**INF1500 – logique des systèmes numériques**

**Laboratoire 2 :**

**Initiation à Vivado**

**Simulation et implémentation d’un circuit en logique combinatoire**

**Groupe : 02**

**Présenté par :**

**Nathan Ramsay-Vejlens (1989944)**

**Louis Dutheil (1994257)**

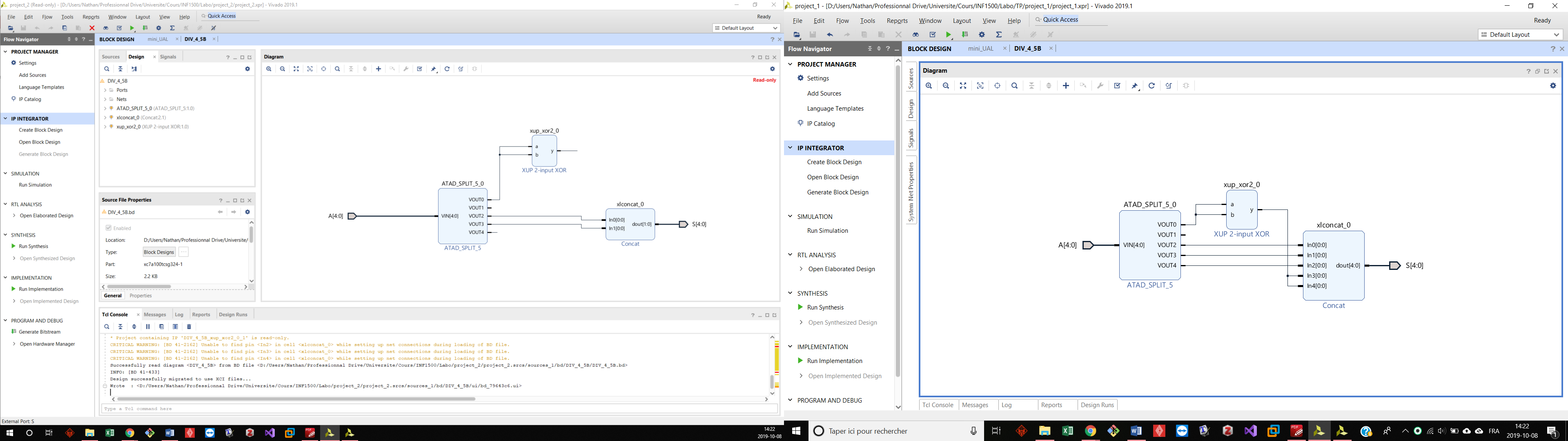
**Octobre 2019**

**Département de génie informatique et de génie logiciel**

**École Polytechnique de Montréal**

**Diviseur par 4**

**Design :**



**Table de vérité diviseur par 4 :**

|  |  |  |
| --- | --- | --- |
| A | S | Justification |
| 11111 | 00111 | 31/4=7 puisqu’on ne veut que des valeurs entières, pas de décimales. |
| 00000 | 00000 | Valeur limite |
| 10101 | 00101 | 21/4=5, valeur entière toujours |
| 00001 | 00000 | ¼=0, valeur entière seulement |

