

الجمهورية الجزائرية الديمقراطية الشعبية

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وَزَارَةُ التَّعْلِيمِ العَالِي وَالْبَحْثِ العِلْمِي

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE  
SCIENTIFIQUE

جَامِعَةُ عَمَّارِ تَلِيدِجِي بِالْأَغْوَاظِ

UNIVERSITE AMAR TELIDJI LAGHOUAT



كلية العلوم

FACULTE DES SCIENCES

DEPARTEMENT DE MATHÉMATIQUES ET INFORMATIQUE

PROJET DE FIN D'ETUDE

(LICENCE)

**Domaine** : Mathématiques et Informatique

**Filière** : Informatique

**Option** : Systèmes et Réseaux Informatiques

**Thème**

---

**Conception et Implémentation d'un unité arithmétique et logique UAL 4 bit avec langage VERILOG et éventuellement transposition sur un FPGA**

---

**Sous l'encadrement du :**

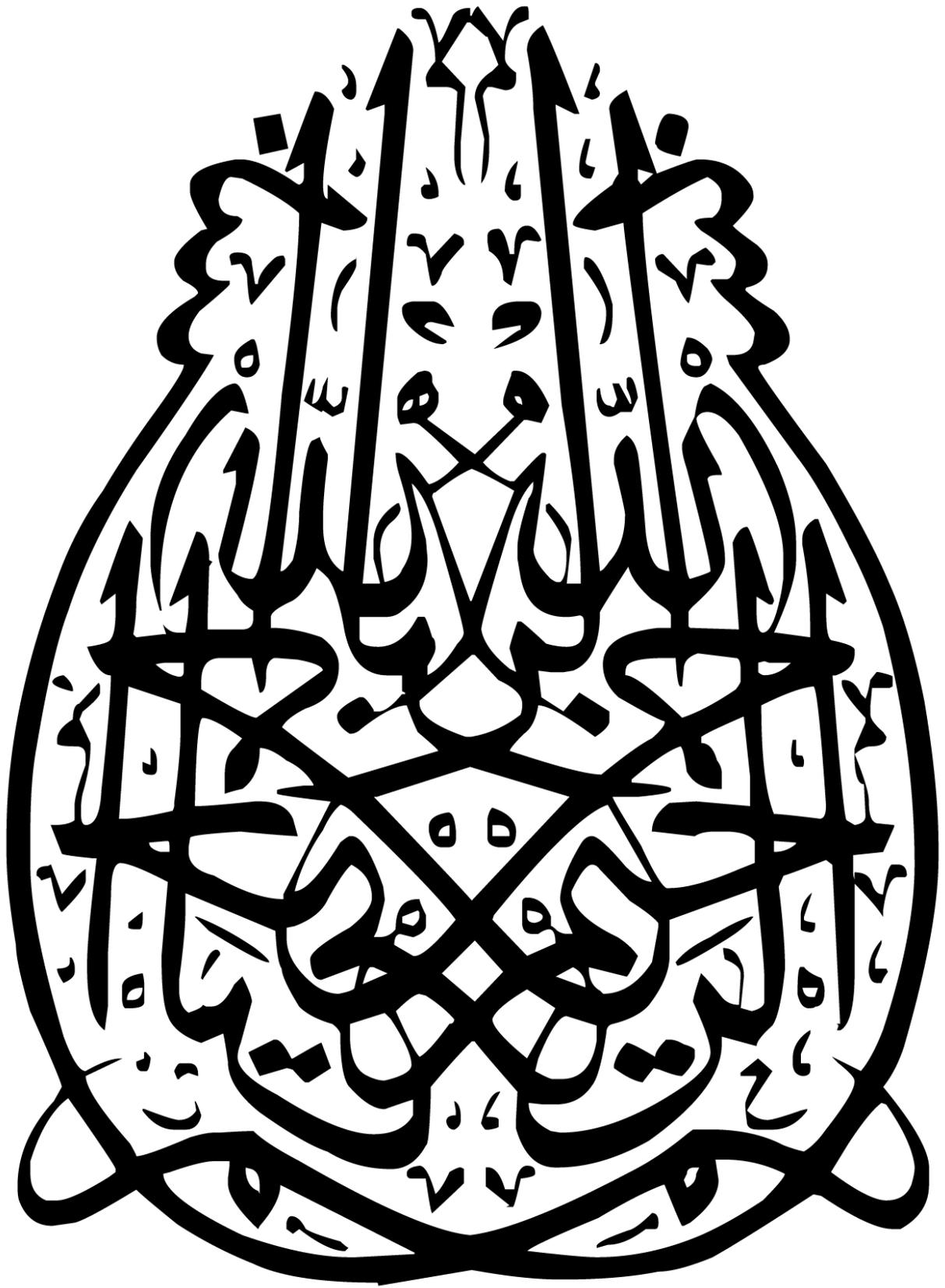
**Hamini Nardjis**

**Par :**

**Bait Marwa**

**Taibi Messaouda Nadjia**

Année Universitaire 2018/2019



# *Dédicaces*

*Ce modeste travail est dédié :*

*A mes chers parents, pour leur soutien, leur éducation ainsi  
leur amour.*

*A tous mes proches de la famille BAIT, et plus  
particulièrement, mes sœurs Khadîdja, Imen et mes frères  
Mohamed, Siradj.*

*A tous mes chères amies proches ou loin et mes collègues de  
l'Université de Laghouat.*

*A tous ceux qui ont sacrifié leur temps pour la science et à  
tous ceux qui utilisent la science pour le bien et la prospérité  
de l'humanité.*

*Marwa*

## *Dédicaces*

*Je dédie ce Mémoire  
A mes très chers parents, dont l'incommensurable  
contribution à mon éducation,  
A mes chères sœurs et chers frères,  
A toute ma chère famille,  
A mes professeurs,  
A mes chers amis et surtout Marwa,  
A tous ceux qui m'aiment,  
A tous ceux que j'aime,  
A tous la promotion 3ème année LMD 2018/2019.*

*Messaouda Nadjia*



## *Remerciement*

*Tout d'abord, nous remercions le Dieu, notre créature de nous avoir donné la force, la volonté et le courage afin d'accomplir ce modeste travail.*

*Ces remerciements vont tout d'abord au corps professoral et administratif de la*

*Faculté Mathématique et Informatique, pour la richesse et la qualité de leur enseignement et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.*

*Nos remerciements vont à notre encadreur Ms HAMINI Nardjes, pour toute sa gentillesse, pour ses précieux conseils et pour sa patience avec nous, ainsi tous ceux qui nous ont aidée et soutenue dans notre travail.*

*Nous remercions vivement les examinateurs, d'avoir accepté l'examen de notre mémoire de fin d'étude et le temps qui l'on réservé pour nous.*

*Remerciements à tous ce qui ont contribué de loin ou de près à l'élaboration de ce mémoire.*

*Finalement, nous tenons à exprimer notre profonde gratitude à nos familles qui nous ont toujours soutenues et à tout ce qui nous ont aider a la réalisation de ce mémoire.*

*Ainsi que l'ensemble des enseignants qui ont contribué à notre formation.*

# Résumé

Le but du projet de fin d'étude est la mise en œuvre du circuit électronique unité arithmétique et logique dans le circuit programmable FPGA nexus 2. Cette FPGA peut être programmée par deux langages de programmation différents VHDL et VERILOG. La méthodologie de conception est la suivante :

Conception des quelques opérations de l'unité arithmétique et logique par LOGISIM.

Conversion les instructions des opérations par langage verilog.

Compilation du programme VERILOG par ISE.

Implémentation du programme VERILOG sur l'FPGA Nexus 2.

**Mots clés :** verilog, FPGA nexus2, unité arithmétique et logique, ISE .

## ملخص

الهدف من مشروع نهاية الدراسة هو تنفيذ الدائرة الإلكترونية للوحدة الحسابية والمنطقية UAL في الدائرة القابلة للبرمجة FPGA nexus2 . حيث يمكن برمجة هذه الدارة بواسطة لغتي برمجة مختلفتين هما VERILOG و VHDL , منهجية التصميم تكون على النحو التالي:

تصميم معظم العمليات لوحدة الحساب UAL بواسطة Logisim.

برمجة العمليات بواسطة VERILOG

تنفيذ برنامج VERILOG بواسطة ISE

تطبيق برنامج VERILOG في شريحة FPGA Nexus2

# Table des matières

Dédicaces	v
Remerciements	iii
Résumé	iv
Table des matières	v
Table des figures	vii
Liste des tableaux	vii
Liste des équations	ix
Les abréviations	ix
<b>Introduction générale</b>	<b>1</b>

## Chapitre 01 : Généralité

1	Introduction	3
2	Définition d'ordinateur	3
3	Histoire de l'ordinateur	3
3.1	Préhistorique	3
3.2	La première génération	4
3.3	La Deuxième génération	4
3.4	La troisième génération	4
3.5	Quatrième génération	5
3.6	Cinquième génération	5
4	L'architecture d'un ordinateur	5
4.1	Architecture de von Neumann	5
4.1.1	Définition	5
4.1.2	Caractéristiques	5
4.2	L'architecture de Harvard	6
4.2.1	Définition	6
4.2.2	Caractéristiques	6
5	Le transistor	6
5.1	Les transistors bipolaires :	7
5.2	Les transistors unipolaires :	7
6	Les micro puce	7
6.1	Définition	7
7	Le processeur (CPU)	8

8	L'unité arithmétique et logique (UAL) .....	8
9	Les étapes d'exécution de programme .....	8
10	Le PLD (Programmable Logic Device).....	9
10.1	SPLDs (Simple Programmable Logic Devices) .....	10
10.1.1	PROM :.....	10
10.1.2	PLA (Programmable Logic Array).....	10
10.1.3	PAL (Programmable Array Logic).....	10
10.1.4	Circuits GAL (Generic Array Logic): .....	10
10.2	HCPLD (High Capacity Programmable Logic Device) .....	10
10.2.1	Les CPLD( Complex Programmable Logic Devices ) .....	10
10.2.2	Les FPGA .....	10

## **Chapitre 02 : conception et implémentation**

11	Les FPGA .....	10
11.1	Définition .....	10
11.2	Architecture de FPGA.....	11
12	Langage .....	12
12.1	Langage Verilog.....	12
12.2	Langage VHDL.....	12
13	Conclusion .....	12
1	Introduction .....	15
2	Détaille de UAL .....	15
3	Détaille des opérations choisit.....	15
3.1	Addition .....	16
3.1.1	Demi-additionneur.....	16
3.1.2	Additionneur complet 1bit.....	17
3.1.3	Additionneur 4 bit.....	19
3.2	Soustraction.....	19
3.2.1	Demi-soustracteur 1 bits .....	20
3.2.2	Additionneur-soustracteur 4bit .....	21
4	Complément à 1 .....	22
5	Complément à 2 .....	23
6	OR .....	23
7	XOR .....	24
8	AND .....	24

9	NOR .....	25
10	Implémentation.....	26
10.1	Le code.....	26
10.2	La liaison avec les ports du FPGA.....	27
10.3	Fichier .bit.....	27
10.4	Configuration de la carte Nexys2 avec ADEPT .....	27
11	Démonstration .....	29
12	Conclusion.....	29
	<b>Conclusion générale.....</b>	<b>28</b>

