الجمهورية الجزائرية الديمقراطية الشعبية

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DR. YAHIA FARES DE MEDEA

FACULTE DES SCIENCES ET DE LA TECHNOLOGIE

Département du Génie Électrique et de l'Informatique



Projet de fin d'étude

Présenté pour l'obtention du Diplôme
De Master en Génie Électrique

Spécialité : AUTOMATIQUE DES SYSTEMES

THEME

IMPLIMENTATION DE LA COMMANDE SCALAIRE DE LA MACHINE ASYNCHRONE SUR UN CIRCUIT FPGA

Proposé et dirigé par :

Mr. KABACHE NADIR

Mr. MOHAMMEDI AMINE

Préparé et présenté par :

Mr. TAIBI ABD EL KRIM.

Année Universities 2015-2016


بسم الله الرحمن الرحيم

Dédicace

- *A l'homme de ma vie, mon exemple éternel, mon soutien moral et source de joie et de bonheur, celui qui s'est toujours sacrifié pour me voir réussir, que dieu te garde dans son vaste paradis, à toi mon père.*

- *A la lumière de mes jours, la source de mes efforts, la flamme de mon cœur, ma vie et mon bonheur ; maman que j'adore.*

- *Aux personnes dont j'ai bien aimé la présence dans ce jour, à tous mes frères, ET A toute ma famille.*

- *A tous mes enseignants de l'UNIVERSITE DE MEDEA qui ont contribués a ma formation master.*

- *Finalement, je remercie tous mes Ami(e)s que j'aime tant, Toufik l'wali ,Abdellah, Houssem, Boualem, Abdelkader, Malak , Taha, zaki, Romaissa, Rafik, mouloud, obama , Hamza, Zaki, Sabrina, Mourad, Abderrahmane, Chouchou, Mohamed ,Saliha , Sofiane , Ali, Islam Fares ,nawal, raouia, fatima, hakoo, sidali, souhaib, iyad, khirelddine, hamid, mustafa,hichem, omar, ali, soufian, dahman, chaima , karim, hani ,alaa, amin , FADI .....*

Pour leur sincère amitié et confiance, et à qui je dois ma reconnaissance et mon attachement.

REMECIEMENTS

- *On remercie dieu le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer ce mémoire.*

- *Nous tenon à exprimer vivement notre profonde gratitude à nos promoteurs : N.KABACHE et A.MOHAMMEDI pour leur confiance, leur encouragement, leur suivi et pour les conseils qu'ils ont apporté pour l'achèvement de ce projet.*

- *Nous remercions également tous les enseignants qui ont contribué de prés ou de loin a notre formation et à cette thèse.*

- *Je tien a remercier très chaleureusement les nombres de jury, qui ont accepté la charge de juger ce mémoire malgré un très court délai a consacre à la lecture.*

- *Nous voudrions associer nos remerciements à toute l'équipe du labo de Médéa,*

TABLE DES NOTATIONS ET DES SYMBOLES

A, b, c	Axes liés aux enroulements triphasés.
d, q	Axes de référentiel de Park.
[P]	Matrice de Park.
A (rad)	Angle entre le stator et le rotor**.**

Grandeurs électriques et magnétiques au stator

Vs (V)	Tension statorique.
Vsabc(V)	Tension statorique phase a, b ou c.
Vsdq(V)	Tension statorique sur l'axe d ou q.
Is(A)	Courant statorique.
Asabc(i)	Courant statorique phase a, b ou c.
Asdq(i)	Courant statorique sur l'axe d ou q.
Rs(Ω)	Résistance statorique par phase.
ls(H)	Inductance propre statorique par phase.
Ms(H)	Inductance mutuelle entre deux phases statoriques.
Ls(H)=ls-Ms	Inductance statorique par phase.
Φsabc(Wb)	Flux statorique phase a, b ou c.
ϕsdq (Wb)	Flux statorique sur l'axe d ou q.

Grandeurs électriques et magnétiques au rotor

Vr(V)	Tension rotorique.
Vrabc(V)	Tension rotorique phase a, b ou c.
Vrdq(V)	Tension rotorique sur l'axe d ou q.
Ir(A)	Courant rotorique.
irabc(A)	Courant rotorique phase a, b ou c.
irdq(A)	Courant rotorique sur l'axe d ou q.
Rr(Ω)	Résistance rotorique par phase.
Φrdq(Wb)	Flux rotorique sur l'axe d ou q.
Mr(H)	Inductance mutuelle entre deux phases rotoriques.

الملخص

يهدف هذا المشروع إلى تنفيذ السيطرة على حلقة مفتوحة بواسطة بطاقة FPGA SPARTAN 3 Eو نريد من هذا العمل أن نحسن أداء الجهاز الغير متزامن. في البداية نقوم بعرض بطاقة FPGA ثم شرح مختلف طرق المحاكات على هذا الجهاز. ثم نبرمج هذا الجهاز بواسطة برنامج VHDL ثم نطبق هذا البرنامج على بطاقة FPGA .

الكلمات المفتاحية: FPGA, VHDL, SPARTAN 3 Eالجهاز غيرمتزامن.السيطرةالعددية .

Résumé

Ce travail a pour but d'améliorer les performances de la machine asynchrone grâce au circuit FPGA, on présente dans notre projet l'implémentation de la commande scalaire en boucle ouverte sur la carte FPGA SPARTANE 3E , en premier lieu on présente le circuit FPGA et leur langage de programmation puis on fait des simulation de la commande scalaire pour la machine asynchrone ensuit on décrit le programme VHDL de la commande on passant par les déférents étapes de l'implémentation telque la simulation, la synthèse, le placement et le routage jusqu'au la création le fichier point bit (.bit) à la fin on a implémenté la commande et on a analysé le comportement de moteur.

Mots clés : FPGA, VHDL, SPARTAN 3E, machine asynchrone, commande scalaire.

Abstract

This work aims to improve the performance of the induction motor through the FPGA, we present our project in the implementation of the scalar open loop control on the FPGA board SPARTANE 3 E, first presents the FPGA and programming language then made simulation of scalar control for induction motor follows describes the VHDL program to control it through the stages of implementation such as simulation, synthesis, placement and routing to create the bit file (.bit) at the end we implemented the control and motor behavior were analyzed.

Key words: FPGA, VHDL, SPARTAN 3 E,ASYNCHRONOUS MACHINE, scalar control

SOMMAIRE

TABLE DES MATIERES

RESUME

TABLE DES MATIERES

TABLE DES FIGURES

LISTE DES ACRONYMES ET NOMENCLATEUR.

INTRODUCTION GENERALE……..1

CAHPITRE I

L'ETAT D'ART DES FPGA ET LEURS APPORTS A LA COMMANDE DES MACHINES

I.1)-Introduction Page 3

I.2)-Les différents types des circuits numériques Page 4

I.3)-Les circuits logiques programmables Page 6

I.4)-Les circuits FPGA Page 7

I.4.1)-Domaines d'application des FPGA Page 7

I.4.2)-Principaux fondeurs d'FPGA Page 8

I.4.3)-Configuration et reconfiguration des FPGA Page 8

I.4.4)-Technologies de programmation des FPGA Page 9

I.4.4.1)-Technologie a base de ram (XILINX et ALTERA) Page 9

I.4.4.2)- Technologie a base d'EEPROM ou FLASH (LATTICE ET ACTEL) Page 10

I.4.4.3)- Technologie a base d'anti-fusibes(ACTEL) Page 10

1.4.5) -Architecture interne des FPGA Page 11

I.4.6)-Avantages et inconvénients des FPGA Page 13

I.5)- Langage de description VHDL Page 13

I.6)-Prototypage des FPGA Page 15

I.6.1)-La programmation des FPGA par les outils HDL Page 15

I.6.1.1)-Spécification du design Page 16

I.6.1.2)-Conception du design Page 16

I.6.1.3)-Synthèse Page 16

I.6.1.4)-Placement et routage Page 16

I.6.1.5)-Intégration et implémentation Page 17

I.6.2)-La programmation des FPGA par XSG/MATLAB-SIMULINK Page 18

I.7)-Conclusion Page 19

CAHPITRE II

LA COMMANDE SCALAIRE

LA MAS

II.1)-Introduction Page 20

II.2)-Constitution et principe de fonctionnement Page 21

II.3)-Modélisation de la machine asynchrone Page 22

II.4) **Equations générales** Page 23

II.5)-transformation de « Park » Page 25

II.6)-Simulation du modèle moteur asynchrone triphasé Page 28

II.7)-Commande scalaire du moteur asynchrone Page 29

II.8) Résultat de simulation de la commande scalaire Page 31

II.9)-Conclusion Page 34

CAHPITRE III

IMPLEMENTATION DE LA COMMANDE SCALAIRE

D'UNE MAS

III.1)-Introduction Page 35

III.2)-Descriptions de la stratégie(MLI) Page 35

III.2.1)-Génération des signaux sinusoïdaux Page 36

III.2.2).Le signal triangulaire (porteuse) Page 37

III.3)-Description VHDL Page 39

III.3.1)- Génération de signal sinusoïdal Page 39

III.3.2)- Génération de signal triangulaire Page 40

III.3.3)- Génération MLI Page 40

III.3.4)- Variation de la vitesse du moteur Page 41

III.5)-Synthèse et simulation de la stratégie MLI: Page 42

III.6)-L'implémentation de la description de la stratégie MLI Page 43

II.7)-Partie De Réalisation Page 46

III.8) Teste experimentaux Page 51

III.8.1)-Essai 1 Page 51

III.8.2)-Essai 2 Page 55

III.8.3)-Essai 3 Page 56

III.9)-Conclusion Page 58

LISTE DES FIGURES ET TABLEAUX

Figure(I.1): Schéma comparatif d'un DSP et d'un FPGA. Page 5

Figure(I.2) : Diagramme des différents types de circuits logiques programmables. Page 6

Figure(I.3) : Statistiques du marché occupé par les vendeurs d'FPGA. Page 8

Figure(I.4): Reprogrammable sur site d'un FPGA. Page 8

Figure(I.5): Caractéristiques des technologies SRAM. Page 9

Figure(I.6): Caractéristiques des technologies FLASH. Page 10

Figure(I.7): Caractéristiques des technologies ANTI-FUSIBLES. Page 10

Figure(I.8): Architecture du FPGA. Page 11

Figure(I.9): Architecture d'un CLB de familles Vertex du Xilinx. Page 12

Figure(I.10) : Différent block de langage VHDL. Page 14

Figure(I.11) : Mode d'exécution matériel de la CAO. Page 15

Figure(I.12) : les étapes de développement une application sur FPGA. Page 17

Figure(I.13): Hardware Co-simulation XSG. Page 18

Figure(II.1) : constitution de moteur asynchrone à cage Page 21

Figure(II.2) : Représentation schématique d'une machine asynchrone triphasée. Page 22

Figure(II.3) : Représentation du passage d'un système triphasé à celui biphasé. Page 25

Figure(II.4): Courant statorique Isa. Page 28

Figure(II.5): Courant statorique Ira. Page 28

Figure(II.6):Couple électromagnétique Ce. Page 28

Figure(II.7):la vitesse de rotation n. Page 28

Figure(II.8):schéma général de la commande scalaire en boucle ouvert. Page 29

Figure(II.9):schéma général de la commande scalaire en boucle fermée. Page 30

Figure(II.10):la vitesse de la commande scalaire Page 31

Figure(II.11):le couple électromagnétique de la commande scalaire Page 31

Figure(II.12):le courant Ids de la commande scalaire Page 32

Figure(II.13):le courant Iqs de la commande scalaire Page 32

Figure(II.14):le flux FDN. Page 33

Figure(II.15):le flux FQN. Page 33

Figure (III.1) : Principe de la MLI sinusoïdaux-triangulaire. Page 36

Figure (III.2): Génération des signaux sinusoïdaux par Matlab. Page 37

Figure (III.3) : Synchronisation des signaux sinusoïdaux et triangulaire. Page 38

Figure (III.4): l'entité d'une description VHDl. Page 39

Figure (III.5) : Création d'un RAM pour le signal sinusoïdal. Page 39

Figure (III.6) : déclaration de signal triangulaire. Page 40

Figure (III.7) : génération de signal MLI. Page 41

Figure (III.8) : représente les pins de commande de la vitesse sur le circuit FPGA Page 41

Figure (III.9) : Vue externe (RTL) de MLI. Page 42

Figure (III.10) : simulation de MLI. Page 43

Figure (III.11) : Forme d'un fichier UCF. Page 44

Figure (III.12) : Processus d'implémentation de la description. Page 44

Figure (III.13) : Configuration de la carte FPGA par un câble JTAG. Page 45

Figure (III.14) : banc expérimental. Page 46

Figure (III.15) : le moteur asynchrone. Page 47

Figure (III.16) : l'encodeur. Page 47

Figure (III.17) : la carte DSPACE. Page 48

Figure (III.18) : la structure externe de la carte FPGA. Page 48

Figure (III.19) : la structure de pupitre électrique. Page 49

Figure (III. 20) : la structure d'onduleur Page 49

Figure (III.21) : interface d'onduleur Page 49

Page 33
Figure (III.22) : Oscilloscope numérique. Page 50

Figure (III.23) : Capteur de courant différentielle Page 50

Figure (III.24) : Sonde différentielle Page 50

Figure (III.25) : Multimètre. Page 50

Figure (III.25) : courant, tension et vitesse. Page 51

Figure (III.26) : vitesse réelle. Page 52

Figure (III.27) : phase de démarrage de vitesse réelle. Page 52

Figure (III.28) : tension réelle. Page 53

Figure (III.29) : zoom de la tension réelle. Page 53

Figure (III.30) : courant réel. Page 54

Figure (III.31) : zoom du courant réel. Page 54

Figure (III.32) : courant, tension et vitesse. Page 55

Figure (III.33) : vitesse réelle. Page 55

Figure (III.34) : courant, tension et vitesse. Page 56

Figure (III.35) : vitesse réelle. Page 56

Figure (III.37) : zoome des signaux de MLI Sa,Sa'. Page 57

Figure (III.36) : les signaux de MLI Sa,Sa'. Page 57

Tableau(I.1): Comparaison des différentes solutions numériques. Page 5

Tableau(I.2): les avantages et les inconvénients de FPGA. Page 13

Tableau(III.1) : Etat du moteur en fonction de l'état des interrupteurs Page 41

LISTE DES ACRONYMES

ASlC: **A**pplication **S**pecific **I**ntegrated **C**ircuit.

CAO: **C**onception **A**ssistée par **O**rdinateur.

CLB: **C**onfigurable **L**ogique **B**loc.

CPLD: **C**omplex **P**rogrammable **L**ogique **D**evise.

CORDI : **C**oordinate **R**otation **D**igital **C**omputer.

CPLD : **C**omplex **P**rogrammable **L**ogic **D**evise.

DSP : **D**igital **S**ignal **P**rocessor.

EPROM: **E**rasable **P**rogrammable **R**ead **O**nly **M**emory.

EEPROM: **E**lectrically **E**rasable **P**rogrammable **R**ead **O**nly **M**emory.

EEPLD: **E**lectrically **E**rasable **P**rogrammable **L**ogic **D**evice.

FPGA: **F**ieled **P**rogrammable **G**ate **A**rray.

IOB: **I**nput **O**utput **B**loc.

LUT: **L**ook **U**p **T**able.

MLI : **M**odulation de **L**argeurs d'**I**mpulsions.

PED: **P**rogrammable **L**ogique **D**evice.

PROM: **P**rogrammable **R**ead **O**nly **M**emory.

PI: **P**roportionnel **I**ntégral.

PAL : **P**rogrammable **A**rray **L**ogic.

PLD: **P**rogrammable **L**ogic **D**evice.

ROM: **R**ead **O**nly **M**emory.

RTL: **R**egister **T**ransfer **L**evel.

SRAM: **S**tatic **R**andom **A**ccess **M**emory.

LCA: **L**ogic **C**ell **A**rray

MAS: **M**achine **As**ynchrone.

GAL: **G**eneric **A**rray **L**ogic

VHDL**: V**hsic **(V**erry **H**ight **S**peed **I**ntegrated **C**ircuit) **H**ardware **D**escription **L**anguage

INTRODUCTION GENERATION

INTRODUCTION GENERAL

Dans plusieurs applications, l'utilisation des microcontrôleurs ou des microprocesseurs pour implémenter les lois de commande d'un système physique est généralement gêner par le fonctionnement séquentiel de ces outils de calcul. A cet effet et durant les dernières années le parallélisme inhérent et la croissance exponentielle des nombres de portes et de blocs logiques ont donné une grande popularité pour les circuits FPGA dans l'implémentation des algorithmes de commande d'identification et d'analyse de plusieurs systèmes réels de différents domaines d'application.

Vu que la machine asynchrone représente l'un des systèmes physiques les plus complexes, l'apparition de ces outils a donné un grand souffle pour améliorer la commande de celle-ci En effet l'amélioration de la commande du MAS conduit à utiliser des algorithmes de commande plus compliqués dont leur réalisation dépasse parfois les capacités des outils à base de microprocesseurs. De là vient l'utilité d'exploiter la puissance de traitement des circuits FPGA pour aller loin dans la commande des MAS.

C'est dans ce contexte que s'inscrit l'objectif du présent travail qui tente de faire un premier pas dans l'exploitation des circuits FPGA pour la commande d'un MAS. En effet, un FPGA est utilisé pour implémenter la commande scalaire en boucle ouverte. Cette commande repose sur la technique de commande MLI d'un onduleur triphasé. Par conséquent, en utilisant le langage de description VHDL sous l'environnement de développement de XILINX cette technique a été décrite synthétisée est implémentée.

Pour valider la réalisation effectuée différents test ont été exécuté.

Le présent manuscrit s'articule sur trois chapitres.

Le premier chapitre est consacré à l'état d'art des circuits reconfigurables FPGA. On traitera l'architecture interne de ces derniers et une comparaison des FPGA et les circuits séquentiels classiques puis les différents modes de prototypage de ces circuits.

Le second chapitre présente brièvement des rappels sur le modèle de la machineAsynchrone tout en utilisant une commande scalaire pour ajuster ses performances.

Le dernier chapitre est consacré à l'implémentation de la commande scalaire en boucle ouverte. L'implémentation envisage principalement la réalisation de la commande MLI en utilisant un circuit FPGA de type SPARTAN-3e. Cette commande permettra de varier la vitesse d'un moteur asynchrone suivant le principe de la commande scalaire (U/F) constant.

CHAPITRE I

L'ETAT D'ARTE DES FPGA ET LEURS APPORTS A LACOMMANDE DES MACHINES

I.1)-Introduction

Le début des circuits intégrés a commencé après l'invention du transistor aux années quarante du siècle passé. Depuis, le monde de l'électronique numérique a entré dans le concept d'intégrer sur un même substrat plusieurs éléments électroniques (transistors, résistances, capacités), c'est l'ère des circuits intégrés. Avec le temps, la capacité d'intégration a vu une énorme évolution, d'après Gordon Moore, le taux d'intégration des transistors dans ces circuits double tous les deux ans. Actuellement, ils existent des circuits numériques très complexes avec des centaines de millions de composants dans une surface très réduite. Avec une telle capacité d'intégration, il est devenu possible de réaliser et d'implémenter des fonctions très compliquées avec des vitesses et des précisions satisfaisantes. Arrivant à cette étape, une question a été posée, concernant la possibilité de réaliser l'analogique avec le numérique. La réponse à cette question était avec l'arrivé des circuits logiques reconfigurables, en particuliers les circuits FPGA. Ces circuits sont susceptibles d'être reconfigurés (Architecture programmée modifiable) partiellement ou entièrement suivant l'application. Via ces circuits, la logique programmable séquentielle a été substituée par une logique câblée tout en restant dans le monde numérique mais avec l'avantage d'exécuter, à la fois, des processus séquentiels et combinatoires. Ces propriétés permettent aux circuits FPGA de résoudre plusieurs problèmes complexes dont la puissance de traitement dépasse la capacité des outils de traitement à base de microprocesseurs.

Ce premier chapitre s'inscrit dans un contexte qui traite les FPGA et leurs positions au sein des autres systèmes digitaux. [1]

I.2)-Les différents types des circuits numériques

La grande partie des circuits numériques programmables sont issues de la célèbre architecture «Von-Neumann » proposée par John Von Neumann en I945 et porte son nom. Ensuite une autre architecture qui vienne pour compenser les lacunes de la précédente dans certains domaines et afin d'améliorer la cadence de calcule où le facteur temps d'exécution est le plus important. Cette architecture nommée «Architecture Harvard» qui porte le nom de l'université américaine qui la proposée sachons que parallèlement l'architecture «Von-Neumann » n'est pas figée mais on évolution. Mais le besoin croissant de composants très rapides a orienté les chercheurs à développer une autre solution qui sera complètement différente des deux précédentes architectures. Cette solution réside dans le mode de programmation qui est devenu architectural à logique câblée inversement aux deux premières architectures précédentes où la programmation est séquentielle

D'une manière générale, il existe deux familles de circuits numériques :

- **Une famille logicielle**: Elle est nommée aussi solutions programmables du type «Processeur » où un traitement séquentiel relativement lent et programmation dépendante du composants. Cette famille regroupe les Microprocesseur, les Microcontrôleur et les DSP.

- **Une famille matérielle:** Elle est nommée aussi solutions programmables du «type Logique »où un traitement parallèle en temps réel et une programmation architecturale avec un langage de description matériel HDL (Méthodologie deConception CAO) indépendante du composant. Cette famille regroupe les ASIC, les CPLD, les FPGA, etc.

Les caractéristiques principale de ces circuits sont (pour une comparaison complète, voir le tableau I.1) :

- **Les circuits du type DSP/Microprocesseurs** : Un rapport performance/coût faible, un temps de conception très court et une grande souplesse d'utilisation.

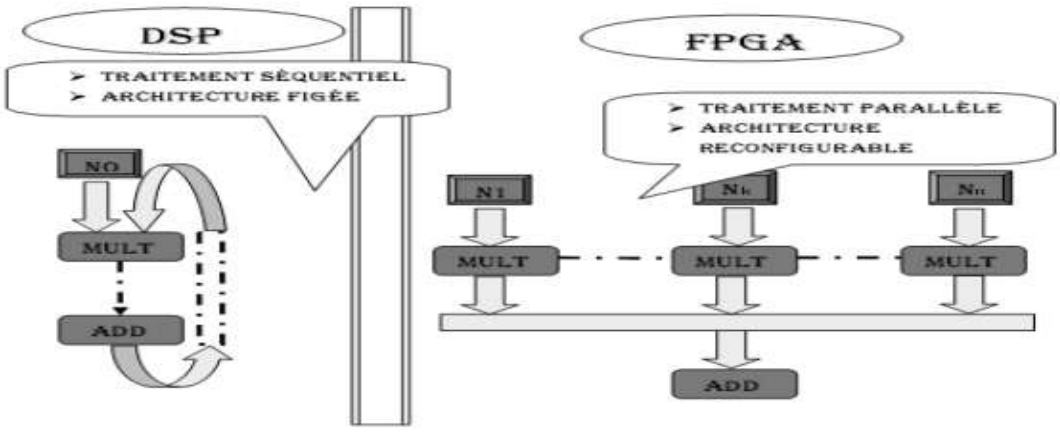
- **Les circuits du type spécialisé ASIC**: Très performants mais avec un cycle de conception long et une architecture figée.

- **Les circuits du type FPGA:** Des performances proches des ASIC, un coût unitaire intermédiaire et un cycle de conception moyen et une architecture modifiable.[1]

	ASICS	FPGA	DSP
Performance	Très élevées	Elevées	Faible
Taille	Faible	Moyen	Faible
Consommation	Faible	Moyen	Très élevées
Intégration	Système sur puce	Système sur puce	Composants supplémentaires
Souplesse	Fonction figées	Reconfigurable	Programmable
Misse en œuvre	Complexe	Complexité moyenne	Complexité moyenne
Cout du composant	Très élevées	Moyen	Faible

Tableau (I.1): Comparaison des différentes solutions numériques.

Le fonctionnement du mode séquentiel dépassent aujourd'hui 2 GHz, la réduction du temps de cycle ne suffira pas à compenser les insuffisances de ce mode de fonctionnement. La fréquence d'horloge ou de fonctionnement des FPGA est relativement faible devant les microprocesseurs et les DSP et ne dépasse pas quelques centaines de Mégahertz, mais cette faiblesse est largement compensée et même surpassée grâce au parallélisme de traitement. L'exploitation du parallélisme est une technique en pleine expansion dans les circuits numériques FPGA qui sont une alternative des DSP.



