République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique
Département d'Electronique
Laboratoire des Dispositifs de Communication
et de Conversion Photovoltaïque

Thèse de Doctorat

En ELECTRONIQUE

Présenté par : **GUELLAL Amar**
Magister en Electronique, ENP

Thème

Contribution à l'étude et à l'implémentation des commandes en temps réel pour MAS

Soutenue publiquement le 13 Décembre 2015 devant le jury composé de :

HADDADI Mourad	Professeur	ENP	Président
LARBES Cherif	Professeur	ENP	Directeur de thèse
AIT- CHEIKH Mohamed Salah	Professeur	ENP	Examinateur
BARAZANE LINDA	Professeur	USTHB	Examinatrice
HADJ ARAB Amar	Directeur de Recherches	CDER	Examinateur
LARABI Abdelkader	Professeur	USTHB	Examinateur

ENP 2015

ملخص:
في السنوات الأخيرة, الكثير من الأبحاث توجهت لدراسة التحكم في سرعة السيارات الكهربائية (EVs) المدفوعة بمحرك لا متزامن. إن تقنية تغيير عرض الدفع المصحوبة بتقنية حذف المركبات التوافقية (SHE PWM) تمثل حلا ممتازا للتحكم في سرعة المحرك اللامتزامن لكنها لم تلق انتشارا واسعا خاصة في مجال السيارات الكهربائية و ذلك بسبب استحالة حساب زوايا التبديل في نفس الوقت الذي يشتغل فيه المحرك .للتخلص من هذا المشكل اقترحنا في هذه الأطروحة تقنية جديدة تعتمد على طريقة الشبكات العصبية(ANN). في هذه الأطروحة نبدأ بشرح دقيق للطريقة ANNSHE PWM المقترحة ثم نجري لها التحاكي. بعد ذلك نجري تحليلا عميقا لدراسة الخطأ في زوايا التبديل و ذلك للتأكد من دقة و نجاعة الطريقة المقترحة. في الأخير قمنا ببرمجة الطريقة المقترحة على بطاقة "FPGA " و استعمالها في منصة تجارب للتحكم في سرعة محرك لا متزامن, وذلك للتحقق من نجاعة هذه الطريقة في التطبيقات الحقيقية. النتائج المستخرجة تؤكد أن الطريقة المقترحة تسمح بالتحكم و الضبط الجيد للتوتر الأساسي كما بينت نجاعتها في حذف المركبات التوافقية في الوقت الحقيقي و ذلك في جميع مجال تغير سرعة المحرك.
كلمات مفاتيح "SHE PWM", الشبكات العصبية, بطاقة "FPGA ",السيارة الكهربائية, المحرك اللامتزامن, العاكس

Résumé :

Au cours des dernières années, un grand intérêt a été donné à la commande de vitesse des véhicules électriques (EV) entrainés par un moteur asynchrone. La modulation de largeur d'impulsion avec élimination sélective des harmoniques (SHE PWM) est une alternative intéressante pour la commande de la vitesse d'un moteur asynchrone. Cependant, son utilisation est impossible dans les applications temps réel, comme celle des véhicules électriques, vu que les angles de commutation ne peuvent être calculés et ensuite générés on-line et en temps réel. Pour résoudre ce problème, dans cette thèse nous proposons un nouvel algorithme PWM on-line basé sur la théorie réseaux de neurones artificiels (ANN) en combinaison avec la technique SHE PWM. Dans cette thèse, l'algorithme ANNSHE PWM proposé est d'abord décrit et simulé. Ensuite, une analyse approfondie des erreurs de calcul des angles de commutation est effectuée afin de vérifier la précision de cet algorithme. Dans le but de valider cet algorithme et de générer les angles de commutation avec précision dans une application en temps réel, une implémentation hardware sur un circuit FPGA de l'algorithme proposé est réalisée et détaillée dans une première étape, puis une mise en œuvre de cette dernière sur un banc d'essai de commande de vitesse d'un moteur asynchrone est effectuée dans une deuxième étape. Les résultats obtenus montrent que l'algorithme proposé ANNSHE PWM contrôle l'amplitude du fondamental et élimine les harmoniques sélectionnés avec précision et en temps réel et ceci dans toute la gamme de variation de vitesse du moteur asynchrone.
Mots clés- SHE PWM, Réseaux de Neurones Artificiels (ANN), FPGA, Véhicule Electrique, moteur asynchrone, onduleur.

Abstract:

During the last decade, many interests were given to the speed control of induction motor based electrical vehicles (EVs). The calculated Pulse Width Modulation technique with Selective Harmonic Elimination and Voltage Control (SHE PWM) is an attractive alternative for speed control of an induction motor. However, its application is unfeasible in real time application, as in EVs, because the switching angles cannot be calculated and then generated online and in real time. To overcome this problem, this thesis proposes a new online PWM algorithm based on the Artificial Neural Network (ANN) theory in combination with the SHE PWM. In this thesis, the proposed ANNSHE PWM algorithm is first described and simulated. Then, an extensive angle error analysis is carried out in order to check the accuracy of this algorithm. Finally, in order to validate this algorithm in a real time application, an FPGA implementation of the proposed algorithm is first presented and detailed. Then, the application of this latter on a variable speed induction motor tests bench is carried out. The results obtained show that the proposed ANNSHE PWM algorithm controls the fundamental voltage and eliminates the desired harmonics efficiently and in real time and this in the whole range of speed variation.

Keywords- SHE PWM, Artificial Neural Network (ANN), FPGA, Electrical Vehicle, Induction Motor, Inverter.

Dédicaces

A mon père

A ma mère

A ma femme et mes enfants

A mes frères et sœurs

A toute la famille GUELLAL

A tous mes amis

Je dédie le fruit de ce modeste travail

Ahmed Wid Aissa

Remerciements

Avant tout, je remercie Dieu, le tout puissant, pour m'avoir donné la volonté et la patience qui m'ont permis de mener à bien ce modeste travail.

J'adresse mes respectueux remerciements à mon directeur de thèse, Monsieur LARBES Cherif, Professeur à l'ENP, pour m'avoir accueilli et dirigé dans le Laboratoire des Dispositifs de Communication et de Conversion Photovoltaïque à l'ENP. Je voudrais exprimer toute ma gratitude et ma reconnaissance pour ses encouragements, son suivi, son aide documentaire, son soutien matériel précieux et sa disponibilité le long de ce travail. A part ses grandes qualités scientifiques, j'ai également beaucoup apprécié ses qualités humaines qui ont fait que ce travail s'est déroulé dans une ambiance agréable.

Mes remerciements et ma profonde gratitude vont également aux membres du jury. Tout d'abord, je tiens à exprimer ma profonde reconnaissance envers Monsieur HADDADI Mourad, Professeur à l'ENP, pour avoir accepté d'examiner mon travail et de présider mon jury de soutenance. Mes remerciements vont également à Monsieur AIT- CHEIKH Mohamed Salah, Professeur à l'ENP, Madame BARAZANE Linda, Professeur à l'USTHB, Monsieur HADJ ARAB Amar, Directeur de Recherches au CDER et Monsieur LARABI Abdelkader, Professeur à l'USTHB, pour l'intérêt qu'ils ont porté à mon travail en acceptant de participer à ma soutenance en tant qu'examinateurs.

Je remercie vivement mon collègue et mon ami KHERCHI Mohamed, Attaché de recherche au CDER, pour ses encouragements et son aide durant la validation pratique de ce travail.

Je voudrai exprimer mon profond respect à tous les Enseignants qui m'ont encadré durant mes études.

Pour finir, je tiens à remercier tous les membres de ma famille pour leurs soutiens, leurs encouragements et leurs patiences durant toute la durée qu'a pris cette thèse. Aussi, je remercie toute personne qui, d'une manière ou d'une autre, m'a aidé dans l'élaboration de ce travail.

Sommaire

Listes des figures

Liste des tableaux

Introduction générale

La consommation excessive d'énergie et l'épuisement des ressources fossiles ainsi que la pollution et le réchauffement climatique, qui sont principalement causés par les véhicules à moteur à combustion, sont derrière l'intérêt récent donné par de nombreux chercheurs et de constructeurs d'automobiles aux véhicules électriques (EV) [1-4]. L'objectif principal de ces recherches est de construire un véhicule sécurisé, plus propre, plus durable, efficace et intelligent. Ces caractéristiques peuvent être réalisées dans un EV. La solution pour atteindre ces objectifs est de développer toutes les technologies existantes dans un EV. En fait, la technologie la plus importante est la batterie ou le stockage d'énergie en général, qui nécessite plus d'attention en vue d'augmenter l'autonomie des EVs [3]. D'autre part, l'électronique de puissance et l'entraînement des moteurs en particulier, faisant l'objet de cette thèse de doctorat, est également une technologie fondamentale, qui devrait être prise en compte afin d'améliorer les performances entières des EVs y compris l'autonomie [2, 4-5]. L'entraînement du moteur, qui est le cœur du système de propulsion de l'EV, requiert une innovation dans toutes ses composantes. Il est constitué d'un moteur électrique, un convertisseur de puissance et une unité de commande électronique. Cette dernière est l'objectif principal de cette thèse. Le choix du convertisseur de puissance et le dispositif de commande électronique dépend souvent du choix du moteur électrique.

Dans le cadre de ce projet, on a préféré le moteur asynchrone (IM) par rapport à son homologue le moteur à courant continu (DCM) en raison de sa robustesse, entretien moindre, une puissance plus élevée par rapport au poids et un coût moindre par rapport à la puissance [6]. Cependant, pour que les moteurs asynchrones soient compétitifs par rapport aux moteurs à courant continu il est nécessaire de développer des onduleurs de tension (VSI) efficaces et à faibles coûts.

En principe, pour commander la vitesse d'un moteur asynchrone il est nécessaire d'utiliser un VSI triphasé avec une sortie sinusoïdale variable en tension et en fréquence. Dans la pratique, les stratégies de commande produisent des harmoniques indésirables dans la sortie variable du VSI [7-10]. Ces harmoniques ont de nombreux effets indésirables sur le fonctionnement du moteur, tels que l'échauffement du moteur qui est dû à l'augmentation des pertes et les pulsations de couple qui deviennent gênantes surtout aux faibles vitesses [9-10].

Une solution bien connue pour résoudre le problème des harmoniques indésirables est l'utilisation de la commande en modulation de largeur d'impulsion avec élimination sélective des harmoniques (SHE PWM) [10-14]. La commande SHE PWM qui a été initialement développée par Patel et Hoft [12-13] offre plusieurs avantages par rapport à la commande PWM sinusoïdale classique (SPWM) : meilleure élimination des harmoniques indésirables, pertes de commutation moindres, la limite minimale de largeur d'impulsion peut être atteinte facilement et la sur-modulation est possible [11]. Compte tenu de ces avantages, la commande SHE PWM pourrait être une alternative intéressante à la commande SPWM, en particulier pour les onduleurs de forte puissance utilisés dans les EVs.

Cependant, l'utilisation de la commande SHE PWM n'est pas évidente puisque le système d'équations utilisé pour calculer les angles de commutation est non-linéaire limitant cette stratégie de contrôle au calcul off-line en utilisant des techniques de calcul numériques telles que la méthode de Newton-Raphson. Cette dernière peut entraîner beaucoup de cycles d'itération si les valeurs initiales ne sont pas choisies correctement et dans certains cas la solution peut ne pas converger du tout. En supposant que tous les angles SHE PWM sont calculés off-line avec succès en utilisant une technique de calcul numérique ; ils sont ensuite stockés dans des mémoires (look-up tables) et appelés à chaque fois que la forme d'onde PWM doit être construite. Cette approche est connue sous le nom de commande SHE PWM programmée ou calculée [15]. Bien que simple, elle nécessite une large mémoire pour stocker tous les angles calculés et atteindre une bonne précision.

Pour un onduleur qui a une fréquence fixe et une tension de sortie fixe comme dans les applications utilitaires, l'utilisation de la commande SHE PWM programmée est acceptable [16-17]. Cependant, dans des applications où la fréquence et la tension nécessitent des changements fréquents, comme dans une application EV, l'utilisation de cette technique est difficile et pas efficace.

Cet inconvénient a poussé les chercheurs à explorer d'autres méthodes permettant le calcul des angles de commutation on-line et en temps réel ; évitant ainsi la nécessité d'une grande mémoire de stockage et le calcul off-line compliqué tout en améliorant les performances globales du moteur. Le travail le plus important a été publié par Taufiq et al qui transforment le système d'équations non linéaires en un système d'équations linéaires en utilisant une approximation sinusoïdale [18]. Avec cette méthode, le système d'équations d'angles de commutation SHE PWM est réduit à une forme linéaire. Sidney a proposé un autre algorithme SHE PWM, basé sur la modulation régulière échantillonnée " Regular Sampled PWM" [19]. Le même auteur a également réalisé un travail similaire pour la modulation vectorielle (Space

vector modulation SVM) [20]. Un algorithme SHE PWM utilisant la commutation bipolaire a été proposé dans [21]. D'autres méthodes, plus complexes ont été proposées en utilisant les fonctions de Walsh [22] et le calcul sur la base d'homotopie "homotopy-based computation"[23]. Plus tard, Chiasson a proposé une conversion du système d'équations SHE PWM en un ensemble de polynômes et en appliquant la théorie de la résultante pour trouver une solution complète des angles de commutation [24]. Cette technique a été appliquée aux onduleurs multi-niveaux [10, 25]. Plus récemment, en utilisant l'algorithme génétique, un algorithme SHE PWM pour les onduleurs multiniveaux a été proposé [26-28]. Un autre algorithme SHE PWM basé sur l'interpolation polynômiale de la trajectoire des angles exacts SHE PWM a été proposé par [11]. Suivant la même approche, dans cette thèse, un nouvel algorithme ANNSHE PWM basé sur la théorie des réseaux de neurones Artificiels (Artificial Neural Network (ANN)) et la commande SHE PWM est proposé afin de calculer les angles de commutation et générer les signaux de commande PWM on-line et en temps réel. Pour vérifier le fonctionnement de cet algorithme ANNSHE PWM proposé, une implémentation sur un circuit FPGA de cette technique est effectuée.

Afin d'atteindre nos objectifs, cette thèse est divisé en cinq chapitres :

Dans le premier chapitre, nous commençons par des généralités sur les véhicules électriques, les machines asynchrones et les onduleurs de tension. Ensuite, on détaille le principe de la commande SHE PWM.

Dans le deuxième chapitre, puisque l'algorithme proposé est basé sur le principe des réseaux de neurones artificiels (ANN), nous introduisons les fondements de cette technique, ainsi on décrit l'étape d'apprentissage qui est l'étape la plus importante dans le développement d'un modèle ANN.

Dans le troisième chapitre, l'algorithme ANNSHE PWM proposé est décrit, simulé et validé pour une implémentation sur un circuit FPGA.

Comme l'algorithme proposé sera implémenté sur un circuit FPGA, dans le chapitre IV on introduit l'architecture des circuits FPGA et on présente la manière de développement d'un projet sur ces circuits.

Dans le chapitre cinq, on explique la façon dont on a procédé pour optimiser l'implémentation de l'algorithme proposé sur un circuit FPGA afin d'améliorer ses performances, ainsi les détails de chaque partie du code VHDL sont présentés. Ce chapitre sera terminé par une validation expérimentale.

Enfin une conclusion générale termine cette thèse.

Chapitre I : La commande en Modulation de Largeur d'Impulsion avec Élimination Sélective des Harmoniques (SHE PWM) appliquée à la commande de vitesse d'une MAS dans un EV

1- Introduction

Les activités de recherche et de développement relatives au transport, réalisées ces dernières années, se concentrent à développer un transport confortable, sécurisé, peu coûteux et respectant l'environnement (réduction des émissions de polluants). Pour répondre à ces exigences, les Véhicules Électriques (EV) et les Véhicules Électriques Hybrides (HEV) sont proposés. Ces véhicules occupent de plus en plus le marché de l'automobile et tentent à remplacer les véhicules conventionnels [1-2].

Le système de propulsion électrique qui est le cœur du EV est constitué d'un actionneur électrique, un dispositif de transmission et des roues. L'entraînement, qui est l'ensemble du moteur électrique et des convertisseurs statiques associés à une commande électronique, est le noyau du système de propulsion dans le EV.