## Table des Figures

Figure 1.1: Pascaline.....	3
Figure 1.2: la machine à 8 chiffres.....	3
Figure 1.3: ENIAC.....	4
Figure 1.4: Différents transistors .....	4
Figure 1.5: 1 <sup>ère</sup> imprimante laser .....	4
Figure 1.6: l'ordinateur pomme (Apple) .....	5
Figure 1.7: Apple .....	5
Figure 1.8 : Schématisation de l'architecture de von Neumann .....	6
Figure 1.9: Schématisation de l'architecture de Harvard.....	6
Figure 1.10: transistor .....	7
Figure 1.11: Exécution d'une instruction.....	9
Figure 1.12: L'architecture d'un FPGA.....	11
Figure 1.13: FPGA Nexys2.....	12
Figure 2.1: Schéma ALU 4 bits.....	15
Figure 2. 2: Schéma du demi additionneur fait avec le simulateur Logisim.....	17
Figure 2. 3: Schéma de add 1 bit.....	19
Figure 2. 4: Schéma du add 4 bit .....	19
Figure 2. 5: Schéma du Demi-soustracteur.....	21
Figure 2. 6: Schéma de l'Additionneur-soustracteur 4bits.....	22
Figure 2. 7: Schéma du Complément a un.....	23
Figure 2. 8: Schéma du Complément a 2 .....	23
Figure 2. 9: Schéma de OR .....	24
Figure 2. 10: Schéma du XOR.....	24
Figure 2. 11: Schéma de AND .....	25
Figure 2. 12: Schéma du NOR .....	25
Figure 2. 13: Schéma du ALU 4 BIT.....	26
Figure 2. 14: Le code Verilog .....	26
Figure 2. 15: Le code de la liaison .....	27
Figure 2. 16: Fichier .bit .....	27
Figure 2. 17: Configuration sur ISE.....	28
Figure 2. 18: Implémentation sur la carte .....	28
Figure 2. 19: Exemple de soustraction sur une FPGA .....	29

## Liste des tableaux

Tableau 1: Tableau de sélection d'opération .....	16
Tableau 2: Table De Vérité Additionneur Complet.....	18
Tableau 3: Table De Vérité Demi-soustracteur .....	20

# Liste des Equation

Équation 1: Expression Carry .....	18
Équation 2: Expression Sum .....	18
Équation 3: Expression de résultat.....	20
Équation 4: Expression de retenu .....	21

## Les abréviations

- UAL** : unité arithmétique et logique.
- ENIAC**: Electronic Numerical Integrator Analyser and Calculator.
- UC** : Unité de Contrôle.
- ASIC** : Application Specific Integrated Circuit.
- SVA** : signe et valeur absolue.
- HDL**: Hardware Description Langage.
- FPGA**: Field-Programmable Gate Array.
- PLD** : Programmable Logic Device.
- SPLDs**: Simple Programmable Logic Devices.
- PLA**: Programmable Logic Array.
- PAL**: Programmable Array Logic.
- GAL**: Generic Array Logic.
- HCPLD**: High Capacity Programmable Logic Device.
- CPLD** : Complex Programmable Logic Devices

# Introduction générale

L'informatique représente la révolution la plus importante et la plus innovante qui a marqué la vie de l'humanité ces dernières décennies. En effet, loin d'être un éphémère phénomène de mode, ou une tendance passagère, l'informatique vient nous apporter de multiples comforts à notre mode de vie. Aucun domaine n'est resté étranger à cette stratégie qui offre tant de services aussi bien pour l'enseignement ou l'administration que pour le personnel et c'est dans ce cadre d'idées que s'inscrit notre projet de fin d'études. Le rôle de l'informatique est de résoudre des problèmes à l'aide d'un ordinateur. Un problème s'exprime sous la forme d'un énoncé qui spécifie les fonctions que l'on souhaite réaliser.

L'ordinateur est une machine électronique permettant le traitement des données. Ce terme provient du latin ordinaire (« mettre en ordre »). un ordinateur est formé par un ensemble de circuits intégrés et d'autres composants connexes, qui permettent d'exécuter une variété de séquences ou de routines d'instructions indiquées par l'utilisateur. Ces séquences sont systématisées en fonction d'une grande variété d'applications pratiques et déterminées, au sein d'un processus connu sous le nom de programmation.

Pour qu'il marche/fonctionne, l'ordinateur a besoin de programmes informatiques (logiciels) qui fournissent des données spécifiques, nécessaires pour le traitement de l'information. Une fois obtenue l'information souhaitée, celle-ci peut être utilisée internement ou transférée à un autre ordinateur ou dispositif électronique.

Cette architecture conçoit quatre sections principales dans un ordinateur : l'unité arithmétique et logique (Arithmetic Logic Unit – UAL), l'unité de contrôle, la mémoire (une séquence de cellules de stockage numérotées, où chacune est une unité d'information connue sous le nom de bit ou chiffre binaire) et les dispositifs d'entrée-sortie. Toutes ces parties sont reliées par un groupe de câbles, soit les bus.

Les circuits électroniques plus complexes sont ceux qui sont inclus dans les chipsets des microprocesseurs modernes, lesquels contiennent une UAL assez importante. Chaque microprocesseur peut avoir de multiples noyaux et ceux-là, à leur tour, peuvent avoir de multiples unités d'exécution (chacune d'entre elles ayant plusieurs UAL).

L'UAL (Unité d'Arithmétique et de Logique) est l'élément de base d'un microprocesseur. Comme son nom l'indique, son rôle est la réalisation d'opérations arithmétiques (additions, soustractions...), logiques (OR, AND, NOR...), mais aussi de décalage, et de transfert. Sous sa forme la plus simple, elle possède une entrée reliée au bus de données du microprocesseur, une autre reliée à un registre interne, et une sortie reliée au même registre et au bus de données. Elle possède également un bus destiné à la sélection des opérations à réaliser.

## Problématique

Le processeur ainsi que les microcontrôleurs, une fois sortis d'usine ne peuvent plus être modifiés et le nombre d'UAL par exemple ne peut pas être augmenté. Certaines applications ont besoin d'une grande vitesse de calcul par exemple les applications en temps réel.

Pour répondre aux besoins du temps réel, d'un grand nombre d'applications de traitement du signal et de commande numérique, des solutions d'implémentations matérielles sur des Plateformes reconfigurables de type FPGA sont de plus en plus utilisées. Les FPGA présentent beaucoup de perspectives pour l'implantation d'algorithmes en temps réel. En plus, les outils de conception assistée par ordinateur servent à passer directement d'une description fonctionnelle (comme le VHDL) à un schéma en porte logique prêt à implémenter sur FPGA

## **Pourquoi nous avons choisi ce sujet ?**

Nous avons choisi ce sujet pour les raisons suivantes :

Les raisons subjectives :

- Le désir personnel de traiter un tel sujet.
- L'enrichissement des connaissances sur FPGA
- Apprendre à programmer des circuits.

Les raisons méthodologiques :

- La disponibilité des références autour de ce sujet.

Les raisons objectives :

- La nouveauté du sujet.
- La propagation des FPGA.

## **Structure du mémoire :**

Premièrement, on a commencé par une introduction générale où nous avons montré la Problématique, puis nous avons introduit deux chapitres qui sont les suivants :

### **Chapitre 01 : Généralité.**

Dans ce chapitre les différentes conceptions liées à l'ordinateur ainsi que les composants d'un processeur dont l'unité arithmétique et logique et défini les notions que nous allons utiliser dans notre application.

### **Chapitre 02 : Implémentation et conception.**

Dans ce chapitre nous allons voir comment concevoir et réaliser une UAL avec le langage VHDL et tester sur une carte FPGA.

Finalement, on se terminera avec une conclusion générale et quelques perspectives intéressantes concernant ce travail.

# **Chapitre N°1**

## **Généralités**

## 1 Introduction

Au cours des quinze dernières années, les méthodes de conception des fonctions numériques ont subi une évolution importante. Dans les années soixante-dix, la majorité des applications de la logique câblée étaient construites autour de circuits intégrés standard, parallèlement,

Les premiers circuits programmables par l'utilisateur (ASICs) pour les fonctions complexes fabriquées en grande série. La complexité de ces derniers a nécessité la création d'outils logiciels de haut niveau qui sont à la description structurelle (schémas au niveau des portes élémentaires)

Aujourd'hui, l'avènement des dernières générations d'FPGAs a permis de mettre la technologie SOC à la portée d'un public nettement plus large. Ceci est particulièrement depuis que les FPGA sont proposés à un prix très faible et raisonnable. Ce prodigieux essor a été rendu possible grâce aux progrès concernant les technologies de fabrication des transistors et les méthodes de conception assistée le rôle des FPGA est d'intégrer des circuits logiques complexes. Ces circuits sont susceptibles d'être reconfigurés (Architecture programmée modifiable) partiellement ou entièrement.

Dans le cadre de ce chapitre, nous allons définir quelques généralités portant sur les méthodes et les outils qui nous permettent la modélisation de notre projet.

## 2 Définition d'ordinateur

- Est un équipement informatique ; il permet de traiter des informations selon des séquences d'instructions prédéfinies ou programmes. Il interagit avec l'environnement grâce à des périphériques (écran, clavier, modem...).
- Un ordinateur est un ensemble de circuits électroniques permettant de manipuler des données sous forme binaire, ou bits.

## 3 Histoire de l'ordinateur

L'ordinateur moderne est passé par bien des phases avant d'être ce qu'il est aujourd'hui

### 3.1 Préhistorique

Les ordinateurs mécaniques

En 1623 William Schichard inventa la première machine mécanique à calculer

- En 1642, Blaise Pascal créa le premier calculateur arithmétique baptisée « Pascaline ».

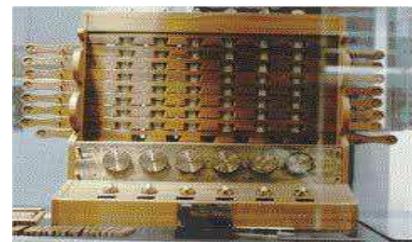


Figure 1.1 : Pascaline.

- En 1834, Charles Babbage invente la machine à différence qui permet d'évaluer des fonctions.



Figure 1.2: la machine à 8 chiffres.

### 3.2 La première génération

Fabuleuse épopée marquée notamment par :

- Tubes à vide.
- Premiers calculateurs électroniques.
- Volumineux.
- Logiciel.
- Utilisation des carte perforées.

ENIAC (Electronic Numerical Integrator Analyser and Calculator) -1945

- Technologie des tubes à vide (18000) ... 30 tonnes
- Construit à l'Université de Pennsylvanie
- Multiplication de 2 nombres de 10 chiffres en 3ms.



Figure 1.3: ENIAC.

1958 er ordinateur commercial entièrement transistorisé par S. Cary.

### 3.3 La Deuxième génération

Un bon en avant grâce à la d »couverte du transistor.

✓ Composants : transistors, mémoire à tores de ferrite, imprimantes, bandes magnétiques.

✓ Logiciels : apparition des systèmes d'exploitation,

langages évolués : Formula Translator (1957)

Common Business Oriented Language (1959)

Fortran et COBOL

✓ Apparition de l'industrie, IBM.

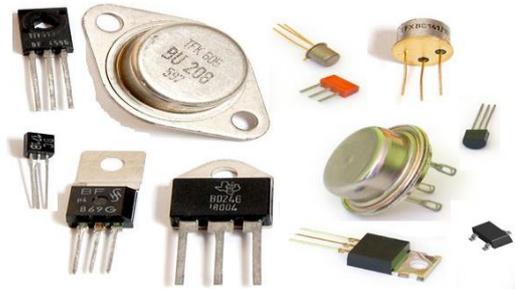


Figure 1.4: Défèrent transistors

DEC, HP.

### 3.4 La troisième génération

Les premiers circuits intégrés voient le jour

✓ Composants :

-circuits intègres, SSI (Small Scale Integration) puis MSI (Medium Scale Integration),

✓ Machine :



Figure 1.5: 1ère imprimante laser

- Faible consommation énergétique,
- Encombrement réduit.
- ✓ Evolution :
  - Multiprocesseur,
  - Temps partagé,
  - Accès interactif,
  - Apparition des réseaux
  - Premiers problèmes de compatibilité



Figure 1.6: l'ordinateur pomme (Apple)

### 3.5 Quatrième génération

- ✓ Caractérisée principalement par la notion de réseaux de machines.
- ✓ Circuits LSI (Large Scale Integration)
- ✓ Stagnation de la sémantique des jeux d'instructions et à la complexité grandissante des langages de programmation

Le 1er *micro*-ordinateur grand public

Processeur 6502.

16 Ko de Rom.

4 Ko de Ram.

Carte graphique couleur.

Clavier : manettes de jeu.

Écran : basic intégré (Figure06).



