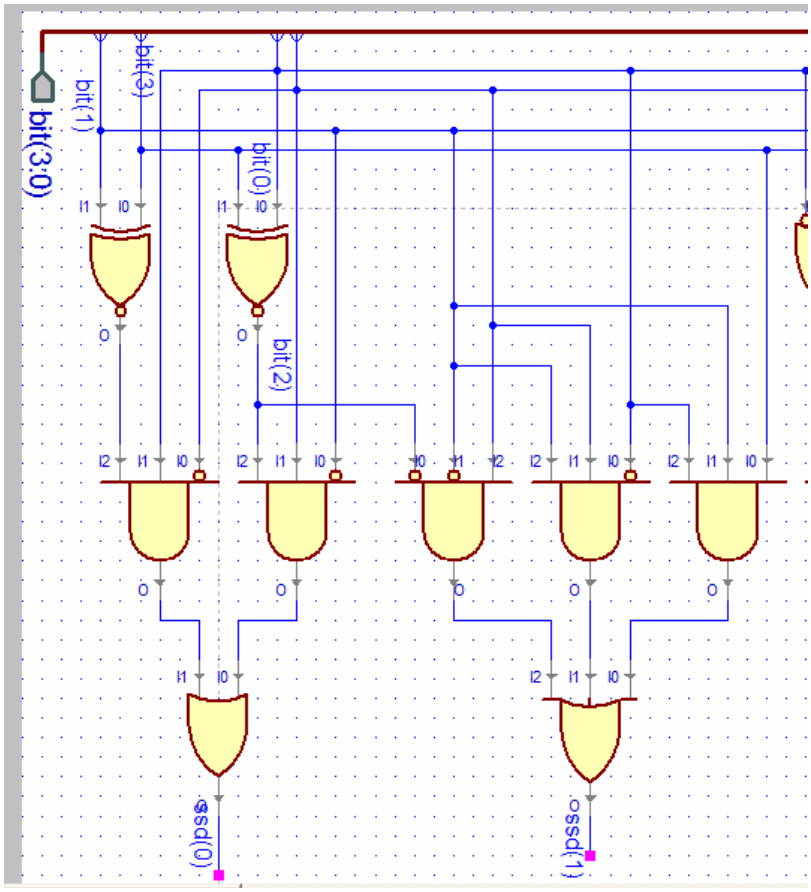
(*sources : http://fr.wikipedia.org/wiki/Afficheur_7_segments*)