Figure(I.1): Schéma comparatif d'un DSP et d'un FPGA.

Ces circuits séquentiels ont une architecture matérielle figée et inversement pour les circuits logiques reconfigurables FPGA, on fait une adaptation de l'architecture du composant en fonction des l'algorithme.

I.3)-Les circuits logiques programmables

A l'époque la distinction était nette entre le logiciel et le matériel, le circuit logique programmable FPGA est venu s'introduire comme un hybride entre les deux approches. Les circuits logiques programmables et reprogrammables architecturalement sont classifiés en trois grandes familles les PLD, CPLD et FPGA. La figure (I.2) illustre les différents types de circuits reconfigurables où :

- **Les PLD (Programmable LogicDevice):**Famille des circuits programmables qui comprend les PAL, GAL.

  - **PAL (Programmable ArrayLogic):** Circuits logiques programmables dans lesquels seules les fonctions ET sont programmables, les fonctions OU ne le sont pas.

  - **GAL (GenericArrayLogic):**Circuits logiques PAL reprogrammables à technologie CMOS.

- **Les CPLD ou EPLD (Erasable Programmable LogicDeVice):**Circuits logiques reprogrammables

- **Les FPGA (Field Programmable GateArray)** : Ces circuits sont une évolution des
  CPLD. Récemment, ils intègrent également des mémoires entières, des multiplieurs Et même des noyaux de processeur.

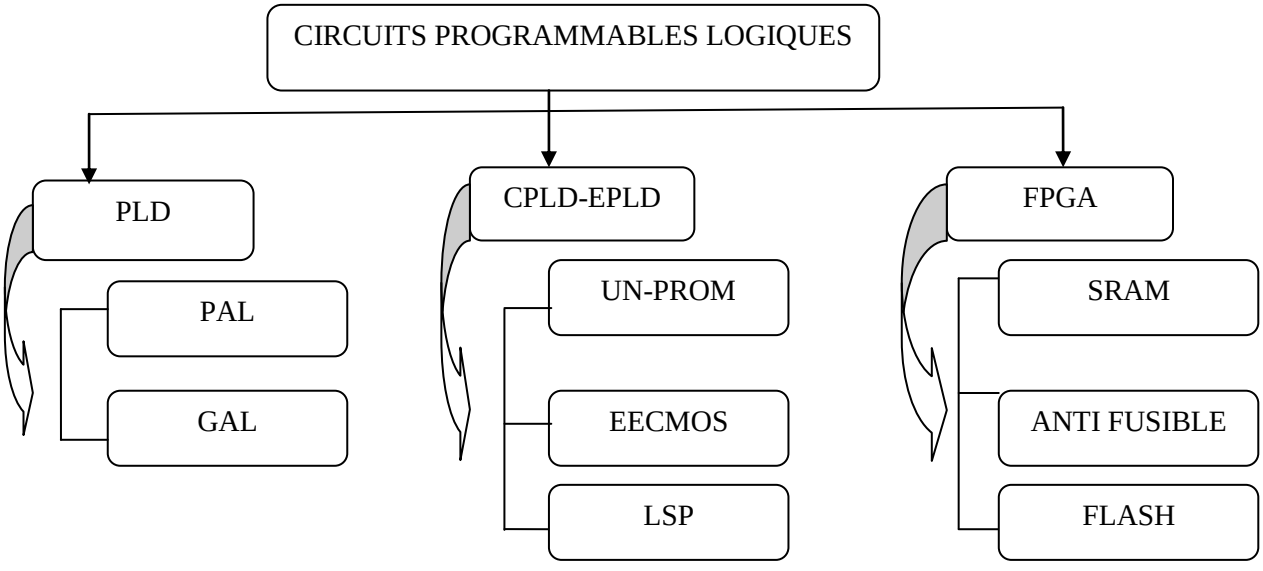


Figure (I.2) : Diagramme des différents types de circuits logiques programmables.

I.4)-Les circuits FPGA

Les FPGA sont développés récemment grâce aux progrès de la technologie VLSI, l'apparition de ce type de circuits est une révolution des systèmes digitaux et ouvrants des perspectives de traitement numérique inaccessibles auparavant. La fin des années 80 a vu l'apparition des premiers circuits FPGA qui sont des circuits intégrés que l'on peut configurer en un temps relativement court pour réaliser n'importe quelle fonction logique « câblée » à bas coût par une programmation de ses cellules logiques et ses interconnexions avec une restriction de ne pas épuiser les ressources du FPGA. Typiquement, un circuit FPGA haute densité peut contenir jusqu'à plusieurs millions d'éléments programmables. Pour réussir une application à base d' FPGA et afin d'obtenir un système plus performant, consommant un minimum de puissance, il est nécessaire de respecter un certain nombre de règles comme :

- Bien connaître les caractéristiques du FPGA ciblé pour assurer son adéquation avec les besoins du projet.
- Elaborer une méthodologie de conception.
- Maîtriser les outils d'implémentation et de choisir des outils de synthèse de qualité.

La conception sur les circuits FPGA est un challenge dans lequel l'objectif est de trouver le bon compromis entre densité, flexibilité et performances temporelles.[1]

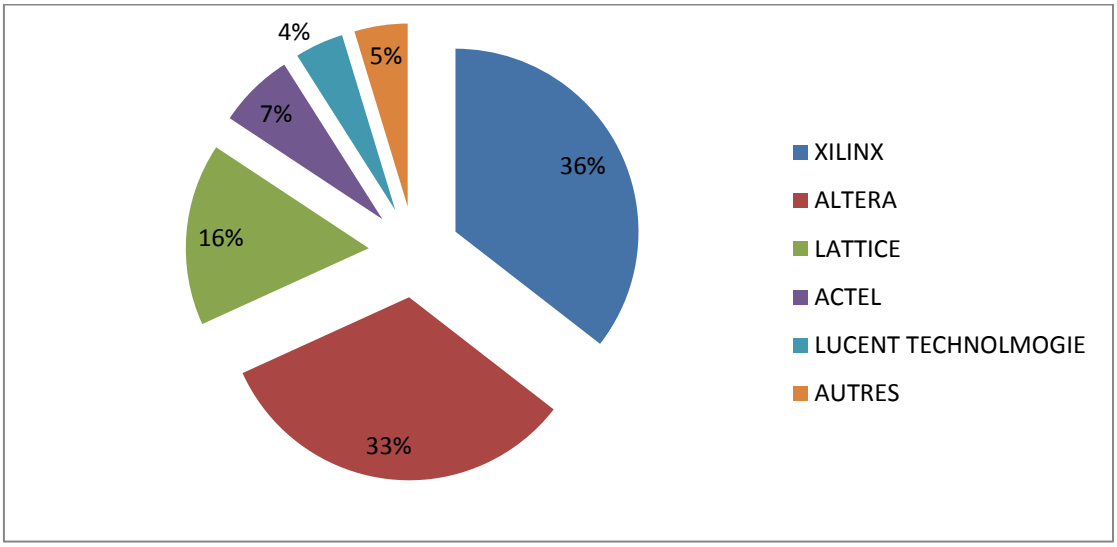
I.4.1)-Domaines d'application des FPGA

Les FPGA ont fait révolutionner certain domaines de contrôle numérique et de plus en plus utilisés pour intégrer des architectures numériques complexes. Ils sont devenus les plus populaires en matière d'implantation et de prototypage des circuits numériques après leur apparition sur le marché en I984.La clé de leurs réussites est l'aspect de programmation de ces derniers. Leurs utilisations actuelles couvrent les deux domaines : civil et militaire. Parmi ces applications nous citons :

- **Informatique :** Périphériques spécialisés.
- **Machinerie industrielle:** Contrôleur pour machines.
- **Télécommunications :** Traitement d'images, Filtrage.
- **Instrumentation** Équipement médical, Prototypage.
- **Transport :** Contrôle d'avions et métros.
- **Aérospatiale :** Satellites.
- **Militaire:** Radar, Communication protégée, la détection ou la surveillance.[1]

I.4.2)-Principaux vendeurs d'FPGA

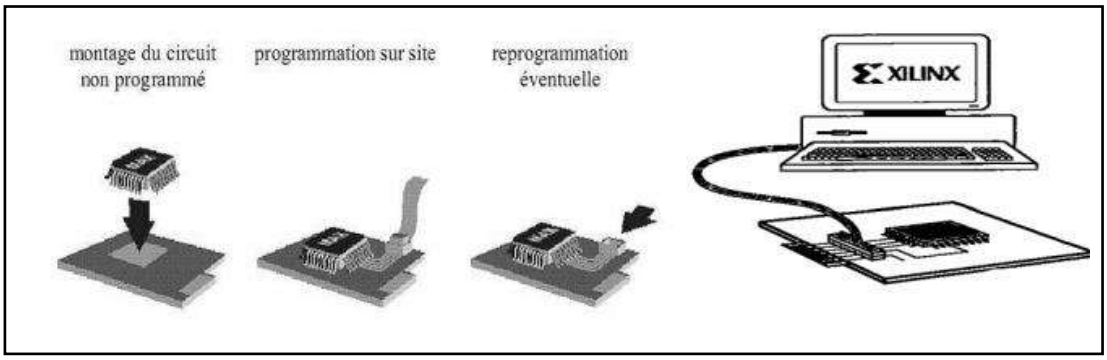
Les fabricants des FPGA ne cessent pas d'améliorer leurs produits par l'efficacité et la puissance. L'ensemble des firmes (Principaux fondeurs) qui conçoivent ce type de circuits sont : **Actel ,Altera , Atmel ,Cypress, Lattice, Minc , QuicLogic, Xilinx** et d'autres (Fig. I.3).



Figure(I.3) : Statistiques du marché occupé par les vendeurs d'FPGA.

I.4.3)-Configuration et reconfiguration des FPGA

Un système reconfigurable est un système qui est constitué de composants ou entités à architecture modifiable afin de répondre à un objectif bien déterminé. Ce système reconfigurable dispose d'un mécanisme permettant de choisir une nouvelle configuration et de la mettre en place dans le cadre du processus de reconfiguration. Les circuits FPGA sont un type de ces circuits reconfigurables. Ils sont programmables ou configurables sur les cartes sur quelles ils sont implantés par l'utilisateur. Le FPGA est une abréviation anglaise de qui signifie réseau des portes programmables sur site ce qui est décrit dans la figure suivante.[2]



Figure(I.4): Reprogrammable sur site d'un FPGA.

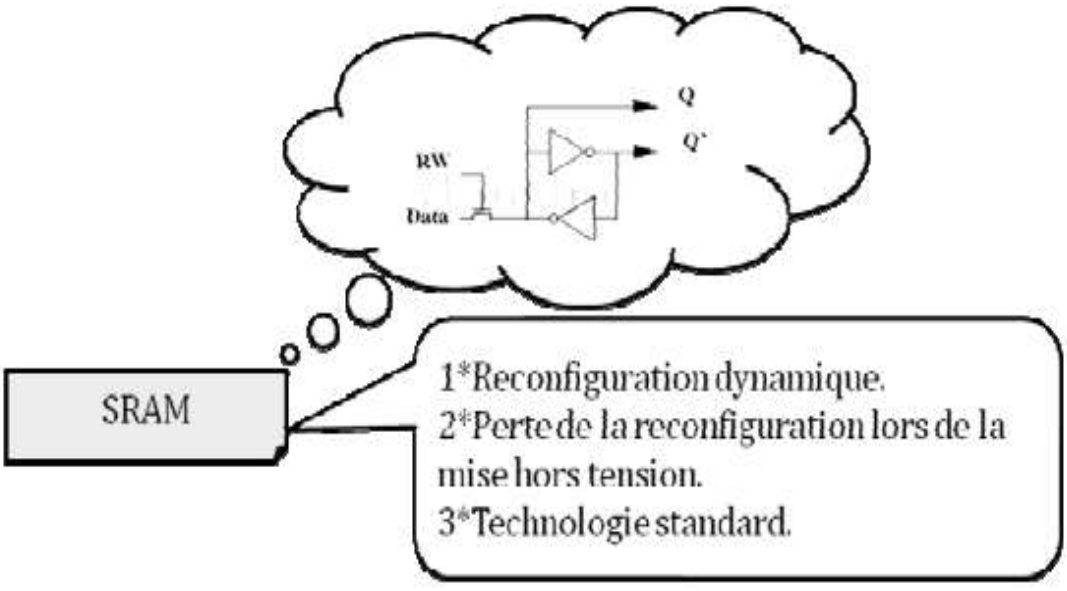
I.4.4)-Technologies de programmation des FPGA

Pour franchir les inconvénients susmentionnés des mémoires, et dans le but de faire un ensemble de technologies complémentaire adaptable suivant l'environnement des cahiers de charges, il existe trois types d'FPGA reprogrammables suivant la technologie de mémorisation pour répondre aux différentes applications. Ces trois principales technologies d'FPGA sont :

- Technologie de programmation par RAM.
- Technologie de programmation par EEPROM ou FLASH.
- Technologie de programmation par ANTI-FUSIBLE.[3]

I.4.4.1)-Technologie a base de ram (XILINX et ALTERA)

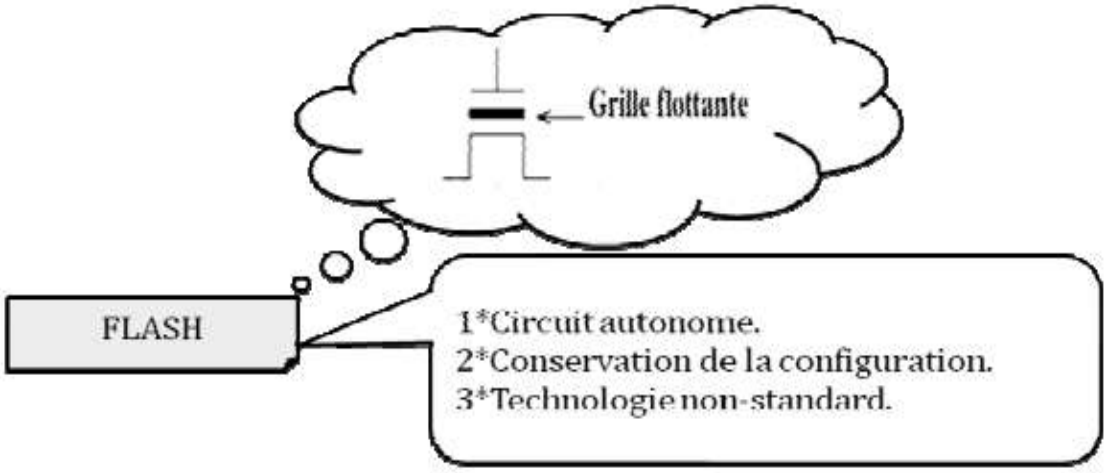
Cette technologie permet d'avoir une reconfiguration rapide des FPGA (Fig. I.5).Les points de connexions sont des ensembles de transistors commandés. L'inconvénient majeur de cette technologie c'est qu'elle nécessite beaucoup de place et il est nécessaire de sauvegarder le design du FPGA dans une autre mémoire Flash.



Figure(I.5): Caractéristiques des technologies SRAM.

I.4.4.2)- Technologie a base d'EEPROM ou FLASH (LATTICE ET ACTEL)

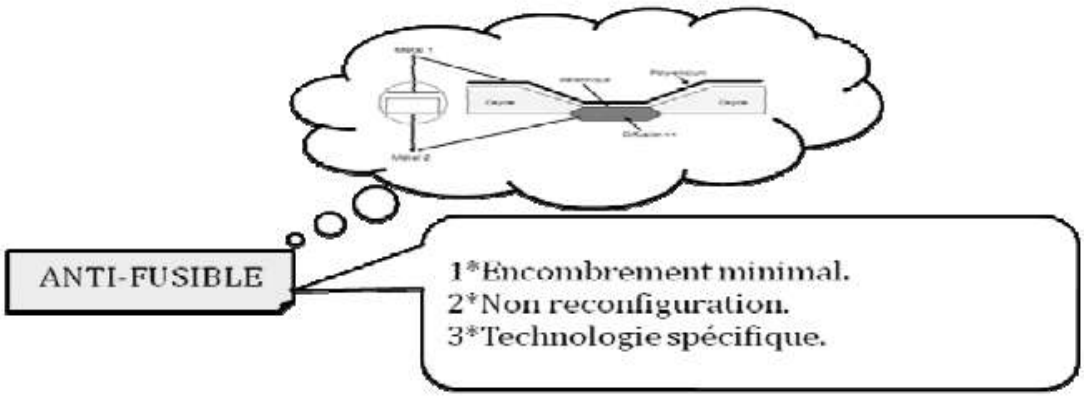
Cette technologie garde sa configuration mais un nombre limité de configuration Avec une configuration plus lente par rapport à SRAM (Fig. I.6).



Figure(I.6): Caractéristiques des technologies FLASH.[4]

I.4.4.3)- Technologie a base d'anti-fusibes(ACTEL)

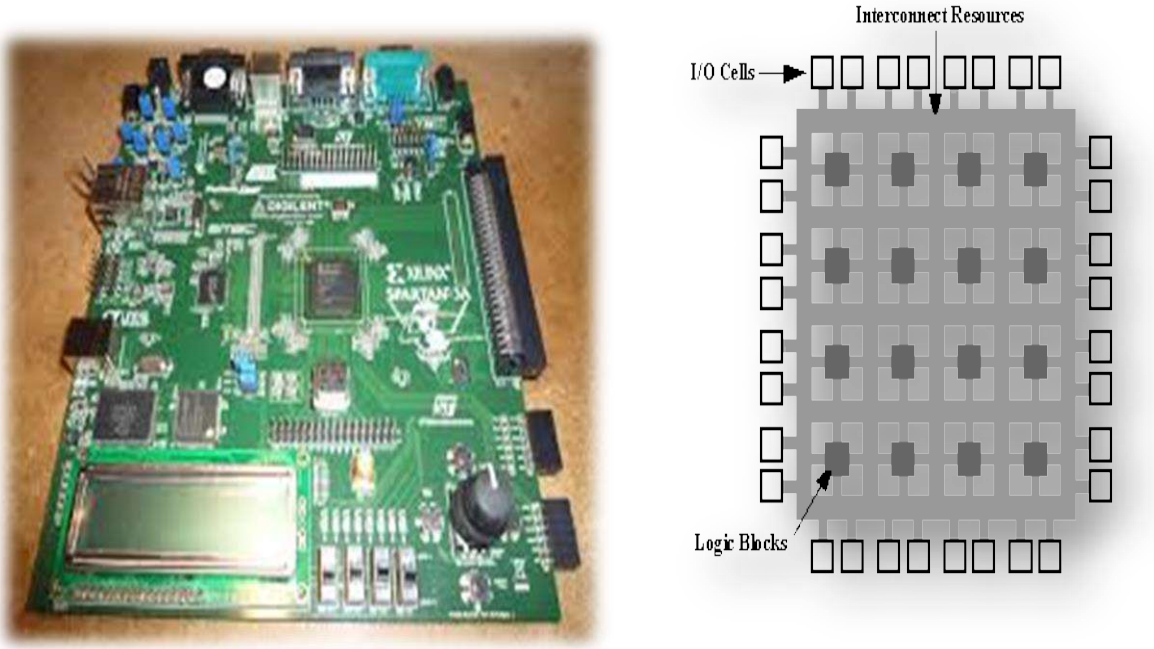
Les points de connexions sont du type ROM, c'est-à-dire que la modification du point est irréversible. Pour comprendre le mécanisme de connexion sans rentrer dans les détails des semi-conducteurs, on considère que le point de connexion est le point de rencontre de deux segments conducteurs ou lignes conductrices. Le non anti-fusible vient du fait que l'état initial du fusible ou la couche isolante est présent et il n'y a pas de contact pour l'établir il faut détruire le fusible ce qui est contradictoire au fonctionnement habituel d'un fusible. Des composants moins génériques mais plus petits et plus rapides ont été développés.



Figure(I.7): Caractéristiques des technologies ANTI-FUSIBLES.

1.4.5) -Architecture interne des FPGA

Un FPGA (Field-Programmable GateArray) est un circuit intégré avec une structure adaptable par l'utilisateur et composée d'un réseau de cellules élémentaires ou d'éléments logiques programmables (CLB) et un ensemble d'entrées/sorties (IOB) (Fig. 1.8). Les CLB répartis régulièrement et reliés entre eux grâce à des connections qui forment une matrice de routage programmable pour obtenir un comportement spécialisé du circuit.Les composants logiques programmables sont des circuits composés de nombreuses cellules logiques élémentaires librement assemblables. Celles-ci sont connectées de manière définitive ou réversible par programmation afin de réaliser les ou les fonctions numériques désirées. Puisque tous les éléments sont programmables, le FPGA peut émuler et réaliser n'importe quel circuit a l'unique condition que celui-ci n'épuise pas les ressources logiques et de routage du FPGA. L'architecture interne des FPGA est différente d'un fondeur à un autre et même entre les différentes gammes du même constructeur mais rien n'empêche que leurs ressemblances peuvent être rassemblées dans le schéma représentatif de la figure suivante :[4]



Figure(I.8): Architecture du FPGA.

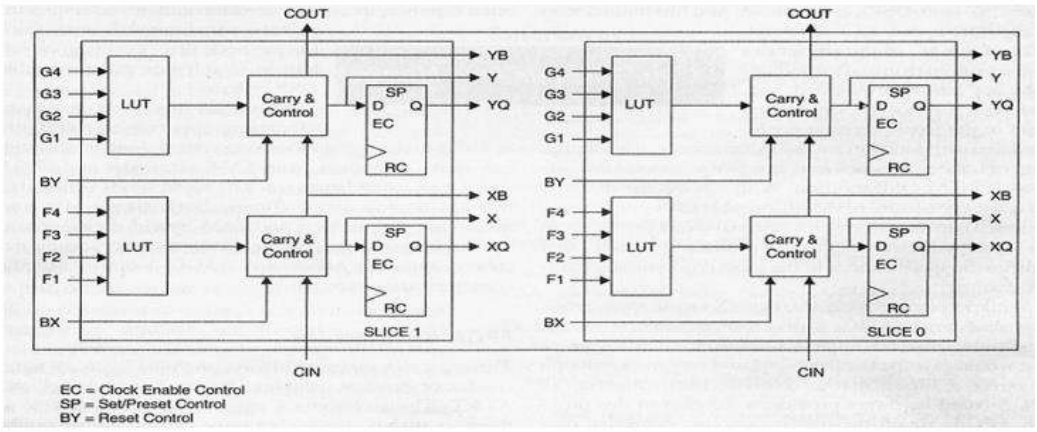
Les macro-cellules internes sont appelées :

- Soit CLB qui est la dénomination adoptée par Xilinx et abréviation anglaise de «Configurable Logic Block », signifiant bloc logique configurable.
- Soit LC qui est le nom choisi par Cyprès et abréviation anglaise de « LogicCell », signifiant cellule logique.
- Soit LE qui c'est l'appellation d'Altéra abréviation anglaise de « Logic Elément» signifiant élément logique.

Les macro-cellules sur la périphérie sont appelées :IOB abréviation anglaise de «Input Output Block », signifiant bloc logique d'entrées sorties.

L'ensemble des points de connexion est appelé PIP, abréviation anglaise de «Programme InterConnect Points ».

La granularité des FPGA par les macro-cellules CLB nous permet d'implémenter des fonctions logiques « combinatoires ou séquentielles » complexes car chaque CLB est constitué d'une partie combinatoire et d'une partie séquentielle (Fig. 1.9). Chaque fonction est décomposée en petites fonctions booliennes qui peuvent être contenues par de petites cellules élémentaires SLICES.Ces dernières comportent des LUT pour la partie combinatoire et une ou des bascules (généralement de type D) pour la partie séquentielle. [4]



Figure(I.9): Architecture d'un CLB de familles Vertex du Xilinx.

I.4.6)-Avantages et inconvénients des FPGA

Les principaux avantages et inconvénients des FPGA sont résumés dans le tableau(I.2)

Les avantages	Les inconvénients
-Technologie «facile » à maîtriser. -Temps de développement réduit. -Reprogrammable. -Idéal pour le prototypage. -Coût peu élevé. -Parallélisme de traitement. -Flexibilité et la possibilité de réduire fortement les délais de développement et commercialisation. -La reconfiguration, parfois en temps réel.	- Performances non optimisées. -Temps de réponse long par rapport aux ASIC.