Figure 1.7: Apple

### 3.6 Cinquième génération

- ✓ Celle des systèmes distribués interactifs. Ce fut en son début, la génération de machines langages dédiées à l'intelligence artificielle
- ✓ Niveau d'intégration VLSI (Very Large Scale Integration)  
Voire même de WSI (Wafer Scale Integration),
- ✓ Le retour à des jeux d'instructions beaucoup plus simples (architecture RISC : Reduced Instruction Set Computers)
- ✓ La génération des architectures à parallélisme massif (Plusieurs milliers ou millions de processeurs élémentaires).

## 4 Architecture d'un ordinateur

### 4.1 Architecture de von Neumann

#### 4.1.1 Définition

Dite architecture de von Neumann, est un modèle pour un ordinateur qui utilise une structure de stockage unique pour conserver à la fois les instructions et les données requises ou générées par le calcul. De telles machines sont aussi connues sous le nom d'ordinateurs à programme stocké en mémoire. La séparation entre le stockage et le processeur est implicite dans ce modèle.

#### 4.1.2 Caractéristiques

Le Modèle pour concevoir et construire des ordinateurs se base sur les trois caractéristiques suivantes :

- ✓ L'architecture de von Neumann décompose l'ordinateur en quatre parties distinctes :

Mémoire : qui contient à la fois les données et le programme qui dira à l'unité de contrôle quels calculs faire sur ces données. La mémoire se divise entre mémoire volatile (programmes et données en cours de fonctionnement) et mémoire permanente (programmes et données de base de la machine).

UAL : Unité Arithmétique et Logique (ALU : Arithmetic/Logic Unit) son rôle est d'effectuer les opérations de base, un peu comme le ferait une calculatrice

UC : Unité de Contrôle (Control Unit) : C'est l'équivalent des doigts qui actionneraient la calculatrice

E/S : Système d'Entrée Sortie(E/S) (Input/Output System (I/O)); dispositifs qui permettent de communiquer avec le monde extérieur

- ✓ Le programme est sauvegardé dans la mémoire durant l'exécution.
- ✓ Les instructions d'un programme sont exécutées séquentiellement.

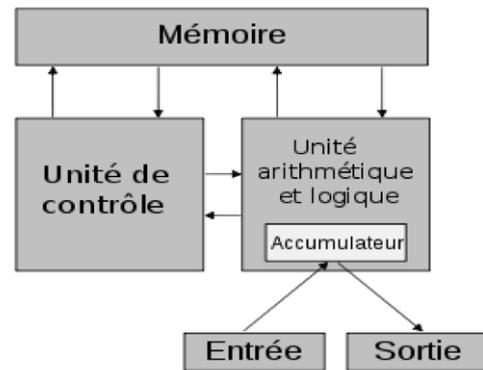


Figure1.8 : Schématisation de l'architecture de von Neumann

## 4.2 L'architecture de Harvard

### 4.2.1 Définition

L'architecture de Harvard est une conception des processeurs qui sépare physiquement la mémoire de données et la mémoire programme. L'accès à chacune des deux mémoires s'effectue via deux bus distincts.

### 4.2.2 Caractéristiques

Avec deux bus distincts, l'architecture dite de Harvard permet de transférer simultanément les données et les instructions à exécuter. Ainsi, l'unité de traitement aura accès simultanément à l'instruction et aux données associées. Cette architecture peut se montrer plus rapide à technologie identique que l'architecture de von Neumann ; le gain en performance s'obtient cependant au prix d'une complexité accrue de structure

architecture de type von Neumann s'oppose à celle de Harvard car elle utilise une unique structure pour stocker à la fois le programme et les données

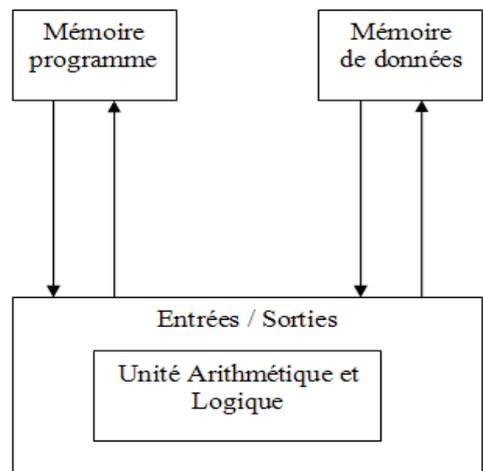


Figure1.9:Schématisation de l'architecture de Harvard

## 5 Le transistor

Un transistor est un dispositif électronique à base de semi-conducteur. Son principe de fonctionnement est basé sur deux jonctions PN, l'une en direct et l'autre en inverse. La

polarisation de la jonction PN inverse par un faible courant électrique (parfois appelé effet transistor) permet de « commander » un courant beaucoup plus important.

Le transistor a été inventé en 1948 par les physiciens Américains J. Bardeen, W. Shockley et W. Brattain, chercheurs à la compagnie Bell Téléphone.

Un transistor est constitué de 2 jonctions PN (ou diodes) montées en sens inverse. Selon le sens de montage de ces diodes on obtient 2 types de transistors :



On distingue plusieurs types de transistors :

### 5.1 Les transistors bipolaires :

Met en jeu le déplacement des deux types de porteurs (électrons et trous)

Exemple : le transistor à jonctions

### 5.2 Les transistors unipolaires :

Met en jeu le déplacement d'un seul type de porteurs (électrons ou trous)

Exemple : les transistors à effet de champ (FET)

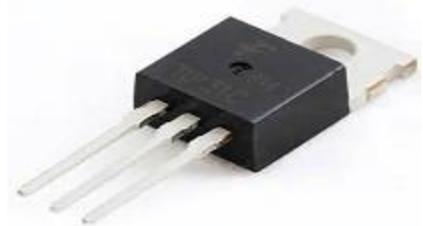


Figure 1.10: transistor

## 6 Les micro puce

### 6.1 Définition

La conception des micropuces est un concept simple avec une explication scientifique complexe.

Une micropuce est également appelée un circuit intégré (IC), qui est souvent raccourci à une simple puce. Il existe un certain nombre de types de puces, y compris les microprocesseurs, les puces de mémoire,

Puces analogiques et numériques et circuits intégrés spécifiques à une application. Une puce ou un circuit intégré, commence par un élément appelé silicium. Grâce à un processus minutieux impliquant plusieurs phases, ce matériau semi-conducteur finit par devenir un ensemble de circuits électroniques sur une petite plaque (puce).

Les micropuces peuvent héberger des milliards de transistors et de composants électroniques pour alimenter nos automobiles, ordinateurs, téléphones cellulaires, appareils numériques et des milliers d'autres produits. Toutes les quelques années, à mesure que la technologie avance, ces puces continuent de réduire leur taille, de nanomètres. Les micropuces sont généralement conçues dans les grandes usines de fabrication, plus communément appelées usines de fabrication. Ces usines disposent de suffisamment de main-d'œuvre et d'équipements pour concevoir et développer des micropuces, y compris plusieurs processus propres, délicats et longs. Les salles blanches sont conçues pour éviter toute contamination de l'environnement. La poussière ou toute substance indésirable peut entraîner une défaillance des produits finis. Le sable est d'abord fondu et raffiné pour produire des lingots de silicium monocristallin presque purs. Les lingots sont sciés en plaquettes épaisses avant d'être

nettoyés. Ensuite, une couche non conductrice de dioxyde de silicium est déposée à la surface de la tranche. La couche est recouverte d'un produit chimique photosensible appelé résine photosensible. Le photoresist est exposé à une lumière ultraviolette qui traverse une "plaque" ou un "masque" qui durcit ensuite les zones exposées à la lumière. Les gaz chauds éliminent les zones non exposées et révèlent la base de dioxyde de silicium en dessous. La base et la couche de silicium sont ensuite gravées à certaines profondeurs. Le photoresist est enlevé. La puce est laissée avec une base en trois dimensions qui répète la conception du circuit du masque. Des produits chimiques sous chaleur et pression peuvent être utilisés pour modifier les copeaux. Ces processus, décrits comme la photolithographie, la gravure et le dopage, peuvent être répétés à maintes reprises sur la même puce, afin de produire des circuits intégrés plus complexes. Ensuite, il faut créer un comportement entre les gravures sur la puce. La puce est recouverte d'une couche de métal tel que l'aluminium. Le processus de lithographie et de gravure est répété pour éliminer toutes les voies de construction, sauf les plus minces. Des couches de conducteurs séparées par des isolants en verre sont souvent utilisées. Les puces sur la plaquette de silicium sont testées pour leur qualité et leurs performances. Si trop de dés sur la plaquette échouent, la plaquette sera mise au rebut. Une fois qu'une plaquette est testée, son épaisseur est souvent réduite grâce à un processus appelé rectification, avant d'être classée, divisée en matrices individuelles (découpage en dés), et préparée pour l'emballage et l'expédition.

## **7 Le processeur (CPU)**

Le processeur (CPU, pour Central Processing Unit, soit Unité Centrale de Traitement) est le cerveau de l'ordinateur. C'est qui organise les échanges de données entre les différents composants (Disque dur, mémoire RAM, carte graphique) et qui fait les calculs qui font que l'ordinateur interagit avec vous et affiche votre système à l'écran. Il permet de manipuler des informations numériques, c'est-à-dire des informations codées sous forme binaire, et d'exécuter les instructions stockées en mémoire.

## **8 L'unité arithmétique et logique (UAL)**

Ce module est chargé de l'exécution de tous les calculs que peut réaliser le microprocesseur. Cette unité est constituée de l'ensemble des circuits arithmétiques et logiques permettant au processeur d'effectuer les opérations élémentaires nécessaires à l'exécution des instructions machine. Elle inclut donc les circuits d'addition, de soustraction, de multiplication, de comparaison, etc. Dans ce module se trouvent également des registres dont l'objet est de contenir les données sur lesquelles vont porter les opérations à effectuer, l'UAL possède deux registres d'entrée (E1 et E2) et un registre de sortie (S).

## **9 Les étapes d'exécution de programme**

L'objet de cette partie est de présenter succinctement comment s'exécute un programme machine sur le matériel que nous venons de définir. Pour être exécutable une instruction doit nécessairement être présente en mémoire centrale. La mémoire centrale contient donc des instructions et des données. De plus toutes les informations en mémoire sont codées sur un alphabet binaire. Les informations sont alors, quelle que soit leur nature, des suites de 0 et de 1. Il faut donc pouvoir différencier instructions et données afin que le registre instruction RI contienne bien des instructions et non des données (et réciproquement). Dans le cas contraire le

décodeur ne pourrait interpréter la nature du travail à faire (RI ne contenant pas une instruction mais une donnée). En général lors du placement du programme machine et des données dans la mémoire centrale les instructions et les données sont séparées et occupent des espaces mémoires différents. À la fin du chargement du programme machine et des données en mémoire le compteur ordinal CO reçoit l'adresse de la première instruction du programme à exécuter.

L'exécution peut alors commencer. Le principe général d'exécution est illustré dans la figure. Les différentes phases de l'exécution d'une instruction sont les suivantes :

1. le contenu du compteur ordinal CO est placé dans le registre d'adresses RAD : il y a sélection de l'instruction à exécuter ;
2. une commande de lecture de la mémoire centrale est déclenchée via le bus de commandes
3. l'instruction est transférée de la mémoire centrale vers le registre instruction RI via le bus de données et le registre de données RDO ;
4. le décodeur analyse l'instruction placée dans le registre instruction RI, reconnaît cette instruction et indique au séquenceur la nature de l'instruction ;
5. le séquenceur déclenche au rythme de l'horloge la séquence de micro-instructions nécessaires à la réalisation de l'instruction.