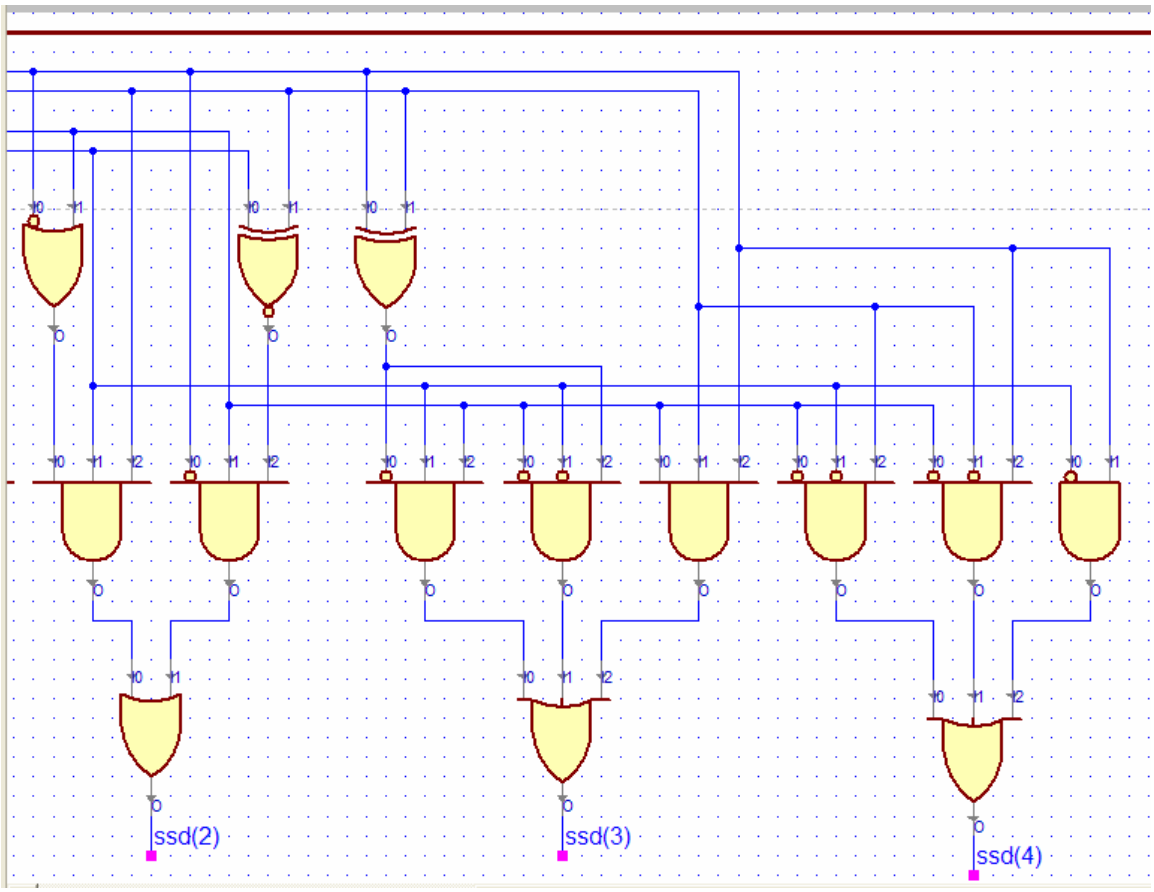
Nombre	**bit3**	**bit2**	**bit1**	**bit0**	segment 'a' **ssd0**	segment 'b' **ssd1**	segment 'c' **ssd2**	segment 'd' **ssd3**	segment 'e' **ssd4**	segment 'f' **ssd5**	segment 'g' **ssd6**	le point 'dp' **ssd7**
0	0	0	0	0								
1	0	0	0	1								
2	0	0	1	0								
3	0	0	1	1								
4	0	1	0	0								
5	0	1	0	1								
6	0	1	1	0								
7	0	1	1	1								
8	1	0	0	0								
9	1	0	0	1								
A	1	0	1	0								
b	1	0	1	1								
C	1	1	0	0								
d	1	1	0	1								
E	1	1	1	0								
F	1	1	1	1								