Tableau (I.2): les avantages et les inconvénients de FPGA.[4]

I.5)-Langage de description VHDL

Le VHDL (Vhsic-VerryHight Speed Integrated Circuit- Hardware Description Language) est l'un des langages de description matérielle le plus utilisé pour la description, La simulation, et la synthèse des circuits logique (circuits électroniques) de grand densité d'intégration et travaillant à haut vitesse. Le VHDL est un langage moderne, puissant et général à cause de sa lisibilité, à sa modularité, à sa sécurité d'emploi et à la fiabilité de ses descriptions [4]. Il autorise plusieurs méthodologies de conception (comportemental, flot de données, structurel) tout en étant d'un très haut niveau d'abstraction en électronique (il est indépendant de la technologie utilisée : FPGA, CPLD, ASIC, etc.). Cette abstraction permet d'ailleurs de le simuler sur ordinateur avant de programmer la moindre puce. Historiquement, le développement

du langage VHDL a passé par plusieurs phases, parmi lesquelles :

• 1980: Début du projet, inancé par le DoD (400M $US)

• 1982: Contrat pour Intermetrics, IBM et Texas

• 1985:Version 7.2 dans le domaine public

• 1987: Standard IEEE 1076 (VHDL-87)

• 1993: Nouvelle version du standard (VHDL-93)

• 2001: Nouvelle version du standard (VHDL-2001)

• 2008: Nouvelle version du standard (VHDL-2008) [5]

I.5.1)-Structure générale d'une description VHDL

Une description VHDL est composée de 3 parties indissociables à savoir (Fig. I.10)

- **Bibliothèque IEEE, (**Library de déclaration)
- **L'entité (ENTITY),** elle définit les entrées et sorties.
- **L'architecture (ARCHITECTURE),** ellecontient les instructions VHDL permettent de réaliser le fonctionnement attendu.

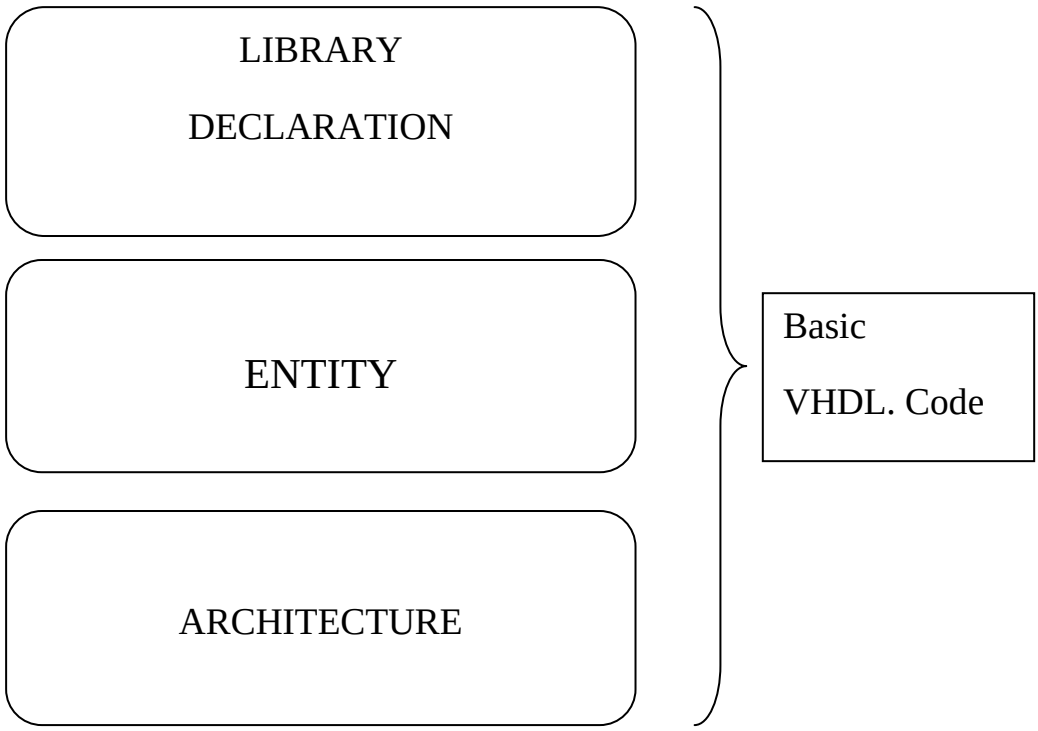


Figure (I.10) : Différent block de langage VHDL.

I.6)-Prototypage des FPGA

Il existe différents moyens pour programmer ou configurer un circuit FPGA, parmi lesquels on peut citer :

- Programmation en utilisant la conception HDL.
- Programmation en utilisant l'interface Matlab-Simulink-XSG (Xilinx System Genrator).

I.6.1)-La programmation des FPGA par les outils HDL

Avant d'arriver à l'étape de configuration d'un circuit FPGA, Le flot de conception d'un design regroupe plusieurs niveaux d'abstraction (Fig I.11). Dans chaque niveau, le concepteur s'intéresse à la résolution d'un problème. Pour les circuits FPGA, ils existent des outils **CAO** spécifiques quisontutiliséspourréaliserles différents niveaux d'abstraction. On trouve l'environnement ISE-XILINX pour les circuits FPGA de la firme XILINX, l'environnement QUARTUS pour les circuits FPGA de la firme ALTERA, etc. [6]

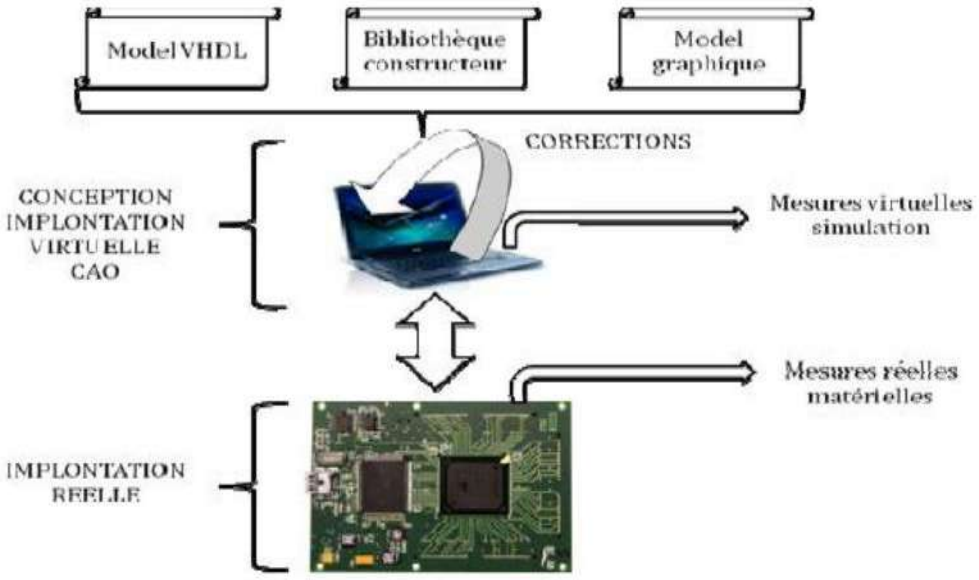


Figure (I.11) : Mode d'exécution matériel de la CAO.

Communément, le développement d'une application sur FPGA suit l'enchainement des étapes suivantes :

I.6.1.1)-Spécification du design

- Le nombre de broches d'entrée-sortie et leur localisation dans la puce **FPGA.**
- La spécification de la fréquence d'horloge du système.
- La spécification de la mémoire requise pour l'application.

I.6.1.2)-Conception du design

C'est une étape essentielle à travers laquelle le design ou l'application envisagée est décrite en utilisant un outil de description matériel approprié, nommé langage HDL. En effet, deux langages HDL sont largement utilisés, il s'agit du VHDL et du Verilog. Ces deux langages sont standardisés, portables et compatibles avec toutes les types des **FPGA**. En plus de la description, certains signaux de test sont créés par le concepteur, dans cette étape, pour simuler le comportement de la description.

I.6.1.3)-Synthèse

La synthèse est le processus qui convertit la représentation du design à partir du code HDL fourni pour produire une représentation au niveau porte logique. Elle s'occupe de déterminer quelles sont les structures susceptibles de répondre au cahier des charges étudié et de produire un code booléen unique sous forme d'un fichier.

I.6.1.4)-Placement et routage

A partir des fichiers de synthèse, l'outil de conception procède au placement et routage. Un algorithme de routage est sensé de faire l'aiguillage des données qu'il reçoit vers leurs destination par action sur les nœuds de routage ce qui est équivalent à définir les chemins qui relient l'ensemble des **CLB** contenus dans la fonction désirée. Ces algorithmes de routage sont différents d'un concepteur à un autre. Plusieurs traitements sont nécessaires pour obtenir un fichier de configuration. [7]

- **Partitionnement**:
- **Placement**
- **Routage**
- **Génération des données numériques de**

I.6.1.5)-Intégration et implémentation

L'implémentation est la réalisation proprement dite qui consiste à mettre en œuvre l'algorithme sur l'architecture du circuit configurable cible, c'est-à-dire à compiler, charger, puis lancer l'exécution sur un circuit FPGA. C'est une étape de programmation physique et de tests électriques qui clôture la réalisation du circuit,

La figure (I.12) résume l'ensemble de ces étapes précédentes.

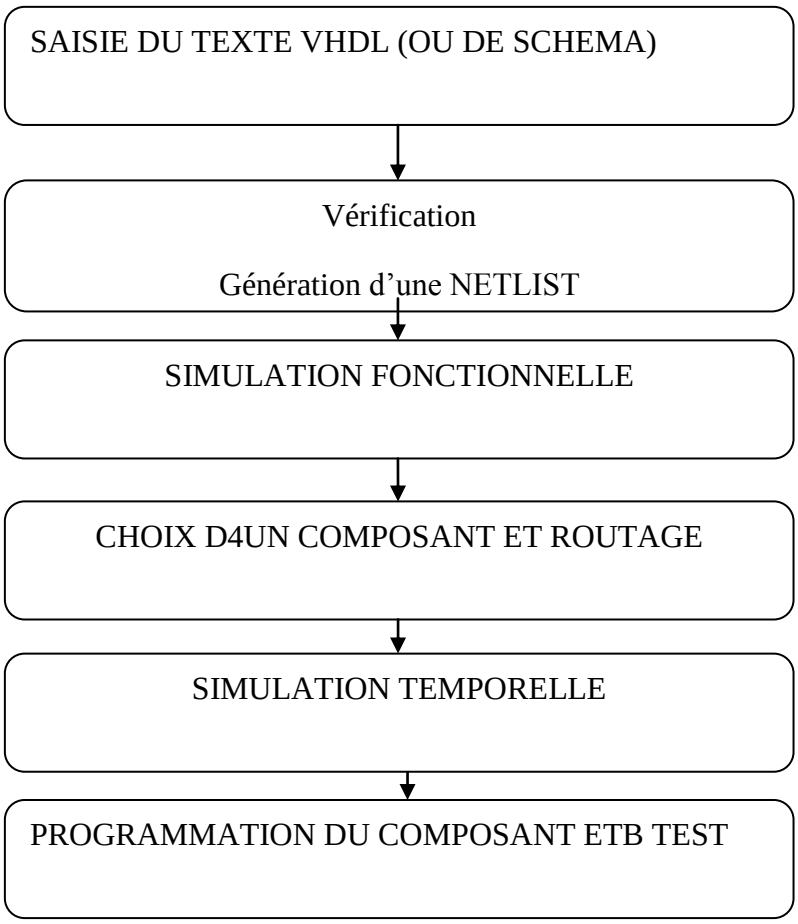
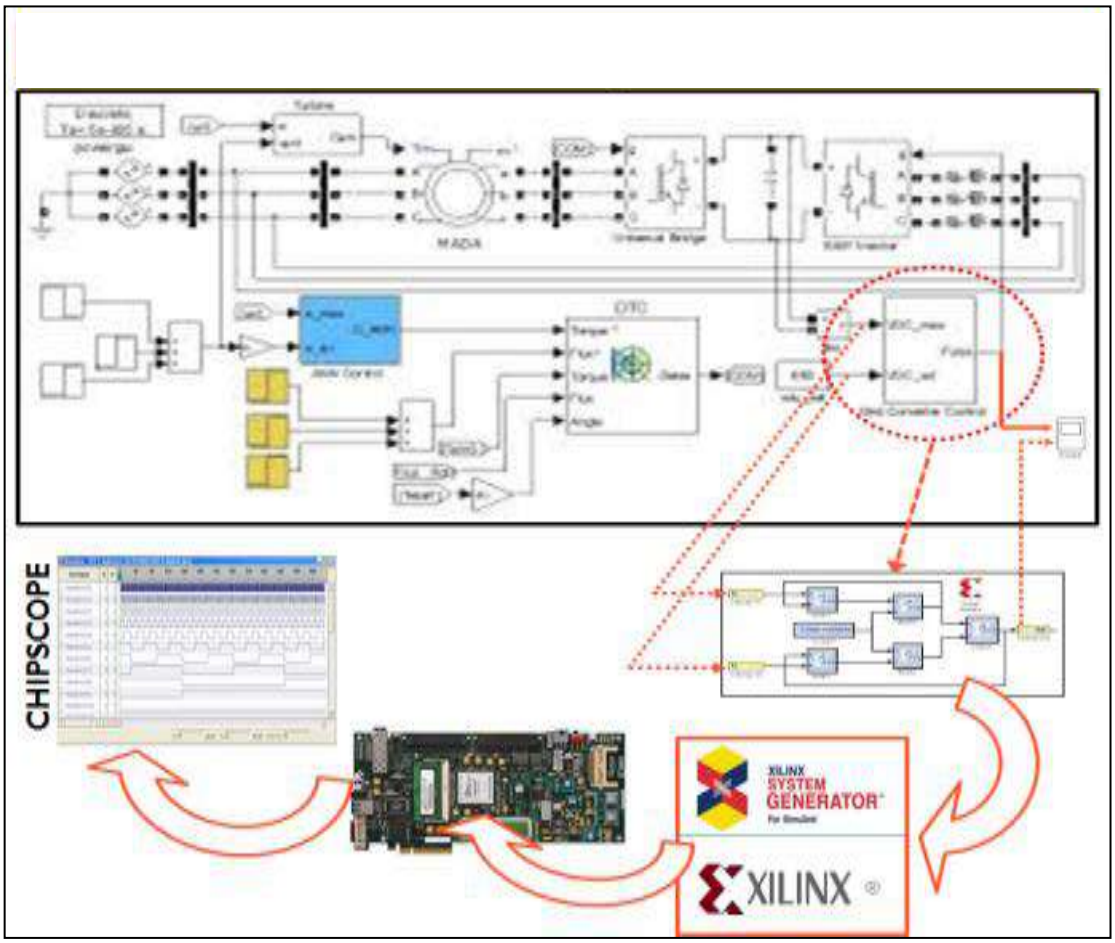


Figure (I.12) : les étapes de développement une application sur FPGA.

I.6.2)-Laprogrammation des FPGA parXSG/MATLAB-SIMULINK

Malgré la puissance des outils traditionnels de conception matérielle (HDL), pour certaines applications, ils sont compliqués à mettre en œuvre et nécessitant des professionnels. D'autre part, les langage matériels ne sont pas très populaires dans le domaine de la commande automatique des systèmes où l'environnement MATLAB/Simulink est le plus utilisé. Heureusement, avec l'incorporation du Toolbox 'Xilinx-System-Generator (XSG)', proposé par Xilinx, dans l'environnement Matlab/Simulink, il est devenu possible de programmer, de simuler et d'implémenter les descriptions matérielles en utilisant l'environnement graphique de Simulink. En plus de cet avantage, le XSG permet d'exécuter ce qu'on appelle 'the Hardware In the Loop (HIL), le Co-design et la Co-simulation. En effet, ces opportunités sont très appréciées du fait qu'elles permettent de tester virtuellement un algorithme de commande ou une partie d'un système compliqué sont aller directement sur le matériel et évitant, par conséquent, les dégâts pouvant être engendrés [7].



Figure(I.13): Hardware Co-simulation XSG.

I.7)-Conclusion

Dans ce chapitre on a donné une brève aperçue sur les circuits numériques reconfigurables. En effet, après présenter l'historique d'évolution dans le monde des circuits numériques, on a donné illustré les différentes caractéristiques et spécification des circuits FPGA tout en les comparant avec les circuits numériques standards. Par la suite, les différents outils de développement utilisés avec ces circuits ont été décrits. A rappelé, la programmation d'un circuit FPGA peut être achevé par les langages de description matériel ou l'environnement Matlab-Simulink en utilisant le Xilinx-System-Generator. Pour notre cas, le langage de description Matériel VHDL avec le software ISE-Xilinx sera utilisé pour réaliser le reste de ce projet.

CHAPITRE II

LA COMMANDE SCALAIRE DE LA MAS

II.1)-Introduction

Les machines électriques ou machines tournantes sont des dispositifs électriques Qui permettent de générer un mouvement ou une énergie mécanique à partir d'une énergie électrique et inversement. Elles occupent une place prépondérante dans tous les Secteurs industriels. Les machines électriques tournantes sont classées en trois catégories qui sont les machines à courant continu, les machines synchrones et les machines asynchrones.

Depuis son invention et sa découverte par NICOLA TESLA, la machine asynchrone est devenue l'actionneur le plus important parmi les machines tournantes de nos jours. Elle est caractérisée par sa simplicité de conception, de fabrication, d'entretien, de robustesse et peu coûteuse avec un excellent rendement. Le moteur asynchrone a des avantages assez connus et assez nombreux. Par contre, il est difficile et onéreux d'en faire une variation de sa vitesse lors du fonctionnement. C'est la raison pour laquelle, le moteur à courant continu est resté longtemps le plus utilisé dans les entraînements à vitesse variable. Avec le développement de l'industrie et les techniques de commande numérique en Temps réel, le moteur asynchrone suscite de plus en plus d'intérêts. Nous consacrerons ce chapitre au moteur asynchrone triphasé qui est notre application. Le principe de fonctionnement est d'abord étudié puis les éléments de contrôle et les boucles de régulation seront décrits et finalement nous présenterons la commande qui pourrait être implantable sur les FPGA. [13]

II.2)-Constitution et principe de fonctionnement

Les moteurs asynchrones, appelles moteur à induction, sont pratiquement tous des moteurs triphasés. Ils sont basés sur l'entraînement d'une masse métallique par l'action d'un champ tournant. En termes de constitution, le moteur asynchrone comprend différents éléments (Fig. II.1).

Le stator ou l'inducteur, se constitue de trois enroulements (bobines) parcourus par des courants alternatifs triphasés et possède un certain nombre de paires de pôles. Les courants alternatifs dans le stator créent un champ magnétique tournant à la pulsation de synchronisme résultant des fréquences des tensions d'alimentation. Pour le rotor, il n'est relié à aucune alimentation tournant avec une pulsation légèrement inférieure à celle du synchronisme. Le rotor peut être à cage d'écureuil constitué de barres conductrices court-circuitées ou bobiné dont l'accès peut assurer par l'intermédiaire de trois bagues et trois balais. Le rotor à cage d'écureuil est plus utilisé en fonctionnement moteur, tandis que, le rotor bobiné est plus utilisé en fonctionnement générateur.

Concernant le principe de fonctionnement, les courants alternatifs dans le stator créent un champ magnétique tournant à la pulsation de synchronisme et le rotor tourne à une vitesse plus petite que la vitesse de synchronisme .On dit que le rotor «glisse» par rapport au champ tournant et ce glissement dépend de la charge. L'interaction entre les courants induits dans le rotor et le champ magnétique tournant génère un couple électromagnétique permettant de faire tourner le rotor. [5]

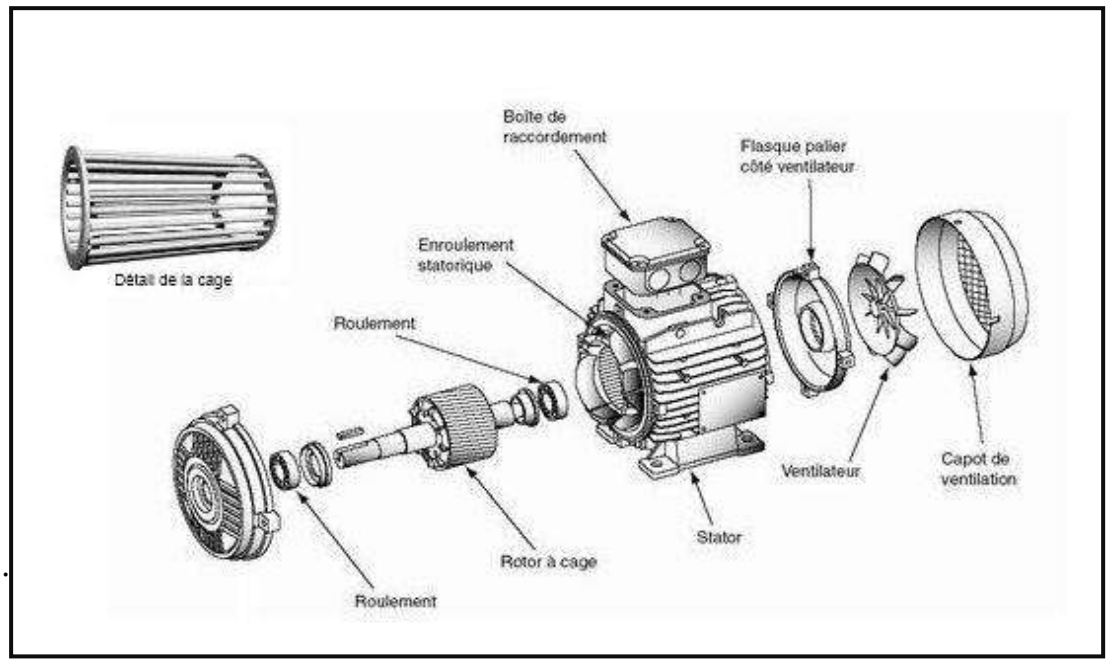


Figure (II.1) : constitution de moteur asynchrone à cage

Le moteur possède plusieurs avantages, entre autres, on trouve :

- La robustesse.
- La simplicité de construction.
- Le bas coût.
- Le rendement important

Toutefois et, en dépit de ces avantages, le moteur asynchrone pose un problème avec sa dynamique qui fortement non-linéaire et couplé. A cet effet, la commande de ce moteur a été un dur défit pour ses adaptes. [9]

II.3)-Modélisation de la machine asynchrone

Avant d'entamer le volet modélisation du moteur asynchrone, il est nécessaire d'adopter certaines hypothèse simplificatrice, telles que :

- La saturation du circuit magnétique n'est pas considérée
- L'effet d'hystérésis est négligé
- La construction mécanique est parfaitement équilibrée
- L'entrefer est lisse
- La dispersion du champ magnétique aux deux bouts de la machine est négligeable.

Au niveau de la méthode de modélisation mathématique, nous considérerons que stator est composé de trois bobines identiques (A, B et C) régulièrement déphasées de $2\pi/3$ radians électriques, la phase A sera prise comme référence angulaire. Le rotor sera considéré comme équivalent à trois bobines identiques (a, b et c) court-circuitées sur elles- de mêmes régulièrement déphasées de $2\pi/3$ radians électriques.[6]

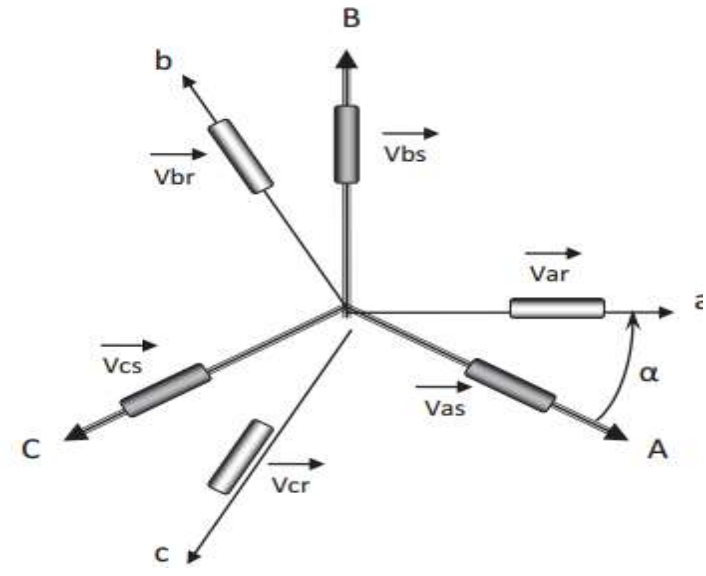


Figure (II.2) Représentation schématique d'une machine asynchronetriphasée.