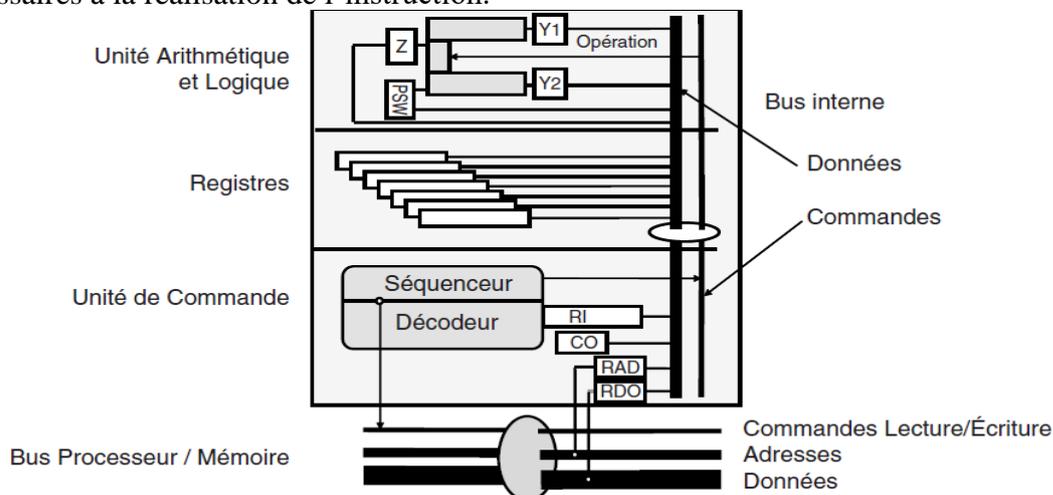


Figure 1.11: Exécution d'une instruction.

## 10 Le PLD (Programmable Logic Device)

Un circuit logique programmable (Programmable Logic Device ou PLD) est un dispositif qui peut être configuré par l'utilisateur pour réaliser une fonction logique quelconque.

Le circuit logique programmable est réalisé comme un assemblage de matrices programmables de portes AND et OR.

## 10.1 SPLDs (Simple Programmable Logic Devices)

### 10.1.1 PROM :

Les premiers circuits programmables apparus sur le marché sont les PROM bipolaires à fusibles. Cette mémoire est l'association d'un réseau de ET fixes, réalisant le décodage d'adresse, et d'un réseau de OU programmables,

### 10.1.2 PLA (Programmable Logic Array)

PLA FPD relativement petit qui contient deux niveaux de logique programmable : un plan « ET » et un plan « OU ». Bien que Les structures PLA sont parfois intégrées à des puces entièrement personnalisées.

### 10.1.3 PAL (Programmable Array Logic)

FPD relativement petit contenant un plan AND programmable suivi d'un plan OU-fixe.

### 10.1.4 Circuits GAL (Generic Array Logic):

- Dispositifs programmables par l'utilisateur ;
- Mis en marché par Lattice Semiconductors en 1985 ;
- Peuvent émuler différents types de PAL.
- Les circuits GAL ont longtemps remplacé les composantes SSI-LSI, mais ne sont plus manufacturés.

## 10.2 HCPLD (High Capacity Programmable Logic Device)

Les CPLD sont de conception plus ancienne et ont généralement des Capacités moindres que les FPGA. Sont deux des types de puces à logique numérique bien connus. En ce qui concerne l'architecture interne, les deux puces sont évidemment différentes.

### 10.2.1 Les CPLD( Complex Programmable Logic Devices )

Sont une extension naturelle des circuits PAL.

- incorpore plusieurs PAL sur une seule puce avec un réseau d'interconnexions.
- Le réseau permet de relier les pattes de la puce à différents blocs internes et de relier les blocs entre eux.

### 10.2.2 Les FPGA

L'abréviation de x, est un type de puce logique programmable. C'est une excellente puce car il peut être programmé pour faire presque n'importe quel type de fonction numérique. L'architecture du FPGA permet à la puce d'avoir une très grande capacité logique. Il est utilisé dans les conceptions nécessitant un nombre élevé de portes et leurs délais sont assez imprévisibles en raison de son architecture

## 11 Les FPGA

### 11.1 Définition

Un FPGA (Field-Programmable Gate Array) est un composant électronique programmable. Il permet de synthétiser n'importe quel circuit numérique (multiplieur, additionneur, processeur, ...), pourvu que le FPGA soit assez complexe pour le contenir.

Un FPGA est composé de centaines de cellules logiques réalisant chacune une fonction logique élémentaire programmable. Ces cellules sont organisées en une matrice et reliées entre elles par

un vaste réseau d'interconnexions programmables. Une fonction logique complexe est implémentée en programmant les fonctions élémentaires de chaque cellule logique et en les reliant entre elles en programmant chaque interconnexion. Elle communique avec les composants électroniques extérieurs via des plots d'entrée/sortie également programmables. Cette flexibilité est le principal avantage des FPGAs sur les ASICs (Application Specific Integrated Circuit).

### 11.2 Architecture de FPGA

Un FPGA est constitué principalement de quatre composants :

- les blocs logiques configurables
- les cellules mémoires
- le réseau d'interconnexions programmables
- les blocs d'entrée/sortie

La plupart des grands FPGA modernes sont fondés sur des cellules SRAM aussi bien pour le routage du circuit que pour les blocs logiques à interconnecter.

Un bloc logique est de manière générale constitué d'une table de correspondance (LUT ou Look-Up-Table) et d'une bascule (Flip-Flop en anglais). La LUT sert à implémenter des équations logiques ayant généralement 4 à 6 entrées et une sortie. Elle est comme une petite mémoire, un multiplexeur ou un registre à décalage. Le registre permet de mémoriser un état ou de synchroniser un signal (pipeline).

Comme la configuration (routage et LUT) est faite par des points mémoire volatils, il est nécessaire de sauvegarder le design du FPGA dans une mémoire non volatile externe, Certains fabricants se distinguent toutefois par l'utilisation de cellules EEPROM pour la configuration, éliminant le recours à une mémoire externe, ou par une configuration par anti-fusibles Cette dernière technologie n'est toutefois pas reconfigurable.

Il y a 3 plans virtuels

Logique	Interconnexion	Programmation
Matrice de cellules	canaux de routage Et commutateurs	mémoire de configuration contient le bitstream

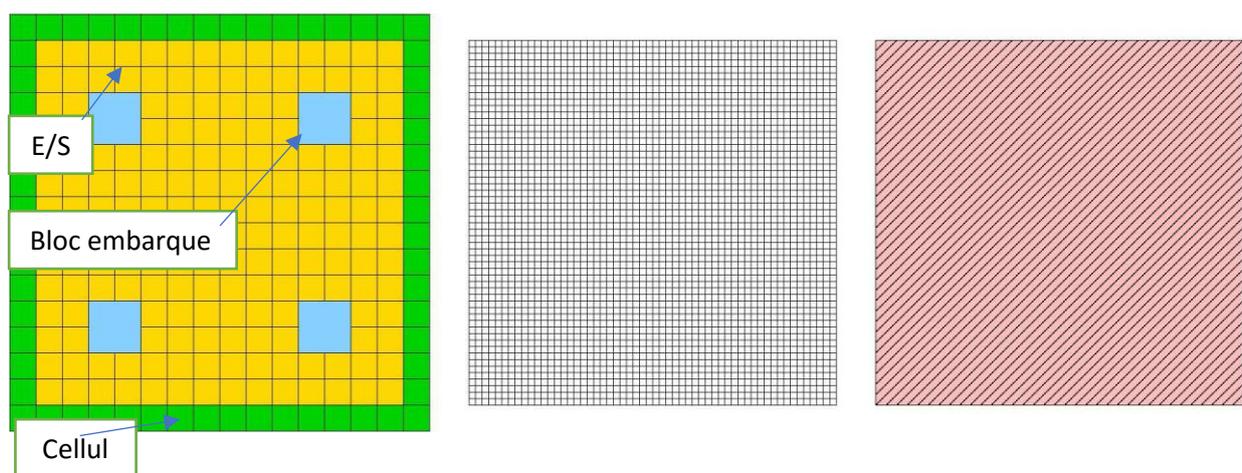


Figure 1.12: L'architecture d'un FPGA

L'interconnexion occupe environ 80% de la surface pour une même fonction, la surface est environ 30 fois celle d'un circuit ASIC

## Nexys2

La carte de circuit Nexys2 est une plate-forme de développement de circuit complète, prête à l'emploi, basée sur un FPGA Xilinx Spartan 3E. Son port USB2 haut débit intégré, ses 16 Mo de mémoire vive et son ROM, ainsi que plusieurs ports et périphériques d'entrée / sortie, en font une plate-forme idéale pour les systèmes numériques de toutes sortes, y compris les systèmes de traitement intégrés basés sur MicroBlaze de Xilinx. Le port USB2 fournit l'alimentation de la carte et une interface de programmation. La carte Nexys2 peut donc être utilisée avec un ordinateur portable pour créer une station de conception réellement portable.



Figure 1.13: FPGA Nexys2

Afin de pouvoir finaliser un FPGA, il est nécessaire d'utiliser un langage de description matériel ou bien un outil de saisie graphique. Après compilation de cette description, on obtient un fichier de configuration pour le FPGA choisi. VHDL et Verilog sont les deux langages de description les plus répandus.

## 12 Langage

Les langages de description de matériel (HDL = Hardware Description Language) font partie des outils de base pour la conception de systèmes logiques intégrés câblés, que le produit final soit construit sur des composants électriquement configurables (FPGA Field Programmable Logic Array) ou des circuits intégrés spécifiques (ASIC Application Specific Integrated Circuit). Il existe deux standards de langages de description de matériel :

- VHDL
- Verilog

### 12.1 Langage Verilog

Le langage de description matérielle Verilog est un langage permettant de décrire le comportement et la structure des circuits électroniques, et est une norme IEEE (IEEE Std.1364-1995). Verilog est utilisé pour simuler la fonctionnalité de circuits électroniques numériques à niveaux d'abstraction allant du comportement stochastique et pur à la porte et niveau de commutation, et est également utilisé pour synthétiser (c'est-à-dire générer automatiquement) le niveau de la porte.

### 12.2 Langage VHDL

Est un langage de description de matériel destiné à représenter le comportement ainsi que l'architecture d'un système électronique numérique. Son nom complet est VHSIC1 Hardware Description Language.

L'intérêt d'une telle description réside dans son caractère exécutable : une spécification décrite en VHDL peut être vérifiée par simulation, avant que la conception détaillée ne soit terminée. En outre, les outils de conception assistée par ordinateur permettant de passer directement d'une description fonctionnelle en VHDL à un schéma en porte logique ont révolutionné les méthodes de conception des circuits numériques, ASIC ou FPGA.

## 13 Conclusion

Nous avons vu dans ce chapitre les différents conception liés à l'ordinateur ainsi que les composant d'un processeur dont « l'UAL » qui est le circuit responsable de l'exécution des opération arithmétiques et logique.

Dans le prochain chapitre nous allons voir comment concevoir et réaliser une UAL avec le langage virelog, puis nous allons la tester sur une carte FPGA.

# **Chapitre N°2**

## **Conception et implémentation**

## 1 Introduction

Dans ce chapitre nous avons à réaliser une ALU 4 bits (Unité arithmétique et logique) et implémenter sur un FPGA en fait, une unité arithmétique et logique, ou ALU, permet aux ordinateurs d'effectuer des opérations mathématiques sur des nombres binaires. Ils se trouvent au cœur de chaque ordinateur numérique et constituent l'une des parties les plus importantes d'un CPU. Nous allons donc en créer une pouvant réaliser pas moins de moins de 8 opérations différentes.

## 2 Détail de UAL

Celle que nous devons réaliser doit répondre à certaines contraintes :

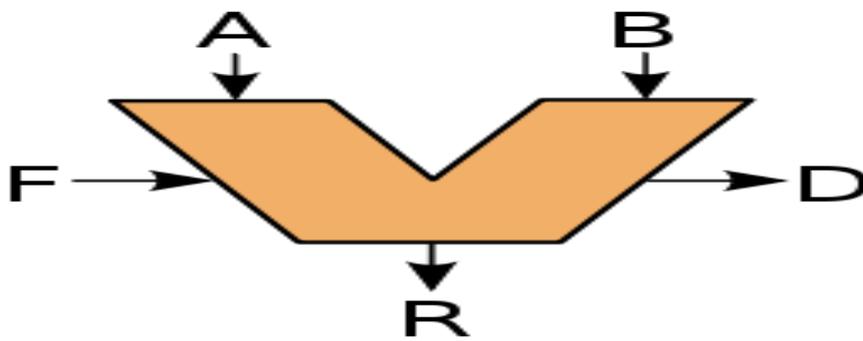


Figure 2. 1: Schéma ALU 4 bits

- A, B : signaux d'entrée codés sur 4 bits.
- F : entrée désigne l'opération à effectuer.
- R : le résultat de l'opération sur 4 bits.
- D : les éventuels drapeaux (qui définissent ou une erreur dans le résultat).