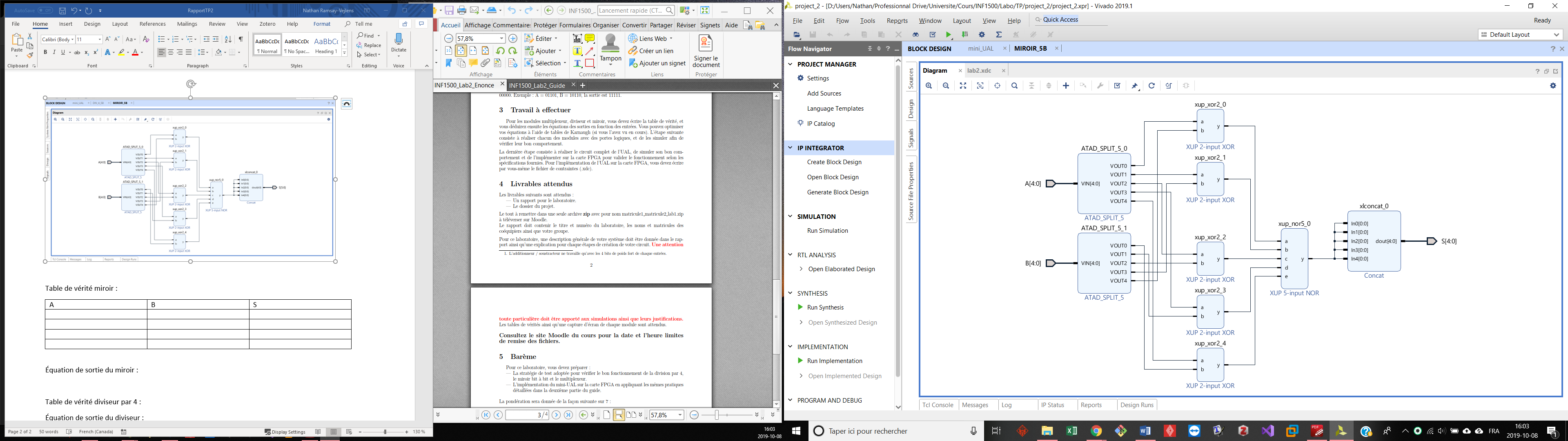
**Équation de sortie du diviseur :** S0=A2 S1=A3 S2=A4 S3=0 S4=0

**Description générale :**

Nous avons simplement décalé les valeurs des bits de A de 2 cases en utilisant un SPLIT et un CONCAT, ce faisant nous divisions par 4 la valeur. Les deux bits supérieurs du résultat ne pouvant être égale à autre chose que 0, ils ont été mis à 0 à l’aide d’un XOR avec une des deux valeurs inférieures de A. Les deux valeurs inférieures de A n’étant pas importantes puisqu’elles ne peuvent donner un résultat supérieur à 3 et les décimales ne sont pas prises en compte dans cette division car on arrondit à la valeur inférieure.

**Miroir 5B**

**Design :**



**Table de vérité miroir :**

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | S | Justification |
| 11111 | 11111 | 11111 | Valeur limite |
| 01101 | 10110 | 11111 | Exemple du TP |
| 10110 | 10110 | 00000 | Même valeur, mais pas miroir doit donner 0. |
| 00000 | 00000 | 11111 | Valeur limite |