D'après l'étude comparative de topologies de moteurs utilisées dans un EV réalisée par [62,63] le moteur asynchrone semble d'être le meilleur candidat pour la propulsion des véhicules électriques. Par ailleurs, le moteur asynchrone peut fonctionner sur une grande plage de variation de vitesse avec de faibles ondulations de couple s'il est associé à une commande adéquate [63].

Les techniques de commande d'entraînements du moteur asynchrone sont bien traitées dans la littérature. Dans la pratique, ces stratégies de commande produisent des harmoniques indésirables dans la sortie de l'onduleur. Une solution bien connue pour résoudre le problème des harmoniques indésirables est l'utilisation de la commande en modulation de largeur d'impulsion avec élimination sélective des harmoniques (SHE PWM) [7-10, 64-66].

Dans ce chapitre, on commence par une brève description des véhicules électriques. Après on présente des rappels sur la machine asynchrone qui est le meilleur candidat pour la propulsion

des véhicules électriques. Les techniques de commande des machines asynchrones sont présentées dans la même section. Par la suite, on montre le principe de fonctionnement des onduleurs et leurs différentes méthodes de commande. Enfin, on introduit le principe de la commande SHE PWM.

2- Le Véhicule électrique

2.1 Introduction

Actuellement, la majorité des travaux de recherche sont orientés pour résoudre les problèmes environnementaux et de consommation d'énergie. Le problème environnemental principal est le changement climatique dû aux rejets des gaz à effet de serre provenant de l'activité humaine. Ces rejets ont augmentés dans les dernières décennies et les conséquences sont de plus en plus évidentes, comme l'augmentation de la température globale de la terre, la diminution des glaciers etc… [3].

D'autre part, le problème de la consommation d'énergie est dû à la nature des ressources fossiles qui sont limitées et sous réserve d'épuisement. Leur épuisement parviendra d'autant plus vite que la consommation est grande. Actuellement, le plus grand consommateur d'énergie fossile est le secteur du transport [1-3].

Ainsi, pour faire face à ces deux problèmes, un intérêt croissant est porté aux énergies renouvelables, ainsi qu'aux transports à traction électrique [61-63, 67].

2.2 Bref historique [61, 67]

Le premier véhicule électrique a fait son apparition dans les années 1830. La première personne à avoir inventé une voiture électrique est Robert Anderson. Il s'agissait plutôt d'une carriole électrique.

En 1911, l'invention du démarreur automatique rendait les véhicules à combustion interne plus commodes, les raisons étaient simples : d'une part, 1 gallon d'essence fournissait la même autonomie qu'une batterie de 270kg, d'autre part, il fallait une minute pour recharger un véhicule thermique, alors qu'une batterie mettait des heures pour être rechargée. Enfin, le prix des batteries à cette époque n'était pas abordable pour le grand public. Néanmoins, les EVs continuèrent à être utilisés dans quelques activités spécifiques pour les avantages qu'elles présentent : absence de bruit et de pollution.

À la fin des années quatre-vingt-dix, l'environnement devenant un problème majeur, l'épuisement annoncé des gisements de pétrole et l'augmentation du prix de ce dernier ont remis en question l'utilisation des véhicules thermiques. Les constructeurs d'automobiles mettent actuellement l'accent sur la recherche dans le domaine des véhicules électriques et des énergies renouvelables.

2.3- Les avantages et les limites des EVs

Les véhicules conventionnels équipés avec des moteurs à combustion interne fournissent une grande autonomie grâce à la grande densité énergétique des carburants pétroliers. Cependant, ces moteurs sont caractérisés par un rendement faible, une pollution environnementale, des émissions de gaz à effet de serre et une grande dépendance des ressources pétrolières [67].

D'autre part, les véhicules tout électriques possèdent de grands avantages par rapport aux véhicules traditionnels. En effet, ils sont [67] :

- sans pollution environnementale locale,
- sans émission des gaz à effet de serre (en supposant que la production d'électricité est propre),
- sans dépendance des ressources fossiles,
- avec un bon rendement global de la chaîne de traction,
- permettent un freinage électrique avec récupération d'énergie,
- une conduite plus souple,
- d'une conception simplifiée.
- silencieux.

Malgré ces avantages, l'inconvénient majeur des véhicules électriques est leur source d'énergie, c'est-à-dire les batteries, qui sont caractérisées par une faible densité d'énergie engendrant ainsi une autonomie réduite et un véhicule aux faibles performances (augmentation du poids et de la durée de rechargement).

2.4- Éléments constituant un véhicule électrique

Il comprend trois systèmes majeurs : un système de traction électrique, une source d'énergie et des accessoires figure 1.1.

2.4.1- Le système de traction électrique

C'est la partie responsable à la transformation d'énergie électrique en énergie mécanique. De plus, lors des phases de décélération ou de freinage elle transforme aussi l'énergie mécanique en énergie électrique. Elle se compose des sous-systèmes suivants [68] :

a- Le moteur électrique

On utilise plusieurs types de moteurs selon l'application, on retrouve souvent les moteurs à courant continu et les moteurs asynchrones.

b- La partie électronique de puissance

À partir de l'alimentation principale, elle module le signal qui va attaquer le moteur électrique.

Pour un moteur à courant continu, on utilise un hacheur. Pour un moteur asynchrone, on utilise un onduleur.

c- La partie contrôle et régulation

Suivant les entrées des systèmes qui sont les pédales de freinage et d'accélération, la pente, et le poids du véhicule, le but de la partie contrôle et régulation est d'asservir et réguler l'alimentation du moteur électrique et recharger les batteries.

d- La partie transmission mécanique

Il existe plusieurs configurations mécaniques du véhicule. Selon ces configurations, la partie transmission mécanique peut être simplement un arbre de transmission qui relie le moteur à la roue.

2.4.2- Le système de distribution et de stockage de l'énergie

Il comprend la source d'énergie, l'unité de gestion d'énergie, et l'unité de ravitaillement en énergie.

2.4.3- Le bloc des systèmes auxiliaires

Ce sont tous les autres systèmes qui sont utilisés afin d'apporter plus de confort et de facilité d'utilisation pour le conducteur. Il comporte au minimum une unité de gestion et transfert de la puissance, plus une unité de contrôle de la température et une unité d'alimentation auxiliaire.

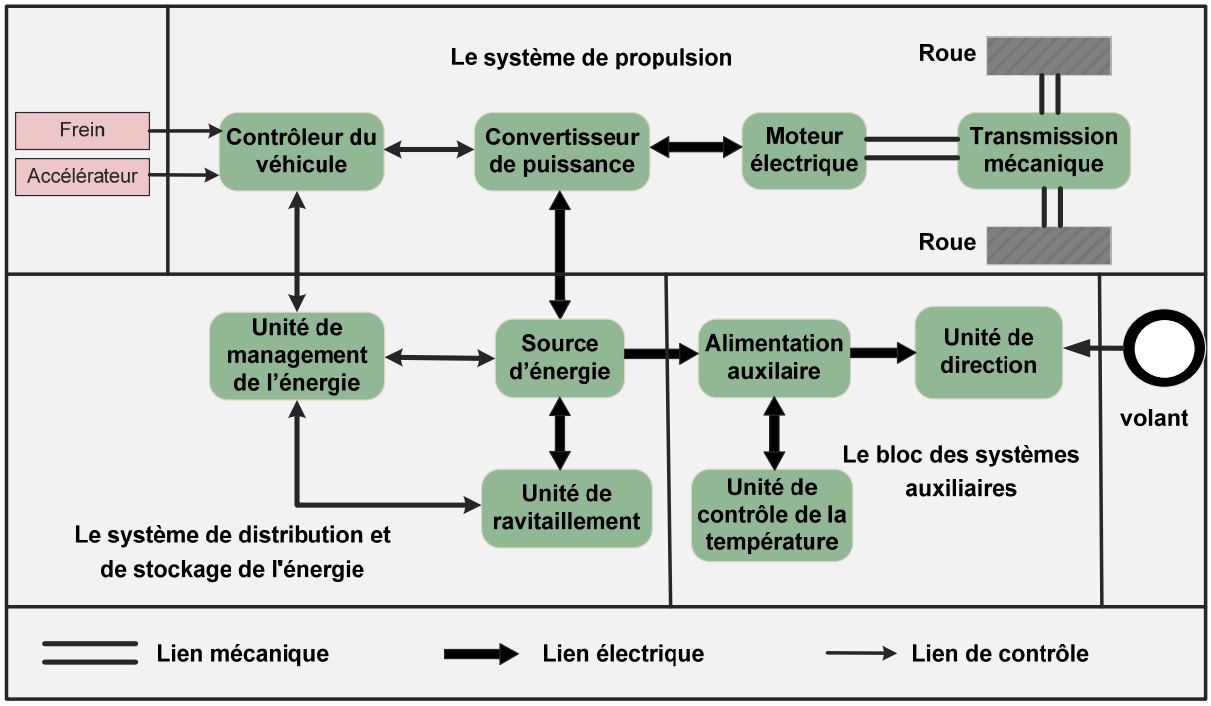


Figure 1.1 : Schéma bloc d'un véhicule électrique.

3- La machine asynchrone

3.1- Introduction

La MAS a de nombreux avantages par rapport aux autres types de machines électriques tournantes, parmi lesquelles nous pouvons citer : la robustesse, le prix relativement bas et l'entretien facile et moins fréquent. La MAS est aujourd'hui la plus utilisée dans les applications industrielles où la variation de vitesse, la haute précision de régulation et les hautes performances en couple sont requises. Cependant il faut noter que ces avantages ont longtemps été inhibés par la complexité de la commande due au couplage non linéaire existant entre le flux magnétique et le couple moteur. L'utilisation à grande échelle à l'heure actuelle est due à l'évolution technologique, notamment en matière de semi-conducteurs (MOSFET, IGBT, …etc.) et circuits électroniques programmables (DSP, FPGA,…etc.).

3.2- Définition, constitution et principe de fonctionnement

Le moteur asynchrone, ou moteur à induction, est une machine électrique alimentée en alternatif. La robustesse, le faible coût, les performances et la facilite d'entretien font l'intérêt de ce moteur. Il est utilisé aujourd'hui dans de nombreuses applications, notamment dans le transport (métro, trains, propulsion de navires, EVs) et dans l'industrie (machines-outils). D'après l'étude comparative de topologies de moteurs utilisées dans un EV réalisée par [62,63] le moteur asynchrone est parmi les meilleurs candidats pour la propulsion des véhicules électriques.

Un moteur asynchrone est une machine à 2p pôles, alimentée à partir du réseau alternatif de fréquence f et qui ne tourne pas exactement à la vitesse synchrone N définie par :

$$N = f / p = \omega / 2\pi p \quad [\text{t/s}] \tag{1.1}$$

Le moteur asynchrone est formé d'un stator, relié à la source triphasée, et d'un rotor constitué d'un enroulement polyphasé en court-circuit.

Le stator crée un flux tournant à la vitesse angulaire synchrone :

$$\Omega = \omega / p. \tag{1.2}$$

Ce flux tournant balaie les enroulements du rotor et y induit des courants. L'interaction entre le flux statorique et les courants induits rotoriques crée le couple de rotation du rotor.

Si le rotor tournait à la même vitesse que le flux tournant, le flux à travers les enroulements du rotor ne varie plus et il n'y aurait plus de courants induits dans le rotor, donc il n'y aurait pas de couple. Cependant, à cause des forces de frottements, le rotor n'atteindra jamais la vitesse de synchronisme. En conséquence, le rotor tourne à une vitesse Ω' plus petite que Ω. L'écart entre Ω' et Ω augmente lorsque le couple résistant sur l'arbre du rotor augmente. On appelle

glissement l'écart des vitesses angulaires synchrone Ω et réelle Ω' rapporté à la vitesse synchrone Ω :

$$g = (\Omega - \Omega')/\Omega = (\omega - \omega')/\omega = (N - N')/N \quad (1.3)$$

Avec : $N = \Omega/2\pi$ et $N' = \Omega'/2\pi$ (t/s) (1.4)

3.3- Commande des moteurs asynchrones

On distingue deux types de commandes : les commandes scalaires et les commandes vectorielles. La commande scalaire est basée sur le modèle en régime permanent, elle est simple à implanter avec une dynamique lente. Elle contrôle les grandeurs en amplitude.

La commande vectorielle est basée sur le modèle transitoire, elle est précise et rapide, elle permet le contrôle du couple à l'arrêt ; elle est cependant chère car elle requiert un encodeur incrémental ou un estimateur de vitesse, un DSPetc.…. Elle contrôle les grandeurs en amplitude et en phase [29-31].

3.4- La commande scalaire

Plusieurs commandes scalaires existent selon que l'on agit sur le courant ou sur la tension. Elles dépendent surtout de la topologie de l'onduleur utilisé (onduleur de tension ou de courant). Actuellement, l'onduleur de tension est le plus utilisé en petite et moyenne puissance avec une commande en V/f constant [29].

a- Contrôle en V/f constant

Son principe est de maintenir V/f constant, ce qui est équivalent à garder le flux maximal constant. Le contrôle du couple se fait par l'action sur le glissement. Dans notre étude on a utilisé ce type de commande.

En effet, d'après le modèle établi en régime permanent, le couple maximum s'écrit [29, 63] :

$$T\max = \frac{3pV^2}{4\pi f\,(Rs + \sqrt{Rs^2 + [2\pi f\,(Ls + Lre)]^2})} \quad (1.5)$$

Avec Rs et Ls respectivement la résistance et l'inductance du stator, Lre l'inductance du rotor ramenée au stator, p le nombre de paires de pôles, V la tension efficace d'entrée du moteur (d'une phase) et f est la fréquence de la tension d'alimentation.

Sachant que, dans l'équation (1.5), les paramètres Rs, Ls, p et Lre sont constants, alors le couple Tmax est fonction seulement des variations de V et f.

En conséquence, pour faire varier la vitesse du MAS, il faut faire varier la fréquence et la valeur efficace des tensions d'alimentation conjointement. Le rapport V/f doit varier en fonction de la fréquence, de la façon suivante :

- Pour des fréquences en dessous de la fréquence nominale Fr, le rapport V/f doit être maintenu constant pour que le flux totalisé reste approximativement constant. Dans ce cas on obtient un couple de sortie maximum Tmax constant, figure 1.2. La tension d'entrée du moteur V devrait être à la valeur maximale Vr quand f est égale à la fréquence nominale Fr.
- Pour des fréquences supérieures à la fréquence nominale Fr, la tension V est maintenue constante à la valeur nominale Vr. Dans ce cas, le couple est proportionnel à $1/f^2$, figure 1.2).

En effet, dans notre étude, on s'intéresse à la région située en dessous de la fréquence nominale Fr.

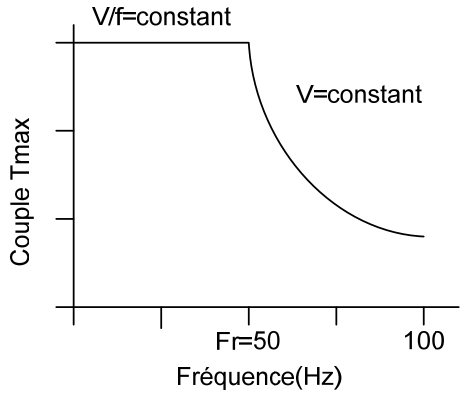


Figure 1.2 : Le couple moteur Tmax en fonction de la fréquence avec le rapport V/f constant puis avec V constant.

b- Contrôle du courant

Cette commande est appliquée dans un onduleur de courant. On impose directement des courants dans les phases de la machine. La valeur du courant Id (courant continu) est égale, à une constante près, à la valeur efficace du courant imposé Is. Cette valeur est imposée par régulation à l'aide d'un pont redresseur contrôlé.

4- Onduleurs de tension

4.1- Définition et principe

Un onduleur est un dispositif permettant de transformer en alternatif une énergie électrique de type continue. Ils sont utilisés pour :

- Fournir des tensions ou courants alternatifs avec une fréquence et une amplitude variable. Ces onduleurs servent à alimenter des moteurs à courant alternatif devant tourner avec une vitesse variable (comme le cas des EVs).