## 3 Détail des opérations choisit

C'est le signal qui permet de sélectionner l'opération courante parmi les 8 possible opération

CODOP	Opération
0000	Addition
0001	Substruction
0010	Logical-And

0011	Logical-Or
0100	Logical-nor
0101	Logical-Xor
0110	Cmp 1
1000	Cmp 2

Tableau 1: Tableau de sélection d'opération

### 3.1 Addition

Pour la conception d'additionneurs, une brique qui peut être utilisée est le demi-additionneur ou full adder

- entrées : deux bits à sommer a et b ;
- sorties : un bit de somme s et un bit de retenue sortante c.

#### 3.1.1 Demi-additionneur

Ce circuit, qui permettrait d'effectuer l'addition des deux bits de plus bas poids est appelé demi-additionneur (Half-Adder). Ecrivons la table de vérité de celui-ci :

a	b	sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Tableau 2: table de vérité Demi-add

Si nous écrivons ces deux fonctions sous leur forme canonique il vient :

$$\begin{cases} D = \bar{A} B + A \bar{B} \\ C = A B \end{cases}$$

$$\begin{cases} D = A \oplus B \\ C = A B \end{cases}$$

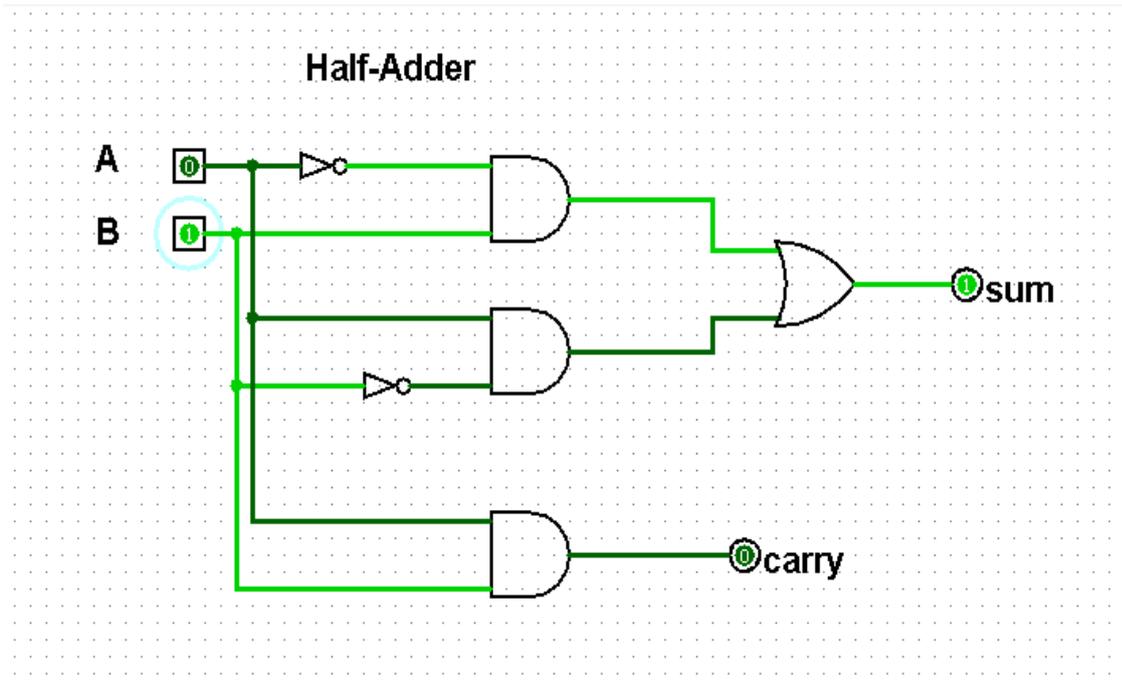


Figure 2. 2: Schéma du demi additionneur fait avec le simulateur Logisim

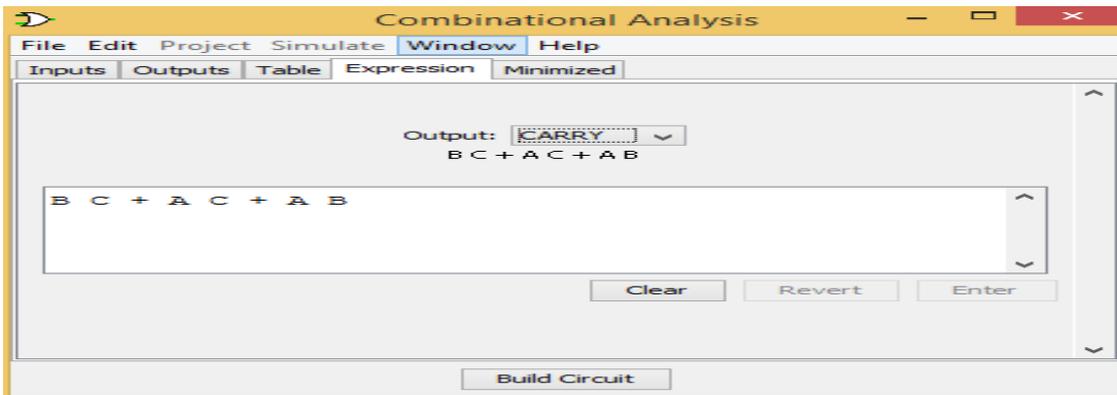
### 3.1.2 Additionneur complet 1 bit

Ecrivons la table de vérité de celui-ci :

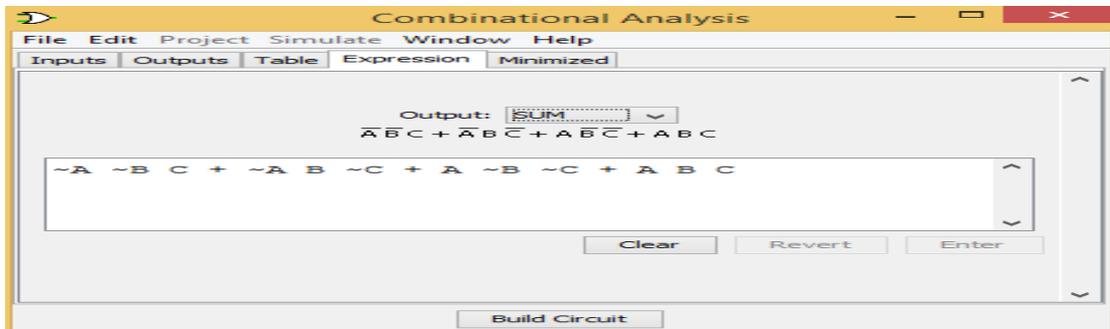
A	B	C	SUM	CARRY
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Tableau 3: Table De Vérité Additionneur Complet

On constate que



Équation 1: Expression Carry



Équation 2: Expression Sum

On déduit des expressions précédentes le circuit d'un additionneur 1 bit complet :

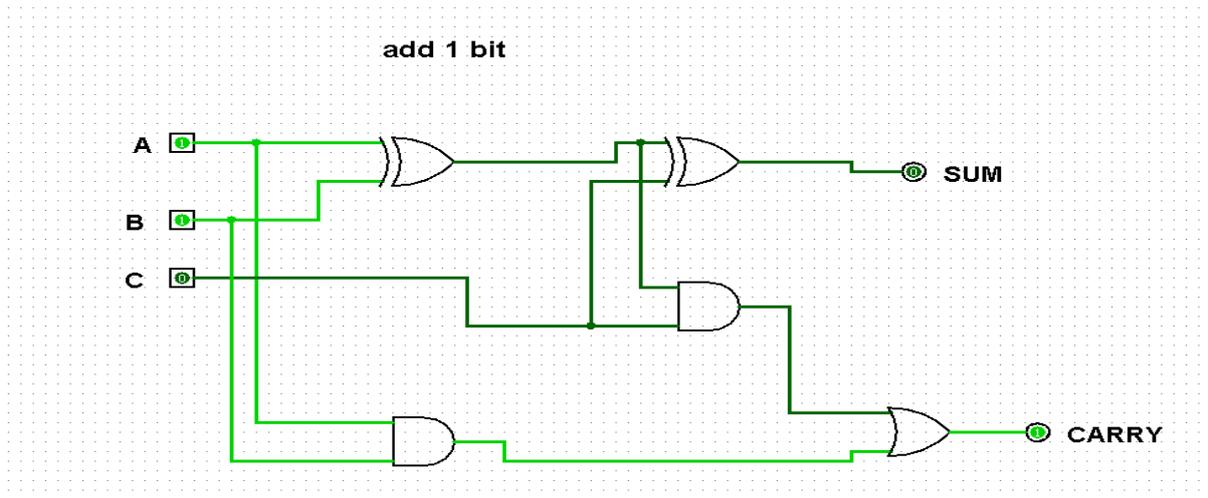


Figure 2. 3: Schéma de add 1 bit

### 3.1.3 Additionneur 4 bit

Un additionneur sur 4 bits est un circuit qui permet de faire l'addition de deux nombres A et B de 4 bits chacun  $A(a_3a_2a_1a_0)$  .B  $(b_3b_2b_1b_0)$  En plus il tient compte de la retenue entrante .En sortie on va avoir le résultat sur 4 bits ainsi que la retenue (5bits en sortie) . Pour effectuer l'addition de deux nombres de 4 bits, il suffit de chaîner entre eux 4 additionneurs 1-bit complets. La retenue est ainsi propagée d'un additionneur à l'autre. Un tel additionneur est appelé un additionneur série.