II.4)Equations générales

La machine asynchrone est définie par trois types d'équations à savoir :

- Les équations électriques.
- Les équations magnétiques.
- Les équations mécaniques.

- Equations électriques

Le système matriciel électrique suivant dans le repère a, b, c :

$$\begin{bmatrix} V_{sa} \\ V_{sb} \\ V_{sc} \\ V_{ra} \\ V_{rb} \\ V_{rc} \end{bmatrix} = \begin{bmatrix} R_s & 0 & 0 & 0 & 0 & 0 \\ 0 & R_s & 0 & 0 & 0 & 0 \\ 0 & 0 & R_s & 0 & 0 & 0 \\ 0 & 0 & 0 & R_s & 0 & 0 \\ 0 & 0 & 0 & 0 & R_s & 0 \\ 0 & 0 & 0 & 0 & 0 & R_s \end{bmatrix} \cdot \begin{bmatrix} i_{sa} \\ i_{sb} \\ i_{sc} \\ i_{ra} \\ i_{rb} \\ i_{rc} \end{bmatrix} + \frac{d}{dt} \cdot \begin{bmatrix} \phi_{sa} \\ \phi_{sb} \\ \phi_{sc} \\ \phi_{ra} \\ \phi_{rb} \\ \phi_{rc} \end{bmatrix} \quad \text{(II.01)}$$

Ou de manière plus raccourcie :

$$[V_{sabc}] = [R_s].[i_{sabc}] + \frac{d}{dt}.[\phi_{sabc}] \quad \text{(II.02)}$$

$$[V_{rabc}] = [R_r].[i_{rabc}] + \frac{d}{dt}.[\phi_{rabc}] \quad \text{(II.03)}$$

Équations magnétiques

Les grandeurs magnétiques au stator et au rotor dans le repère a, b, c :

$$\begin{bmatrix} \phi_{sa} \\ \phi_{sb} \\ \phi_{sc} \\ \phi_{ra} \\ \phi_{rb} \\ \phi_{rc} \end{bmatrix} = \begin{bmatrix} l_s & M_s & M_s & M_1 & M_3 & M_2 \\ M_s & l_s & M_s & M_2 & M_1 & M_3 \\ M_s & M_s & l_s & M_3 & M_2 & M_1 \\ M_1 & M_3 & M_2 & l_r & M_r & M_r \\ M_2 & M_1 & M_3 & M_r & l_r & M_r \\ M_3 & M_2 & M_1 & M_r & M_r & l_r \end{bmatrix} \cdot \begin{bmatrix} i_{sa} \\ i_{sb} \\ i_{sc} \\ i_{ra} \\ i_{rb} \\ i_{rc} \end{bmatrix} \quad \text{(II.04)}$$

Pour l'écriture condensée, on met:

$$[L_{ss}]=\begin{bmatrix} l_s & M_s & M_s \\ M_s & l_s & M_s \\ M_s & M_s & l_s \end{bmatrix} \qquad \text{(II.5)} \qquad [L_{rr}]=\begin{bmatrix} l_r & M_r & M_r \\ M_r & l_r & M_r \\ M_r & M_r & l_r \end{bmatrix} \qquad \text{(II.6)}$$

On aura :

$$[M_{sr}]=[M_{rs}]^{t}=M_{sr}\begin{bmatrix} \cos(\alpha) & \cos(\alpha+2\pi/3) & \cos(\alpha-2\pi/3) \\ \cos(\alpha-2\pi/3) & \cos(\alpha) & \cos(\alpha+2\pi/3) \\ \cos(\alpha+2\pi/3) & \cos(\alpha-2\pi/3) & \cos(\alpha) \end{bmatrix} \qquad \text{(II.07)}$$

On aura finalement :

$$[V_{sabc}] = [R_s].[i_{sabc}] + \frac{d}{dt}.([L_{ss}].[i_{sabc}] + [M_{sr}].[i_{rabc}]) \qquad \text{(II.08)}$$

$$[V_{rabc}] = [R_r].[i_{rabc}] + \frac{d}{dt}.([L_{rr}].[i_{rabc}] + [M_{rs}].[i_{sabc}]) \qquad \text{(II.09)}$$

-Équations mécaniques

Il faut ajouter l'équation de mouvement au système d'équations différentielles Pour étudier les phénomènes transitoires électromécaniques avec une vitesse rotorique variable (par exemple le démarrage, le freinage, la variation de la charge à l'arbre, etc….).

$$J.\frac{d\Omega_r}{dt}=C_e - C_r - f\Omega_r \qquad \text{(II.10)}$$

La vitesse électrique du rotor est donnée par l'expression suivante :

$$w_r = p\Omega_r \qquad \text{(II.11)}$$

Cette modélisation précédente présente l'inconvénient de la présence des paramètres variables dépendant de l'angle de rotation mécanique (les matrices [M sr] et [Mrs]).

II.5)-transformation de « Park »

La transformé de Park est utilisé pour rendre le système (modèle de La machine) plus simple. On utilise alors cette transformation mathématique qui permet de décrire le comportement de la machine à l'aide d'équations différentielles à coefficients constants. Elle consiste à faire des projections sur deux axes orthogonaux de toutes les grandeurs du système (Fig. II.2). La condition qui permet de remplacer la machine triphasée par son modèle biphasé est sa symétrie. La matrice de Park et sont inverse sont les suivantes [10]

$$[P]=\sqrt{\frac{2}{3}}\begin{bmatrix}\cos\theta & \cos(\theta-\frac{2\pi}{3}) & \cos\left(\theta-\frac{2\pi}{3}\right)\\ -\sin\theta & -\sin(\theta-\frac{2\pi}{3}) & -\sin(\theta-\frac{4\pi}{3})\\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\end{bmatrix} \quad \text{(II.12)}$$

$$[P^{-1}]=\sqrt{\frac{2}{3}}\begin{bmatrix}\cos\theta & -\sin\theta & \frac{1}{\sqrt{2}}\\ \text{os}(\theta-\frac{2\pi}{3}) & -\sin(\theta-\frac{2\pi}{3}) & \frac{1}{\sqrt{2}}\\ \cos\left(\theta-\frac{2\pi}{3}\right) & -\sin(\theta+\frac{2\pi}{3}) & \frac{1}{\sqrt{2}}\end{bmatrix} \quad \text{(II.13)}$$

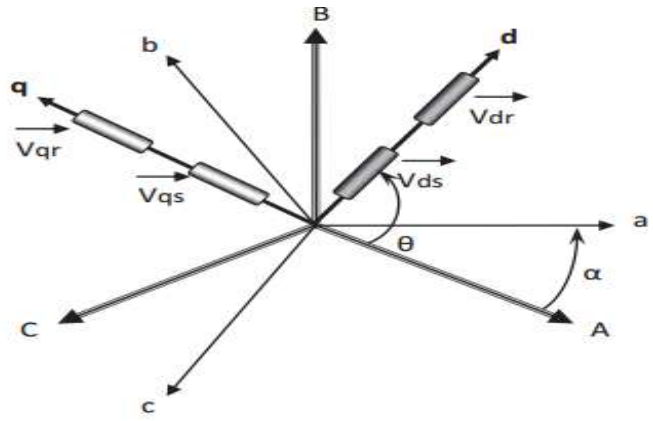


Figure (II.3) : Représentation du passage d'un système triphasé à celui biphasé.

- **Équations électriques**

$$V_{ds}=R_s.i_{ds}+\frac{d\phi_{ds}}{dt}-w_s.\phi_{qs} \quad \text{(II.14)}$$

$$V_{qs}=R_s.i_{qs}+\frac{d\phi_{qs}}{dt}-w_s.\phi_{ds} \quad \text{(II.15)}$$

$$V_{dr}=R_r.i_{dr}+\frac{d\phi_{dr}}{dt}-(w_s-w_{r)}.\phi_{qr}=0 \quad \text{(II.16)}$$

$$V_{qr}=R_r.i_{qr}+\frac{d\phi_{qr}}{dt}-(w_s-w_{r)}.\phi_{dr}=0 \quad \text{(II.17)}$$

-Equations magnétiques

$$\phi_{ds=L_s.i_{ds}}+M.i_{dr} \quad \text{(II.18)}$$

$$\phi_{ds=L_r.i_{dr}}+M.i_{ds} \quad \text{(II.19)}$$

$$\phi_{qs=L_s.i_{qs}}+M.i_{qr} \quad \text{(II.20)}$$

$$\phi_{ds=L_r.i_{dr}}+M.i_{ds} \quad \text{(II.21)}$$

-Équations mécaniques

$$J.\frac{d\Omega_r}{dt}=C_e-C_r-f\Omega_r \quad \text{(II.22)}$$

$$C_e=(3/2)p.M/L_r\,(i_{rd}.i_{sq}-i_{sd}.i_{rq}) \quad \text{(II.23)}$$

$$w_r=p\Omega_r \quad \text{(II.24)}$$

Avec.

J: Moment d'inertie du rotor.
f: Coefficient de frottement visqueux.
Ce: Couple électromagnétique.
Cr: Couple résistant.
P: Nombre de paires de pôles.

- **Modèle globale du moteur asynchrone**

Suivant la vitesse attribuée au référentiel biphasé, on distingue trois variantes pour le modèle du moteur asynchrone triphasé :

- Modèle lié stator ou au référentiel fixe

$$we = 0$$

- Modèle lié champ tournant

$$we = ws$$

- Modèle lié rotor

$$we = wr$$

En considérant le référentiel fixe, le modèle du moteur peut être donné par le système d'équations suivant :

$$\begin{cases} \dfrac{dI_{S\alpha}}{dt} = -\left(\dfrac{R_S}{\sigma L_S} + \dfrac{M^2}{\sigma L_S L_r}\right) I_{S\alpha} + \dfrac{M}{\sigma L_S L_r T_r}\Phi_{r\alpha} + \dfrac{M}{\sigma L_S L_r}\omega_r \Phi_{r\beta} + \dfrac{1}{\sigma L_S} V_{S\alpha} \\ \dfrac{dI_{S\beta}}{dt} = -\left(\dfrac{R_S}{\sigma L_S} + \dfrac{M^2}{\sigma L_S L_r}\right) I_{S\beta} - \dfrac{M}{\sigma L_S L_r}\omega_r \Phi_{r\alpha} + \dfrac{M}{\sigma L_S L_r T_r}\Phi_{r\beta} + \dfrac{1}{\sigma L_S} V_{S\beta} \\ \dfrac{d\Phi_{r\alpha}}{dt} = \dfrac{M}{T_r} I_{S\alpha} - \dfrac{1}{T_r}\Phi_{r\alpha} - \omega_r \Phi_{rq} \\ \dfrac{d\Phi_{r\beta}}{dt} = \dfrac{M}{T_r} I_{S\beta} - \dfrac{1}{T_r}\Phi_{r\beta} + \omega_r \Phi_{r\alpha} \\ \dfrac{d\Omega}{dt} = \dfrac{1}{J}(C_{em} - K_f \Omega - C_r) \end{cases}$$

Avec :

$$C_{em} = P\frac{M}{L_r}(\Phi_{r\alpha} I_{S\beta} - \Phi_{r\beta} I_{S\alpha})$$

Où :

$\sigma = 1 - \dfrac{M^2}{L_S L_r}$; Coefficient de dispersion total.

$T_r = \dfrac{L_r}{R_r}$; Constant du temps rotorique.

II.6)-Simulation du modèle moteur asynchrone triphasé

La machine étant alimentée par un système de tensions sinusoïdales, Va, Vb et Vc. Après démarrage, un couple de charge de 6N.m a été appliqué à l'instant 0.6s. Les figures (II.3, II.4, II.5, II.6) illustrent les réponses dynamiques du moteur avec les conditions de fonctionnement appliquées.

Pour la vitesse (Fig. II.4),en régime transitoire, on remarque un croissement linéaire de vitesse jusqu'à la vitesse nominale. Avec l'application de la charge, la vitesse chute et maintient sa nouvelle valeur vue l'absence de la commande. Pour le couple électromagnétique (Fig. II.4), dans l'intervalle de temps 0 et 0.6 on observe une allure caractéristique type de tous les moteurs asynchrones à cage. En charge, le moteur développe un couple équivalent au couple de charge appliqué. Pour les courants (II.5 et II.6), après un régime transitoire caractérisé par des fortes oscillations, les courants prennent des formes sinusoïdales.

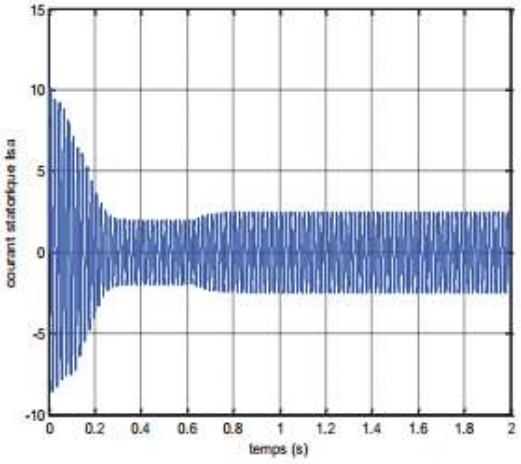


Figure (II.4): Courant statorique Isa

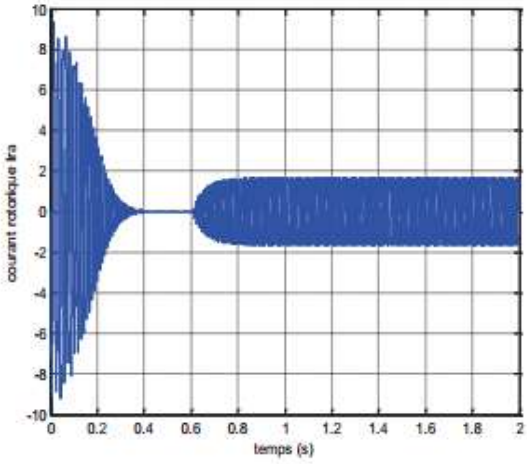


Figure (II.5): Courant rotorique Ira

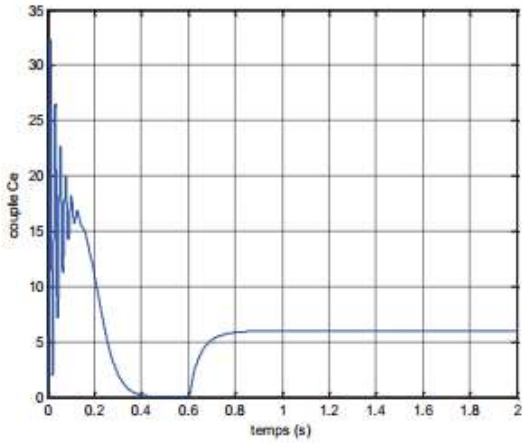


Figure (II.6):Couple électromagnétique Ce

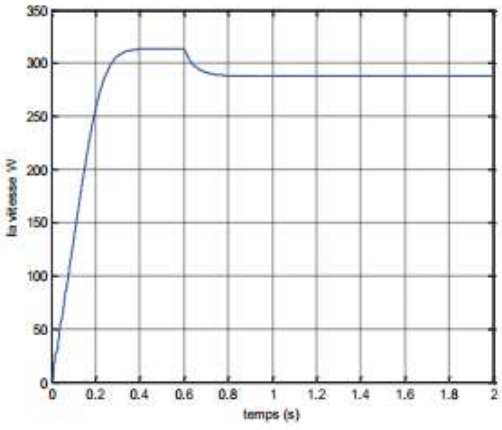


Figure (II.7):la vitesse de rotation n

II.7)-Commande scalaire du moteur asynchrone

La commande scalaire est la première commande qui a été appliquée pour varier la vitesse des moteurs asynchrone. Dans cette commande, on ne s'intéresse qu'à l'amplitude de la variable contrôlée et non à sa phase. Ondistingue deux type de commande scalaire [5, 6,7]

- La commande de scalaire indirecte ou le flux magnétique est contrôlé en en imposant le rapport amplitude /fréquence de la tension ou du courant
- La commande de scalaire direct ou le flux magnétique est contrôlé à partir de son estimation ou de sa mesure.

La deuxième méthode est plus difficile à mettre en pratique et nous nous intéressons uniquement à la première approche qui de par sa simplicité est la plus utilisée [5, 12,10]. Pour cette variante, on trouve de voies d'application : en boucle ouverte et en boucle fermée.

Pour la commande en boucle ouverte (Fig. II.8), la vitesse est variée directement en agissant directement sur l'onduleur tout en maintenant un rapport fixe entre la tension et la fréquence pour maintenir constante l'amplitude du flux du moteur.

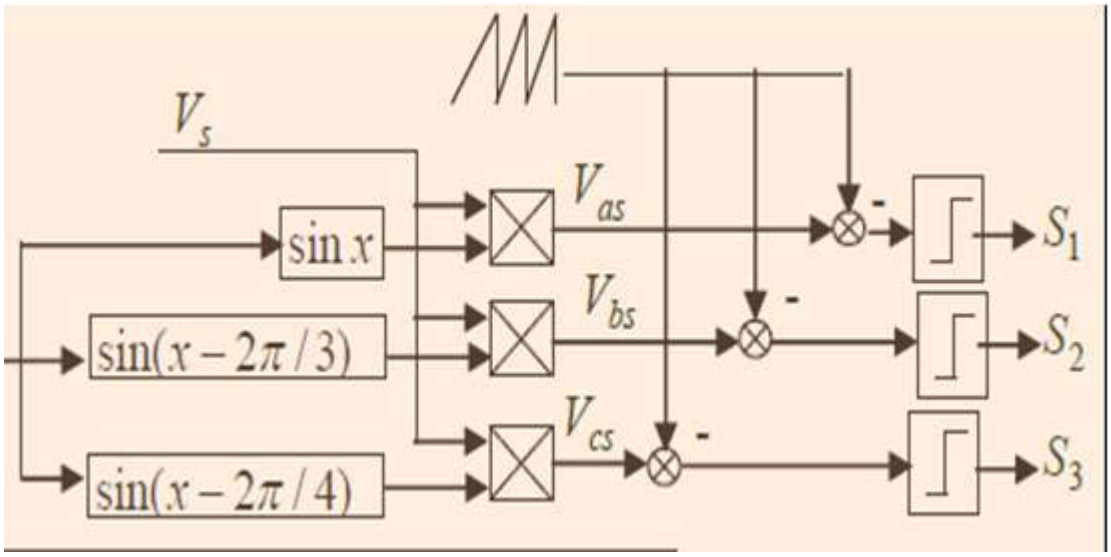


Figure (II.8):schéma général de la commande scalaire en boucle ouverte.

Pour la commande en boucle fermée (Fig. II.9), L'erreur de poursuite de la vitesse permet, par l'intermédiaire d'un régulateur PI, de générer la pulsation de glissement qui est de son tour ajoutée à la pulsation rotoriquepermet d'avoir la pulsation (fréquence) des tensionsstatoriques. La pulsation générée est utilisé à la fois pour calculer l'amplitude des tensionsstatoriques en utilisant la loi (u/f) constante ainsi que pour la commande de la fréquence de l'onduleur [1].

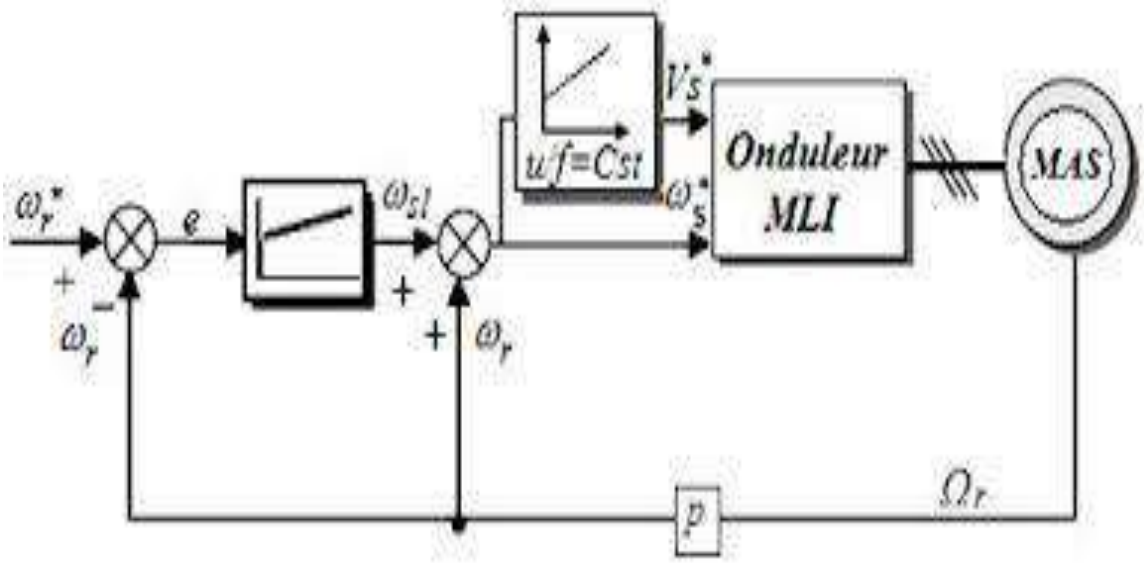


Figure (II.9):schéma général de la commande scalaire en boucle fermée.

La commande scalaire permet d'obtenir des résultats satisfaisants pour les régimes permanents de la vitesse. Toutefois, pour les régimes transitoires, les performances se dégradent où le flux oscille fortement avec des grandes amplitudes et son module est variable. Ceci explique pourquoi les relations utilisées pour calculer cette commande ne sont valables que dans le régime permanant. Ces oscillations vont se répercutent sur la qualité du couple et la vitesse. A cet effet, ce type de commande n'est donc employé que pour des applications où la variation de vitesse n'est pas grande telle en pompage ou en ventilation. Pour de meilleures performances, il faut faire recours à d'autres commandes plus performantes telles que la commande vectorielle. [2]

II.8) Résultat de simulation de la commande scalaire

Les résultats de simulation de la commande sont présentés sur les figures (10, 11, 12, 13, 14, 15).La vitesse a un temps de réponse très rapide de 0.22s, le couple électromagnétique suit bien le comportement de la vitesse.

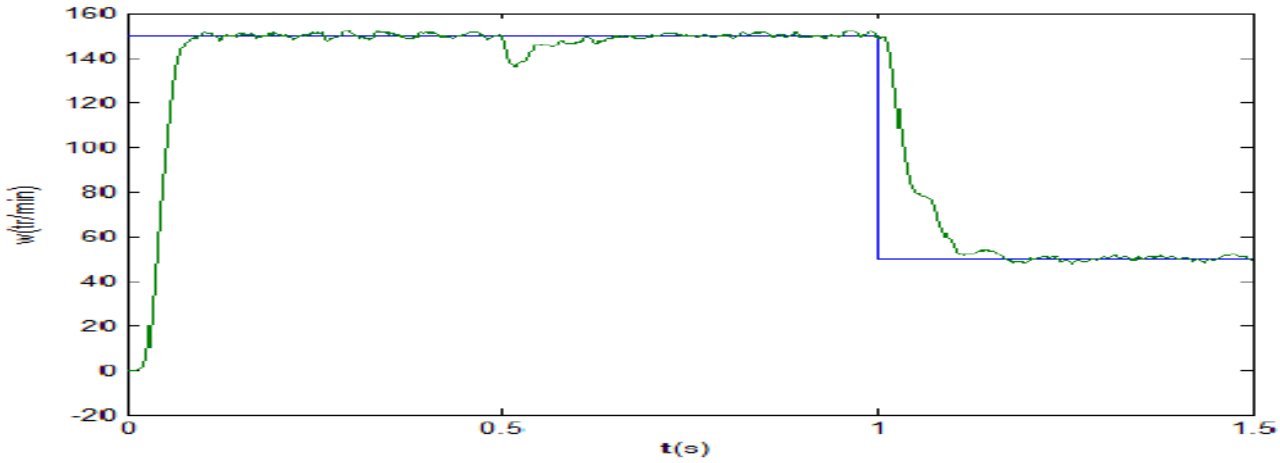


Figure (II.10):la vitesse de la MAS.