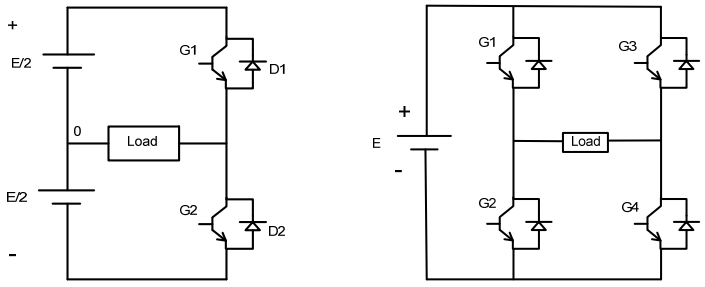
- Fournir une ou des tensions alternatives de fréquence et d'amplitude fixes. Ces onduleurs sont appliqués, en particulier, dans les alimentations de sécurité destinées à se substituer au réseau en cas de défaillance de celui-ci.

On distingue les onduleurs de tension et les onduleurs de courant, en fonction de la source d'entrée continue : source de tension ou source de courant. La technologie des onduleurs de tension est la plus maîtrisée et on la trouve dans la plupart des systèmes industriels et dans toutes les gammes de puissance (quelques Watts à plusieurs MW).

4.2- Types d'onduleurs de tension

a- Les Onduleurs monophasés

Ce type d'onduleurs est destiné à alimenter des charges alternatives monophasées, on distingue deux configurations de base : en demi-pont ou en pont complet, figure 1.3.



(a) (b)

Figure 1.3. Schéma des onduleurs monophasés : a- demi-pont, b- pont complet

b- Les onduleurs triphasés

Les onduleurs monophasés sont utilisés pour des applications de faible puissance, alors que les onduleurs triphasés couvrent la gamme des moyennes et des fortes puissances comme le cas des véhicules électriques.

L'objectif de cette topologie est de fournir une source de tension triphasée, dont l'amplitude, la phase et la fréquence sont contrôlables. Dans notre étude on a utilisé cette topologie. La figure 1.4 montre une MAS commandée par un onduleur triphasé

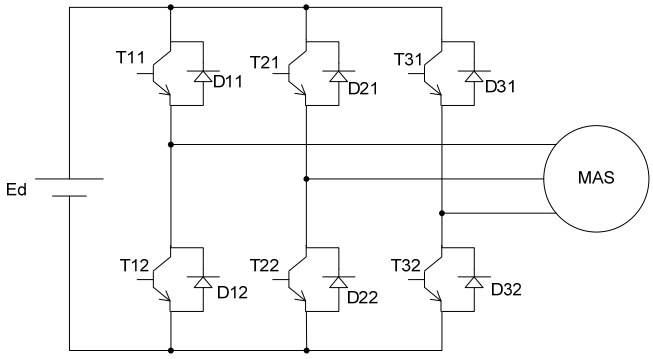


Figure 1.4. Schéma de principe d'un onduleur triphasé de tension

c- Les onduleurs multi-niveaux

Par définition, l'onduleur de tension multi-niveaux possède trois ou plusieurs niveaux. Dans la littérature on trouve plusieurs topologies multiniveaux, les plus utilisées sont la topologie à diode de bouclage, la topologie au condensateur flotteur et la topologie en cascade.

4.3- La commande des onduleurs

Le rôle de la fonction de commande est de déterminer les instants de commutation et les ordres de commande logique des interrupteurs afin d'obtenir une séquence de commutation de ces derniers. Le choix d'une stratégie de modulation peut s'effectuer en fonction des performances souhaitées par l'utilisateur. Toute les stratégies ont des avantages et des inconvénients et peuvent être réalisées par programmation logicielle ou matérielle.

Plusieurs stratégies de commande des onduleurs ont été développées dans la littérature dont les principes consistent soit [29-31] :

- La génération des signaux de commande des interrupteurs de puissance par l'asservissement de la tension de sortie de l'onduleur à une référence de tension sinusoïdale : c'est la commande dite implicite, technique analogique telle que principalement la MLI engendrée, la modulation Delta.

- La détermination des instants de commutation des composants de puissance formant l'onduleur par le biais du développement en série de Fourier des formes d'onde souhaitées en sortie répondant à des critères bien définies (taux d'harmoniques, valeur du terme fondamental,..). C'est la commande dite explicite où la commande des interrupteurs peut être analogique ou numérique telle que la technique de la Sortie Sinusoïdale Synthétisée ou la modulation programmée.

4.4- La commande en Modulation de Largeur d'Impulsion MLI

La technique de modulation en largeur d'impulsion MLI (Modulation de Largeur d'Impulsion ou PWM : (Pulse Width Modulation) est l'essor et le fruit du développement de l'électronique de puissance à la fin du siècle dernier. Elle est le cœur du contrôle des convertisseurs statiques. L'objectif de la technique PWM pour commander un onduleur de tension est d'avoir une réponse rapide et des performances élevées.

Le choix de la technique dépend aussi du type de la machine à commander, du type des semi-conducteurs, de la puissance mise en jeu et la simplicité ou la complexité d'algorithmes à implanter.

Il existe plusieurs techniques PWM. Une brève description de l'ensemble de ces techniques est résumée dans les paragraphes suivants.

a. PWM engendrée :

Parmi les variantes de la modulation PWM engendrée, la plus populaire étant la modulation sinusoïdale "Modulation sinus-triangle SPWM (Sinusoïdal PWM)".

Cette technique consiste à comparer une tension de référence de fréquence Fr, image du signal souhaité à la sortie, appelée modulante, avec une porteuse triangulaire ou en dent de scie de fréquence Fp. Les points d'intersection entre la modulante et la porteuse correspondent aux instants de commutation au moment desquels l'onduleur change d'état.

b- MLI Vectorielle :

La modulation vectorielle SVM (Space Vector Modulation) est une technique numérique. Les ordres de commutations des interrupteurs sont déterminés par un algorithme et sont calculés analytiquement à travers des équations mathématiques avec un vecteur de tension de contrôle qui est calculé globalement et approximé sur une période de modulation, par un vecteur de tension moyen, puis les ordres de commandes adéquats sont appliqués aux interrupteurs.

c- MLI calculée (programmée) :

Cette technique consiste à calculer les instants de commutation des interrupteurs (séquences de fonctionnement) de manière à répondre à certains critères portant sur le spectre fréquentiel de l'onde délivrée par l'onduleur. Ces séquences de fonctionnement sont en général mémorisées et restituées cycliquement pour assurer la commande des interrupteurs. Les critères usuellement retenus sont : l'élimination d'harmoniques de rangs spécifiés ou l'élimination d'harmoniques dans une bande de fréquences spécifiée.

Parmi ces techniques on trouve la technique SHE PWM (selective harmonics elimination PWM) qui consiste à éliminer les premiers harmoniques et à contrôler le fondamentale. Cette technique sera détaillée dans la section suivante.

5- La technique SHE PWM

5.1- Introduction

Le principe de la technique de modulation par élimination d'harmoniques SHE PWM a été introduit pour la première fois par Turnbull en 1964 puis développé par Patel et Hoft [12-13].

Cette technique consiste à former l'onde de sortie d'une succession de créneaux de largeurs variables et contrôlables. Les angles de commutation sont déterminés de façon à éliminer certains harmoniques gênants dans l'onde de sortie. Améliorant ainsi le rendement du système onduleur-machine par la réduction des ondulations du couple, ainsi que des pointes de courant et des pertes dans la machines [64-66].

Malgré la difficulté de calcul des angles de commutation, la technique SHE PWM présente plusieurs avantages par rapport à la PWM engendrée à modulation sinusoïdale (SPWM) [11,

70-73]. Meilleure élimination des harmoniques indésirables, pertes de commutation moindres, la limite minimale de largeur d'impulsion peut être atteinte facilement et la sur-modulation est possible.

5.2- Principe :

La technique SHE PWM est basée sur l'algorithme de Patel et Hoft [12-13]. Dans cette technique, il est possible d'asservir le fondamental de la tension et d'annuler les amplitudes des (m-1) premiers harmoniques.

La tension de sortie de l'onduleur, figure 1.3-a, est définie en fonction des angles exacts de commutation $\alpha_1,...,\alpha_m$, figure 1.5, qui correspondent aux instants de commutation de la tension d'une valeur positive +E/2 à une valeur négative –E/2 ou inversement. L'indice m est le nombre d'angles de commutation de la tension de sortie de l'onduleur par quart-d'onde.

La tension de sortie de l'onduleur est construite de façon à présenter une symétrie demi-onde (fonction impaire par rapport à l'angle π). Cette symétrie permet de supprimer certains types d'harmoniques, ce qui simplifie le développement en série de Fourier de cette tension et réduit le taux d'harmoniques. Ensuite, on fixe l'amplitude du fondamental à la valeur im et on annule les amplitudes des (m-1) premiers harmoniques.

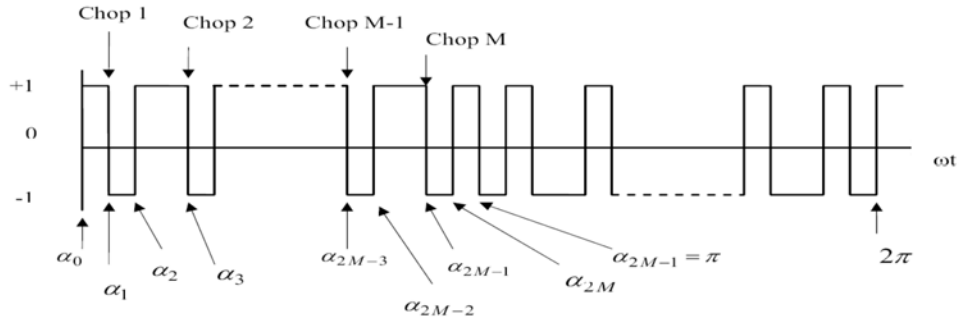


Figure 1.5 : La tension normalisée de sortie de l'onduleur demi-pont

Le taux im est le taux de modulation défini par :

$$im = \frac{V}{(E/2)} \tag{1.6}$$

Avec V la tension du fondamental.

D'après la figure 1.5 qui représente la tension normalisée de sortie de l'onduleur demi-pont de la Figure 1.3-a, les angles de commutation impairs $\alpha_1,\alpha_3,...$ définissent des transitions négatives, tandis que les angles de commutation pairs $\alpha_2,\alpha_4,...$ définissent des transitions positives. On suppose que la tension de sortie est périodique et d'amplitude unité. Dans ce cas, la tension de sortie f(α) peut s'écrire en série de Fourier sous la forme :

$$f(\alpha) = a_0 + \sum_{n=1}^{\infty}(a_n \sin(n\alpha) + b_n \cos(n\alpha)) \tag{1.7}$$

Les coefficients a_n et b_n sont donnés par :

$$\begin{aligned} a_0 &= \frac{1}{2\pi}\int_0^{2\pi} f(\alpha)d\alpha \\ a_n &= \frac{1}{\pi}\int_0^{2\pi} f(\alpha)\sin(n\alpha)d\alpha \qquad n=1,2,3,4,\ldots \\ b_n &= \frac{1}{\pi}\int_0^{2\pi} f(\alpha)\cos(n\alpha)d\alpha \end{aligned} \tag{1.8}$$

Comme f(α) présente une symétrie demi-onde : f(α+ π)= - f(α)

La valeur moyenne a_0 est nulle et seulement les harmoniques impairs existent. Par conséquent, l'indice n prend les valeurs impaires 1, 3, 5, 7, 9,...

Les coefficients a_n et b_n sont alors donnés par :

$$\begin{aligned} a_0 &= 0 \\ a_n &= \frac{2}{\pi}\int_0^{\pi} f(\alpha)\sin(n\alpha)d\alpha \\ b_n &= \frac{2}{\pi}\int_0^{\pi} f(\alpha)\cos(n\alpha)d\alpha \end{aligned} \tag{1.9}$$

Remplaçons f(α) par sa valeur dans l'équation (1.9) :

$$a_n = \frac{2}{\pi}\left[\int_{\alpha_0}^{\alpha_1}(-1)^0 \sin(n\alpha)d\alpha\right] + \ldots\ldots\ldots\ldots\ldots + \frac{2}{\pi}\left[\int_{\alpha_{2M}}^{\alpha_{(2M+1)}}(-1)^{2M}\sin(n\alpha)d\alpha\right]$$

D'où :

$$a_n = \frac{2}{\pi}\left[\sum_{k=0}^{2M}\int_{\alpha_k}^{\alpha_{(k+1)}}(-1)^k \sin(n\alpha)d\alpha\right] = \frac{2}{n\pi}\left[\sum_{k=0}^{2M}(-1)^k\left(\cos(n\alpha_k) - \cos(n\alpha_{k+1})\right)\right]$$

Avec $\alpha_{2M+1} = \pi$ et $\alpha_0 < \alpha_1 < \alpha_2 < \ldots < \alpha_{2M+1}$.

$$a_n = \frac{2}{n\pi}\left[\cos(n\alpha_0) - \cos(n\alpha_{2M+1}) + 2\sum_{K=1}^{2M}(-1)^K \cos(n\alpha_K)\right]$$

Comme : $\alpha_0 = 0$, $\alpha_{2M+1} = \pi$

On déduit : $\cos(n\alpha_0) = 1$, $\cos(n\alpha_{2M+1}) = (-1)^n$

D'où : $a_n = \frac{2}{n\pi}\left[1-(-1)^n + 2\sum_{K=1}^{2M}(-1)^K \cos(n\alpha_K)\right]$

De même pour le coefficient b_n, on trouve, après simplifications, le résultat suivant :

$$b_n = \frac{-4}{n\pi}\sum_{K=1}^{2M}(-1)^K \sin(n\alpha_K)$$

Comme n doit être impair on peut écrire :

$$a_n = \frac{4}{n\pi}\left[1+\sum_{K=1}^{2M}(-1)^K \cos(n\alpha_K)\right] \quad \text{n impair} \qquad (1.10a)$$

$$b_n = \frac{4}{n\pi}\left[-\sum_{K=1}^{2M}(-1)^K \sin(n\alpha_K)\right] \quad \text{n impair} \qquad (1.10b)$$

D'autre part la forme d'onde f(α) présente une symétrie quart-d'onde c'est-à-dire :

f (α)=f ($\pi-\alpha$)

Et d'après la figure 1.5 on a :

$\alpha_K = \pi - \alpha_{2M-K+1}$ K=1,2,…,M

D'où :

$$\sin(n\alpha_{K)}) = \sin(n(\pi-\alpha_{2M-K+1})) = \sin(n\pi)\cos(n\alpha_{2M-K+1}) - \cos(n\pi)\sin(n\alpha_{2M-K+1})$$

Pour n impair on a :

$\sin(n\pi) = 0$ et $\cos(n\pi)$= -1

D'où : $\sin(n\alpha_K)$=sin ($n\alpha_{2M-K+1}$) K=1,2,…, M (1.11)

Remplaçons (1.11) dans (1.10b) on trouve : $b_n = \frac{4}{n\pi}\sum_{k=1}^{M}\left(\sin(n\alpha_k) - \sin(n\alpha_{2M-k+1})\right) = 0$

D'autre part : $\cos(n\alpha_k) = \cos(n(\pi-\alpha_{2M-k+1})) = \cos(n\pi)\cos(n\alpha_{2M-k+1}) + \sin(n\pi)\sin(n\alpha_{2M-k+1})$

D'où : $\cos(n\alpha_k) = -\cos(n\alpha_{2M-k+1})$ (1.12)

Remplaçons (1.12) dans (1.10a), on obtient : $a_n = \frac{4}{n\pi}\left[1+2\sum_{k=1}^{M}(-1)^k \cos(n\alpha_k)\right]$ (1.13)

Dans notre étude on a utilisé un onduleur triphasé, donc les harmoniques trois et multiples de trois sont éliminés automatiquement [11, 14, 18]. Ainsi, n prend des valeurs impaires différentes d'un multiple de 3.

5.3- Calcul des valeurs exactes des angles de commutation

Chaque équation (1.13) possède m variables inconnues $\alpha_1, \alpha_2, \alpha_3, ..., \alpha_m$. Le problème est de calculer les valeurs de ces angles de commutation qui permettent d'annuler les amplitudes a_n des (m-1) premiers harmoniques f_n et d'assigner une valeur im appelée indice de modulation, à l'amplitude a_1 du fondamental f_1.

D'autre part, il faut éliminer deux harmoniques de tension pour éliminer un harmonique de courant [18].