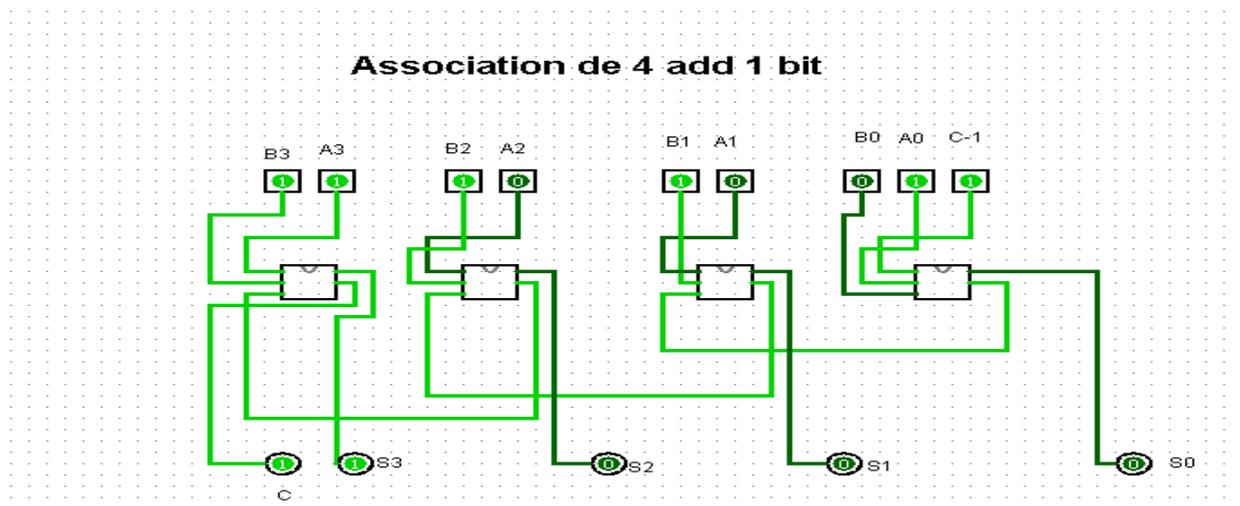


Figure 2. 4: Schéma du add 4 bit

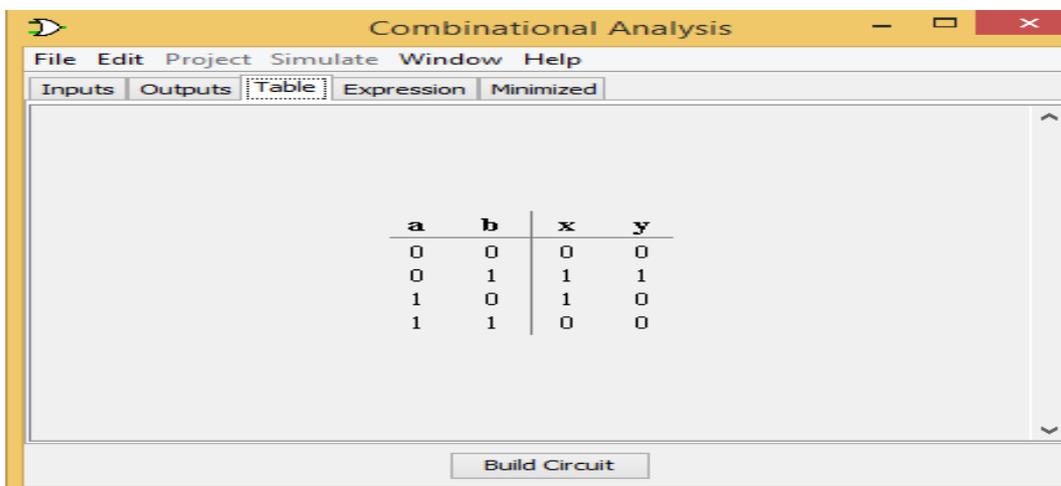
## 3.2 Soustraction

Il n'y a pas de circuit soustracteur dans un processeur parce que l'on peut implémenter la soustraction à l'aide de l'additionneur avec des modifications mineures. Pour ce faire, on exploite les propriétés du complément à 2 et le fait que le bit de poids faible de l'additionneur n'a pas de retenue d'entrée. En effet, effectuer  $X - Y$  en complément à 2, est équivalent à  $X + Y' + 1$ . Pour

effectuer la deuxième addition (+1), il suffit d'injecter un 1 en guise de retenue dans l'additionneur de poids faible. On peut donc supposer que l'on dispose d'un signal de contrôle  $c$  qui vaut 0 lorsque l'on veut faire une addition, et 1 lorsque l'on veut faire une soustraction. On utilise ce signal  $c$  comme retenue du bit de poids faible de l'additionneur. Enfin, pour obtenir  $Y'$ , il suffit de rajouter un inverseur (une porte XOR) en entrée de chacun des additionneurs 1-bit :  $y_i \oplus c$  ; lorsque  $c$  vaut 0, la valeur d'entrée de l'additionneur  $i$  est  $y_i$ , et lorsque  $c$  vaut 1, la valeur d'entrée est  $y_i'$ . Donc, lorsque  $c$  vaut 0, l'opération effectuée par le circuit est  $X + Y$ , et lorsque  $c$  vaut 1, l'opération effectuée est  $X + Y' + 1$ .

### 3.2.1 Demi-soustracteur 1 bits

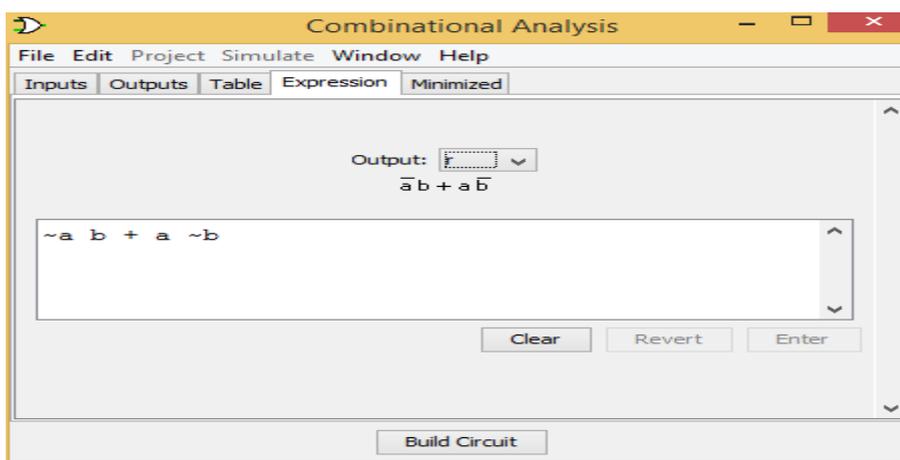
La table de vérité pour un demi-soustracteur est la suivante



a	b	x	y
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

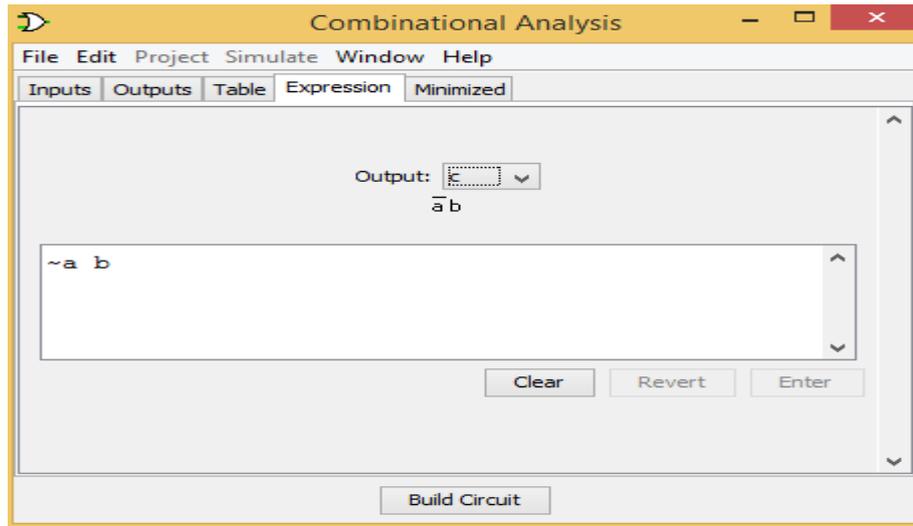
Tableau 4: Table De Vérité Demi-soustracteur

On constate que



Output:  $a b + a \sim b$

Équation 3: Expression de résultat



Équation 4: Expression de retenu

Et le schéma correspondant

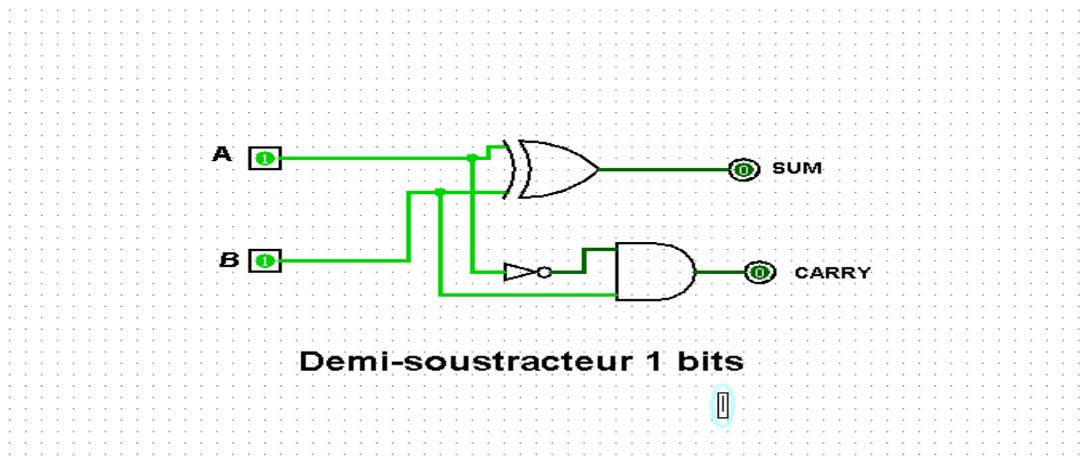


Figure 2. 5: Schéma du Demi-soustracteur

### 3.2.2 Additionneur-soustracteur 4bit

Voici le schéma d'un Additionneur-soustracteur 4bits, on a donc 4 additionneurs 1 bit complets, des portes xor et 1 bit de sélection entre l'addition et la soustraction comme expliqué au paragraphe 2.



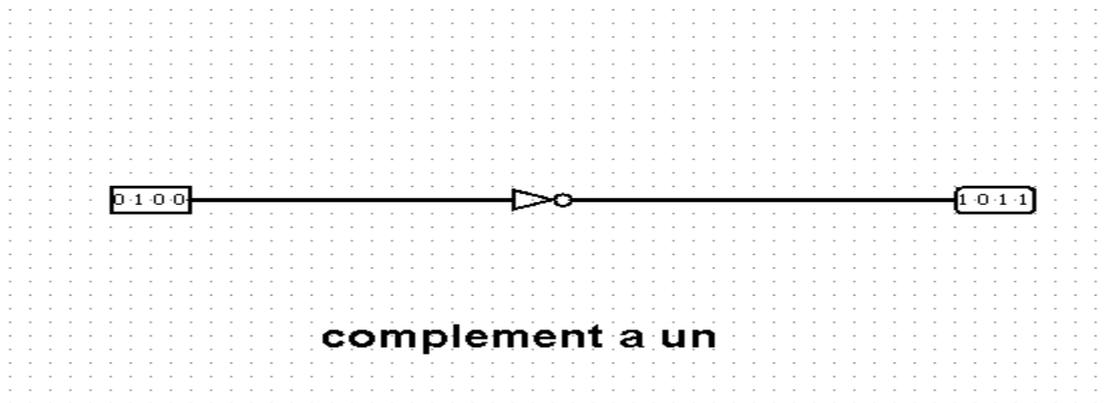


Figure 2. 7: Schéma du Complément a un

## 5 Complément à 2

Le complément à 2 (complément arithmétique) d'un nombre entier négatif est obtenu en rajoutant +1 à sa représentation CP1. - Cette représentation donne une configuration de plus car le zéro aura une seule représentation.

Le schéma correspondant :

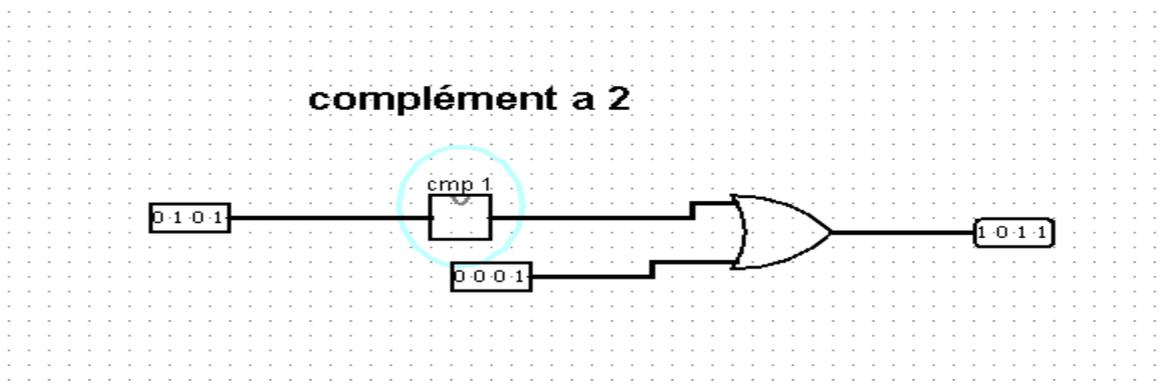


Figure 2. 8: Schéma du Complément a 2

## 6 OR

Le OR est une opération logique. Les circuits logiques sont élaborés à partir de composants électroniques – transistors. Toute fonction logique peut être réalisée à l'aide d'un nombre de fonctions logiques de base appelées portes.