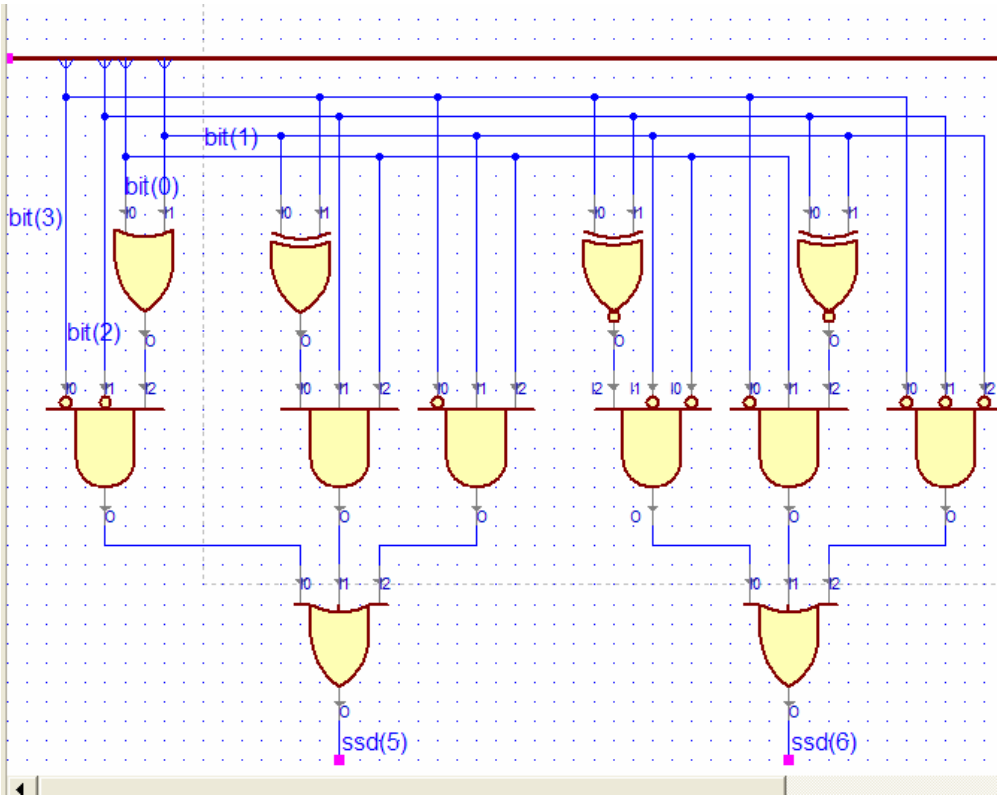
II-3 Circuit

Le circuit permettant le contrôle de l'afficheur à 7 segments est présenté ci-dessous. Vous devez dessinez individuellement le circuit utilisant le logiciel Active-HDL en suivant la même procédure que celle décrite dans le guide pratique v2p (laboratoire 1). La simulation du circuit n'est pas demandé bien qu'elle soit fortement encouragée.


bit(3:0)
bit(1)
bit(3)
bit(0)
bit(2)
ssd(0)
ssd(1)
ssd(2)
ssd(3)
ssd(4)
bit(1)
bit(3)
bit(0)
bit(2)
ssd(5)
ssd(6)
VCC
ssd(7)
ssd(7:0)


bit(3:0)
bit(1)
bit(3)
bit(0)
bit(2)
I0
I1
I2
O
ssd(0)
ssd(1)


ssd(2)
ssd(3)
ssd(4)


bit(3)
bit(2)
bit(1)
bit(0)
ssd(5)
ssd(6)

II.4 Ce que vous devez remettre

Vous devez présenter au chargé de laboratoire au début de la période un rendu avec les éléments suivant (inscrivez dessus votre nom et matricule) :

1- La table de vérité dument remplie.
2- Les équations booléennes sous la forme somme des produits obtenues à partir de la table de vérité correspond a chaque sortie (ssd(7)..ssd(0)).
3- Les équations booléennes obtenues à partir du diagramme des portes logiques.

Vous devez montrer au chargé de laboratoire le dessin du circuit sous active-HDL (ne pas l'imprimer) que vous avez fait au préalable. Vous pouvez le sauvegarder sur une clé USB ou vous l'envoyer par email.

II.4 Circuit d'affichage alterné (signé et non signé)

Pensez à concevoir un circuit permettant de faire un affichage alterné en signé (complément à 2) et non signé. Ce circuit doit avoir en entrée un bus de 4 bits (**bit(3:0)**) qui représente le nombre à afficher et un signal de sélection (**R**) entre les deux représentations. La sortie de ce circuit reste toujours un bus de 8 bits (**ssd(7:0)**) où chaque bit correspond à un segment de l'afficheur à 7 segments.

Sachant que la représentation binaire du nombre signé "-1" (1111) est le complément à 2 de la représentation binaire du nombre non signé "1" (0001), on pourra se servir de l'additionneur-soustracteur 4 bits réalisé dans le premier laboratoire afin de concevoir ce circuit.

N.B.: Vous **n'aurez pas** à présenter ce circuit au début du laboratoire.

III- Procédure à suivre au laboratoire

III.1 Étape 1

Lancer ActiveHDL et dans votre **Workspace** (ou créer une nouvelle "Workspace") ajouter un nouveau **design** (Laboratoire3 par exemple).

*N.B.: **N'utilisez jamais** des espaces dans les noms de vos fichiers ou directoires.*

Créer un nouveau schéma appelé "**Afficheur7Segments**" et entrer le circuit correspondant à vos équations réduites trouvées dans la section "**II.2**" (travail préparatoire).