Comme l'amplitude du fondamental doit être fixée à une valeur déterminée, ceci fixe la première valeur de m à 3 (m étant le nombre de commutations par quart d'onde ou nombre de découpages par demi-onde). Par conséquent lorsque m augmente successivement par pas égal à 2, le nombre d'harmoniques de courant qui seront éliminés augmente par pas égal à 1[18].

On doit signaler que la valeur de l'indice de modulation im assignée au fondamental est un indice sans dimension variant de 0 à 1. Pour obtenir la valeur correspondante en volt, il faut multiplier im par E/2, équation (1.6), pour un l'onduleur demi-pont figure 1.3a et par E pour l'onduleur triphasé, figure 1.4.

En conséquence, les équations (1.13) forment un système de m équations non linéaires à m inconnues $\alpha_1, \ldots, \alpha_m$.

Ce système peur être résolu par la méthode itérative de Newton-Raphson. Pour que cette méthode converge on assigne une valeur négative (-im) au fondamental. Ce qui correspond à un déphasage de π du fondamental. Ce déphasage est sans effet sur le moteur [11-14,18].

Enfin on doit résoudre le système d'équations (1.14).

$$
\begin{aligned}
f_1(\alpha_1,\alpha_2,\alpha_3,..,\alpha_m) &= \frac{4}{\pi}\left[1+2\sum_{k=1}^{m}(-1)^k\cos(\alpha_k)\right]+im=0\\
f_2(\alpha_1,\alpha_2,\alpha_3,..,\alpha_m) &= \frac{4}{5\pi}\left[1+2\sum_{k=1}^{m}(-1)^k\cos(5\alpha_k)\right]=0\\
f_3(\alpha_1,\alpha_2,\alpha_3,..,\alpha_m) &= \frac{4}{7\pi}\left[1+2\sum_{k=1}^{m}(-1)^k\cos(7\alpha_k)\right]=0\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
f_m(\alpha_1,\alpha_2,\alpha_3,..,\alpha_m) &= \frac{4}{n\pi}\left[1+2\sum_{k=1}^{m}(-1)^k\cos(n\alpha_k)\right]=0
\end{aligned}
\qquad (1.14)
$$

Ce système d'équations peut s'écrire sous la forme itérative suivante [11-13, 18, 70] :

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix}_{k+1} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix}_{k} - \frac{8}{\pi} \begin{bmatrix} +\sin(\alpha_1) & -\sin(\alpha_2) & \cdots & +\sin(\alpha_m) \\ +\sin(5\alpha_1) & -\sin(5\alpha_2) & \cdots & +\sin(5\alpha_m) \\ \vdots & \vdots & \ddots & \vdots \\ +\sin(n\alpha_1) & -\sin(n\alpha_2) & \cdots & +\sin(n\alpha_m) \end{bmatrix}^{-1} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix} \quad (1.15)$$

Pour assurer la convergence de la méthode de Newton-Raphson, on doit avoir un bon estimé initial de la solution 'exacte' recherchée. On peut utiliser la méthode du gradient mais, celle-ci étant lente (convergence linéaire). On utilisera plutôt l'algorithme de Taufik, Mellitt et Goodman [18] pour estimer rapidement les valeurs initiales de la solution du système non linéaire.

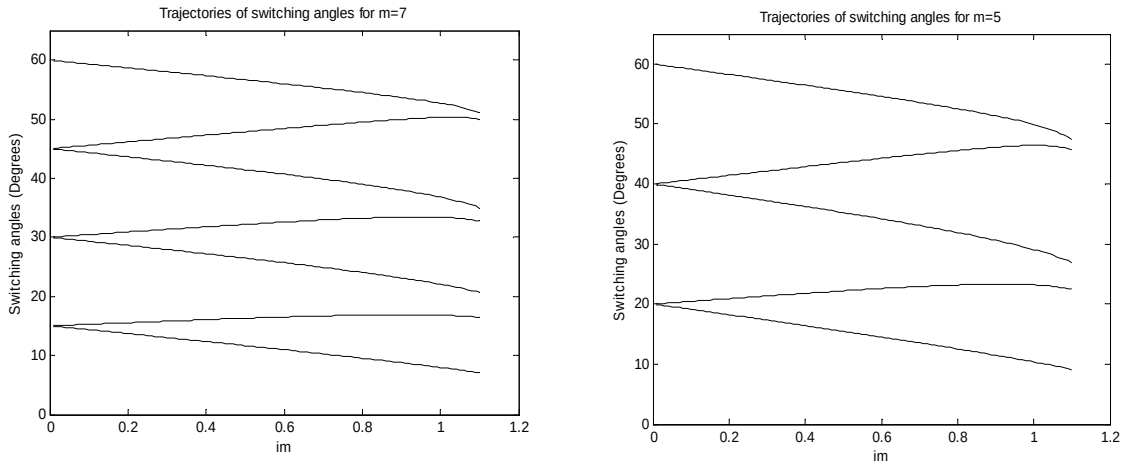
La méthode itérative est arrêtée lorsque l'une des conditions suivantes est vérifiée :

- $k \geq K_{MAX}$
- $\left| f_i\left(\alpha^{(k+1)}\right) \right| \leq E0$

Où E0 est une borne supérieure de l'erreur fixée à priori et K_{MAX} le nombre maximum d'itérations admissible.

En effet, nous avons mis au point un programme écrit en langage Matlab, pour calculer les angles de commutation exacts quel que soit la valeur de l'indice m.

La figure 1.6 donne, à titre d'exemple, le graphe des angles de commutation exacts calculés pour m égal à 5 et 7



(a) (b)

Figure 1.6 Courbes des angles de commutation exacts : a- m =7 et b- m = 5.

6- Conclusion

D'après ce chapitre, on conclut que les problèmes environnementaux et la consommation d'énergie sont derrière l'intérêt récent et croissant donné par de nombreux chercheurs et de constructeurs d'automobiles aux véhicules électriques. L'entraînement du moteur, qui est le cœur du système de propulsion de l'EV, est constitué d'un moteur électrique, un convertisseur de puissance et une unité de commande électronique. En raison de sa robustesse, entretien moindre, coût moindre par rapport à la puissance, la MAS est le meilleur candidat pour la propulsion des véhicules électriques.

Compte tenu de ses avantages, meilleure élimination des harmoniques indésirables, pertes de commutation moindres, la limite minimale de largeur d'impulsion peut être atteinte facilement et possibilité de la sur-modulation, la commande SHE PWM semble être candidat adéquat pour commander la vitesse d'une machine Asynchrone dans un EV. Cependant, le calcul des valeurs exactes des angles de commutation, lors de l'utilisation de cette technique, exige la résolution d'un système de m équations non linéaires à m inconnues en temps réel.

Pour la résolution de système on a utilisé la méthode itérative de Newton-Raphson. De plus pour assurer la convergence on a utilisé l'algorithme de Taufik, Mellitt et Goodman pour estimer les valeurs initiales de la solution.

Le calcul des angles de commutation par cette méthode est précis mais nécessite un temps de calcul très élevé qui empêche une commande de vitesse en temps réel "online". En conséquence, la technique SHE PWM est une technique "off-line". Autrement dit, les valeurs exactes des angles de commutation sont d'abord calculées sur ordinateur "off-line", ensuite ces valeurs sont stockées en mémoire pour pouvoir être utilisées pour la commande de vitesse du moteur. Un tel procédé requiert une large mémoire pour stocker tous les angles calculés et atteindre une bonne précision. Cet inconvénient a poussé les chercheurs à concevoir d'autres techniques permettant le calcul des angles de commutation de la SHE PWM on-line et en temps réel. Dans l'introduction générale on a mentionné les différents travaux qui ont traités ce sujet. Suivant la même approche, dans cette thèse, on propose un nouvel algorithme ANNSHE PWM basé sur la théorie des Réseaux de Neurones Artificiels (Artificial Neural Network (ANN)) et la commande SHE PWM permettant de calculer les angles de commutation et générer les signaux de commande PWM on-line et en temps réel.

Vu que l'algorithme proposé est basé sur la théorie des Réseaux de Neurones Artificiels, on a jugé utile d'introduire un chapitre pour expliquer brièvement le principe de cette théorie.

Chapitre II : Les Réseaux de Neurones Artificiels

1- Introduction

L'arrivée des ordinateurs permet à l'homme d'effectuer diverses tâches avec deux capacités : la rapidité et la précision. Cependant l'exécution d'une tâche par un ordinateur nécessite sa programmation préalable par l'homme. Cette caractéristique fait apparaître les ordinateurs comme des machines exécutant des ordres.

La poursuite de construire des machines ressemblant à des humains intelligents, capables d'apprendre, de raisonner et de réfléchir sans intervention, a mené à la naissance de réseaux de neurones artificiels (ANN). Ce sont des recherches basées sur le fonctionnement du cerveau qui ont constituées le point de départ. Ensuite, des recherches en neurobiologie ont montré que le cerveau est constitué d'un nombre extrêmement élevé d'unités de traitement élémentaire de l'information, les neurones biologiques, fortement interconnectées. En conséquence, les réseaux de neurones artificiels sont inspirés de l'étude du système nerveux et leur fonctionnement imite celui des réseaux de neurones biologiques.

Après l'arrivée de cette technique, beaucoup de travail basé sur des simulations dans l'ordinateur a démontré la capacité des ANN de mapper, modéliser, et de classer les systèmes non linéaires. Les caractéristiques particulières des ANN telles que la capacité d'apprentissage à partir des bases de données, les adaptations, le parallélisme, la robustesse au bruit, et la tolérance aux défauts ont ouvert leur application à divers domaines de l'ingénierie, la science, l'économie, etc [37].

Dans ce chapitre on présente des généralités sur les réseaux de neurones artificiels. Au début, on montre un bref historique sur le développent de cette technique. Après, on explique rapidement les bases biologiques dont elle est originaire. Ensuite, on présente le modèle mathématique d'un neurone artificiel et les différentes topologies des réseaux de neurones artificiels. Par la suite, on décrit l'étape d'apprentissage qui est l'étape la plus importante dans le développement d'un modèle ANN. Enfin, on cite les différents domaines d'application des ANN.

2- Bref historique [57,58]

Les réseaux de neurones artificiels sont inspirés du fonctionnement des cellules biologiques, au départ le but est la modélisation mathématique du cerveau humain. Les premiers qui proposent un modèle mathématique sont deux biophysiciens McCulloch et Pitts. Ces deux chercheurs montrèrent, en 1943, que des réseaux de neurones formels simples peuvent théoriquement réaliser des fonctions logiques, arithmétiques. Quelques années plus tard, en 1949, le physiologiste Hebb propose une formulation du mécanisme d'apprentissage, sous la forme d'une règle de modification des connexions synaptiques "règle de Hebb". Le premier réseau de neurones artificiels apparaît en 1957, grâce aux travaux de Rosenblatt qui conçoit le fameux "perceptron". C'est un réseau mono-couche basé sur la règle de Hebb et inspiré du système visuel. Ce réseau a été utilisé pour identification des formes simples et dans le calcul de certaines fonctions logiques. En 1982, Hopfield démontre l'intérêt d'utiliser des réseaux récurrents dits "feed-back" pour la compréhension et la modélisation des processus mnésiques. Ces réseaux récurrents constituent alors la deuxième grande classe de réseaux de neurones, avec les réseaux type perceptron (dits "feed-forward"). En Suite, des travaux de Hopfield, Werbos conçoit l'algorithme de rétropropagation de l'erreur qui offre un mécanisme d'apprentissage pour les réseaux multi-couches de type Perceptron (Multi-layer Perceptron). Cet algorithme de "back-propagation" a été développé et diffusé en 1986 par Rumelhart.

3- Le neurone biologique

Avant de commencer la présentation des réseaux de neurones artificiels, on a jugé utile d'introduire une section pour expliquer très brièvement les bases biologiques dont ils sont originaires.

Le neurone est le processeur élémentaire du cerveau, chaque neurone traite l'information qui lui parvient localement, puis transmet aux autres neurones qui lui sont connectés l'information qu'il a traitée ; ces cellules peuvent apprendre en changeant l'intensité de leurs connexions avec d'autres cellules ou détruire ou même créer de nouvelles connexions, elles peuvent aussi changer leurs règles de traitement de l'information. Ce processus de changement est appelé apprentissage et joue un rôle fondamental dans le comportement du neurone [56].

D'après la figure 2.1, nous pouvons constater que le neurone est composé des parties suivantes :

Le Corps Cellulaire : Il contient le noyau de neurone et effectue la transformation biochimique nécessaire à la synthèse des enzymes et d'autres molécules qui assurent la vie de neurone. Sa forme est pyramide ou sphérique dans la plupart des cas.

Les Dendrites : Chaque neurone possède une chevelure de dendrites celle-ci est de fines extensions tubulaires de quelques dizaines de micros. Elles sont les récepteurs principaux d'un neurone pour capter les signaux qui lui parviennent.

L'axone : L'axone sert de moyen de transport pour les signaux émis par le neurone. En effet il est généralement plus long que la dendrite.

La synapse : Représente la jonction entre un axone et une dendrite.

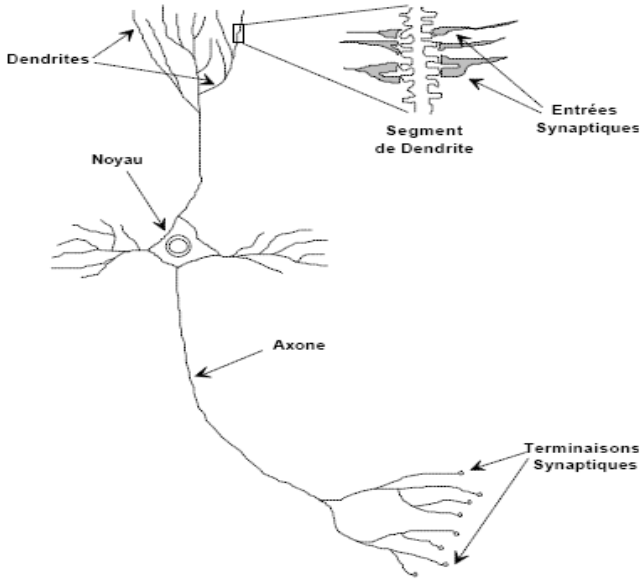


Figure 2.1 Représentation d'un neurone biologique [56,57]

4- Le modèle Mathématique d'un neurone artificiel

Dans la section précédente on a montré la structure de base d'un neurone bioligique qui est l'origine d'un neurone artificiel. En se basant sur cette structure, en 1943, MCCulloch et Walter Pitts ont inventé le neurone formel qui symbolise le modèle mathématique simplifié du neurone biologique [59]. Ces deux chercheurs montrèrent que des réseaux de neurones formels simples peuvent théoriquement réaliser des fonctions logiques, arithmétiques et symboliques complexes. Le tableau 2.1 montre la correspondance entre le neurone biologique et le neurone artificiel. De plus la figure 2.2 représente la modèle mathématique d'un neurone artificiel.

Tableau 2.1 Analogie entre le neurone biologique et le neurone artificiel [57]

Neurone biologique	**Neurone artificiel**
Corps cellulaire (Somma)	Fonction d'activation
Axones	Signal de sortie
Synapses	Poids synaptiques (connexions)
Dendrites	Signal d'entrée

Comme il est illustré sur la figure 2.2, chaque neurone est un processeur élémentaire qui reçoit un nombre variable d'entrées x_i en provenance des neurones amont. A chacune de ces entrées

est associé un poids *w*. Chaque processeur élémentaire est doté d'une sortie unique *y*, qui se ramifie ensuite pour alimenter un nombre variable de neurones avals. En conséquence, le neurone formel est un processeur élémentaire qui réalise une somme pondérée des entrées. La valeur de cette sommation est comparée à un seuil et la sortie de ce réseau est une fonction en générale non linéaire donnée par l'équation (1), cette fonction est appelée fonction d'activation.

$$y = f\left(\sum_{1}^{n} w_i x_i - \theta\right) \tag{2.1}$$

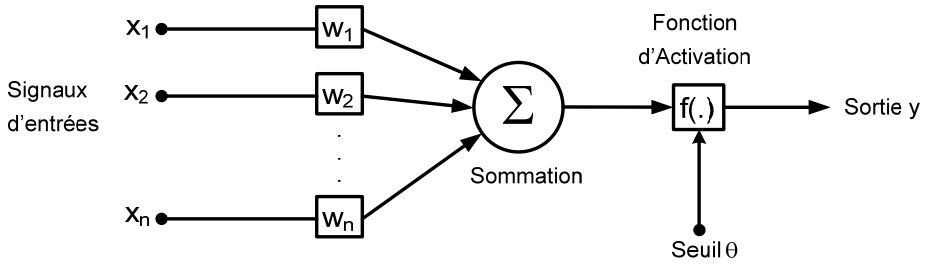


Figure 2.2 : Modèle Mathématique du neurone artificiel.