**Équation de sortie du miroir :** S1=S2=S3=S4=S5=((A1⊕B5)+(A2⊕B4)+(A3⊕B3)+(A4⊕B2)+(A5⊕B1))’

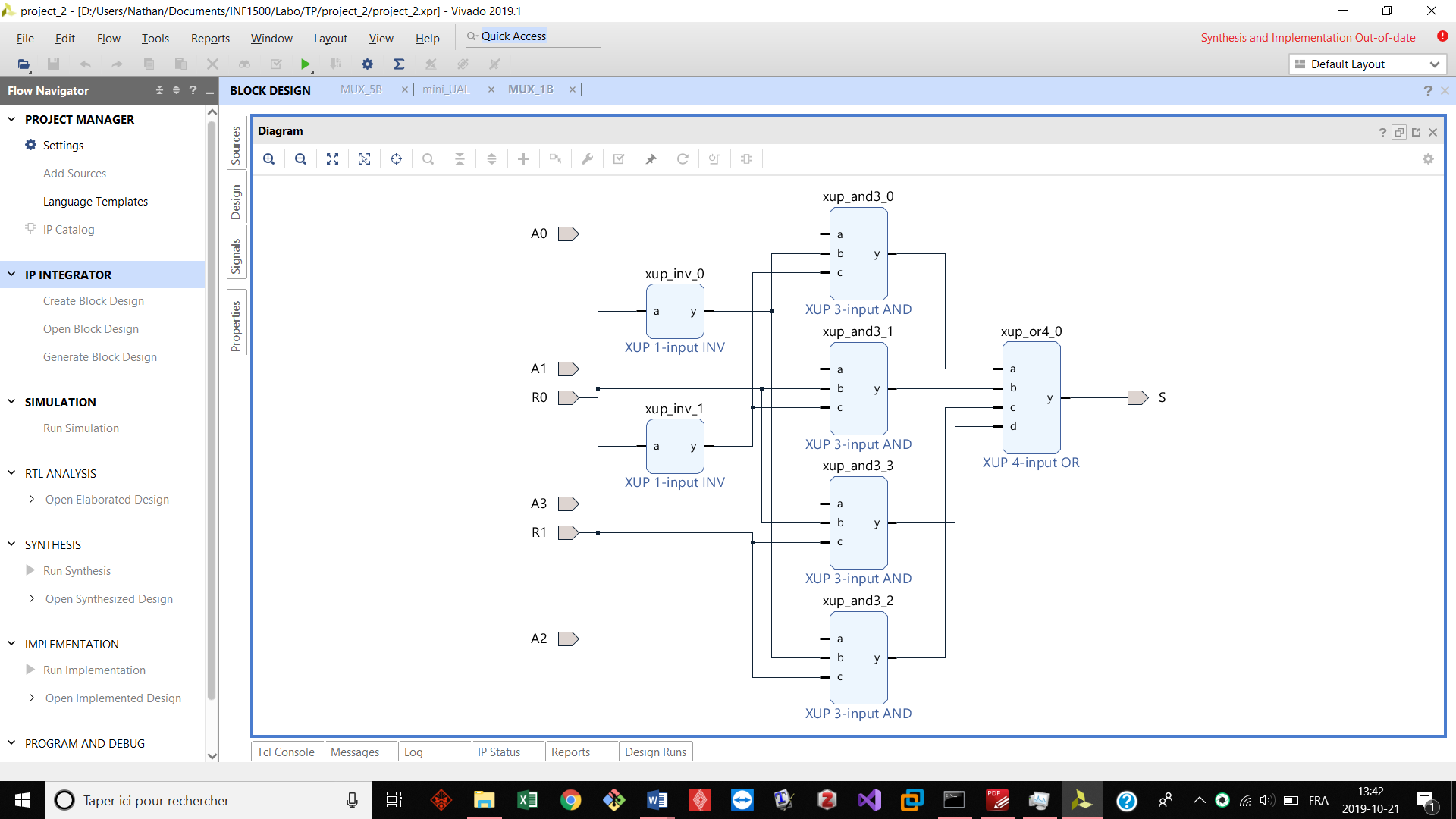
**Description générale :**

Pour le miroir, nous avons commencé par SPLIT les vecteurs en 5 bits. Par la suite, l’utilisation de plusieurs XOR à 2 entrées, nous a permis de vérifier la correspondance des bits. Le 5ième bit de A étant vérifié avec le 1er de B. Ensuite, le 4ième bit de A avec le 2ième de B ainsi de suite. Finalement les 5 résultats de XOR ont été comparés dans un NOR. Il fallait donc que les 2 bits comparés soit identique afin de sortir 0. Si c’était bien le cas alors le NOR sortait 1 qui était alors concaténé 5 fois afin de donner un vecteur de 5 bits.