Représenter par ce schéma :

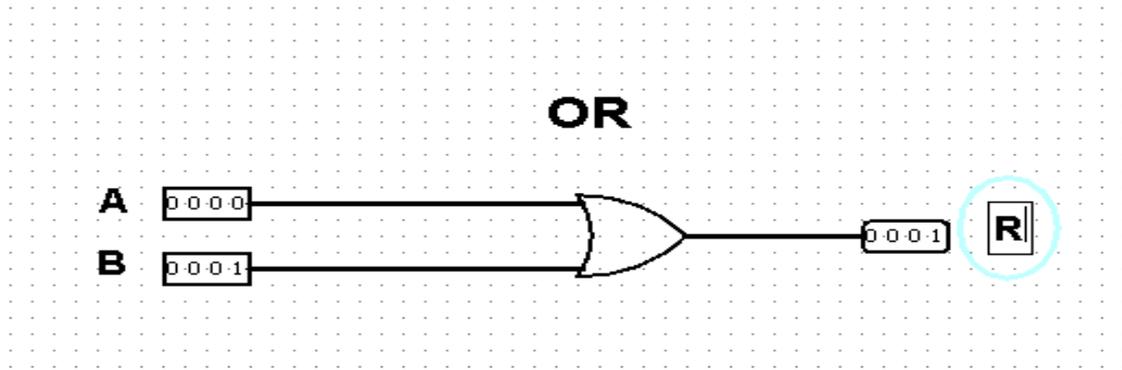


Figure 2. 9: Schéma de OR

On constate que

$$R=A + B$$

## 7 XOR

Représenter par ce schéma :

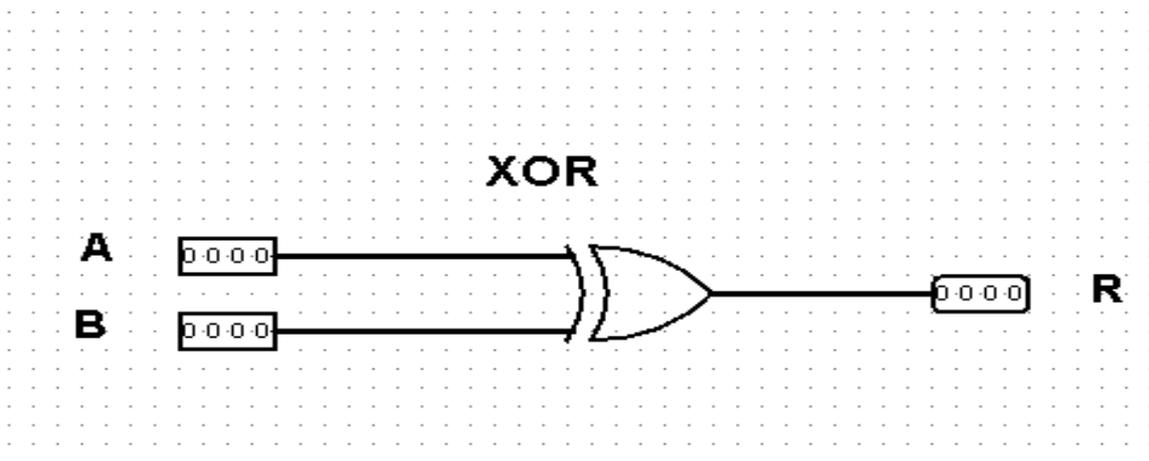


Figure 2. 10: Schéma du XOR

On constate que

$$R=A \text{ xor } B$$

## 8 AND

Représenter par ce schéma :

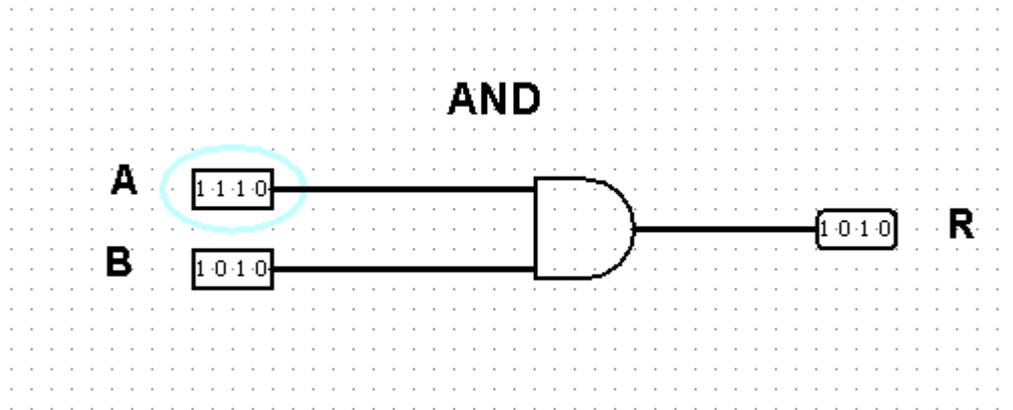


Figure 2. 11: Schéma de AND

On constate que

$$R=A*B$$

## 9 NOR

Représenter par ce schéma :

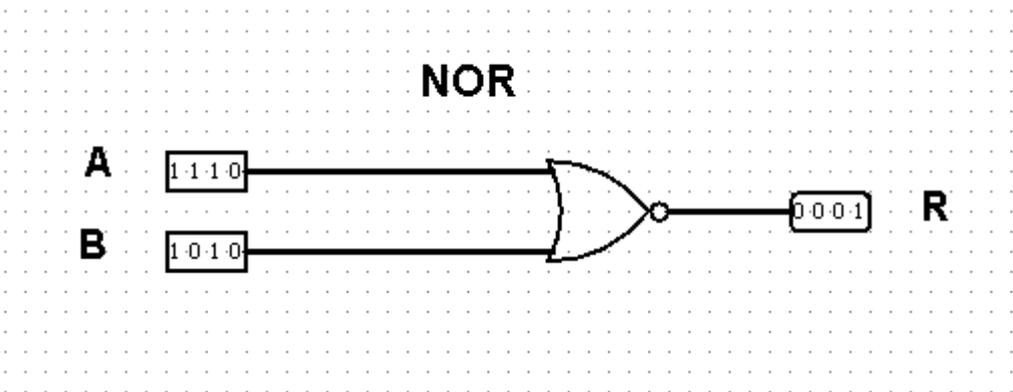


Figure 2. 12: Schéma du NOR

On constate que

$$R=\text{Not}(A+B)$$

Schéma Générale de L'UAL :

Le schéma globale est un ensemble des opérations 4 bits précédentes entre sur le multiplexeurs 8 vers 1 après a l'aide de bit de sélection il choisit la fonction .

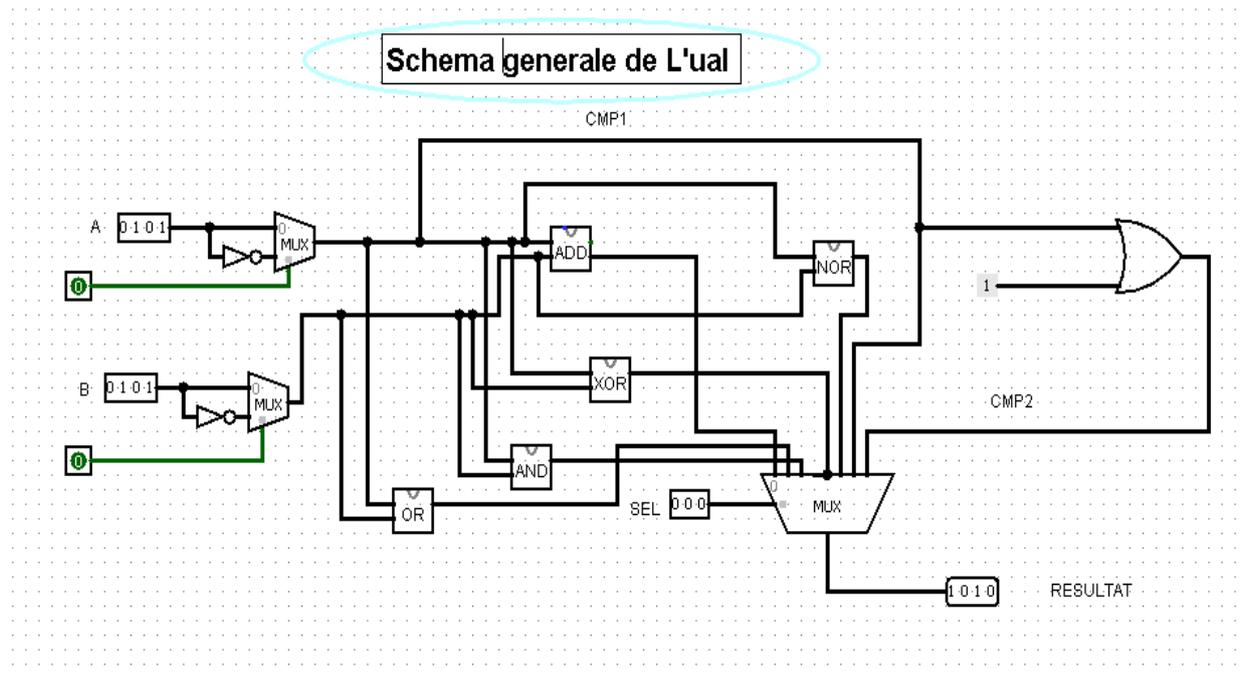


Figure 2. 13: Schéma du ALU 4 BIT

## 10 Implémentation

Il faut savoir que l'implémentation du circuit sous cadence s'effectue avec le logiciel ISE Project Navigator (P.20132013) et le langage de programmation Verilog.

### 10.1 Le code

Pour le code, nous avons deux variables en entrée nommées a et b en plus du code op pour la sélection de l'opération à effectuer en utilisant la variable sel. En sortie nous avons la variable z en plus du drapeau c pour la retenue. L'implémentation sous Verilog se fait sous la forme de modules.

```

ISE Project Navigator (P.20131013) - C:\Users\marwa7\find\find.xise - [alu4bit.v]
ess Tools Window Layout Help
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21 module alu4bit(z,a,b,sel
22 );
23 input [3:0]a,b;
24 input [3:0]sel;
25 output [3:0]z;
26 reg [3:0]z;
27 always@(sel,a,b)
28 begin
29 case (sel)
30 4'b0000: z=a+b; // Addition
31 4'b0001: z=a-b; // Subtraction
32 4'b0010: z=a&b; // Logical and
33 4'b0011: z=a|b; // Logical or
34 4'b0100: z=~(a|b); // Logical nor
35 4'b0101: z=a^b; // Logical xor
36 4'b0110: z=~a; // Logical not
37 4'b1000: z=~a|1;
38 endcase
39 end
40 endmodule
41
42
43
44
45
46
47

```

Figure 2. 14: Le code Verilog

## 10.2 La liaison avec les ports du FPGA

L'assignation des ports se fait par l'entremise d'un fichier de contraintes avec l'extension « .ucf » ( user constraints file).

Nous avons assigné aux variables a et b les interrupteurs de 0 à 3 et de 4 à 7 respectivement. à la variables sel nous avons assigné les boutons poussoirs de 0 à 3. Quant au résultat il s'affiche sur les leds de 0 à 4.

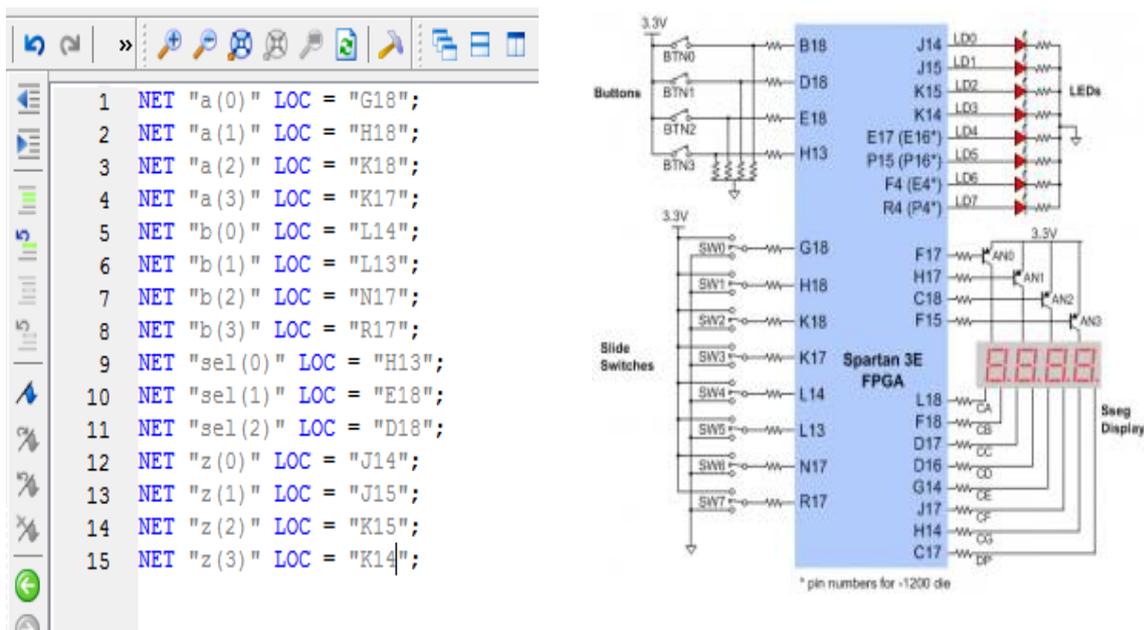


Figure 2. 15: Le code de la liaison

## 10.3 Fichier .bit

Fichier généré par BitGen, un programme utilisé pour générer les trains de bits requis par les FPGA Xilinx enregistré dans un format binaire et contient les informations de configuration pour le circuit ; utilisé pour télécharger les données de configuration sur le périphérique FPGA, ce qui peut être effectué à l'aide de l'interface graphique iMPACT fournie avec le logiciel Xilinx[2].