Dans la littérature on dénombre un ensemble de fonctions d'activations. Les plus courantes sont présentées sur la figure 2.3 :

- la fonction binaire à seuil (figure 2.3a).
- la fonction linéaire, linéaire à seuil ou seuils multiples (figure 3b, figure 3c et figure 3d).
- la fonction sigmoïde (figure 2.3e).
- Une fonction à base radiale (figure 2.3f).

Chacune des fonctions d'activation est adaptée à certaines applications. Par exemple, la fonction binaire est bien adaptée pour l'organisation et la distribution classification des entrées. Les fonctions linéaires, sigmoïdes, à base radiale et plus généralement des fonctions continues et dérivables sont appropriées pour l'approximation des fonctions continues [56].

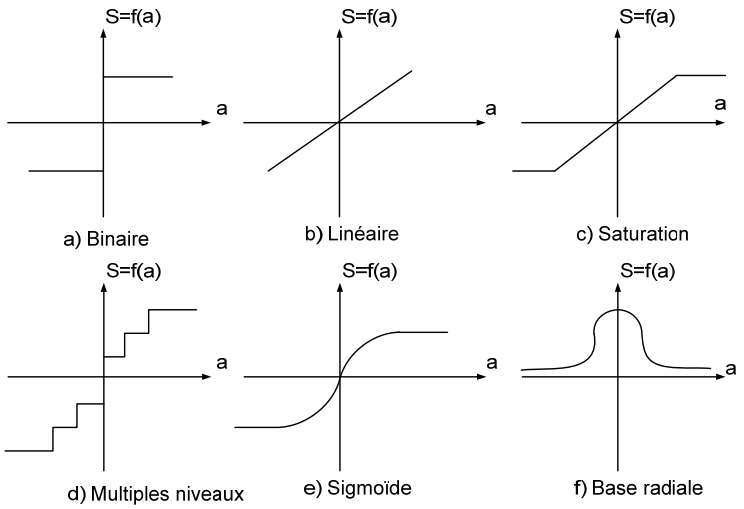


Figure 2.3 Types de fonctions d'activation

5- Architecture des réseaux de neurones [37, 56-58]

Les réseaux de neurones artificiels sont des réseaux fortement connectés de neurone formel fonctionnant en parallèle. Chaque neurone formel calcule une sortie unique sur la base des informations qu'il reçoit des autres neurones. Les connexions entre les neurones qui composent le réseau décrivent sa topologie ou son architecture. Dans les paragraphes qui se suivent on liste les types d'architectures les plus utilisées.

5.1- Réseau multicouche (MLP)

Cette topologie est appelé MLP (Multi Layer Perception). Dans cette architecture, les neurones sont arrangés par couches, il n'y a pas de connexion entre les neurones d'une même couche et les connexions ne se font qu'avec les neurones des couches avales, ceci nous permet d'introduire la notion de sens de parcours de l'information, figure 2.4. Par Définition, on appelle couche d'entrée l'ensemble des neurones d'entrée, couche de sortie l'ensemble des neurones de sortie. Les couches intermédiaires sont appelées couches cachées.

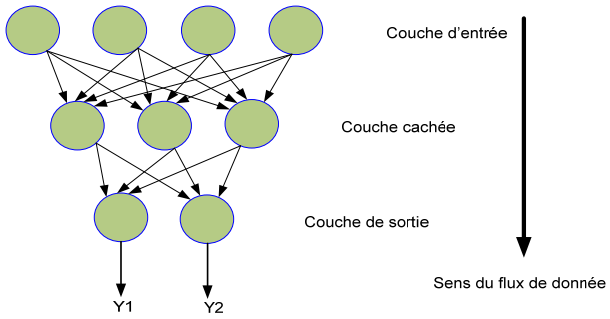


Figure 2.4 Réseau multicouche (MLP)

5.2- Réseaux récurrents "feed-back"

Les connexions récurrentes ramènent l'information en arrière par rapport au sens de propagation défini dans un réseau multicouches, figure 2.5.

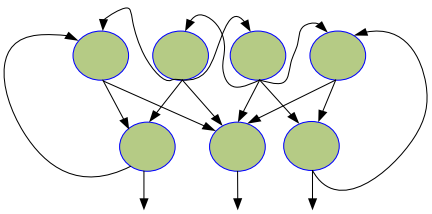


Figure 2.5 Réseau récurrent.

5.3- Réseau à connexion complète ou de Hopfield

Les réseaux de Hopfield sont des réseaux récurrents entièrement connectés. Dans ce type de réseau, chaque neurone est connecté aux autres neurones et il n'y a aucune distinction entre les neurones d'entrée et de sortie, figure 2.6.

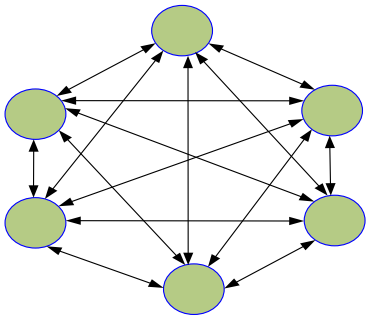


Figure 2.6 Réseau de Hopfield.

5.4 Choix d'une Architecture neuronale

Un problème qu'on doit impérativement résoudre avant d'utiliser un réseau de neurones est la définition de sa structure. Pour une topologie multicouche MLP, le nombre de neurones d'entrée/sortie du réseau de neurones est imposé par la structure de fonctionnement globale où il sera inséré, tandis que le nombre de couches cachées ainsi que leurs nombres de neurones correspondant à chaque couche, ne sont pas limités. Si l'objectif final étant une implémentation matérielle, il est nécessaire de développer une architecture neuronale aussi petite que possible.

6- Apprentissage des réseaux de neurones [37, 56-60]

L'apprentissage est la propriété la plus intéressante des réseaux neuronaux. Durant cette phase, le réseau de neurones modifie sa structure interne jusqu'à l'obtention du comportement désiré. Dans la majorité des algorithmes actuels, les variables modifiées pendant l'apprentissage sont les poids des connexions.

6.1- Les types d'apprentissage

Selon le degré de supervision utilisé dans l'apprentissage, on distingue trois catégories :

a- Apprentissage supervisé *:* les paramètres du réseau sont ajustés via un comportement de référence précis, les exemples sont des couples (Entrée, Sortie associée). Le réseau de neurones est entraîné de façon à reproduire le plus fidèlement possible les réponses désirées à un certain nombre de vecteurs d'entrée d'apprentissage. Parmi les algorithmes à apprentissage supervisé, on reconnaît dans la littérature l'algorithme à Rétro-propagation du gradient. Cette catégorie est la plus utilisée.

b- Apprentissage non supervisé : l'ajustement des paramètres repose sur des critères internes au réseau, aucune information de référence n'est disponible, on ne dispose que des valeurs d'entrée. Ce type est généralement utilisé pour la classification automatique des entrées, où le réseau apprend les caractéristiques des données d'entrée.

c- Apprentissage semi-supervisé ou apprentissage par renforcement : l'utilisateur ne possède que des indications imprécises sur le comportement final désiré (correct/incorrect). Il est

généralement utilisé dans les systèmes de navigation autonome où un robot doit choisir un chemin selon un critère de type «bon/mauvais chemin ».

6.2- Centrage des données

Avant tout apprentissage, il est indispensable de normaliser et de centrer toutes les données de la base d'apprentissage, afin qu'ils soient actifs, en moyenne sur la partie linéaire de la fonction d'activation qui en générale une sigmoïde.

En pratique, il est donc recommandé, d'appliquer pour chaque vecteur d'entrée e dans l'intervalle [e_{min}, e_{max}] la normalisation définit par la fonction suivante :

$$f_{norm}(e) = x = \frac{x_{max} - x_{min}}{e_{max} - e_{min}}(e - e_{min}) + x_{min} \qquad (2.2)$$

6.3- L'algorithme d'Apprentissage "rétropropagation du gradient" du Réseau multicouche MLP

Un réseau MLP est conçu pour réaliser une tâche souhaitée définit par une base de données d'apprentissage Database. Chaque élément de cette base de données est appelé exemple d'apprentissage et se présente sous la forme d'un couple (x, y*) où x est une valeur d'entrée du réseau et y* la valeur désirée correspondante pour les sorties. L'architecture du réseau, la structure de ses connexions, ainsi que les fonctions d'activation, peuvent être fixées en fonction de la tâche que doit remplir le réseau.

Le but de l'apprentissage est donc de déterminer les valeurs w* de la matrice W des poids des connexions du réseau de telle sorte que les sorties (y) soient proches des valeurs désirées y*.

Il existe plusieurs méthodes d'apprentissage, l'algorithme de la rétropropagation du gradient est très connu et le plus utilisé dans les applications des réseaux de neurones de type MLP. Cette technique d'apprentissage supervisé utilise une procédure de descente du gradient, travaillant sur l'erreur quadratique entre la sortie du réseau et la sortie désirée. Elle calcule les dérivées partielles de l'erreur en sortie par rapport à tous les poids du réseau puis applique une procédure de gradient pour minimiser l'erreur. À chaque itération, on retire un exemple d'apprentissage (x, y*) et on calcule une nouvelle estimation des poids W(t). Cette itération est réalisée en deux phases :

a- Propagation

À chaque itération, un élément de l'ensemble d'apprentissage est introduit à travers la couche d'entrée. L'évaluation des sorties du réseau se fait couche par couche, de l'entrée du réseau vers sa sortie.

b- Rétro propagation

Cette étape est similaire à la précédente. Cependant, le calcul s'effectue dans le sens inverse.

À la sortie du réseau, on forme le critère de performance en fonction de la sortie réelle du système et sa valeur désirée. Puis, on évalue le gradient de cette performance par rapport aux différents poids en commençant par la couche de sortie et en remontant vers la couche d'entrée.

7- Domaines d'applications des réseaux de neurones [57,60]

Les réseaux de neurones artificiels sont aujourd'hui appliqués dans divers domaine de l'ingénierie, la science, l'économie, etc. dans ce paragraphe on cite les plus importants :

- Défense : Direction des armes, poursuite des cibles, radars, etc.
- Industrie : contrôle qualité, control de processus, diagnostic de panne, système de guidage automatique des véhicules, etc.

Finance : prévision et modélisation du marché (cours de monnaies...), prévision des indicateur économiques, sélection d'investissements, prévision des prix, etc.

- Télécommunications et informatique : analyse du signal, élimination du bruit, reconnaissance de formes (bruits, images, paroles), compression de données, etc.
- Médical : analyse des signaux, analyse du cancer, etc.
- Environnement : évaluation des risques, analyse chimique, prévisions et modélisation météorologiques, gestion des ressources, etc.

8- Conclusion

D'après l'étude qu'on décrit dans ce chapitre, on conclut qu'un neurone artificiel est une modélisation mathématique du neurone biologique. Dans ce modèle, un neurone réalise une somme pondérée des entrées qui lui parviennent. Le résultat de cette sommation est comparée à un seuil et passe dans une fonction en générale non linéaire, appelée fonction d'activation, pour générer la sortie du neurone.

Les réseaux de neurones se distinguent par trois caractéristiques principales :

- L'architecture du réseau qui décrit la structure interne du réseau (nombre de couches, nombre de neurone dans chaque couche, les connections entre les différentes couches) et définit la direction de propagation des informations.
- Les fonctions d'activation des neurones qui dicte leur comportement.
- L'algorithme d'apprentissage du réseau qui permet de générer les poids associés à chaque neurone

Les propriétés les plus importantes des ANNs sont : la non linéarité, le parallélisme, l'apprentissage et la généralisation. Ces propriétés rendent les réseaux de neurone souhaitable pour l'identification des systèmes non linéaires qui lui permet d'être appliqué dans divers domaine de l'ingénierie, la science, l'économie, etc.

En se basant sur les propriétés des ANNs, dans le chapitre suivant on propose un nouvel algorithme permettant de calculer les angles de commutation et générer les signaux de commande PWM on-line et en temps réel. Ainsi, les différentes étapes qu'on a suivies pour développer et tester cet Algorithme seront présentées.

Chapitre III : L'algorithme MLI à élimination d'harmoniques sélectives basé sur les réseaux de neurones artificiels "ANNSHE PWM"

1- Introduction

Dans le chapitre I on a montré que la résolution du système d'équation du Patel et Hoft pour éliminer des harmoniques sélectionnés, en utilisant l'algorithme itératives du newton-Raphson, est précise mais nécessite un temps de calcul très élevé qui empêche le contrôle de la vitesse en temps réel donc l'algorithme de Patel et Hoft est 'off-line'. D'autre part, dans l'introduction on a cité les différentes recherches qui ont abordé ce sujet. Dans notre laboratoire l'étude a été commencé en 2003 où un nouvel algorithme basé sur l'interpolation polynomial a été proposé [29] ; les résultats de cet algorithme ont été significatif mais il ne était pas possible d'implémenter cet algorithme sur un microcontrôleur. Cependant, en 2007, on a réussi à implémenter cet algorithme sur un circuit FPGA [30-31]. Avec l'apparition et le développement des nouvelles techniques de commande intelligentes (Logique flou, Réseau de neurones Artificiel, les algorithmes génétiques, … etc.), on a proposé un nouvel algorithme basé sur les réseaux de neurones Artificiels (ANN) [31, 70-73], le but de cet Algorithme est de calculer les instants de commutation du signal PWM avec une précision très proche de ceux calculés par l'algorithme de Patel et Hoft. Cet Algorithme sera implémenté sur un circuit FPGA à fin de générer le signal PWM en temps réel. Dans ce chapitre on montre les différentes étapes qu'on a suivies pour développer et tester cet Algorithme.

2- L'Architecture de l'ANNSHE PWM

Le but de l'algorithme proposé et de construire un ANN multicouches (Multi Layer Perceptron MLP) qui sera par la suite implémenté sur un circuit FPGA ; pour simplifier l'implémentation et réduire l'espace consommé dans le circuit FPGA, on a choisi un réseau composé d'une couche d'entrée, une couche cachée et une couche de sortie, de plus la couche cachée contient un seul neurone. Cette architecture permet aussi de réduire l'erreur et le temps lors de calcul des angles de commutation puisque si on augmente le nombre de couches ou le nombre de

neurones dans les couches cachées l'erreur dans les angles de commutation et le temps de calcul seront multipliés [70].

Concernant les fonctions d'activation on a choisi la fonction tangente sigmoïde entre la couche d'entrée et la couche cachée pour présenter la non linéarité du système et une simple fonction linéaire entre la couche cachée et la couche de sortie. On a placé la fonction non linéaire entre la couche d'entrée et la couche cachée pour que cette fonction soit calculée une fois puisque si on la place entre la couche cachée et la couche de sortie cette fonction sera calculée pour chaque angle de commutation ; donc ce choix permet de réduire la complexité et le temps du calcul.

L'architecture de notre algorithme ANNSHE PWM est présentée sur la Figure 3.1 où :

im : l'indice de modulation qui est l'entrée du réseau

W1 : Matrice des poids (Weights) entre la couche d'entrée et la couche cachée.

W2 : Matrice des poids (Weights) entre la couche cachée et la couche de sortie.

b1 : Matrice des seuils (bias) entre la couche d'entrée et la couche cachée.

b2 : Matrice des seuils (bias) entre la couche cachée et la couche de sortie.

α : Matrice des angles de commutation qui présente la sortie de notre réseau.