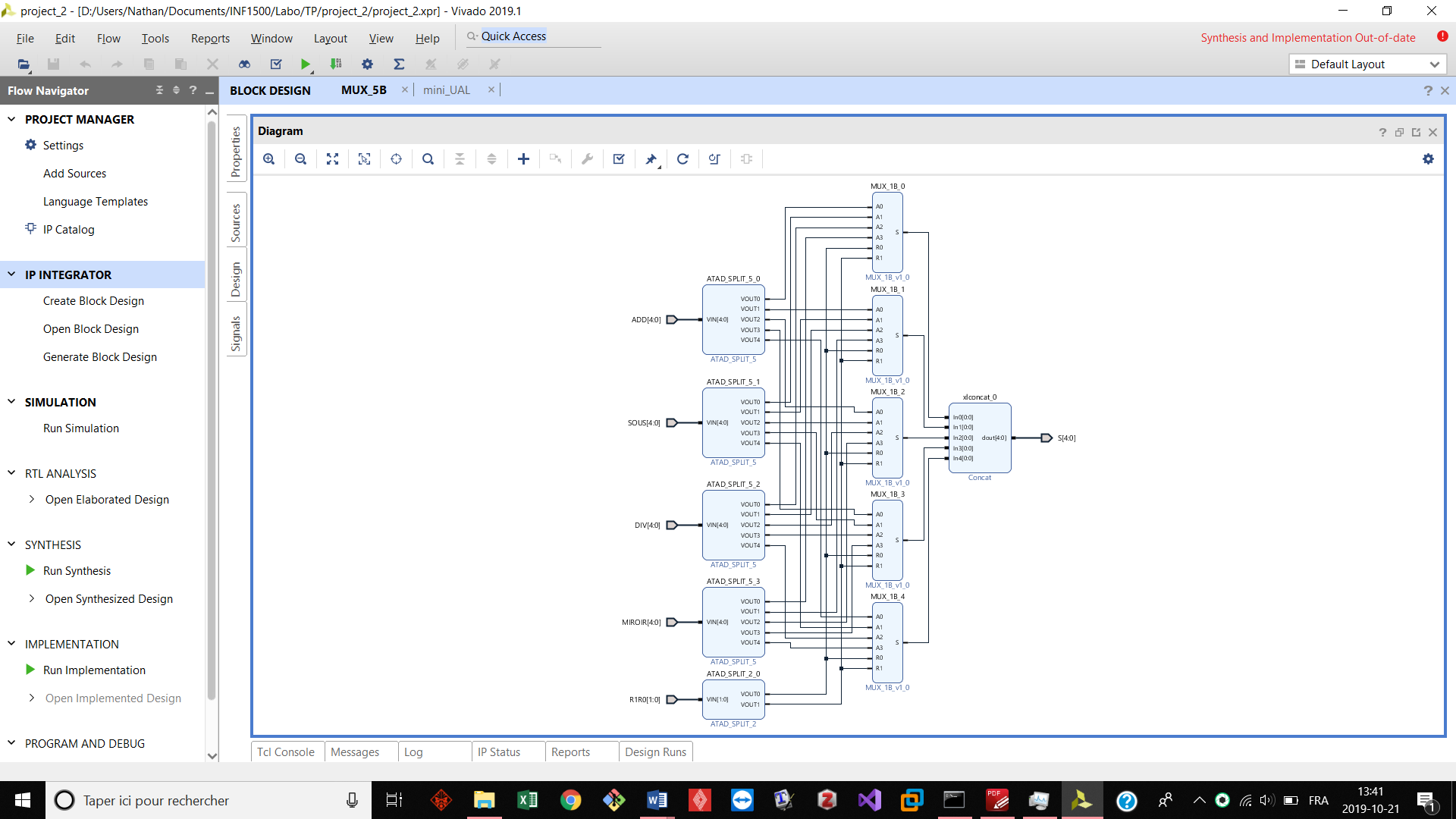
**MUX**

**Design :**

1Bits :



5Bits :



**Table de vérité MUX :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| R1R0 | ADD | SOUS | DIV | MIROIR | S | Justification |
| 00 | 1 | 0 | 0 | 0 | 1 | Addition (R0=R1=0) : 1 |
| 00 | 0 | 1 | 1 | 1 | 0 | Addition (R0=R1=0) : 0 |
| 01 | 0 | 1 | 0 | 0 | 1 | Soustraction (R0=1, R1=0): 1 |
| 01 | 1 | 0 | 1 | 1 | 0 | Soustraction (R0=1, R1=0): 0 |
| 10 | 0 | 0 | 1 | 0 | 1 | Division (R0=0, R1=1): 1 |
| 10 | 1 | 1 | 0 | 1 | 0 | Division (R0=0, R1=1): 0 |
| 11 | 0 | 0 | 0 | 1 | 1 | Miroir (R0=1, R1=1): 1 |
| 11 | 1 | 1 | 1 | 0 | 0 | Miroir (R0=1, R1=1): 0 |

**Équation de sortie du multiplexeur :**

(Addition)S1=R0’R1’