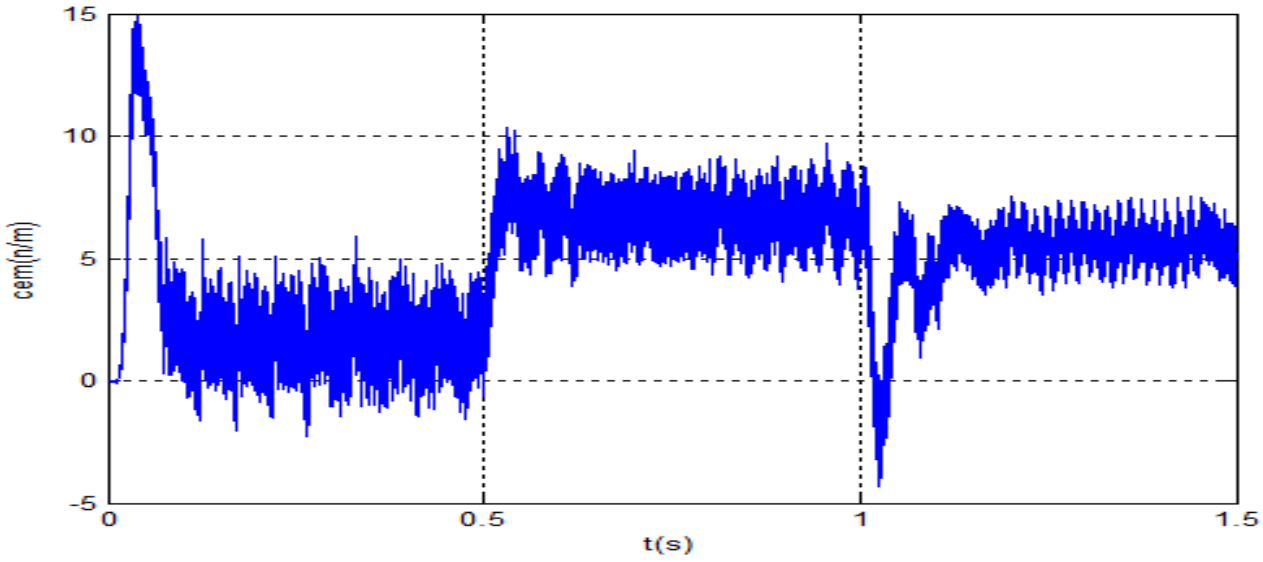


Figure (II.11):le couple électromagnétique de la MAS.

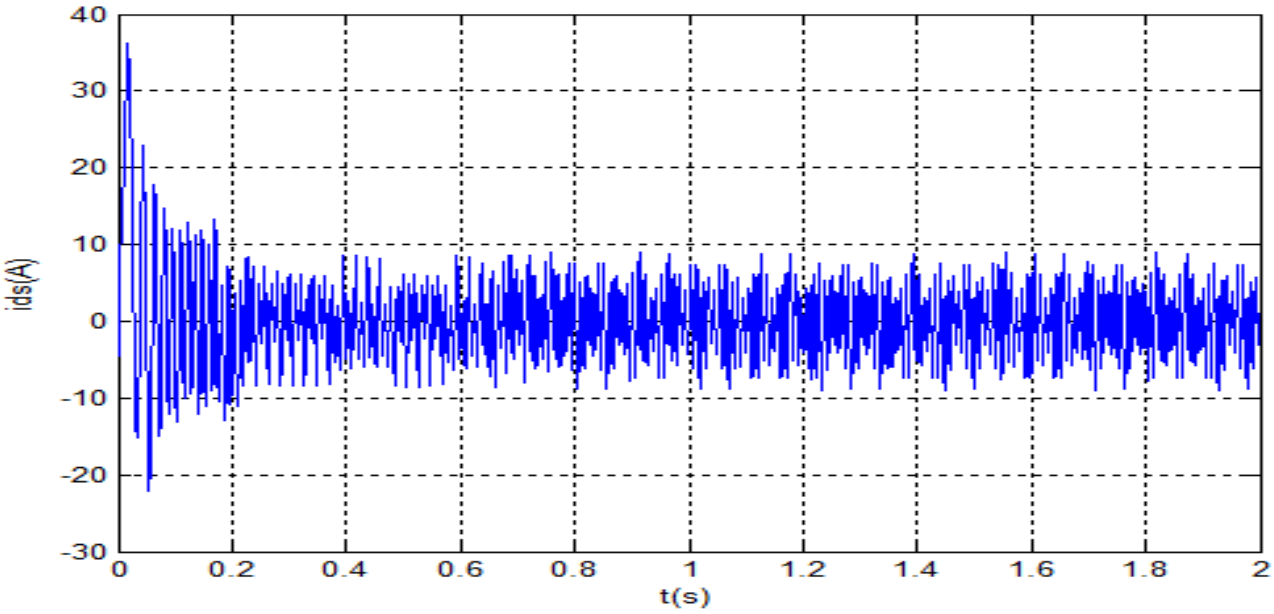


Figure (II.12):le courant statorique Ids de la MAS.

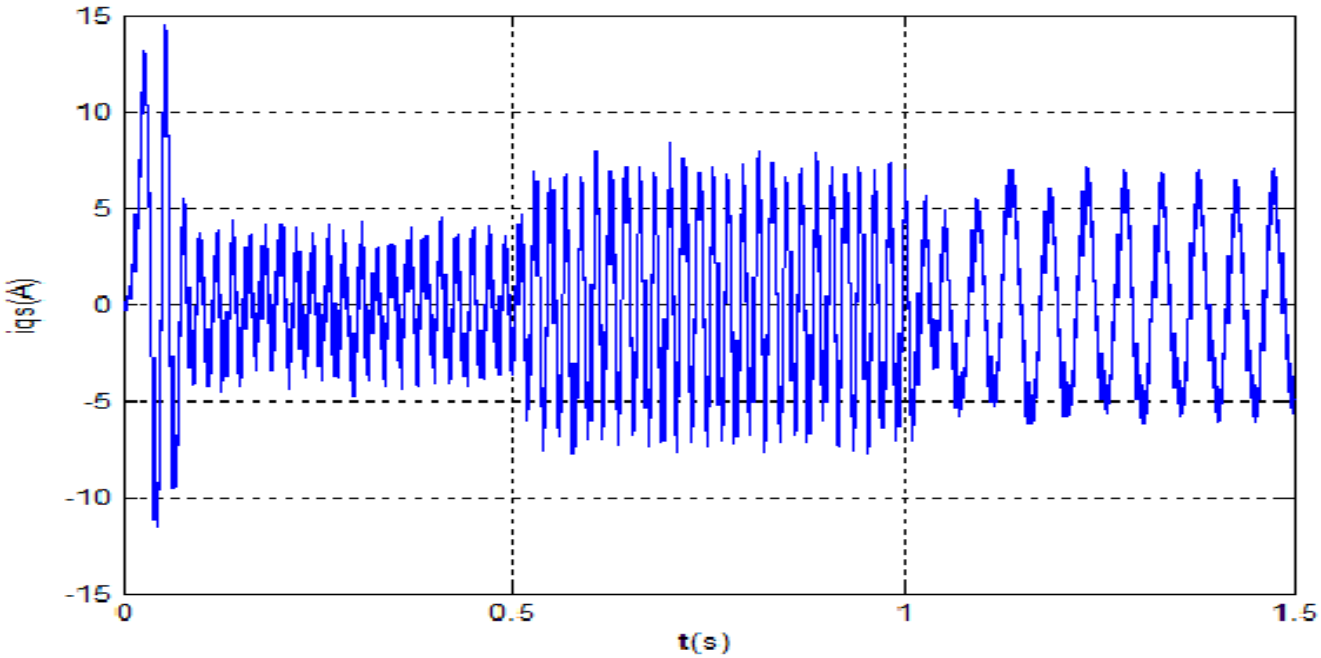


Figure (II.13):le courant stattorique Iqs de la MAS.

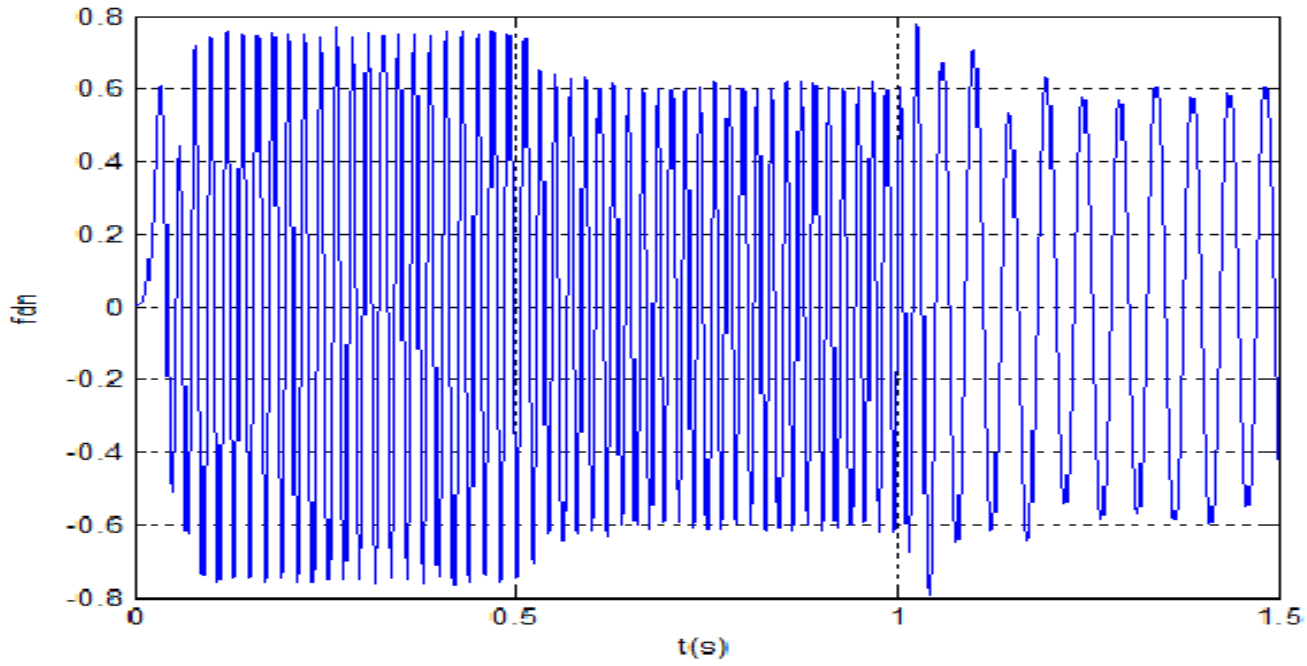


Figure (II.14):le flux statorique dans l'axe d.

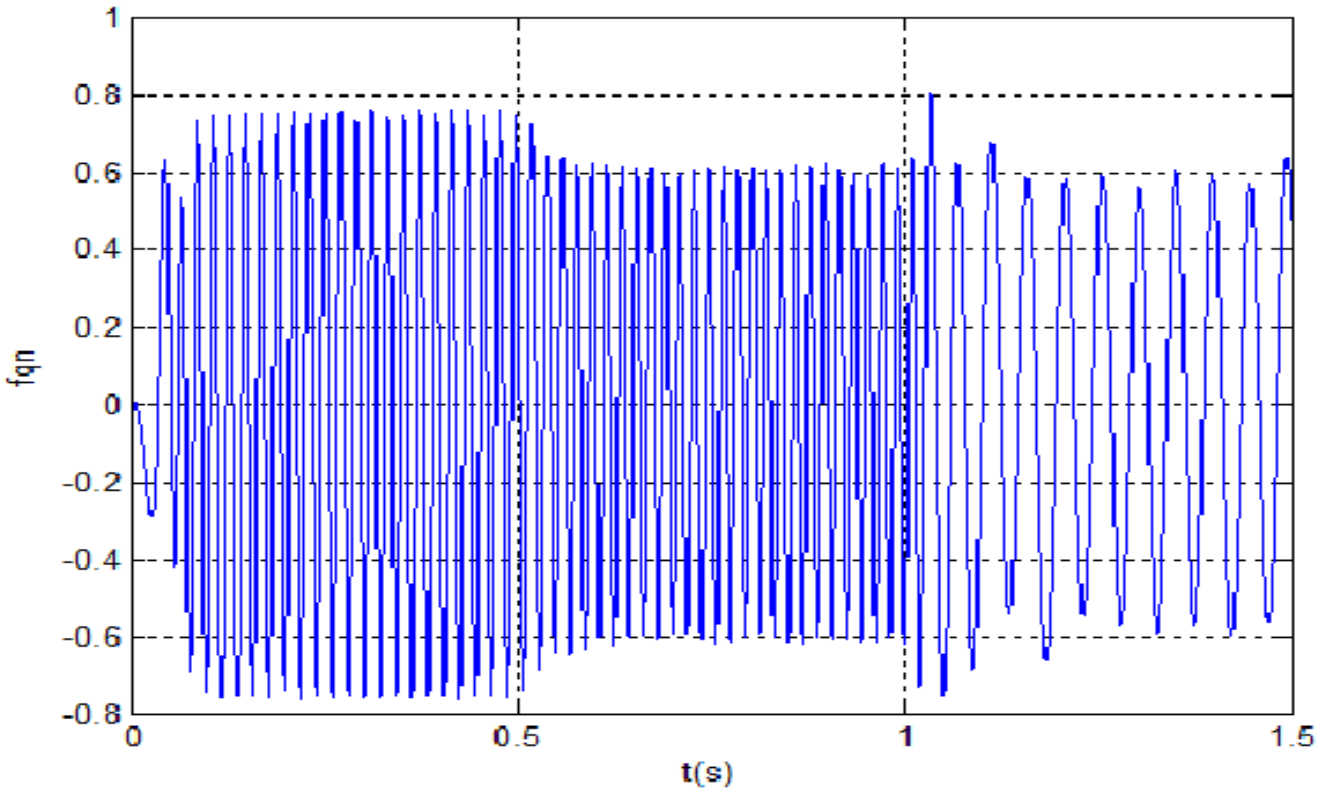


Figure (II.15):le flux statorique dans l'axe q.

II.9)-Conclusion

Dans ce chapitre nous avons présenté la modélisation du moteur asynchrone alimenté par une source de tension sinusoïdal triphasée et équilibrée à fréquence constante établi sous des hypothèses simplificatrices. Après, nous avons utilisé la transformation du Park pour éviter la complexité des équations différentielles. Par la suite, la commande scalaire a été appliqué pour commande le moteur asynchrone. En effet cette commande fera l'objet du chapitre suivant où un circuit FPGA sera utilisée pour l'implémenter.

CHAPITRE III

IMPLEMENTATION DE LA COMMANDE SCALAIRE D'UN MAS

III.1)-Introduction

Dans de nombreuses applications, le mode de fonctionnement séquentiel limite l'utilisation des microprocesseurs ou les microcontrôleurs pour mettre en œuvre les contrôleurs en raison du nombre de cycles machine nécessairespour l'exécution des instructions du programme. Par conséquent, au cours des dernières décennies, le traitement parallèle et le nombre élevé de portes ont conduit à une augmentation rapide de la popularité des circuits reconfigurables, en particulier les circuits FPGA, dans les systèmes embarqués en vue de substituer les moyens à base de microprocesseurs (microcontrôleurs, DSP)[11].

Pour profiter des avantages des circuits FPGA, le présent chapitre est consacré à l'implémentation de la commande scalaire en boucle ouverte. L'implémentation envisage principalement la réalisation de la commande MLI en utilisant un circuit FPGA de type SPARTAN-3e. Cette commande permettra de varier la vitesse d'un moteur asynchrone suivant le principe de la commande scalaire (U/F) constant.

III.2)-Descriptions de la stratégie(MLI)

Le principe de la stratégie MLI est de commander les bras de l'onduleur par une décision livrée par un algorithme au début de chaque période d'échantillonnage. Le principe se repose sur une simple comparaison, pour chaque bras, entre un signal de référence sinusoïdal (la modulante) et un signal triangulaire de fréquence plus élevée (la porteuse), on l'appel : MLI sinusoïdaux-triangulaire (Fig. 3.1). La fréquence de porteuse définie la fréquence de découpage et les points d'intersection entre la modulante et la porteuse correspondent aux instants de commutations de l'onduleur.

Il est bien entendu que la stratégie sinusoïdaux-triangulaire nécessite la génération des deux signaux de comparaison, à savoir, les trois signaux sinusoïdaux at le signal triangulaire. Pour réaliser ces taches, des descriptions VHDL sont développés dans l'environnement XILINX. Il faut noter, que cet outil de développement permet d'exécuter toutes les étapes d'implémentation de l'application envisagé, de la description jusqu'à la programmation du circuit en passant par la simulation et la synthèse.[12]

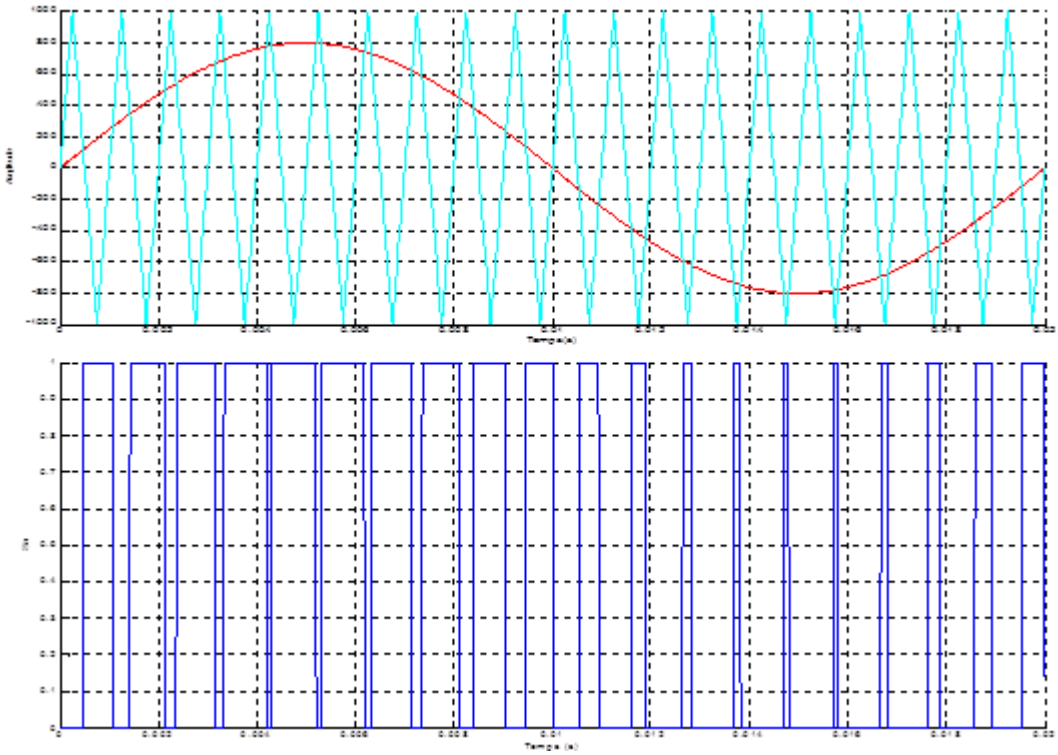


Figure (III.1) : Principe de la MLI sinusoïdaux-triangulaire.

III.2.1)-Génération des signaux sinusoïdaux

Il existe plusieurs méthodes pour générer un signal sinusoïdal avec une description hardware, entre autres, on trouve : les CORDIC, le développement Taylor, les RAM [13].

Pour les CORDIC, ils sont des programmes prédéfinis pour calculer certaines fonctions particulières telles que les fonctions trigonométriques, la division, la racine carré, etc. Le problème pour ces CORDIC réside dans leur complexité et le nombre important de cycles d'horloge nécessaires pour le fonctionnement, chose qui présente une contrainte pour les synchroniser avec les autres parties de la description. En plus, pour les fonctions trigonométriques, il existe une autre contrainte qui impose que l'entré (l'angle) doit être entre 0 etπ. En effet, cette contrainte impose l'utilisation des valeurs avec virgule ce qui est difficile avec un circuit FPGA

Pour le développement Taylor, il faut aller à des ordres supérieurs pour avoir une précision satisfaisant toute en travaillant avec un intervalle restreint d'angle. En plus de la complexité, on trouve toujours le problème des valeurs avec virgule.

Pour la dernière variante, il s'agit de discrétiser le signal sinusoïdal et le stocker dans une RAM. Ensuite, un programme est développé pour lire la valeur de la sinusoïde suivant l'angle ciblé. Cette méthode est simple à mettre en œuvre vue que les circuits FPGA disposent pleinement de RAM. Pour cette raison, cette méthode est retenue pour la réalisation de la MLI.

La figure (III.2) représente un simple programme MATLAB permettant de discrétiser un signal est obtenu un nombre suffisant de points. Le nombre de points dépend de la précision voulue pour le signal sinusoïdal. Pour notre, On essayer de discrétiser ces signaux pour un temps d'échantillonnage ($t = 4.64 * 10^{-5}$s), donc avec une période (T=0.02s,pour le cas de f=50Hz) on obtient 432 (0 à π) points.

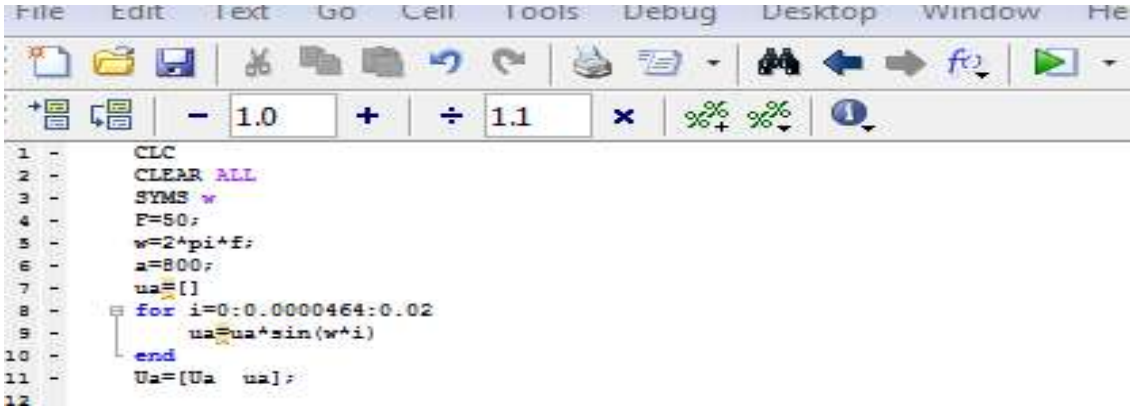


Figure (III.2): Génération des signaux sinusoïdaux par Matlab.

III.2.2).Le signal triangulaire (porteuse)

Les dents du porteuse (signal triangulaire) s'appellent dents de scie, on veut avoir un nombre de dents prédéfinie N dans chaque période de modulante. La période de N dents et la période de modulante doit être égaux, mais avec une amplitude supérieure à l'amplitude de signal de référence (pour notre cas on a pris un rapport de 0.8 (Fig. 3.3).

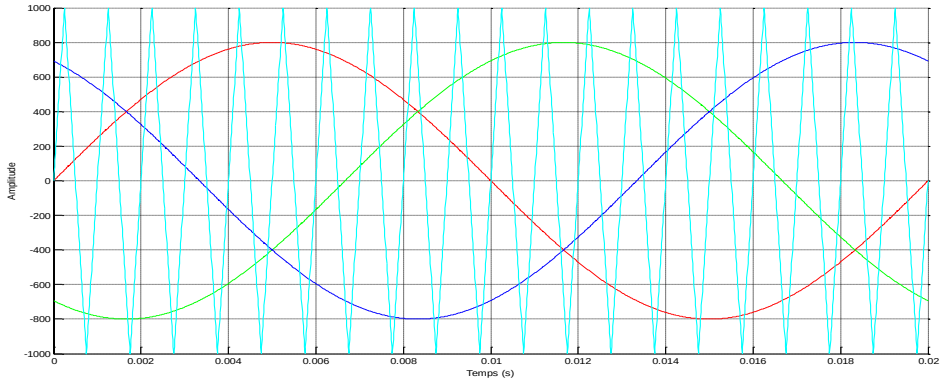


Figure (III.3) : Synchronisation des signaux sinusoïdaux et triangulaire.