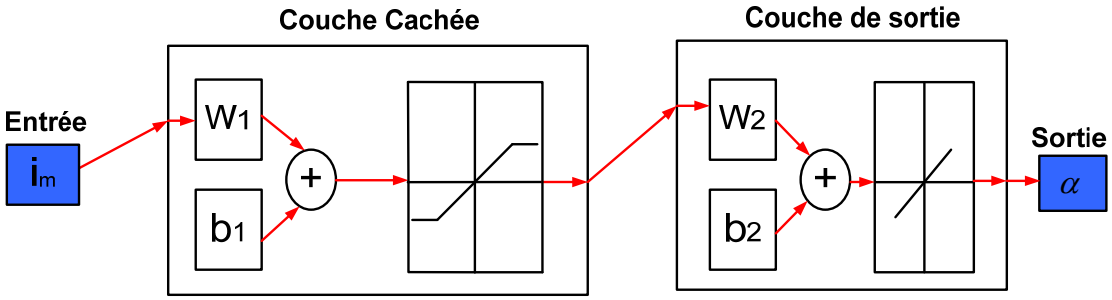


Figure 3.1 Architecture de l'ANNSHE PWM

3-La base de données

Dans cet étape on calcul la base de données qui sera utilisée par la suite dans l'étape d'apprentissage. Cette base de données montre la relation entre l'entrée de notre réseau qui est l'indice de modulation im qui varie de 0 à 1 et la sortie de notre réseau qui est la matrice α des angles de commutation. On veut construire une base de données de 100 valeurs donc im doit varie par pas de 0.01 et pour chaque valeur de im, on résout le système d'équation de Patel et Hoft (Equation 1.9) afin de trouver la matrice α des angles de commutation correspondante.

De plus pour que l'algorithme soit efficace c'est-à-dire les angles calculés par cet algorithme sont très proche des valeurs exactes calculées par la méthode itérative de newton-Raphson, et pour permettre la convergence de l'étape d'apprentissage, on divise l'intervalle de variation de im en six et pour chaque intervalle on construit un ANN spécifique (poids et seuils). Le choix de nombre d'angles de commutation (c'est-à-dire le nombre d'harmoniques à éliminer) dans

chaque intervalle dépend de la valeur de l'indice im et puisque l'effet des harmoniques augmente lorsque im diminue on prend le choix approprié qui est illustré dans le Tableau 3.1 ; ce choix permet aussi d'optimiser notre algorithme et donc minimiser l'espace consommé lors de l'implémentation [70-73].

Tableau 3.1 : les caractéristiques des réseaux ANNSHE PWM

L'indice de modulation im	**ANN**	**Nombre d'angles de commutation m**	**Nombre de neurones dans la couche cachée**	**Nombre de neurones dans la couche de sortie**	**L'algorithme d'apprentissage**	**Les fonctions d'activation**	**Nombre d'éléments dans la base de données d'apprentissage**	**Nombre d'éléments dans la base de données de test**	**Paramètres d'apprentissage**	
									performance	Nombre d'itérations
0< im <0.1	ANN-1	23	1	23	Back-propagation Algorithme	Hyperbolique Tangent Sigmoïde + linéaire	23x10	23x20	2.2 10^{-5} max 10^{-5}	64938 max 10^5
0.1< im <0.2	ANN -2	19	1	19			19x10	19x20	2.3 10^{-5} max 10^{-5}	61686 max 10^5
0.2< im <0.4	ANN-3	15	1	15			15x20	15x40	2.4 10^{-5} max 10^{-5}	58740 max 10^5
0.4< im <0.6	ANN-4	7	1	7			7x20	7x40	5.9 10^{-5} max 10^{-5}	23954 max 10^5
0.6< im <0.8	ANN-5	5	1	5			5x20	5x40	0.0004 max 10^{-5}	12532 max 10^5
0.8< im <1	ANN-6	3	1	3			3x20	3x40	0.0013 max 10^{-5}	06874 max 10^5

Pour trouver les paramètres de six ANN (poids et seuils), on a développé un programme sur MATLAB qui résout le système d'équation de Patel et Hoft (Equation 1.9) et génère les Six bases de données ; ces dernières seront utilisées dans l'étape d'apprentissage. Le Tableau 3.2 montre un exemple des angles de commutation calculés par le programme pour le réseau ANN-2 (le nombre d'angles de commutation m = 19) lorsque im varie de 0,11 à 0,15, de plus la Figure 3.2 montre la variation des angles de commutation dans les six intervalles en fonction de im.

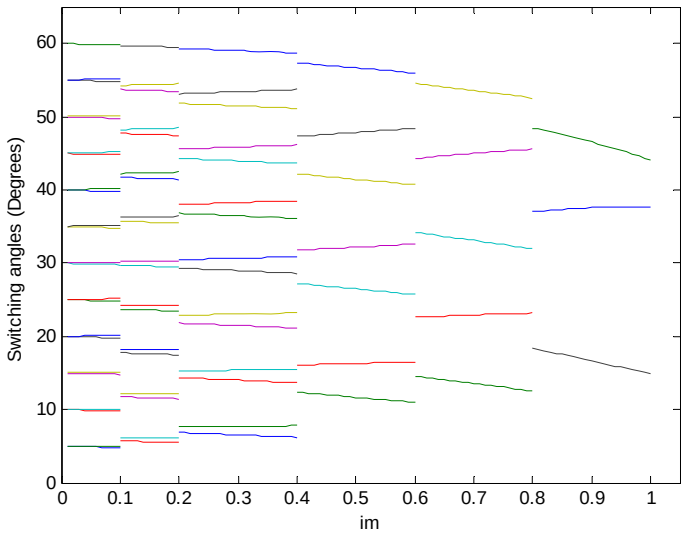


Figure 3.2 : Variation des angles de commutation dans les six intervalles en fonction de im

Tableau 3.2 : Base de données d'angles de commutation pour le réseau ANN-2 lorsque im varie de 0.11 à 0.15

im	0.11	0.12	0.13	0.14	0.15
α_1	5,71795	5,69226	5,66655	5,640839	5,61511
α_2	6,05435	6,05928	6,06420	6,069112	6,07401
α_3	11,71582	11,68985	11,66384	11,637820	11,61176
α_4	12,09877	12,10764	12,11650	12,125343	12,13415
α_5	17,71126	17,68479	17,65829	17,631742	17,60515
α_6	18,13658	18,14882	18,16104	18,173221	18,18536
α_7	23,70628	23,67931	23,65229	23,625223	23,59810
α_8	24,16932	24,18450	24,19964	24,214749	24,22980
α_9	29,70209	29,67472	29,64729	29,619817	29,59228
α_{10}	30,19779	30,21555	30,23326	30,250944	30,26857
α_{11}	35,69948	35,67188	35,64423	35,616531	35,58877
α_{12}	36,22241	36,24243	36,26240	36,282348	36,30225
α_{13}	41,69901	41,67140	41,64375	41,616050	41,58830
α_{14}	42,24344	42,26541	42,28734	42,309259	42,33113
α_{15}	47,70106	47,67369	47,64629	47,618850	47,59137
α_{16}	48,26102	48,28464	48,30825	48,331846	48,35541
α_{17}	53,7059	53,67905	53,65217	53,625263	53,59833
α_{18}	54,27525	54,30024	54,32523	54,350221	54,37520
α_{19}	59,71376	59,68768	59,66160	59,635520	59,60942

4- L'apprentissage des réseaux ANNSHE PWM

Dans cette étape les paramètres de six réseaux (ANN-i) qui sont les poids et les seuils seront calculés. La base de données de chaque réseau ANN-i, calculée dans la section précédente, est utilisée après normalisation comme entrée dans un programme d'apprentissage basé sur la méthode du gradient. Ce programme d'apprentissage a deux conditions d'arrêt, la performance et le nombre d'itérations. Lorsque l'une de ces conditions est vérifiée, d'apprentissage est arrêté pour générer les paramètres (poids et seuils) appropriés. Les paramètres de l'apprentissage ainsi que les caractéristiques de chaque réseau ANN-i sont montrés dans le Tableau 3.1. Le Tableau 3.3 montre, à titre d'exemple, les poids et les seuils calculés par le programme pour le réseau ANN-2 c'est-à-dire lorsque 0.1< im <0.2 et le nombre d'angles de commutation m=19.

5- Les résultats de simulation

Avant l'implémentation de l'algorithme ANNSHE PWM proposé sur FPGA on passe par l'étape de simulation. Cette étape permet de vérifier l'efficacité et la précision de l'algorithme. Pour faire une étude comparative entre les angles PWM exactes calculés par l'algorithme de newton-Raphson et ceux calculés par l'algorithme ANNSHE proposé, la Figure 3.3 montre les courbes de $\alpha 3$ (im) et α 7 (im) pour m = 7(ANN-4) et illustre la variation de l'erreur entre les angles de commutation PWM exactes et les angles de commutation ANNSHE pour $\alpha 3$ et $\alpha 7$. Le Tableau 3.4 donne, à titre d'exemple, l'erreur entre les angles de commutation exactes et

ceux calculés par l'algorithme ANNSHE pour im = 0,415 et im = 0,575(ANN-4). Le Tableau 3.5 donne une vue générale sur l'erreur moyenne et maximale entre les angles de commutation PWM exactes et ceux calculés par l'algorithme ANNSHE pour les six réseaux de neurones ANN-i. La Figure 3.4 montre la variation de l'erreur moyenne entre les angles de commutation PWM exactes et ceux de l'algorithme ANNSHE dans toute la gamme de variation de l'indice im c'est-à-dire pour les six réseaux ANN-i. De plus, la Figure 3.5 montre le signal ANNSHE PWM et son spectre correspondant obtenu par simulation pour im = 0,5 et im = 0,2.

Tableau 3.3 : Les poids et les seuils calculés par le programme pour le réseau ANN-2

W1	W2	b1	b2
-1.93639	0.66680	0.87904	0.49633
	-0.58300		0.50461
	0.66686		0.49675
	-0.64725		0.50568
	0.66695		0.49709
	-0.65798		0.50591
	0.66704		0.49729
	-0.66197		0.50584
	0.66711		0.49736
	-0.66397		0.50561
	0.66714		0.49731
	-0.66512		0.50528
	0.66714		0.49716
	-0.66584		0.50490
	0.66710		0.49693
	-0.66632		0.50449
	0.66701		0.49664
	-0.66665		0.50409
	0.66687		0.49629

Tableau 3.4 : l'erreur entre les angles de commutation exactes et ceux calculés par l'algorithme ANNSHE pour im = 0,415 et im = 0,575(ANN-4)

im= 0.415			**im= 0.575**		
Les angles exactes	**Les angles ANNSHE**	**Erreur**	**Les angles exactes**	**Les angles ANNSHE**	**Erreur**
12.25313	12.24218	0.01095	11.14998	11.14643	0.00355
16.09470	16.10522	-0.01051	16.46374	16.46603	-0.00228
27.10632	27.09751	0.00880	25.90467	25.90172	0.00294
31.83408	31.84794	-0.01385	32.48118	32.48518	-0.00400
42.08539	42.07688	0.00850	40.87523	40.87237	0.00286
47.38242	47.39558	-0.01315	48.28394	48.28802	-0.00408
57.25323	57.24281	0.01041	56.14964	56.14624	0.00339

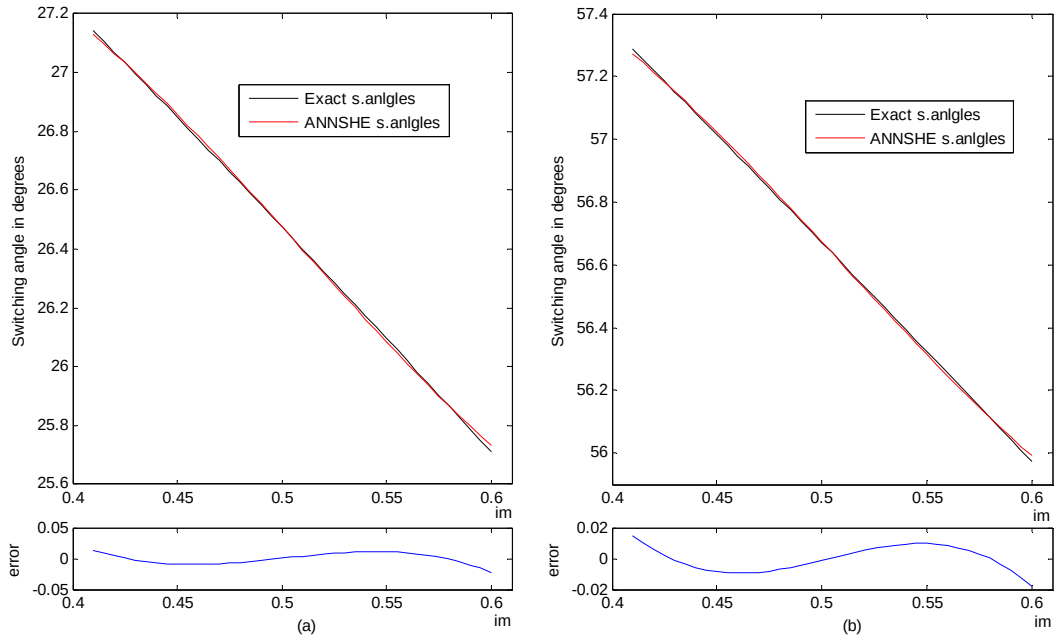


Figure 3.3 : Comparaison entre les angles de commutation exactes et ceux de l'algorithme ANNSHE PWM pour (a) : α3 (m=7) and (b) : α7 (m=7) (ANN-4)

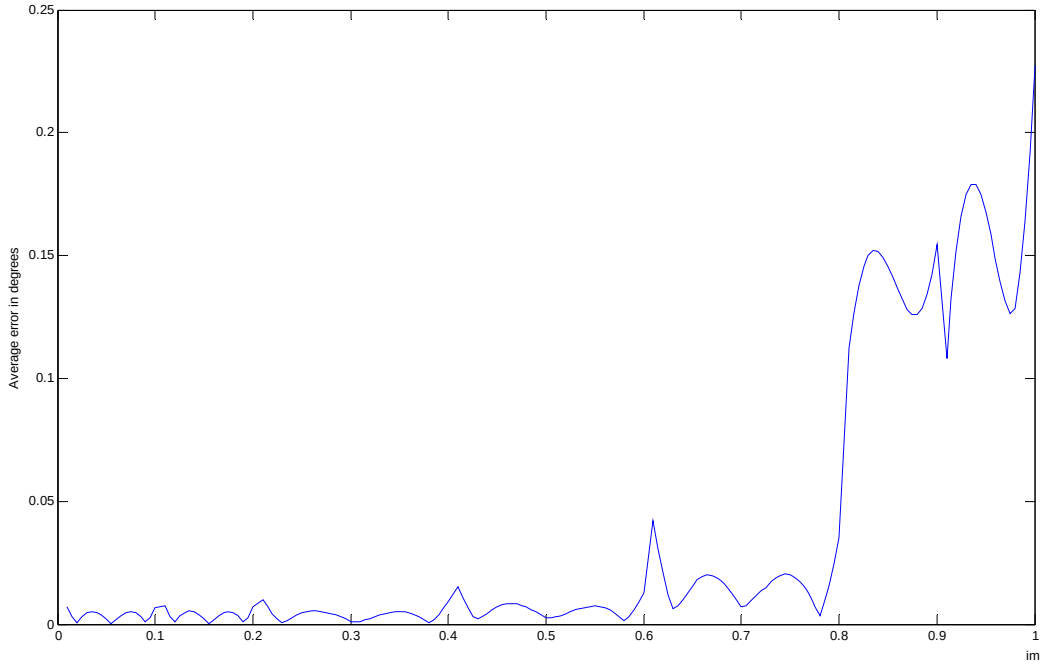


Figure 3.4 : la variation de l'erreur moyenne entre les angles de commutation PWM exactes et ceux de l'algorithme ANNSHE dans toute la gamme de variation de l'indice im

Tableau 3.5 : l'erreur moyenne et maximal entre les angles de commutation PWM exactes et ceux calculés par l'algorithme ANNSHE

ANN	Nombre d'angles de commutation m	Indice de modulation im	L'erreur moyenne	L'erreur maximale
ANN-1	23	0< im <0.1	0.0035	0.0087
ANN-2	19	0.1< im <0.2	0.0037	0.0091
ANN-3	15	0.2< im <0.4	0.0038	0.0137
ANN-4	7	0.4< im <0.6	0.0060	0.0231
ANN-5	5	0.6< im <0.8	0.0161	0.0625
ANN-6	3	0.8< im <1	0.1429	0.3050

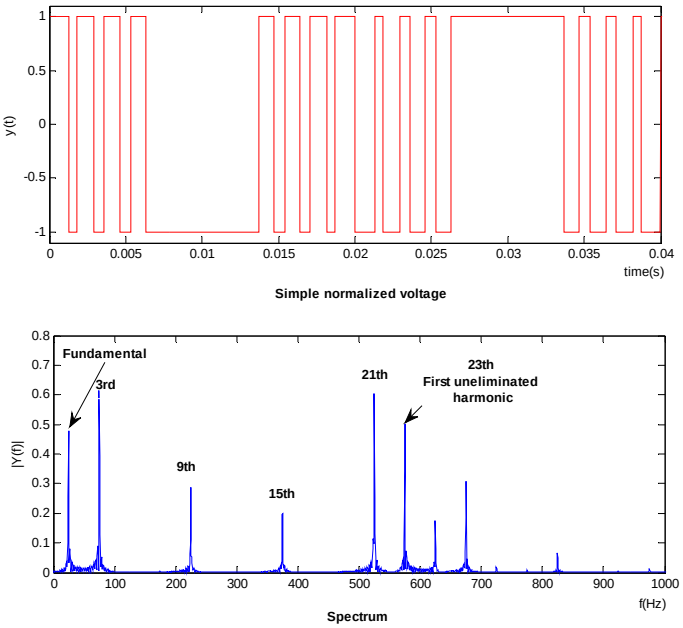
6- Interprétation

D'après les Tableaux 4.4 et 4.5 et les Figures 3.3 et 3.4, on voit bien que les angles de commutation calculés par l'algorithme ANNSHE PWM proposé sont très proches des angles exacts calculés par la méthode itérative. Le Tableau 3.4 montre que l'erreur maximale est inférieure à 0,02 pour im = 0,415 et im = 0,575. Ainsi, le Tableau 3.5 montre que l'erreur moyenne est inférieure à 0,0161 degrés et l'erreur maximale est inférieure à 0,0625 degrés pour les cinq premiers réseaux ANN-1 à ANN-5 ; pour le réseau ANN-6 on voit que l'erreur augmente un peu par rapport les autres réseaux à cause de l'augmentation de la non linéarité de la variation des angles de commutation. La Figure 3.3 montre que l'erreur entre les angles ANNSHE et les angles exacts pour ANN-4 est inférieur à 0.05 pour l'angle $\alpha 3$ et 0.02 pour l'angle $\alpha 7$. De plus la Figure 3.4 montre que l'erreur moyenne est inférieure à 0.05 quand im inférieur à 0.8 et elle est entre 0.1 et 0.25 lorsque im dépasse 0.8 ; cette augmentation est due à l'augmentation de la non linéarité dans la variation des angles de commutation.