(Soustraction) S2=R0R1’

(Division)S3=R0’R1

(Miroir) S4=R0R1

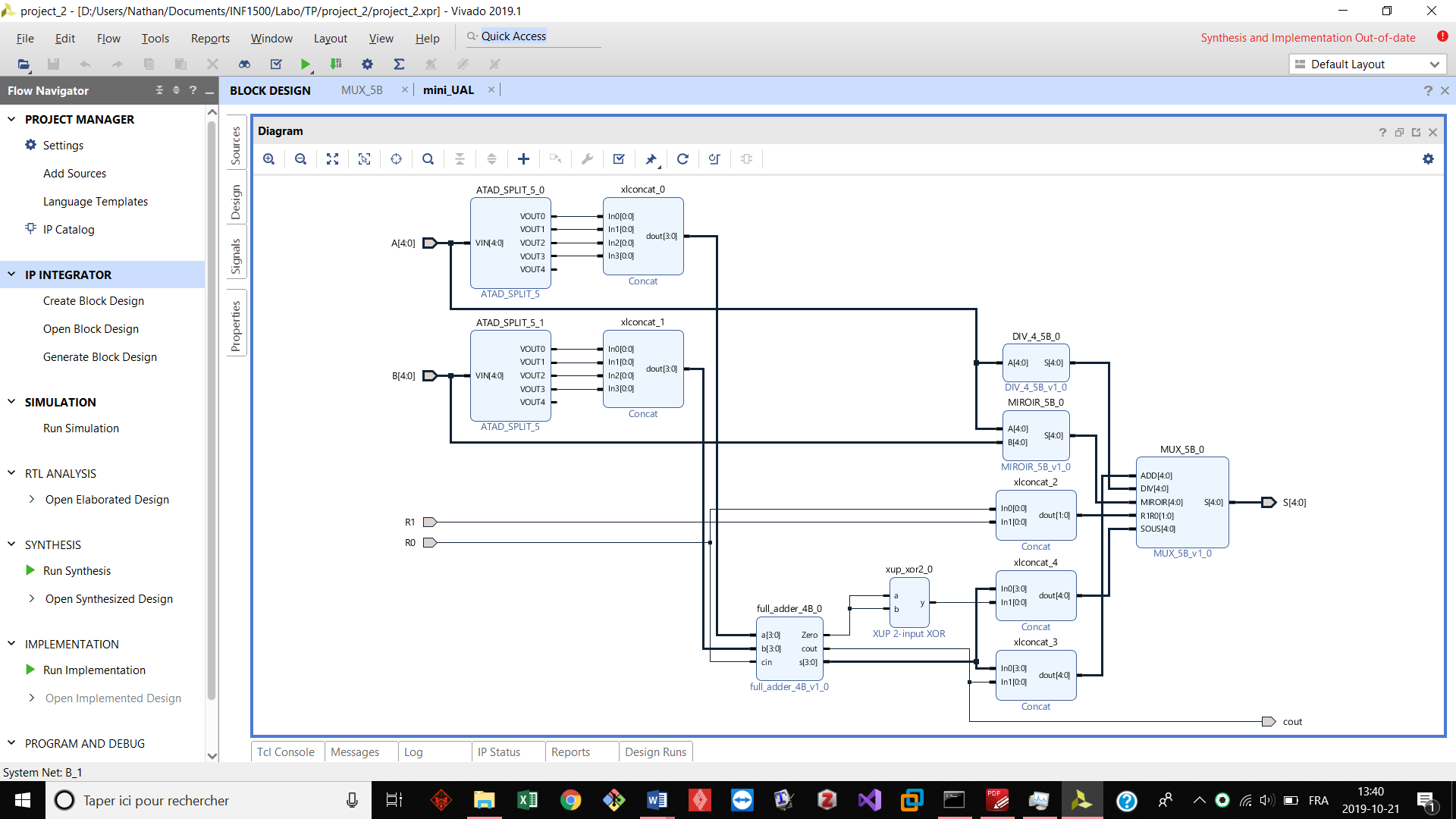
**Description générale du circuit :**

Pour créer un MUX à 1 bit avec 4 entrées autre que les sélecteurs, nous avons utilisé 4 portes ET prenant en entrée les 2 valeurs du sélecteur (inversé ou pas selon le cas) ainsi que la valeur passée. Par la suite, le résultat des portes ET est passé à une porte OU.