Pour une efficace comparaison, il faut faire un compromis entre N etle nombre de points discrétiser, ce compromis résumer par :

X : le nombre de points dans un période (modulante).

N : le nombre de dents dans un période.

On a :

$\boldsymbol{M = {}^{X}/_{N}}$ Représente le nombre de points dans une dent de scie.

Pour une fréquence f=50Hz, on a le nombre de points X=432*2=864 pts, donc :

$$M = {}^{\mathbf{864}}/_{\boldsymbol{N}}$$

Pour N = 36 (une fréquence de commutation 50*36=1.8kHz), le nombre de points pour chaque dent de scie est M = 24.

Pour d'autres fréquences et, en maintenant la même fréquence de commutation, on peut avoir :

- Cas de **25Hz** :

  $\boldsymbol{N=72 \Rightarrow M \approx 12}$ **pts,**

- Cas de **16.66Hz** :

  $\boldsymbol{N=108 \Rightarrow M \approx 8}$ ***pts,***

III.3)-Description VHDL

Apres la création de nouveau projet et la sélection des propriétés de la carte FPGA cible, il faut désigner les entrées/sorties de l'application à réaliser (Fig. 3.4).

```
47
48  entity ml0 is
49
50  port(
51
52        clk      : IN std_logic;
53        res      : IN std_logic;
54        F1, F2   : IN std_logic;
55        iiii       : OUT std_logic_vector ( 9 downto 0);
56        FA, FB, FC : OUT std_logic;
57        sinxa,sinxb, sinxc, tr     : OUT std_logic_vector ( 9 downto 0)
58        );
59
60  end ml0;
```

Figure (III.4): l'entité d'une description VHDl.

III.3.1)- Génération de signal sinusoïdal

Pour générer le signal sinusoïdal on utilise la méthode de sauvegarde les valeurs de signal dans une mémoire de type RAM.Pratiquement, une RAM représente une matrice d'une dimension appropriée (Fig. 3.5). [14]

```
type sn_ram is array (0 to 431) of integer;
 signal snx : sn_ram :=
 (0, 1, 3, 5, 7, 9, 11, 13, 14, 16, 18, 20, 22, 24, 26, 27, 29, 31, 33, 35, 37, 38, 40, 42, 44, 46, 48, 4
  57, 59, 60, 62, 64, 66, 68, 69, 71, 73, 75, 76, 78, 80, 82, 84, 85, 87, 89, 91, 92, 94, 96, 97, 99,
  101, 103, 104, 106, 108, 109, 111, 113, 114, 116, 118, 119, 121, 123, 124, 126, 127, 129, 131, 132, 134
  140, 142, 143, 145, 146, 148, 149, 151, 152, 154, 155, 157, 158, 160, 161, 163, 164, 165, 167, 168, 170
  175, 177, 178, 179, 181, 182, 183, 184, 186, 187, 188, 189, 191, 192, 193, 194, 196, 197, 198, 199, 200
  205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225
  228, 229, 230, 231, 232, 232, 233, 234, 235, 235, 236, 237, 237, 238, 239, 239, 240, 241, 241, 242, 243
```

Figure (III.5) : Création d'un RAM pour le signal sinusoïdal.

Pour économiser les ressources de la carte FPGA on a utilisé une seule ram pour générer les trois signaux (Va, Vb, Vc). Pour différencier les trois signaux sinusoïdaux, on a joué sur l'ordre de lecture à partir de la ram en faisant des décalages correspondant aux décalages naturelle entre les trois phases.

III.3.2)- Génération de signal triangulaire

Par la même manière précédente, des RAM ont été utilisées pour générer les signaux triangulaires. En effet, le nombre de signaux triangulaires dépend du nombre de vitesses qu'on veut avoir pour le moteur asynchrone. Pour notre cas on a tester le programme avec trois vitesse (fréquences) correspondant au fréquences 50Hz, 25Hz et 16.66Hz. A cet effet, on a besoin de trois signaux triangulaire (Fig. 3.6): [14]

- 50Hz avec 24 point
- 25 Hz avec 12 point
- 16.66Hz avec 8point

```
type tr_ram is array (0 to 23) of integer;
signal trr50 : tr_ram :=
(0,54,108,162,216,270,324,270,216,162,108,54,0,-54,-108,-162,-216,-270,-324,-270,-216,-162,-108,-54);
signal trr25 : tr_ram :=
(0,108,216,324,216,108,0,-108,-216,-324,-216,-108,0,108,216,324,216,108,0,-108,-216,-324,-216,-108);
signal trr16 : tr_ram :=
(0,162,324,162,0,-162,-324,-162,0,162,324,162,0,-162,-324,-162,0,162,324,162,0,-162,-324,-162);
```

Figure (III.6) : déclaration de signal triangulaire.

III.3.3)- Génération MLI

Le programme de MLI fait la comparaison entre le signal triangulaire et les trois signauxsinusoïdaux pour générer un signal logique valent 0 ou 1 (Fig. 3.7).

```
240
241   process(clk, k)
242
243   begin
244       IF rising_edge(clk) then
245
246           IF (trr(9) = '1') then
247
248               IF (sin1(9)= '0' ) then
249                   FAA <= '0';
250               else
251                   IF (sin1 >= trr) then
252                       FAA <= '0';
253                   else
254                       FAA <= '1';
255                   end IF;
256               end IF;
```

Figure (III.7) : génération de signal MLI.

Dans ce programme on a basé sur bite MSB (le bit le plus significatif) pour détecter si le signal est négatif ou positif.

III.3.4)- Variation de la vitesse du moteur

Pour commander la vitesse de rotation du moteur on a utilisé les interrupteurs disponibles sur la carte FPGA (Fig. 3.8). La vitesse du moteur change en. Le tableau (III.1) représente les états de F1 et F2 en fonction de fréquence et la vitesse.[15]

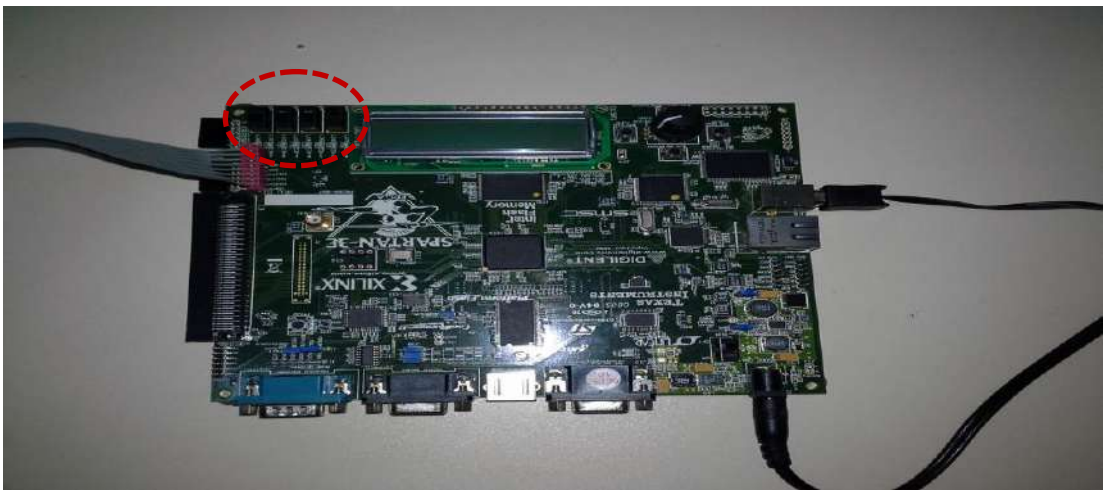


Figure (III.8) : représente les pins de commande de la vitesse sur le circuit FPGA.

F1	F2	FREQUENCE (Hz)	VITESSE (tr/min)
1	1	50	-1500
1	0	50	1500
0	1	25	700
0	0	16.66	400

Tableau(III.1) : Etat du moteur en fonction de l'état des interrupteurs

III.5)-Synthèse et simulation de la stratégie MLI:

Au cours de cette étape, les instructionsécrites, de haut niveau, se traduisent par une configuration logique (RTL) pour obtenir un composant réalisant les fonctions désirées à l'aide d'éléments logiques concrets tel que les portes logiques, les registres, etc. (Fig. 3.9). Ceux-ci seront implémentés, selon la technologie utilisée, soit directement en transistors (dans le cas d'un ASIC), ou en se basant sur les éléments programmables (CLB) des FPGA.[14]

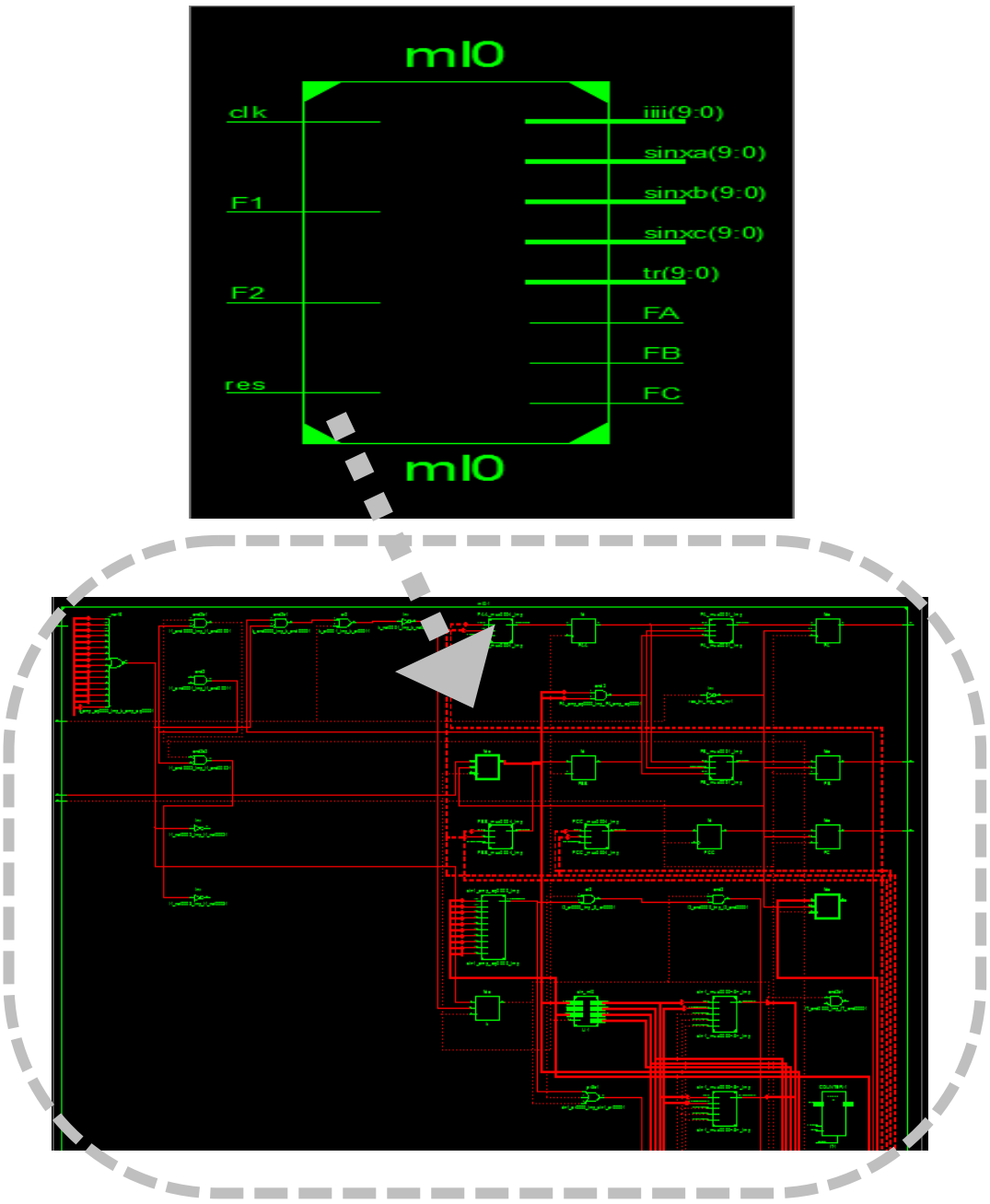


Figure (III.9) : Vue externe (RTL) de MLI.

Pour vérifier le comportement du circuit réalisé, une étape de simulation est nécessaire. Nous allons réaliser une simulation manuelle en appliquant des valeurs sur les entrées et en vérifiant visuellement l'état des sorties.La figure (III.10) représente la simulation de la MLI.

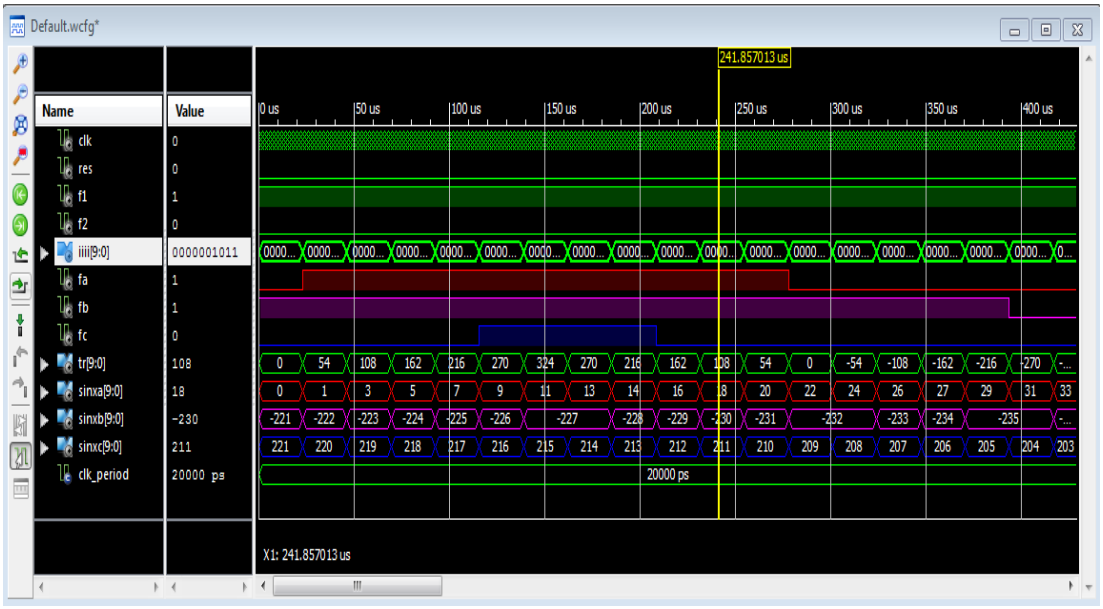


Figure (III.10) : simulation de MLI.

III.6)-L'implémentation de la description de la stratégie MLI

Apres la partie de simulation, on passe à l'étape de l'implémentation, cette dernière est une étape de programmation physique et de teste électrique consiste à mettre en œuvre l'algorithme sur l'architecture de circuit configurable ciblé, c'est-à-dire à compiler, charger, puis lancer l'exécution sur la carte FPGA cible. Avant d'entamer l'étape d'implémentation, il faut spécifier les pins physique sur carte FPGA où seront connectée les entrées et les sorties de l'application réalisée. Cette étape consiste à créer un fichier utilisateur appelé fichier UCF (User Constraints File). Lafigure (III.11) la forme du fichier UCF utilisé pour l'application à réaliser. Il faut noter que la carte FPGA SPARTAN-3E dispose des pins pouvant être utilisés en tant que sortie MLI sans passer par des convertisseurs numérique/Analogique ce qui simplifier énormément le processus d'implémentation. [17]

```
# PlanAhead Generated physical constraints

    NET "clk" LOC = C9;
    NET "res" LOC = N17;
    NET "F1" LOC = L13;
    NET "F2" LOC = L14;
    NET "FA" LOC = D7;
    NET "FB" LOC = C7;
    NET "FC" LOC = F8;
```

Figure (III.11) : Forme d'un fichier UCF.

Après l'assignement des pins d'entrée et de sortie, l'étape d'implémentation peut être exécutée (Fig. 3.12). A la fin de cette, un fichier binaire appelé fichier BitStream est généré, ce fichier est utiliser pour programmer la carte. En effet, le transfert de la solution finale peut être assuré par un câble JTAG qui relie le micro-ordinateur et la carte FPGA (Fig. 3.13). Une fois, le fichier de configuration est charger dans la carte, le fonctionnement de cette dernière devient autonome et la liaison avec le PC peut être éliminée. [18]

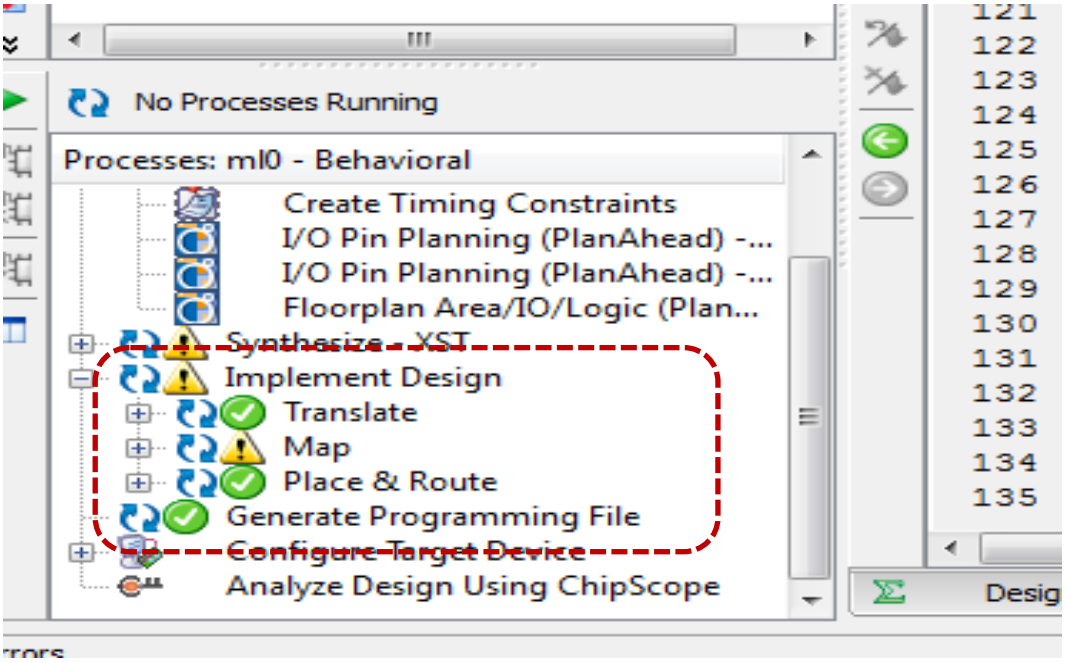


Figure (III.12) : Processus d'implémentation de la description.

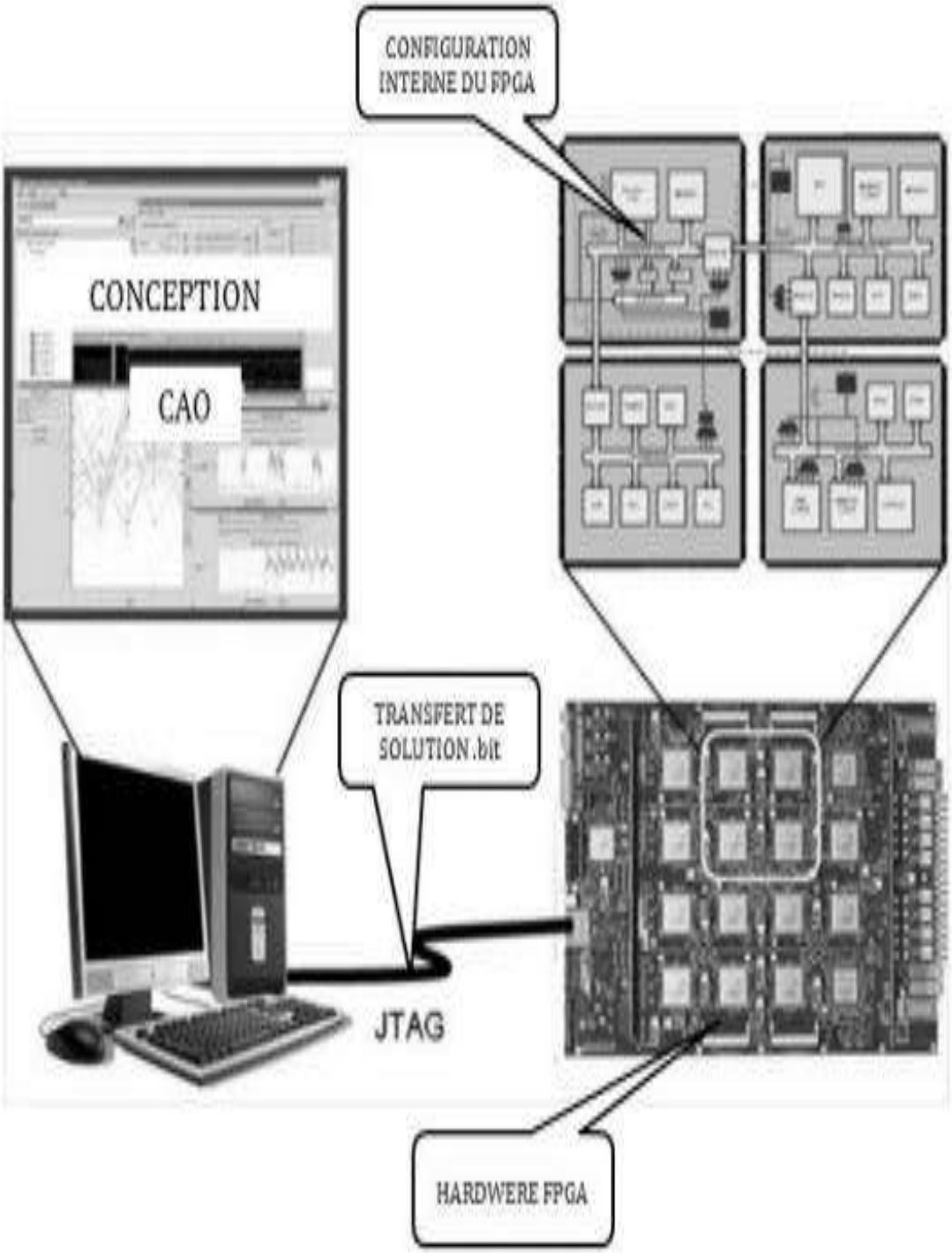


Figure (III.13) : Configuration de la carte FPGA par un câble JTAG.

II.7)-Partie De Réalisation

Pour valider expérimentalement la technique de commande présentée dans le cadre de ce travail, nous construirons un banc d'essais complet, ce dernier est constitué des bras de l'onduleur à deux niveaux. Les interrupteurs de chaque bras sont constitués des transistors IGBT montés en antiparallèles avec des diodes. Les IGBT sont associés à des modules de commande gérant les temps morts et la protection contre les courts circuits. Les signaux de commande des IGBT sont issus d'une carte FPGA SPARTANE-3E. La figure (III.14) représente le banc expérimental réalisé.