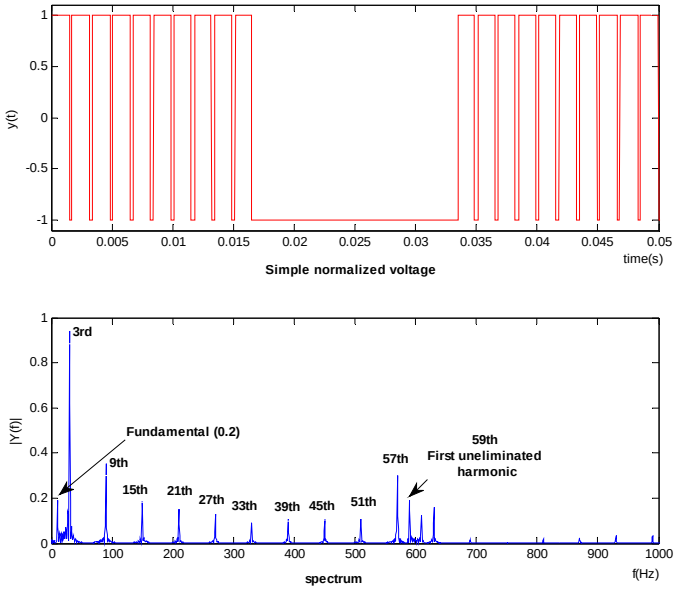
La Figure 3.5 montre que les tensions de phase sont périodique avec une fréquence de 25 Hz (T = 0,05 s) pour im = 0,5 et une fréquence de 10 Hz (T = 0,1 s) pour im = 0,2. Également, on voit clairement l'existence de sept et dix-neuf angles de commutation par quart de période pour m = 7 et m = 19, respectivement.

L'analyse spectrale de la Figure 3.5.a montre, comme prévu, l'élimination des harmoniques 5, 7, 11, 13, 17 et 19 pour m = 7. L'analyse spectrale de la Figure 3.5.b montre que la première harmonique non-éliminée pour m=19 est le 59ème. De plus, la Figure 3.5 montre que l'amplitude du fondamental est presque égale à im.

Par conséquent, l'algorithme ANNSHE PWM proposé a une très grande précision dans le calcul des angles de commutation, et également il est très efficace dans l'élimination des harmoniques sélectionnés et l'asservissement du fondamental.



(a) im = 0.5 et m =7 (f=25Hz)



(b) im = 0.2 et m =19 (f=10Hz)

Figure 3.5 : le signal ANNSHE PWM et son spectre correspondant obtenu par simulation pour (a) : im=0.5, m=7, f=25Hz et (b) : im=0.2, m=19, f=10Hz

7- Implémentation de l'algorithme ANNSHE PWM sur un circuit FPGA

7.1- Description

Dans cette section, pour mettre en œuvre pratiquement l'algorithme ANNSHE PWM proposé et vérifier ainsi sa précision et son efficacité, une implémentation sur un circuit FPGA est décrite.

La figure 3.6 montre l'organigramme de l'algorithme ANNSHE PWM en vue de sa conception sous VHDL où im est l'indice de modulation et clk est l'horloge d'entrée. Dans la conception on a utilisé clk = 50 MHz.

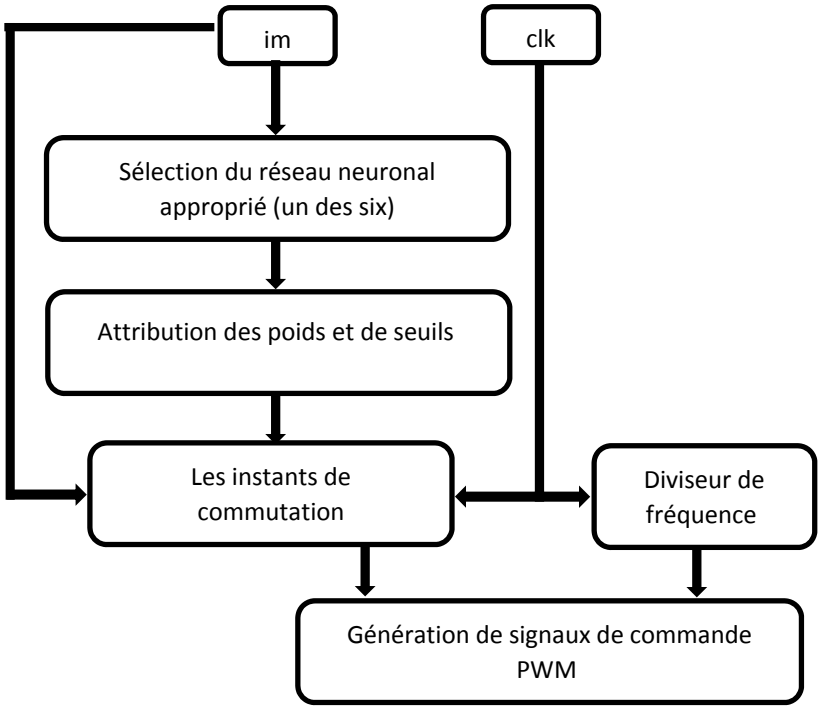
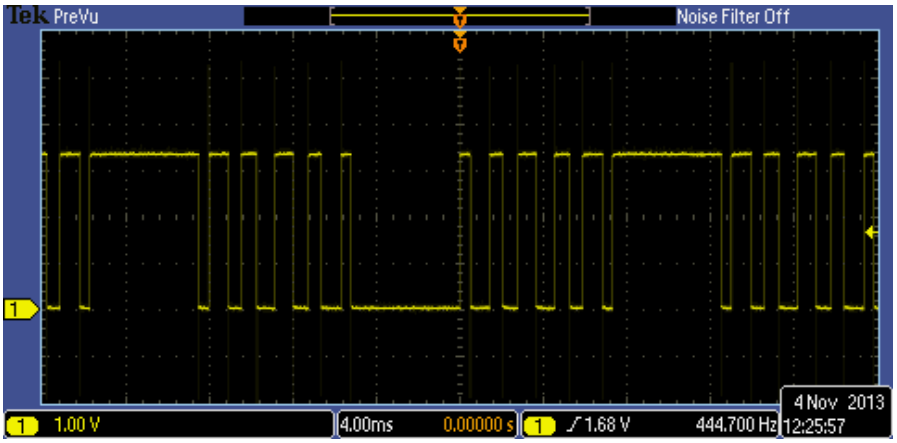


Figure 3.6 : Organigramme de l'algorithme ANNSHE PWM proposé

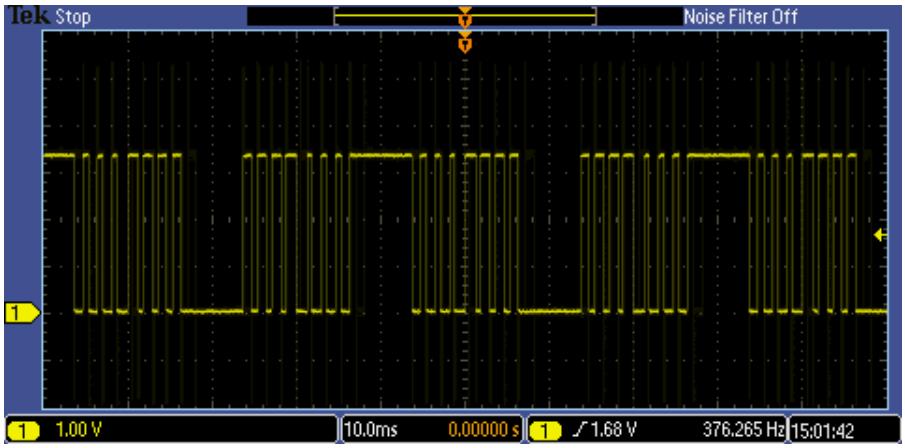
Le programme comporte cinq modules principaux :

- *La sélection du réseau neuronal approprié (un parmi six) :* ce module permet de choisir le réseau de neurones approprié. Selon la valeur de l'entrée im et en se basant sur le Tableau 3.1, un ANN-i est sélectionné.
- *L'attribution des poids et de seuils :* ce module permet d'attribuer pour le réseau ANN-i précédemment sélectionné ses paramètres appropriés (poids et seuils).
- *Le calcul des instants de commutation :* Ce module calcule les instants de commutation en utilisant l'entrée im, l'horloge clk et les paramètres du réseau ANN-i sélectionnés dans la partie précédente.
- *Le diviseur de fréquence :* Ce module est un diviseur de fréquence qui divise la fréquence d'entrée par 50 ; par conséquent, la fréquence de sortie est de 1 MHz. Cette horloge sera utilisée pour générer les signaux PWM.
- *Génération de signaux de commande PWM :* ce module génère les trois signaux de commande PWM avec un déphasage de 120° en utilisant l'horloge et les instant de commutation calculés dans les étapes précédentes.

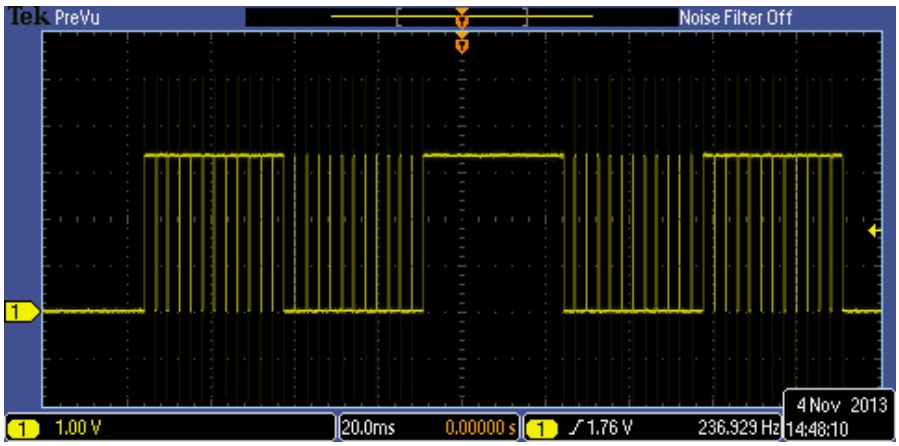
Pour vérifier le fonctionnement de la commande ANNSHE PWM proposé, la conception décrite ci-dessus a été implémentée sur la carte de développement FPGA Spartan-3E. Les Figures 3.7 et 3.8 montrent la visualisation des signaux ANNSHE PWM obtenus sur l'oscilloscope Tektronix MSO 2024B pour im = 0,8, 0,5 et 0,1, tandis que les Figures 3.9 et 3.10 représentent l'analyse des signaux enregistrés pour im = 0,5 et m = 7.



(a) m=5 et im=0.8 (f=40Hz)



(b) m=7 et im=0.5 (f=25Hz)



(c) m=23 et im=0.1 (f=5Hz)

Figure 3.7 : Visualisation des signaux ANNSHE PWM sur l'oscilloscope

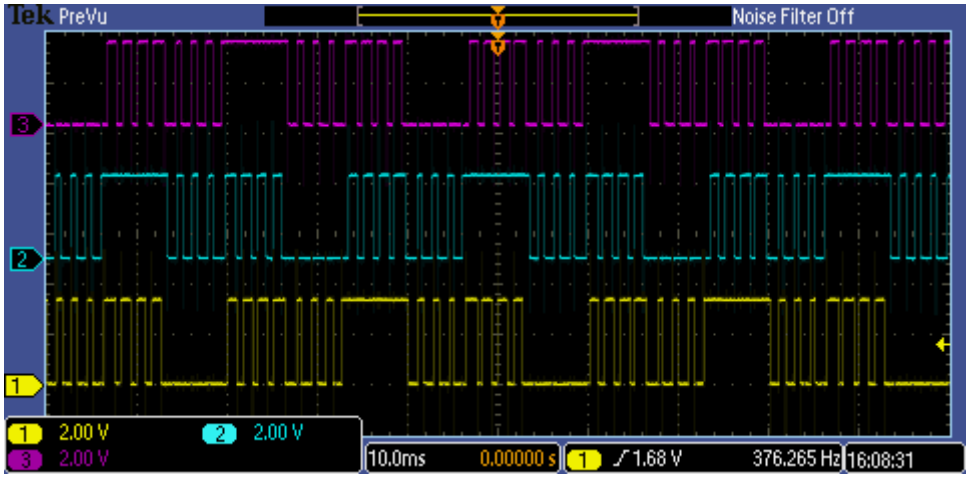


Figure 3.8: Visualisation des trois signaux ANNSHE PWM déphasé de 120° pour m = 7 et im = 0,5 sur l'oscilloscope