Pour créer le MUX à 5 bits, nous avons utilisé plusieurs MUX à 1 bit dans lesquels nous avons passés les valeurs des 4 vecteurs entrés. Il faut diviser les vecteurs et concaténer les valeurs des portes ET à la fin. Nous avons aussi séparés la valeur du vecteur R1R0.

**mini\_UAL**

**Design :**



**Table de vérité mini\_UAL:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A | B | R0 | R1 | S | Cout | Justification |
| 1 1111(1F) | 1 0101(15) | 0 | 0 | 1 0100(14) | 1 | Addition (ne pas tenir compte du cout) des 4 premiers bits(R0=R1=0) : 1111+0101=1 0100 (14) |
| 1 1111 (1F) | 1 1111(1F) | 0 | 0 | 1 1110(1e) | 1 | 1111+1111=1 1110 (1e) |
| 0 0000 | 0 0000 | 0 | 0 | 0 0000 | 0 | 0000+0000=0 0000 |
| 1 1111(1F) | 0 0101(5) | 0 | 1 | 0 1010(0a) | 0 | Soustraction (cout indique négatif) des 4 premiers bits (R0=1, R1=0): 1111-0101=1010 |
| 0 0101(5) | 1 1111(1F) | 0 | 1 | 0 0110(06) | 1 | 5-15(f)=-10  0110 (6) en complément à 2 donne 1010 (10), négatif parce que cout=1 |
| 0 0101(5) | 0 0101(5) | 0 | 1 | 0 0000 | 0 | 5-5=0 |
| 0 0101(5) | 0 0111(7) | 1 | 0 | 0 0001 | 0 | Division par 4 (R0=0,R1=1): 5/4=1.25 ,mais nous ne tenons pas compte des décimales donc 1 |
| 1 1111 | 0 0101(5) | 1 | 0 | 0 0111(7) | 1 | 31/4=7.75, donc 7.  (Ne pas tenir compte du cout parce qu’il est relié à l’addition des 2 termes et non la division) |
| 0 0000 | 0 1010(5) | 1 | 0 | 0 0000 | 0 | 0/4=0 |
| 0 0000 | 0 1010(5) | 1 | 1 | 0 0000 | 0 | Miroir (R0=R1=1): Les 2 termes ne sont pas miroir donc, 0 0000. |
| 0 1101(C) | 1 0110(16) | 1 | 1 | 1 1111 | 0 | Les termes sont miroir alors 1 1111 |
| 0 1101(C) | 0 1101(C) | 1 | 1 | 0 0000 | 0 | Les termes sont identiques, mais pas miroir alors 0 0000 |

**Description générale du circuit :**

Pour utiliser l’additionneur/soustracteur, il faut d’abord transformer le chiffre binaire à 5 bits en un chiffre à 4 bits, c’est à quoi servent les deux SPLIT et CONCAT au début. Par la suite, une addition ou soustraction est exécuté selon que la valeur de R0 soit 0 ou 1. Si c’est une soustraction, la sortie est ensuite concaténée avec un 0 à l’endroit du bit le plus significatif. Ce qui est exécuté à l’aide du XOR sur la valeur de ZERO. Si c’est une addition, le cout qui est l’équivalent de la retenue est ajouté en tant que bit le plus significatif. Finalement le chiffre concaténé est passé en valeur au a (pour l’addition) et au b (pour la soustraction) du MUX. C’est alors que le R1 et R0 entrent en compte pour choisir quelle valeur est passée. Si R0=1, alors c’est une soustraction qui a eu lieu et si R0=0, c’est une addition. Si R1=0 alors ce sont les valeurs de sortie de l’additionneur/soustracteur qui sont passées dans le MUX. Si le R1=1 et le R0=1, c’est la sortie de miroir qui est passée et si le R0=0, c’est la division par 4 qui sera passé.