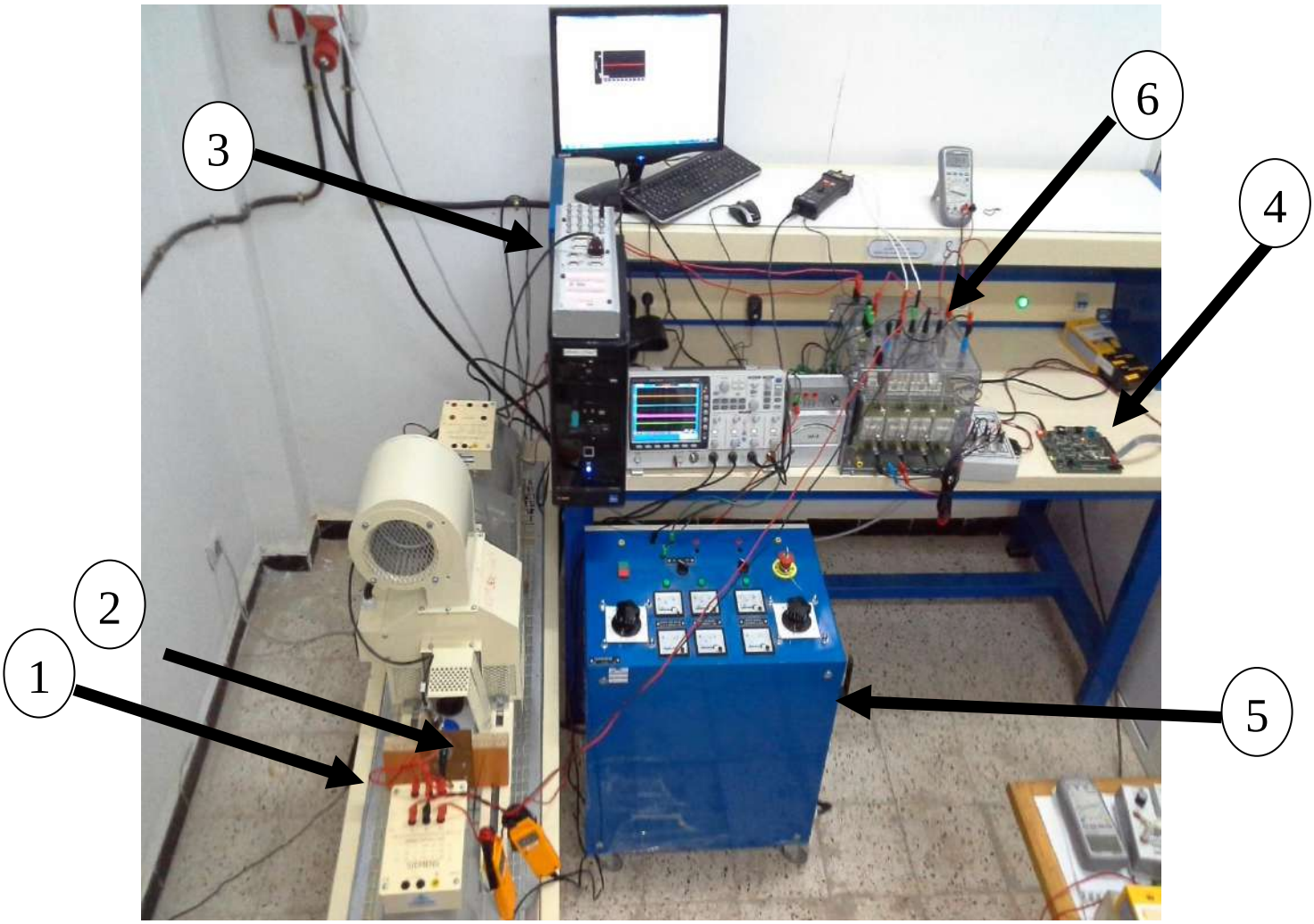


Figure (III.14) : banc expérimental.

1- Moteur Asynchrone

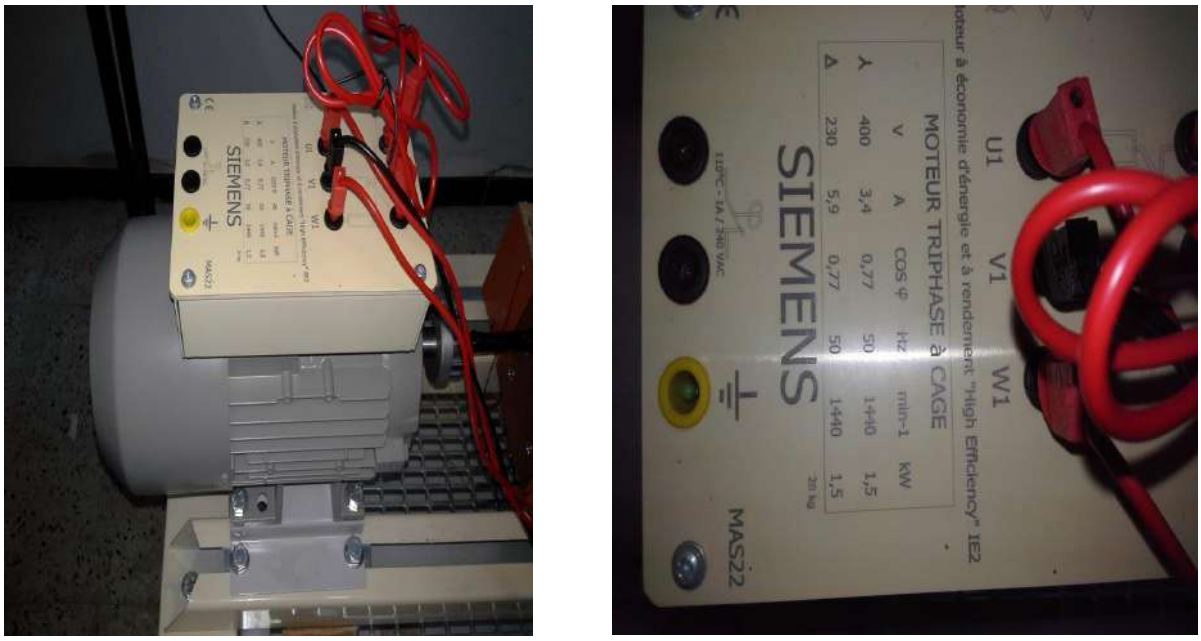


Figure (III.15) : le moteur asynchrone.

2-L'encodeur

L'encodeur est utilisé pour mesurer la vitesse du moteur.



Figure (III.16) : l'encodeur.

3-La Carte DSPACE Clp 1104

Pour la carte dSPACE (DS-1104)on l'a fait recours seulement pour enregistrer la vitesse mesurer afin de la visualiser et de la sauvegarder.[16]

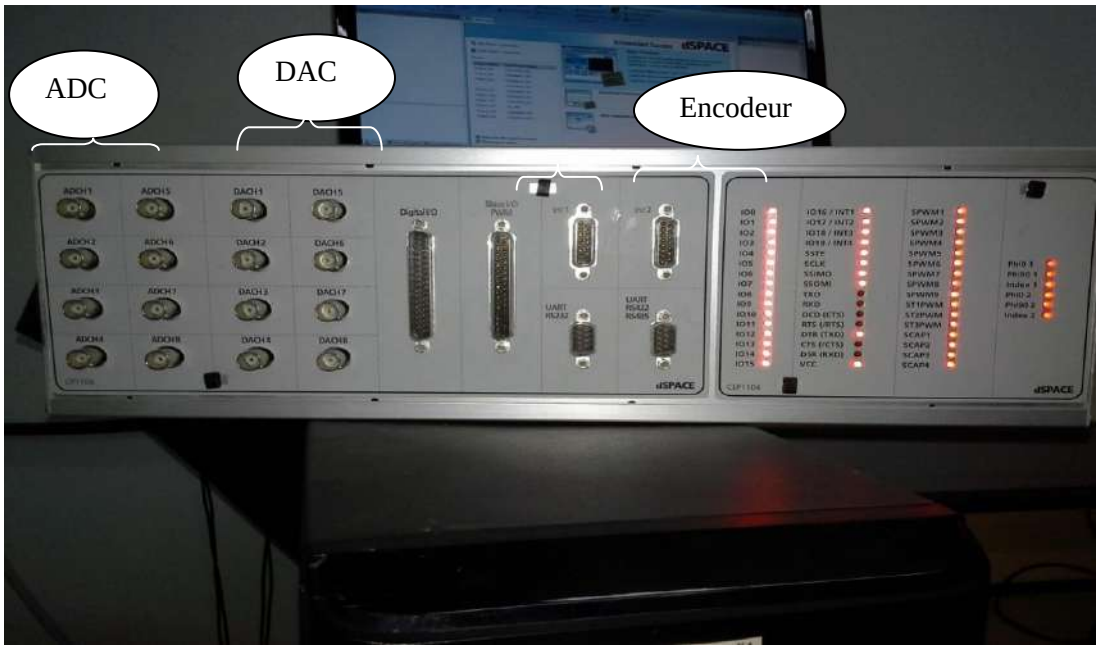


Figure (III.17) : la carte DSPACE.

4-La Carte FPGA

cette carte est le noyeau de notre travail, L'acronyme FPGA (**F**ield **P**rogrammable **G**ate **A**rrays) désigne un circuit intégré composé d'un réseau de cellule programmable. Chaque cellule est capable de réalisé une fonction, choisi parmi plusieurs possible, les interconnexions sont également programmable, la carte utilisé est de type SPARTAN-XC3S500E-FG320.[15]

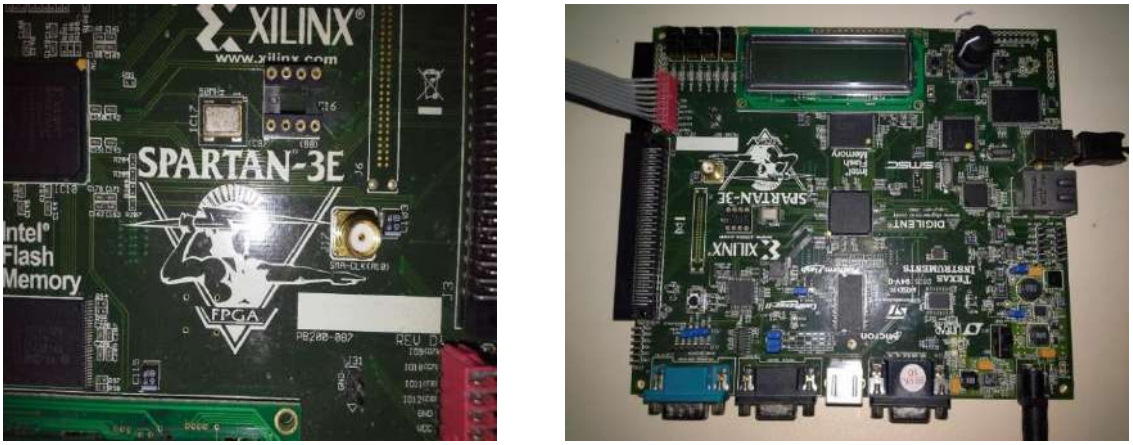


Figure (III.18) : la structure externe de la carte FPGA.

5-Pupitre Electrique

On a utiliser le pupitre pour alimenter l'onduleur avec la tension covenable .



Figure (III.19) : la structure de pupitre électrique.

6-L'onduleur

Cet onduleur figure(III.20)de deux niveau est la source qui alimente notre moteur dont la commande s'effectuevia une interface (III.21) permettant d'adapter le niveau des signaux de commande de la carte FPGA (3V) avic le niceau de tension exigée pour les transistors de l'onduleur (15V). Cette interface reçoit trois signaux (sa,sb,sc) de de la carte FPGA (3V) est délivre 6 sorties(sa,sa',sb,sb'sc,sc')de 15V qui vont commander les six transistor de l'onduleur. [19]

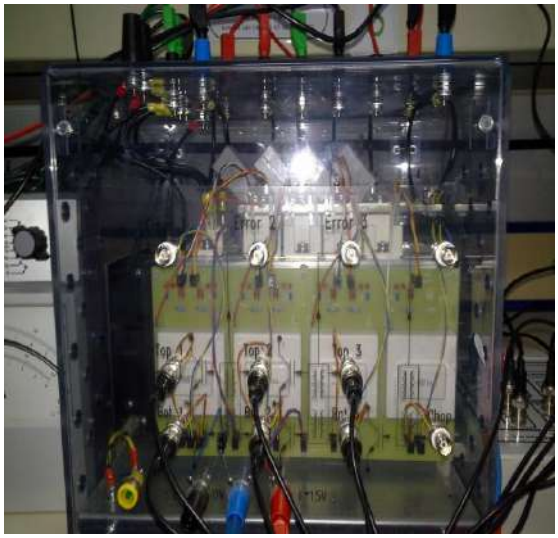


Figure (III. 20) : la structure d'onduleur



Figure (III.21) : interface d'onduleur

En plus, quelques appareils de mesure et d'affichage ont été utilisées (Fig. 3.22 à 3.25).



Figure (III.22) : Oscilloscope numérique.



Figure (III.23) : Capteur de courant.



Figure (III.24) : Sonde différentielle



Figure (III.25) : Multimètre.

III.8) Teste experimentaux

Dans cette partie , nous allons présenter quelques résultats expérimentaux obtenus par la commande scalaire en boucle ouverte réalisée en utilisant la carte FPGA. Les signaux de commande des interrupteurs obtenus par la technique MLI réalisée sont utilisés pour commander le moteur asynhcone via l'onduleur triphasé.

III.8.1)-Essai 1

Les figures(III.25) Jusqu'a la figure(III.29) Montrent l'evolution de la vitesse , de la tension et du courant de la machine asynchrone lorsque la commande scalaire est appliquée. on a appliquée 4 phases de vitesse 0tr/min, 1500tr/min, 700tr/min 400tr/min.

On remarque sur la figure (III.25) que l'amplitude de courant a une relation inverse avec la vitesse, et la frequence de la tension a une relation de corrélation directe avec la vitesse.

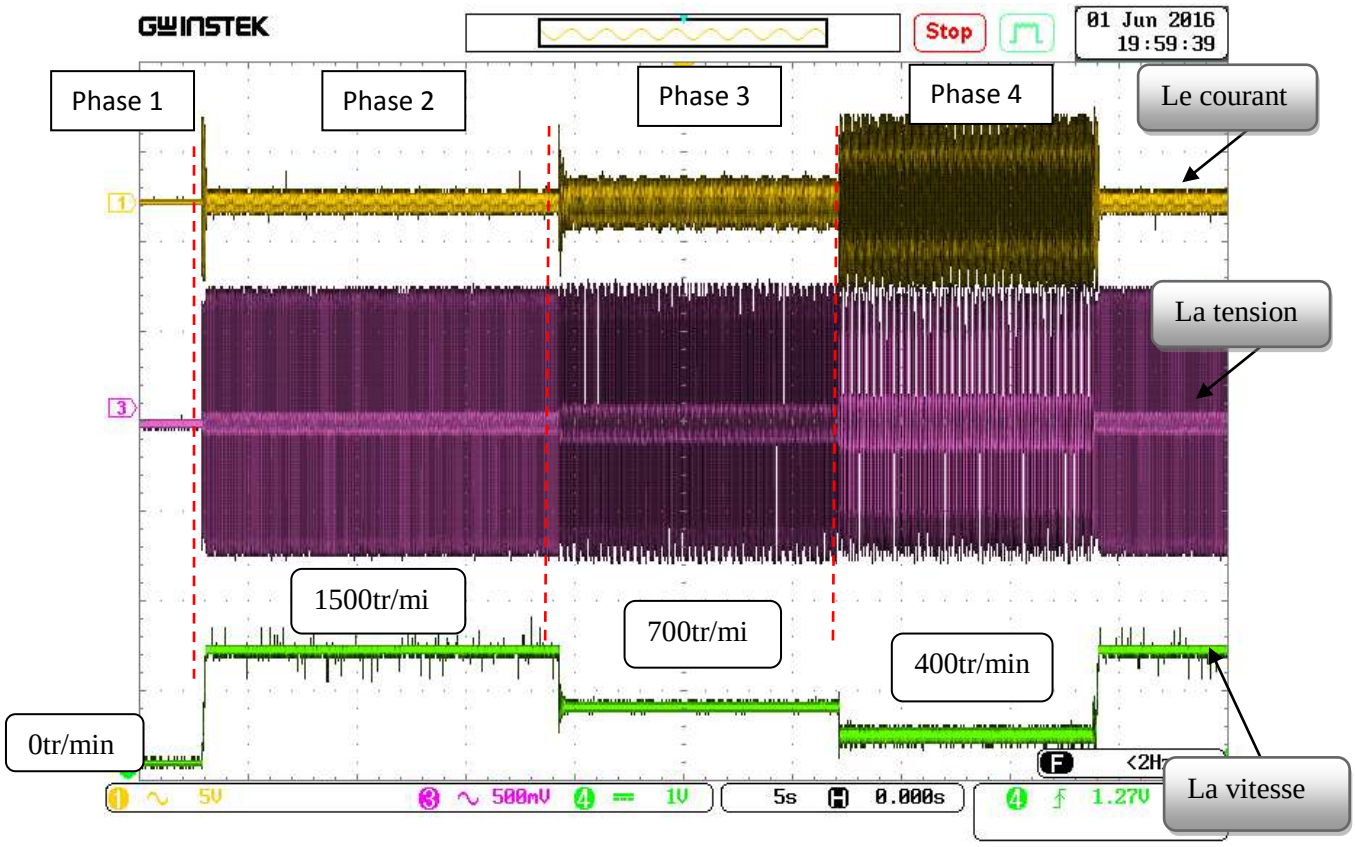


Figure (III.25) : courant, tension et vitesse.

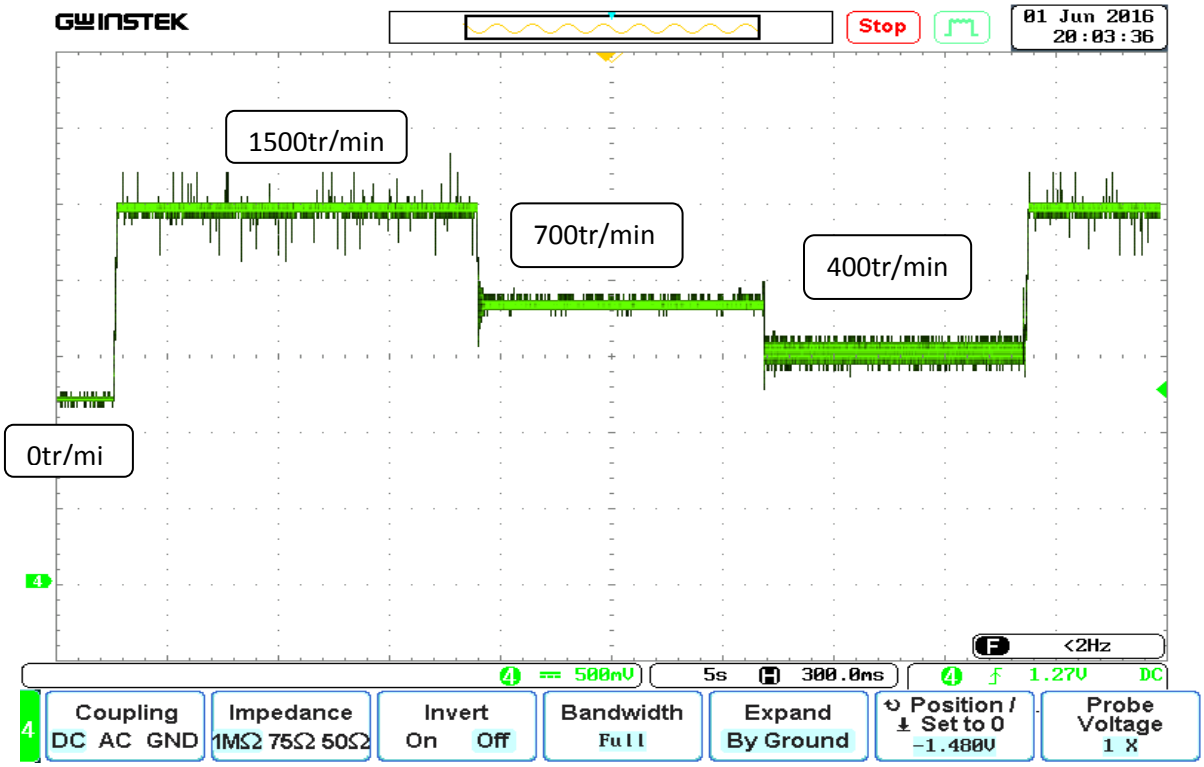


Figure (III.26) : vitesse réelle.



Figure (III.27) : phase de démarrage de vitesse réelle.

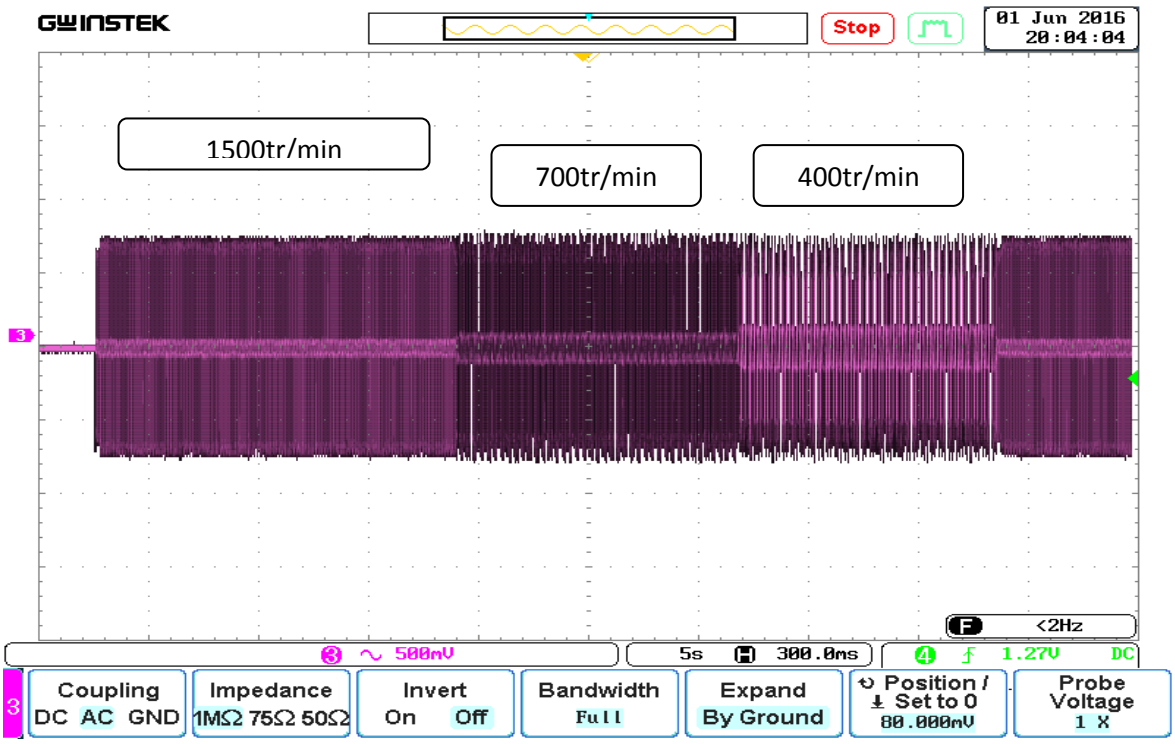


Figure (III.28) : tension réelle.

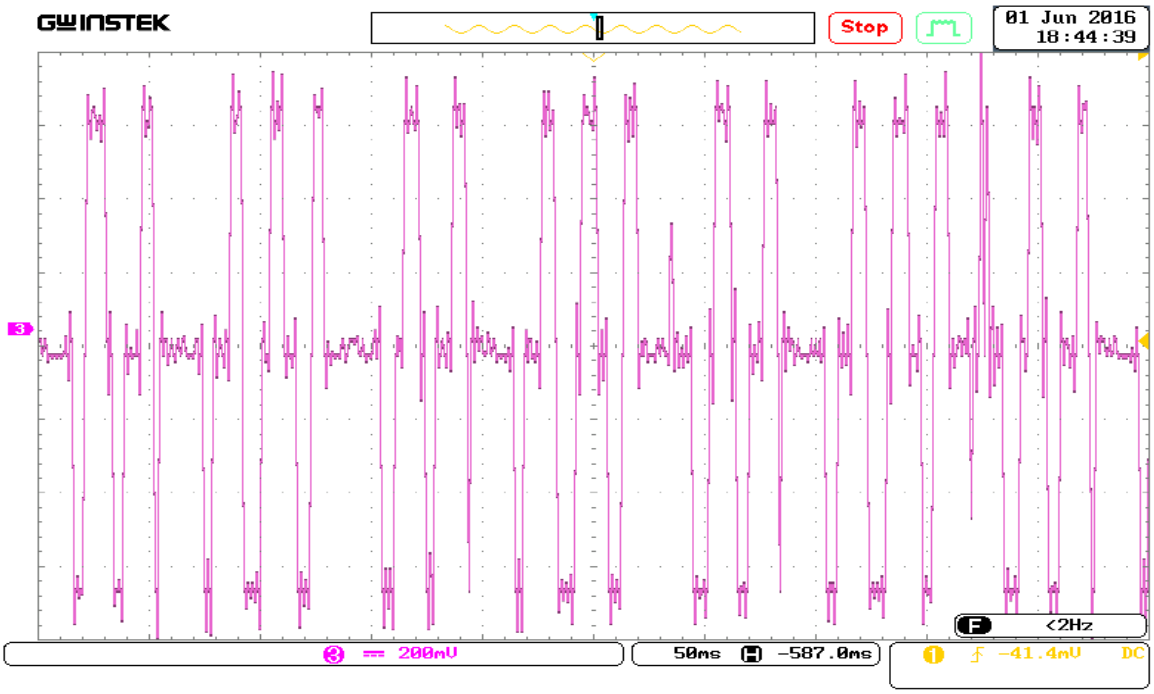


Figure (III.29) : zoom de la tension réelle.

