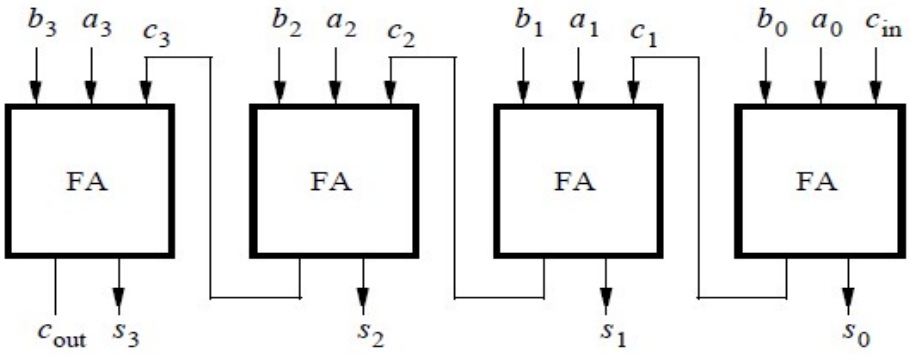
GE34
TP2

Conception d'un additionneur et d'un additionneur/soustracteur sur FPGA

1) On veut écrire le code VHDL d'un circuit qui réalise l'addition à 4 bits. Les deux opérandes A et B sont à 4 bits et la somme, sum, est aussi à 4 bits. Il doit y avoir aussi la sortie cout qui indique le dépassement pour les nombres non signés et ovf qui indique l'overflow pour les nombres signés. La conception de l'additionneur doit se faire en structurelle à base de quatre additionneurs à un bit (fig. 1).



a) Simulation
   i. Créer un nouveau projet Add_4bits dans le dossier C:\GE34\TP2\Add_4bits
   ii. Écrire un fichier VHDL Add_4bits.vhd qui fournit les fonctionnalités nécessaires de l'additionneur à 4 bits. L'entité du design est la suivante

```
library ieee;
use ieee.std_logic_1164.all;
entity add_4bits is
        port(   A, B    :   in    std_logic_vector(3 downto 0);
                sum     :  out  std_logic_vector(3 downto 0);
                cout, ovf       : out std_logic);
end add_4bits
```

   iii. Compiler le circuit et utiliser la simulation fonctionnelle pour vérifier le bon fonctionnement de l'additionneur.

b) Implémentation sur la carte DE1 SOC
   i. Créer un nouveau projet Add_4bits_DE1 dans le dossier C:\GE34\TP2\Add_4bits_DE1

ii. Écrire un fichier VHDL Add_4bits_DE1 qui fournit les fonctionnalités nécessaires de l'additionneur de bits. Utiliser les switches SW7-4 et SW3-0 pour représenter les inputs A et B, respectivement. LEDR3-0 affiche la somme. LEDR(9) et LEDR(7) affichent respectivement ovf et cout.
L'entité du design est la suivante :

```
library ieee;
use ieee.std_logic_1164.all;
entity add_4bits_DE1 is
    port(  SW   : in std_logic_vector(7 downto 0);
           LEDR : out std_logic_vector(9 downto 0));
end add_4bits_DE1;
```

iii. Compiler le circuit et le télécharger sur la carte FPGA. Tester le circuit en manipulant les switches et en vérifiant les ledr.

2) On veut écrire le code VHDL d'un circuit qui réalise l'addition et la soustraction à 4 bits. Les deux opérandes A, B sont à 4 bits et la somme/soustraction, sum, est aussi à 4bits. Il doit y avoir aussi la sortie cout qui indique le dépassement pour les nombres non signés et ovf qui indique l'overflow pour les nombres signés. La conception de l'additionneur/soustracteur doit se faire en structurelle. Le circuit réalise la soustraction si OP=1 et effectue l'addition si OP=0
L'entité du design est la suivante

```
library ieee;
use ieee.std_logic_1164.all;
entity Add_Sub_4bits is
      port( A, B      : in std_logic_vector(3 downto 0);
            OP        : in std_logic;
            Sum       : out std_logic_vector(3 downto 0);
            cout, ovf       : out std_logic);
end Add_Sub_4bits;
```

c) Simulation
   i. Créer un nouveau projet Add_Sub_4bits dans le dossier C:\GE34\TP2\Add_Sub_4bits
   ii. Écrire un fichier VHDL Add_Sub_4bits.vhd qui fournit les fonctionnalités nécessaires de l'additionneur/soustracteur à 4 bits
   iii. Compiler le circuit et utiliser la simulation fonctionnelle pour vérifier le bon fonctionnement de l'additionneur/soustracteur.

d) Implémentation sur la carte DE1 SOC
   i. Créer un nouveau projet Add_sub_4bits_DE1 dans le dossier C:\GE34\TP2\Add_Sub_4bits_DE1
   ii. Écrire un fichier VHDL Add_Sub_4bits_DE1 qui fournit les fonctionnalités nécessaires de l'additionneur/soustracteur de 4 bits. Utiliser

les switches SW7-4 et SW3-0 pour représenter les inputs A et B, respectivement. SW9 précise la valeur de OP. LEDR3-0 affiche la somme. LEDR(9) et LEDR(7) affichent respectivement ovf et cout.

L'entité du design est la suivante :

```
library ieee;
use ieee.std_logic_1164.all;
entity Add_Sub_4bits_DE1 is
    port( SW: in std_logic_vector(7 downto 0);
                    LEDR: out std_logic_vector(9 downto 0));
end Add_Sub_4bits_DE1;
```

iii. Compiler le circuit et le télécharger sur la carte FPGA. Tester le circuit en manipulant les switches et en vérifiant les ledr.