Pour les sorties, utiliser un bus à 8 bits avec le nom **"ssd(7:0)"**, et assigner les différents segments selon la numérotation montrée au tableau de vérité précédent (i.e., ssd(0) correspond au segment 'a', ssd(1) correspond au segment 'b', etc.).

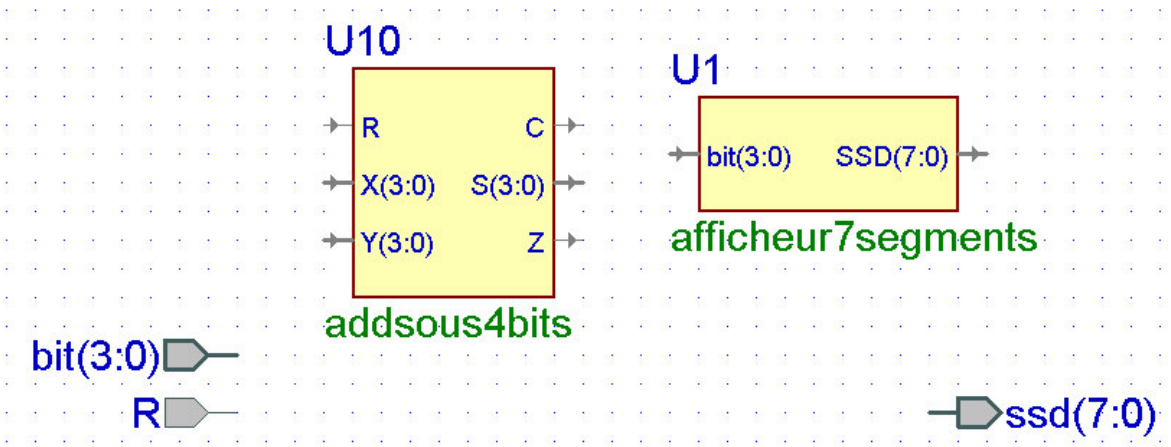
Pour les entrées, utiliser un bus à 4 bits, que vous pouvez appeler **"bit(3:0)"**, où bit(3) correspond au bit3, bit(2) correspond au bit2, etc.

Une fois votre schéma complété, compilez le, un symbole est automatiquement crée dans l'onglet "Symbol Toolbox". Simuler votre circuit pour vous assurer que les sorties correspondent bien à votre tableau de vérité.

III.2 Étape 2

Ajouter le fichier "add1bit.bde" (disponible sur le site de cours) à votre design et Compiler. Ajouter le fichier " addSous4bits.bde" (aussi disponible sur le site de cours) à votre design et Compiler. Un symbole de l'additioneur-soustracteur 4 bits (addSous4bits) est automatiquement crée dans l'onglet "Symbol Toolbox".

Créer un nouveau schéma appelé "**AfficheurComplement2**" permettant de faire l'affichage alterné en signé (complément à 2) et non signé à partir de l'**Afficheur7Segments** (créé dans III.1 ci-haut) et de l'**addSous4bits** et les portes logiques que vous jugez être utiles.



Comme montré dans la figure ci haut, ce circuit doit avoir en entrée un bus de 4 bits (**bit(3:0)**) qui représente le nombre à afficher et un signal de sélection (**R**) entre les deux représentations (signé R = 1 et non signé R = 0). La sortie de ce circuit reste toujours un bus de 8 bits (**ssd(7:0)**) où chaque bit corresponde à un segment de l'afficheur à 7 segments (consulter II.2.2).

Ajouter à votre projet une copie du fichier *XUPV2P_DIO4_UCF.txt*. Enlever les '#' devant les lignes correspondants aux commutateurs SW (Entrées) et aux signaux SSD (Sorties) que vous utilisez.

Synthétiser (*N'oublier pas de spécifier le synthétiseur, l'outil d'implémentation et le type de FPGA dans la fenêtre "Design flow → Flow Settings". Consulter* le *guide pratique V2P page 3*), implémenter, programmer et tester sur la planchette. Vérifier que l'affichage fonctionne correctement pour un seul chiffre.

Démontrer le fonctionnement de votre circuit au chargé de laboratoire.

N.B.: Exemple de **connexion entre deux bus** (ssdin(7:0) et ssd(7:0)):