La figure(III.30) représente la variation de courant, et montre que le démarrage du moteur asynchrone consomme plus de courant.

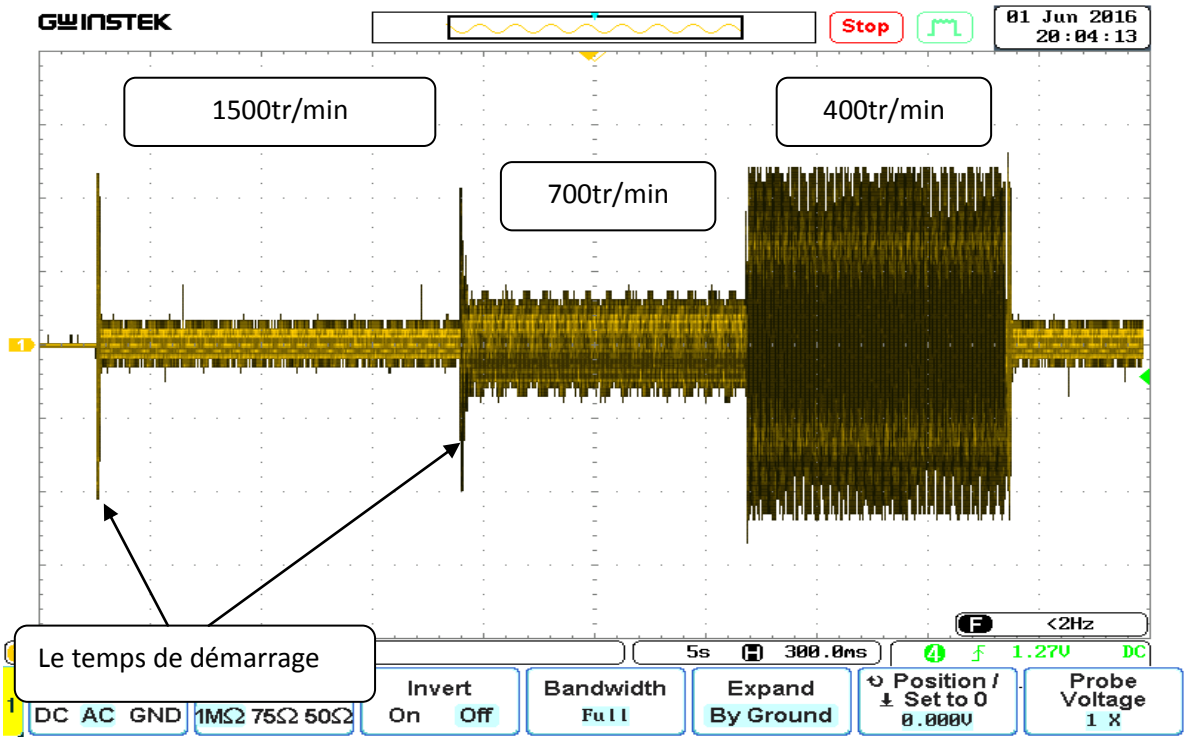


Figure (III.30) : courant réel.

On remarque que le signal de courant est proche de la forme sinusoïdale (Fig. III.31).

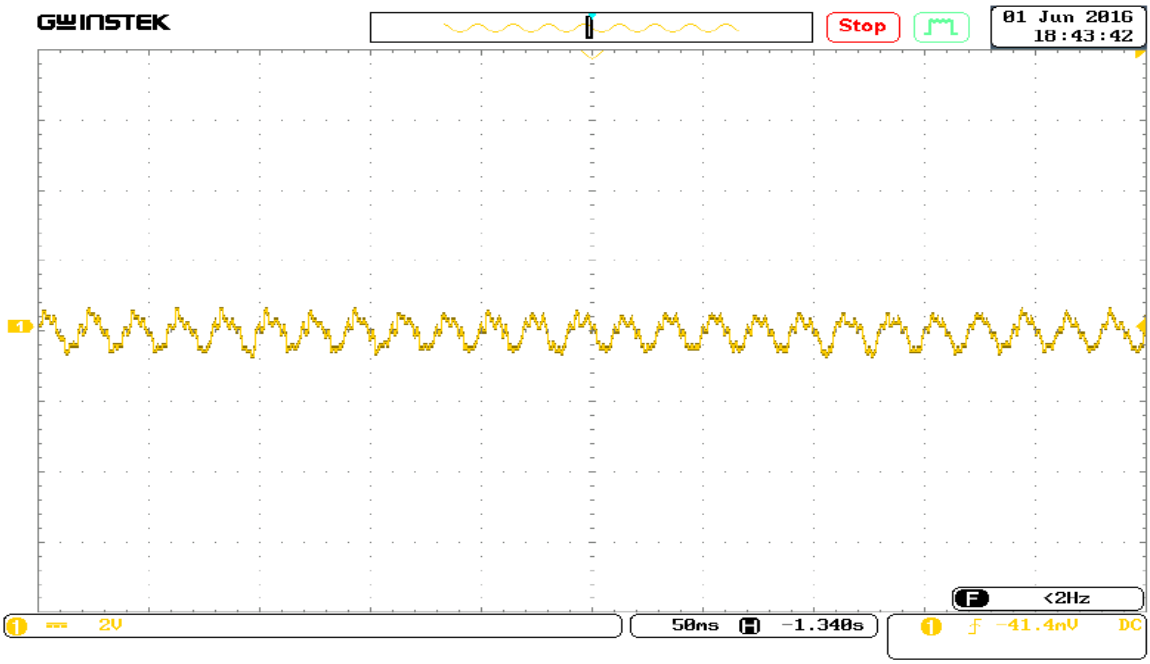


Figure (III.31) : zoom du courant réel.

III.8.2)-Essai 2

Les figures(III.32) et (III.33) Montrent l'evolution de la vitesse , de la tension et du courant de la machine asynchrone lorsque la commande scalaire est appliquée pour faire fonctionner le moteur avec les deux sens de rotation(1500 et -1500 tr/min)



Figure (III.32) : courant, tension et vitesse.

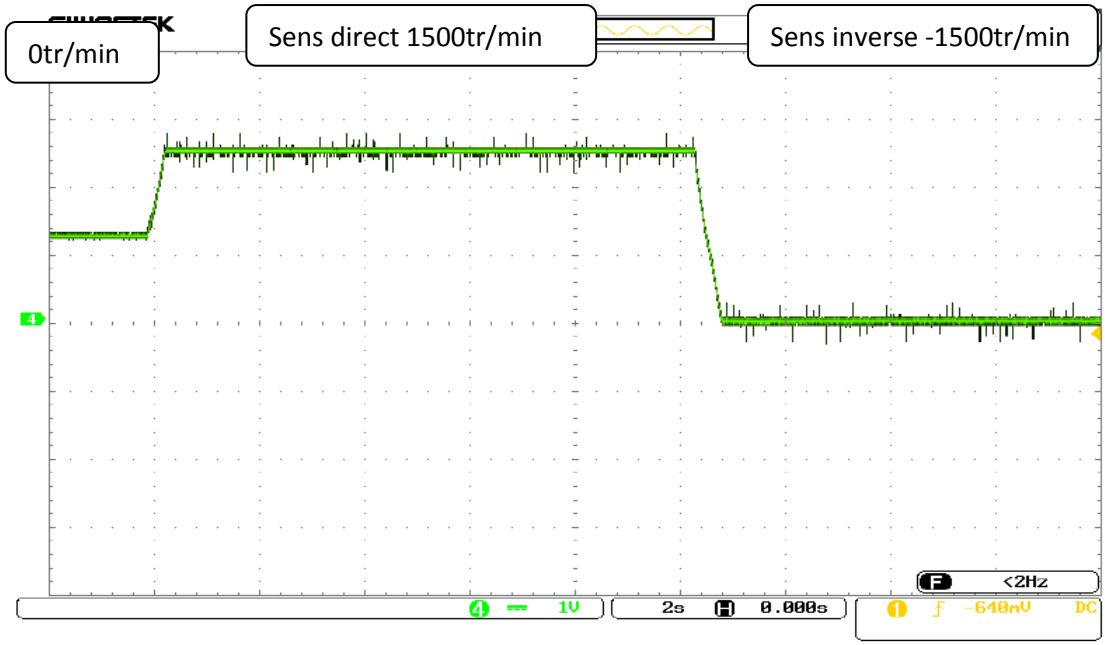


Figure (III.33) : vitesse réelle.

III.8.3)-Essai 3

Les figures(III.34) et (III.35) Montrent l'evolution de la vitesse , de la tension et du courant au borne de la machine asynchrone lorsque la commande scalaire est appliquée.dans cet essai on applique les deux sens de rotation sur le rotor sens direct(1500tr/min) et le sens inverse (-1500tr/min) avec des variation de vitesse (700tr/min, 600tr/min)

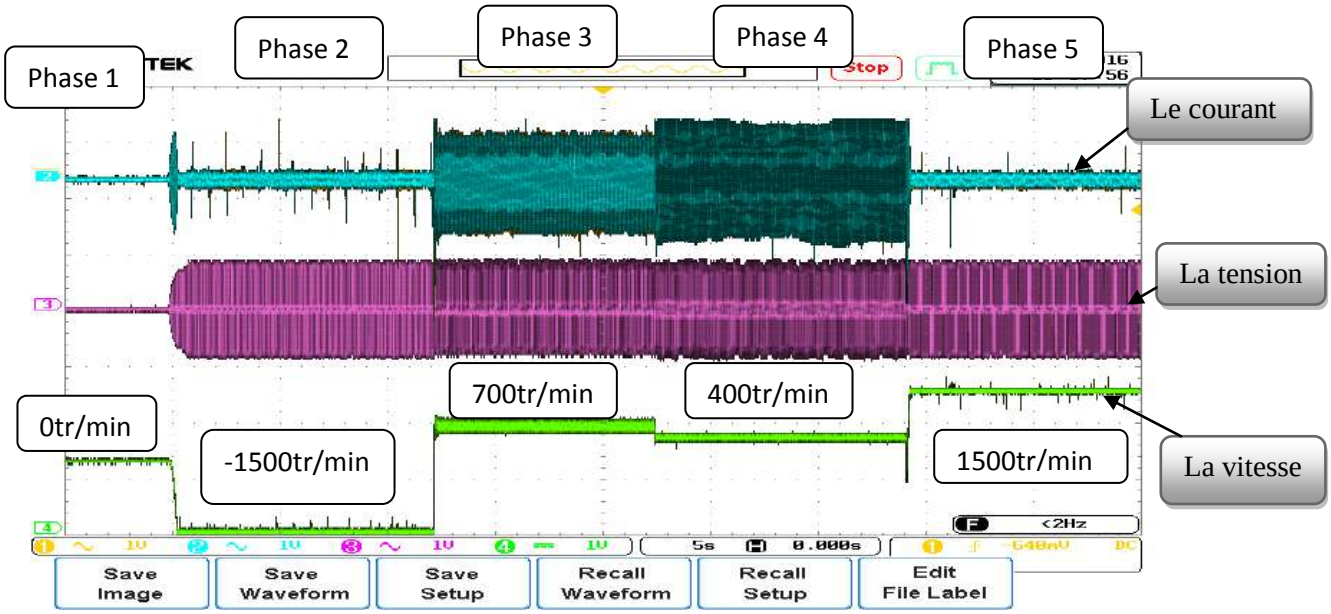


Figure (III.34) : courant, tension et vitesse.

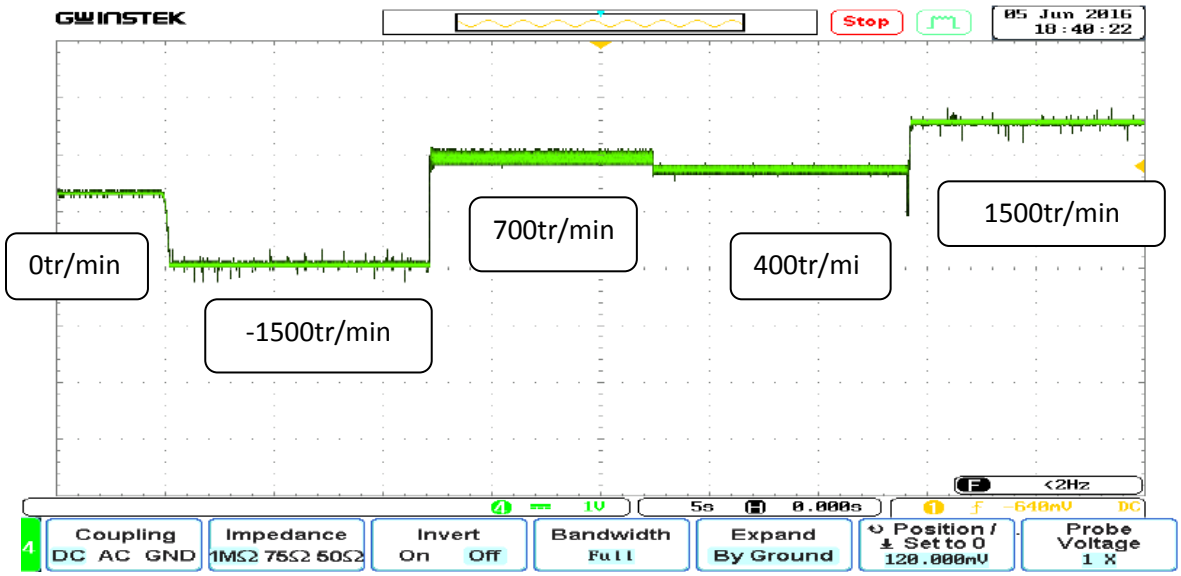


Figure (III.35) : vitesse réelle.

La commande scalaire de la MAS a donné des performances acceptable, l'évolution de la vitesse montre un temps de réponse de 0.45s mais le courant prend 2.5A come valeur maximal, la tension dépend directement du comportement de la vitesse.

Pour les signaux MLI généré par la carte FPGA, les figures (III.36) (III.37)représentent les résultats expérimentaux pour les signaux de commande du premier bras de l'onduleur.

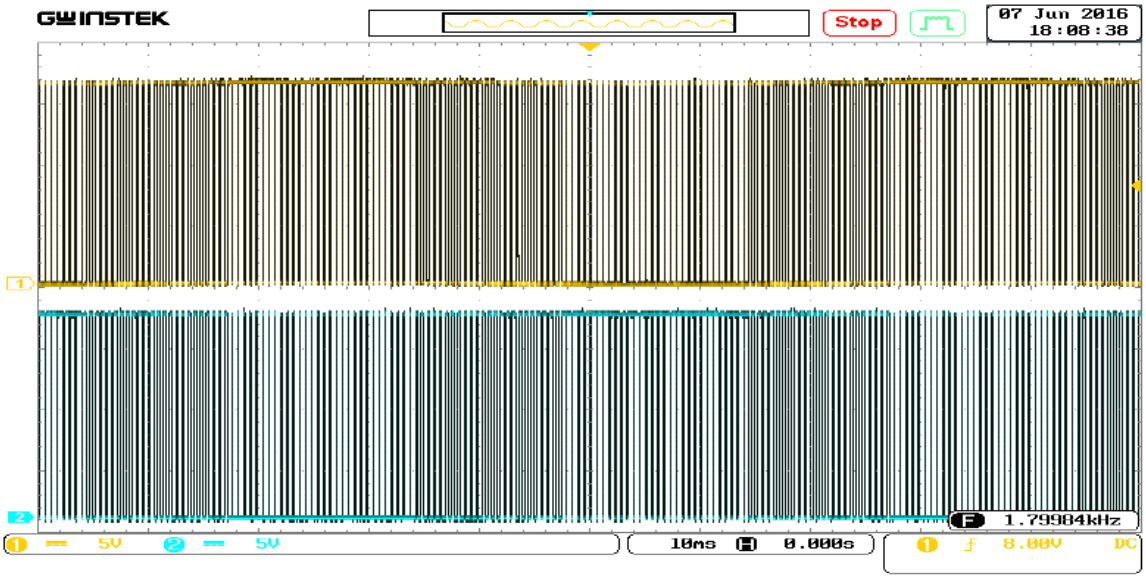


Figure (III.36) : les signaux de MLI Sa,Sa'.

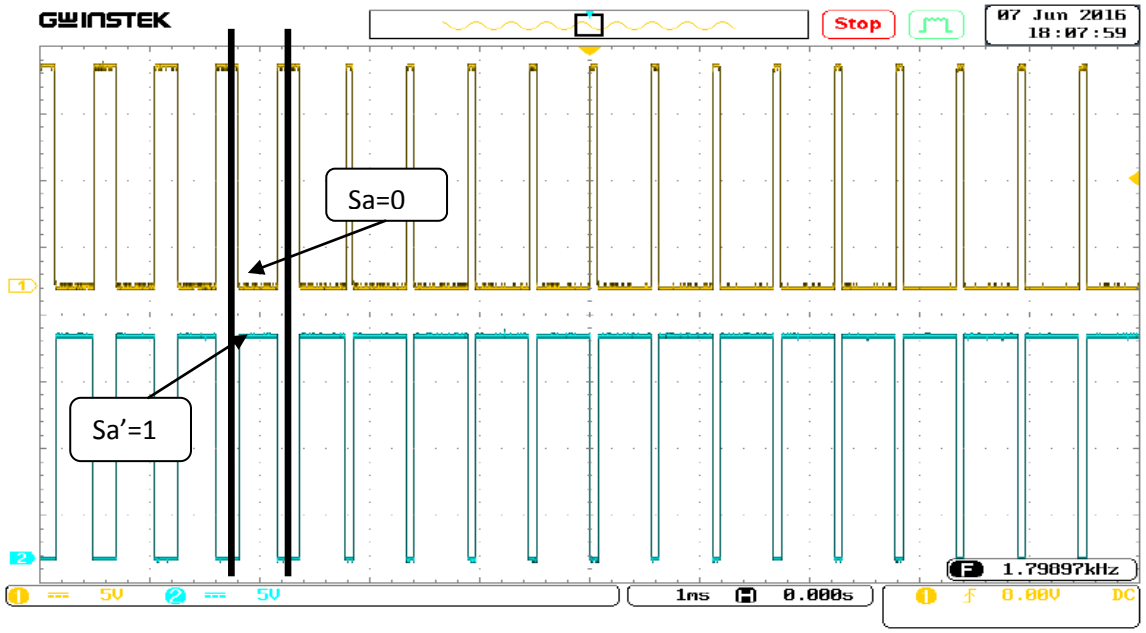


Figure (III.37) : zoome des signaux de MLI Sa,Sa'.

III.9)-Conclusion

Ce dernier chapitre a été consacré l'implémentation de la commande scalaire en boucle ouverte. Premièrement, la technique de commande MLI avec tous ses composantes a été décrit et synthétiser en utilisant le langage de description matériel VHDL sous l'environnement de développement de XILINX. Après la validation par simulation de la description réalisée, on a opté pour l'implémentation tout en vérifiant les différentes contraintes physiques imposées. Une fois le fichier de configuration (BitStream) est généré, il a été chargé dans la carte FPGA et cette dernière devient prête pour commander l'onduleur qui alimentera le moteur asynchrone. Les différents tests expérimentaux effectués ont montré bien la validité de la réalisation élaborée pour les différentes conditions de fonctionnement du moteur

Conclusion Generation

CONCLUSION GENERALE

Le travail présenté dans ce mémoire concerne l'implémentation de la commande scalaire d'un moteur asynchrone triphasé en utilisant un circuit FPGA.

Au premier lieu un aperçu a été donné sur les circuits reconfigurables tout en montrant leurs avantages par rapport aux autres circuits numériques à base de microprocesseur. En effet, l'architecture des FPGA, leur caractéristique ainsi que les moyens de prototypage et de configuration ont été présentés.

En suit, la moteur asynchrone triphasé, la cible de notre application, a été modélisé et simulé en appliquant la commande scalaire.

Dans le dernier volet, l'implémentation de la commande scalaire en boucle ouverte a été abordée. La stratégie de commande MLI avec tous ses composantes a été décrit et synthétiser en implémentée sur une carte FPGA en utilisant le VHDL sous l'environnement de développement de XILINX. Pour valider l'application implémentée, différents essais expérimentaux ont été exécutés. Les résultats obtenus ont montrés clairement la validité du travail effectué.

Comme perspectifs, les résultats obtenus nous permettent d'étendre les objectifs de la réalisation effectuée vers :

- Compléter le schéma de commande en introduisant une boucle de vitesse pour réaliser la commande scalaire en boucle fermée et améliorer les performances de plus
- Essayer d'implémenter d'autres commandes plus performantes pour le moteur asynchrone telles que la commande directe du couple (DTC) et la commande vectorielle.

BIBLIOGRAPHIE

[1] zahir ait ouali «application des FPGA à la commande d'un moteur asynchrone» Thèse magister 2010 UNIV de tizi ouzou

[2]B.K.Bose, « modern power electrics and ac drive» ,prentice hall, upper saddle river,2002.

[3] Corzine.K.A,wielebski.M.K, peng .F.Z ,wang.j« control of multi-level inverter »:IEE Power ElectricsSociety, version 2003

[4] R. Woods, J. Mcallister, G. Lightbody, and Y. Yi, "FPGA-based implementation of signal processing systems", Edition John Wiley & Sons, Ltd., Pulication, 2008.

[5] Pong P. Chu,"FPGA Prototyping by VHDL examples",Edition John Wiley & Sons, Ltd., Pulication, 2008.

[6] XILINX, Spartan-3E FPGA Starter Kit Board User Guide: Complete Data Sheet,http://www.xilinx.com,v1.2 January 20, 2011.

[7] /people.irisa.fr/Arnaud. Tisserand/docs/semmim-at-fpga.pdf

[8]bellatrache « implémentation de la commande vectorielle par un onduleur NPC»Thèse magister 2010 UNIV de Médéa

[9] k.bouhoune « application de nouvelle technique de commande aux machine asynchrones : monophasée et triphasée », UNIV houari Boumediene, alger, algerie, 2005

[10]N.khenfer «machine asynchrone : ses modèles, son identification et sa commande» thèse de doctorat en génie électrique, institut national polytechnique de lorraine, INPL ,Nancy, France ,1995

[11]Kirank.Ggunda « adjustable speed drives laboratorybased on DSPACE controller»bechalor of technology, Jawaharlal Nehru technological univ, 2005 august 2008,

[12] MR NACHEF TOUFIK« implémentation d'une instrumentation sur un fpga» mémoire de magister 2011.

[13]Y.DIAF « modélisation et commande d'un onduleur triphasé pilote par MLI a structure multi niveaux» mémoire de master 2010.

[14]T. Sutikno ,N. Idris, A. Jidin, andN. Cirstea, "An Improved FPGA Implementation of Direct Torque Control for Induction Machines", IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 9, NO. 3,pp. 1280-1290, 2013.

[15] T. Sutikno, A. Jidini, N.R. Idrisi, "New approach FPGA-based implementation of discontinuousSVPWM", Turk. J. Electr. Eng. Comput. Sci., vol. 18, p. 6,2010.

[16] DS1104pc controller board, feature, realize 5.1, May 2006, DSPACE

[17]Me. Messaoudi «Stratégie de modulation pour la réduction des perturbations conduites générées par un variateur de vitesse à redresseur MLI».18 décembre 2015

[18]MR Kaci yahia. «étude d'une nouvelle approche MLI(PWM) en temps réel basée sur les principes des réseaux de neurone destiné aux véhicule électrique » le 15 janvier 2015.

[19]T. Abdelkrim,E.M. Berkouk K. Aliouane, K. Benamrane et T. Benslimane «Etude et réalisation d'un onduleur à trois niveaux commandé par MLI vectorielle»accepté le 25 Juin 2011