Figure 2. 16:Fichier .bit

## 10.4 Configuration de la carte Nexys2 avec ADEPT

Après avoir fait la synthèse, le mapping et le placement & routage du projet :

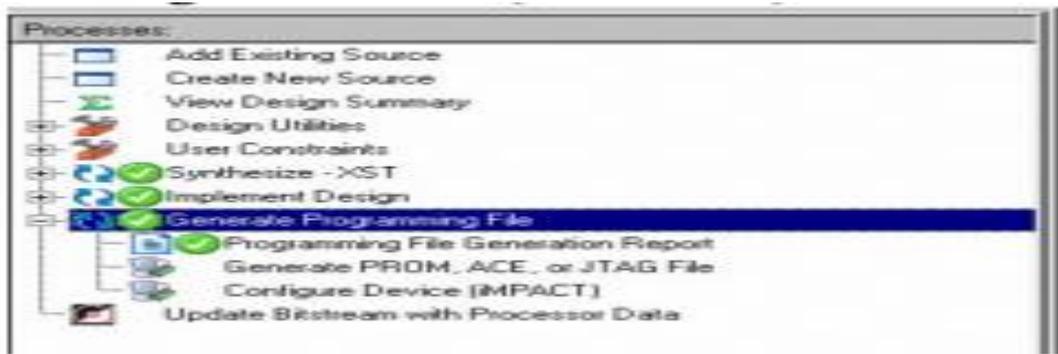


Figure 2. 17: Configuration sur ISE

Vous allez générer le fichier de programmation BIT pour configurer la carte Nexys2. [3]

Vous allez utiliser le logiciel ADEPT pour programmer le FPGA, lancez le logiciel, vous obtenez cette figure, il suffit de retrouver le fichier alu4bit.bit dans votre espace de travail et ensuite cliquez sur Program.

Le FPGA est maintenant programmé avec le projet. Entrez la séquence en appuyant successivement sur les boutons BTN 0, BTN 1,

BTN 2 et BTN 3 et vérifiez que les Leds LD0 à LD7 s'activent en rafale.

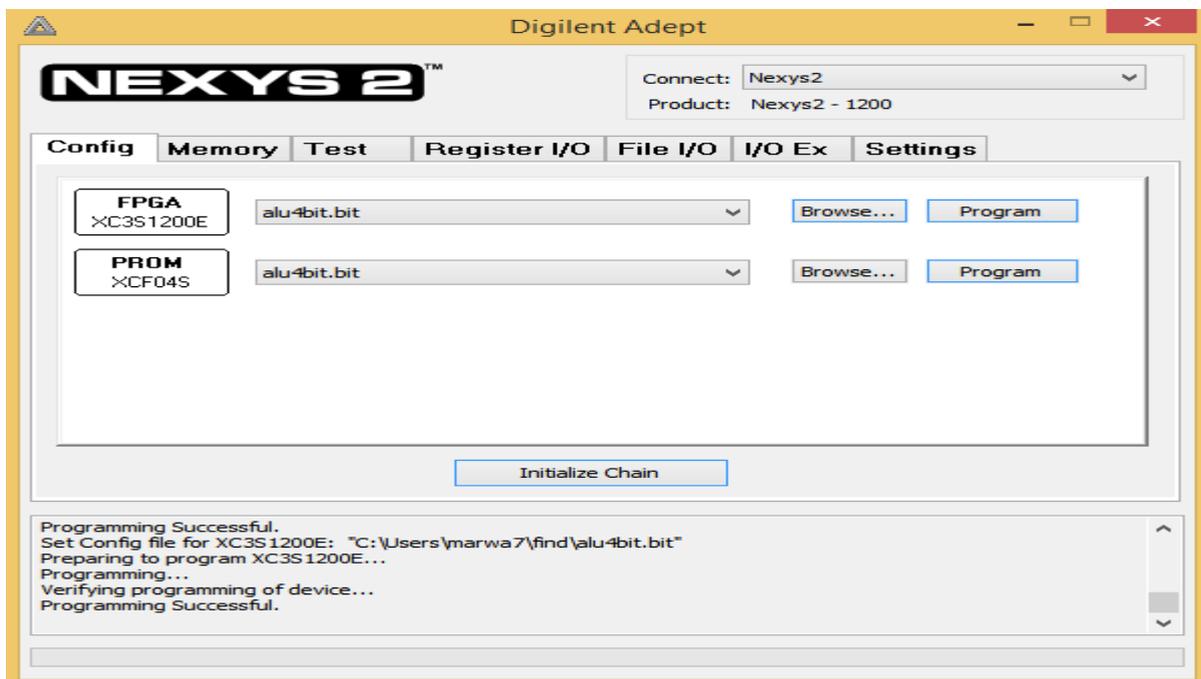


Figure 2. 18: Implémentation sur la carte

## 11 Démonstration

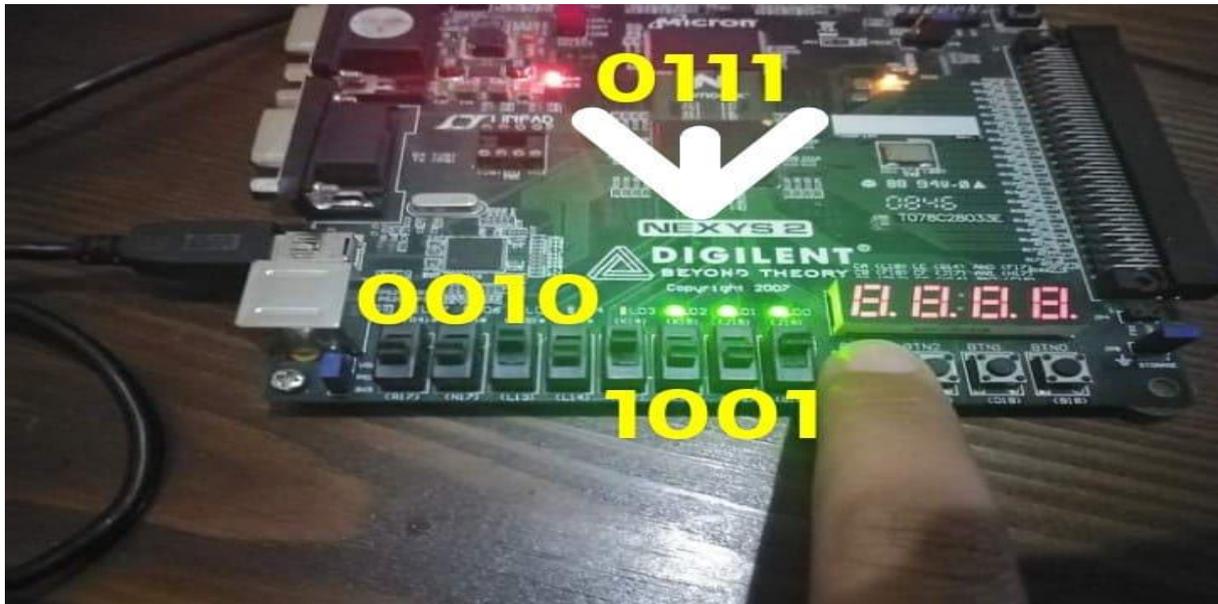


Figure 2. 19:Exemple de soustraction sur une FPGA

## 12 Conclusion

Nous avons vu dans ce chapitre la réalisation d'une unité arithmétique et logique (UAL) codée sous le langage verilog. L'UAL travail sur des nombres de taille de 4 bits. Cette dernière effectue 8 opérations testées avec succès avec le FPGA Nexys2.

# Conclusion générale

Au bout de notre cursus en Licence « Informatique - Systèmes d'information », nous avons été chargés de réaliser un projet de fin d'études.

L'objectif de notre travail s'est basé sur l'implémentation d'une UAL 4 bit en Verilog et simuler les fonctions du UAL sur le circuit programmable "FPGA". Nous avons fait la conception d'une UAL par le simulateur Logisim et le logiciel ISE après l'implémentation sur le FPGA c'était par Digilent Adept. La conception d'une unité arithmétique et logique nous a permis de maîtriser les schémas sous Cadence ainsi que la simulation. Les opérations de l'ALU ont toutes été réalisées avec succès.

Finalement, l'approche traitée au cours de ce travail peut-être avantageusement améliorée et facilement étendue sur d'autres types de commande et les chercheurs du domaine vont donc devoir relever des défis encore plus importants.

Comme perspectives, on propose de simuler d'autres fonctions de CPU comme l'unité contrôle

# Bibliographie

- [1] <https://reference.digilentinc.com/reference/programmable-logic/nexys-2/reference-manual>
- [2] <https://commentouvrir.com/extension/bit>
- [3] <http://systemesembarques.e-monsite.com/medias/files/fascicule.de.tp.fpga.pdf>
- [4] [http://library.euroaquae.eu/bde2015/jahia/webdav/site/bde/shared/Utile/Cours/ELEC/ELEC4/projet\\_electronique\\_alu\\_2007.pdf](http://library.euroaquae.eu/bde2015/jahia/webdav/site/bde/shared/Utile/Cours/ELEC/ELEC4/projet_electronique_alu_2007.pdf)
- [5] [https://fr.wikipedia.org/wiki/Unit%C3%A9\\_arithm%C3%A9tique\\_et\\_logique](https://fr.wikipedia.org/wiki/Unit%C3%A9_arithm%C3%A9tique_et_logique)
- [6] [http://lewebdephilou.free.fr/RESEAUX-TELECOM/Cours-Telecom/Transmission/Electronique-Numerique\\_EISTI\\_Guy-Almouzni.pdf](http://lewebdephilou.free.fr/RESEAUX-TELECOM/Cours-Telecom/Transmission/Electronique-Numerique_EISTI_Guy-Almouzni.pdf)
- [7] <http://systemesembarques.e-monsite.com/medias/files/fascicule.de.tp.fpga.pdf>
- [8] <http://genie-indus.e-monsite.com/medias/files/tp1.pdf>
- [9] [https://fr.wikipedia.org/wiki/Circuit\\_logique\\_programmable](https://fr.wikipedia.org/wiki/Circuit_logique_programmable)
- [10] <http://www.dicodunet.com/definitions/materiel/ual.htm>
- [11] <http://lesdefinitions.fr/ordinateur>
- [12] Jean luc danger, architecture d'un FPGA
- [13] Architecture processeur
- [14] Alain Cazes Joelle Delacroix, Architecture des machie et des système informatique, Dunod,2018
- [15] Emmanuel Boutillon, Les FPGA
- [16] cours de L'histoire-ordinateur
- [17] Adam Foust, Programmable Logic Devices
- [18] Stephen Brown and Jonathan Rose, Architecture of FPGAs and CPLDs.
- [19] Jim Duckworth, VHDL and Verilog ECE Department, WPI.
- [20] Lagra ,cours architecture ordinateur .