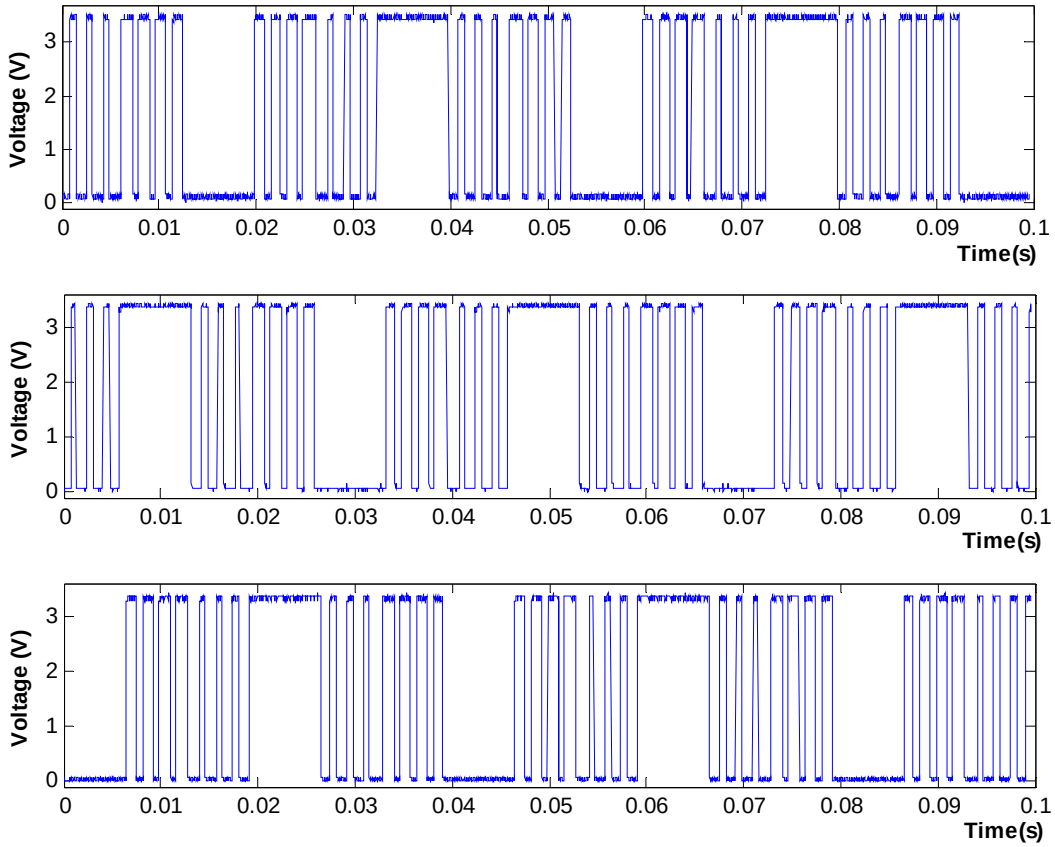


Figure 3.9 : Les trois signaux ANNSHE PWM enregistrés pour m=7 et im=0.5

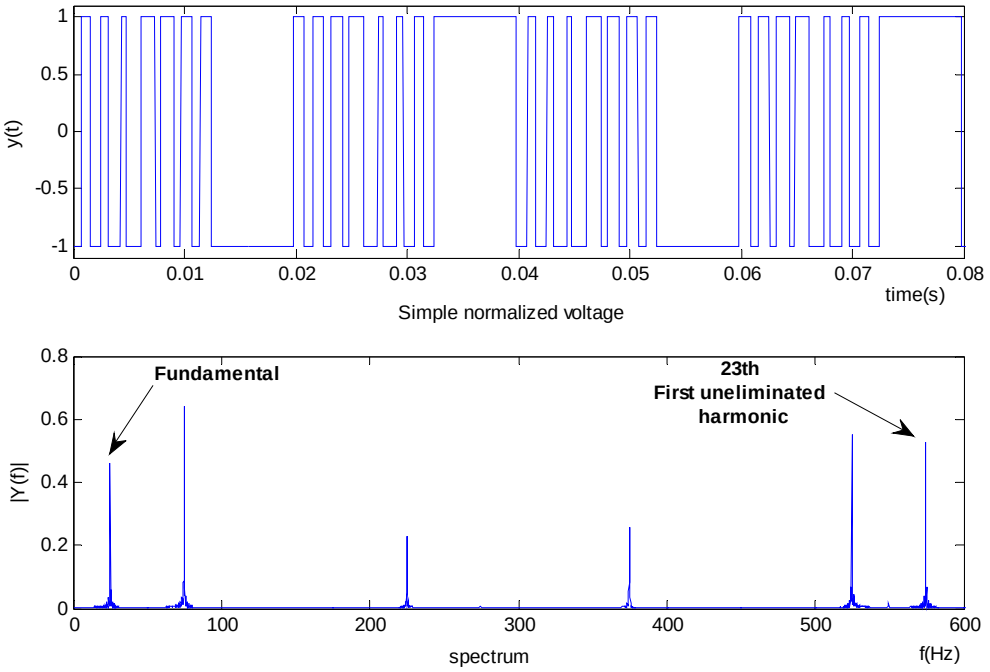


Figure 3.10 : La tension de phase normalisée et son spectre fréquentiel pour m=7 et im=0.5

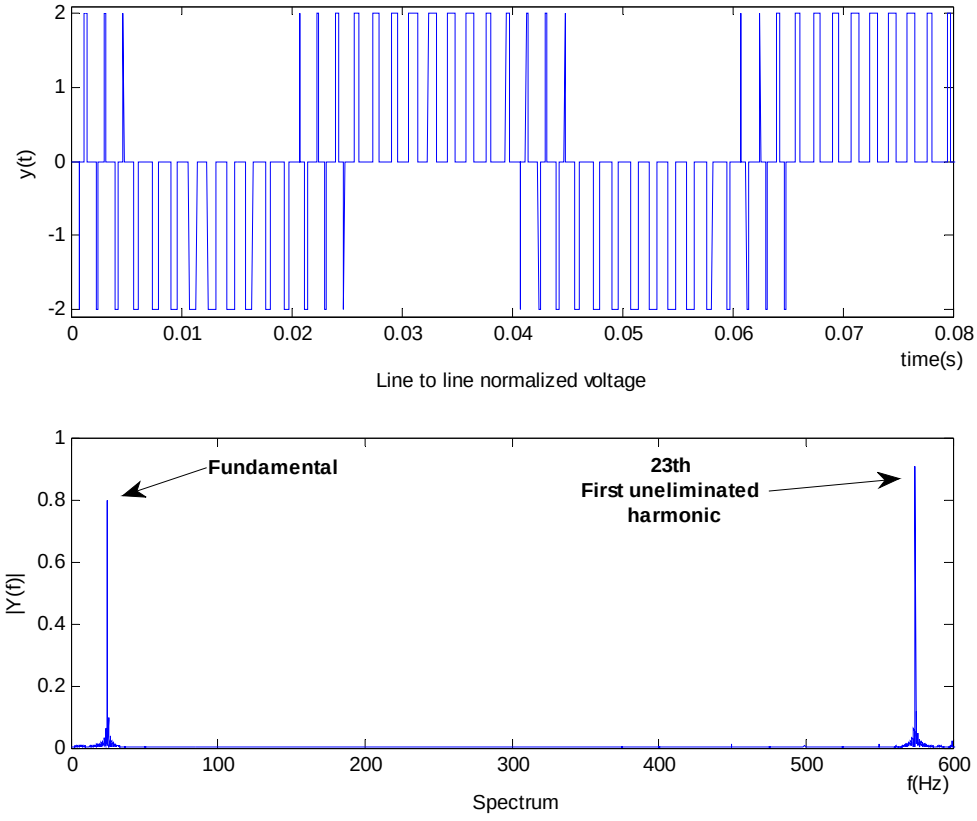


Figure 3.11 : La tension entre phase normalisée et son spectre fréquentiel pour m=7 et im=0.5

7.2- Interprétation

La Figure 3.7 montre que les tensions de phase obtenues sont périodique et de fréquence 40 Hz pour im = 0,8, 25 Hz pour im = 0,5 et 5 Hz pour im = 0,1. Ainsi, on voit clairement l'existence de cinq, sept et vingt-trois angles de commutation par quart de période pour m = 5, 7 et m = 23 respectivement. De plus, les Figures 3.8 et 3.9 montrent que les trois signaux de commande ANNSHE PWM sont déphasés de 120°.

L'analyse spectrale dans les Figures 3.10 et 3.11 montre que la première harmonique non éliminée pour m = 7 est le 23ème. Ainsi, d'après la Figure 3.11, il ttest clair que pour la tension entre phases les harmoniques multiples de trois sont éliminées. De plus, de la figure 3.10 on voit bien que l'amplitude de la tension fondamentale est presque égale à im.

Donc l'implémentation sur le circuit FPGA confirme que l'algorithme ANNSHE PWM proposé reste toujours très précis et efficace dans le contrôle de la tension fondamentale et dans l'élimination des harmoniques sélectionnés.

8- conclusion

Dans ce chapitre, on a présenté les différentes étapes suivies pour développer et tester l'algorithme ANNSHE PWM basé sur la théorie des réseaux de neurones artificiels et la commande SHE PWM. Les résultats de simulation et de l'implémentation montrent que les angles de commutations calculés par l'algorithme ANNSHE PWM sont très proches de ceux calculés par l'algorithme itératif de Newton-Raphson. De plus, ils montrent l'élimination effective des harmoniques sélectionnés et l'efficacité du contrôle de la tension fondamentale à la sortie de l'onduleur.

Par la suite, on explique la façon dont on a procédé pour optimiser l'implémentation de l'algorithme proposé sur l'FPGA ainsi les détails de chaque partie du code VHDL seront présentés. Avant de procéder l'étape de l'optimisation on introduit un chapitre pour expliquer les étapes à suivre pour développer un projet sur un circuit FPGA. Aussi, l'architecture des circuits FPGA et le langage de description VHDL seront présentés.

Chapitre IV : Les circuits FPGA architecture et méthode de développement

1- Introduction

L'électronique moderne se tourne de plus en plus vers le numérique qui présente de nombreux avantages sur l'analogique : grande insensibilité aux parasites, reconfigurabilité, facilité de stockage de l'information etc. De nombreuses familles de circuits numériques programmables et reprogrammables sont apparues depuis les années 70 [31]. La figure 4.1 donne une classification possible de ces circuits.

D'après cette figure on distingue trois types principaux. D'abord, les circuits logiques standards qui comportent les circuits combinatoires et les bascules. Ensuite, les circuits à fonctionnement programmable qui comportent les microprocesseurs, les microcontrôleurs et les DSP (Digital Signal Processor). Dans cette catégorie, on trouve une unité arithmétique et logique qui exécute un programme situé dans une mémoire de programme en utilisant une horloge de synchronisation. Finalement, les circuits à architecture programmable qui comportent les ASIC (Application Specific Integrated Circuit), les PLD (Programmable Logic Device) et les FPGA (Field Programmable Gate Arrays) constituent le troisième type [48-51]. Dans cette catégorie, on fait une conception d'un circuit correspondant à nos propres besoins, l'utilisation d'une horloge n'est pas obligatoire, de plus on peut implémenter facilement les applications qui exigent le parallélisme.

Par définition, les circuits ASIC regroupent tous les circuits dont la fonction peut être personnalisée d'une manière ou d'une autre en vue d'une application spécifique, par opposition aux circuits standards dont la fonction est définie et parfaitement décrite dans le catalogue des composants. L'utilisation d'un ASIC conduit à de nombreux avantages provenant essentiellement de la réduction de la taille des systèmes comme la réduction du nombre des composants sur le circuit imprimé, de la consommation et de l'encombrement. Le concept ASIC assure une optimisation maximale du circuit à réaliser. Enfin, on a un circuit intégré

correspondant réellement à nos propres besoins qui donne une confidentialité au concepteur et une protection industrielle. L'inconvénient majeur des ASIC réside dans le fait du passage obligatoire chez le fondeur ce qui implique des frais et un temps de développement élevés du circuit. En conséquence, les ASIC sont généralement plus convenables pour les productions en série de conceptions déjà vérifiées et non pour les prototypes.

Les PLD sont des chips qui peuvent être programmés pour se comporter comme une conception arbitraire. Un PLD peut être programmé pour une implémentation aussi simple qu'une opération en logique combinatoire comme il pourrait l'être pour des conceptions beaucoup plus importantes. Il Comprend en générale une matrice de portes AND connectée à une matrice de portes OR [51].

Les circuits FPGAs sont des composants entièrement reconfigurables ce qui permet de les reprogrammer à volonté afin d'accélérer notablement certaines phases de calculs. Ils sont constitués d'une matrice de blocs logiques programmables entourés de blocs d'entrée/sortie programmables. L'ensemble est relié par un réseau d'interconnexions programmable. Les FPGAs ne sont pas optimisés pour une application bien déterminée, par conséquent ils consomment plus d'énergie que les ASICs. Par contre ils sont beaucoup plus simples à programmer et à reprogrammer, ce qui raccourcit les cycles de conception et permet de suivre l'évolution de l'application pour laquelle, ils ont été conçus. L'avantage de ce genre de circuit est sa grande souplesse qui permet de les réutiliser à volonté dans des algorithmes différents en un temps très court. Les FPGAs sont plus convenables pour les prototypes et pour les productions en série limitées qui ne sont pas de la qualité des ASICs. Cette technologie permet aussi d'implanter un grand nombre d'applications et offre une solution d'implantation matérielle à faible coût pour des compagnies de taille modeste pour qui, le coût de développement d'un circuit intégré spécifique implique un trop lourd investissement. L'inconvénient majeur des circuits FPGA est qu'ils ne sont pas très sécurisés sur le plan de la confidentialité, puisqu'il suffit d'analyser le contenu de la ROM associée pour remonter à la schématique imaginée [48-53].

Dans l'algorithme proposé on a besoin du parallélisme, pour l'implémentation des réseaux de neurones et la génération des signaux PWM, on a donc choisi un circuit FPGA Spartan-3E de Xilinx [54] pour implémenter et générer les signaux PWM.

Dans ce chapitre, on introduit principalement l'architecture des circuits FPGA et on présente la manière de développement d'un projet sur ces circuits. Nous commençons d'abord par un bref historique et une présentation de Xilinx qui est le premier fabricant des circuits FPGA. Nous détaillons par la suite l'architecture d'un circuit FPGA et on montre une présentation de la carte

de développement Spartan 3E de Digilent qu'on a utilisé pour développer notre projet. Ensuite, nous présentons le langage VHDL qu'on a utilisé pour la conception de notre application. Enfin, nous citons les étapes nécessaires qu'on doit suivre pour développer un projet sur un circuit FPGA.

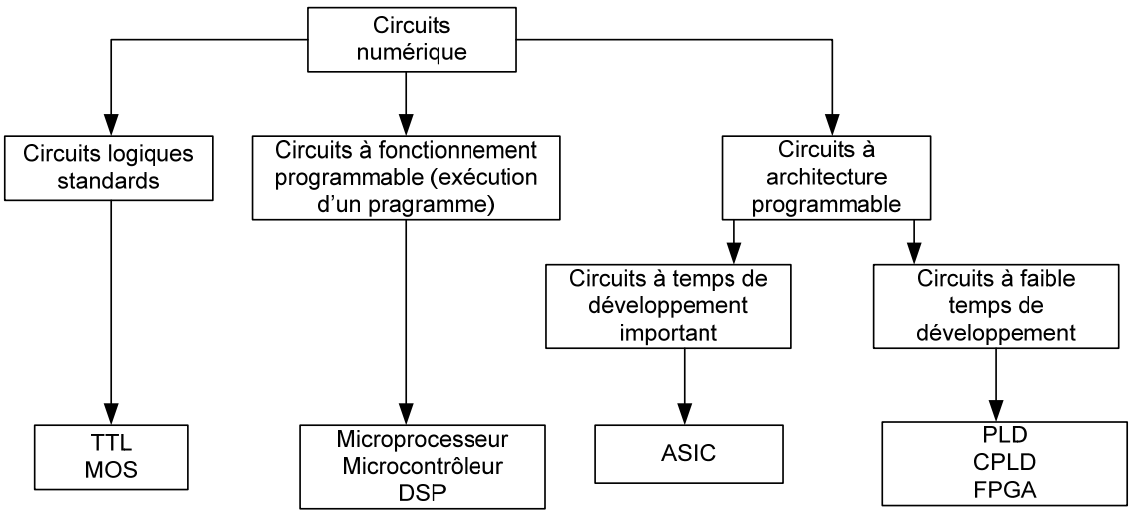


Figure 4.1 : Classification des circuits numériques

2- Les Circuits FPGA

2.1- Historique

Le principe de la logique programmable remonte au début des années 1960, le concept ayant été proposé par Estrin. Mais ce n'est qu'au cours des années 1980 que les premières réalisations matérielles sont introduites sur le marché. L'apparition de ce type de circuit s'est d'abord faite au travers de circuits logiques programmables simples de type PAL (Programmable Array Logic), qui se programment comme des mémoires de type ROM et sont utilisés pour implémenter des fonctions combinatoires simples, telles des décodeurs d'adresse ou des contrôleurs de bus. Avec les évolutions en microélectronique, différentes familles de circuits programmables ont vu le jour : les CPLD (Complex Logic Programmable Device), puis les FPGA (Field Programmable Gate Arrays), introduits par la société Xilinx en 1985 [48].

2.1.1- Technologie du premier fondeur de circuits FPGA « XILINX »

Le plus gros constructeur du marché est XILINX qui a introduit la série XC2000 (de 600 à 1500 portes) entre 1984 et 1985. D'autres séries sont apparues par la suite comme les XC3000, XC4000, XC5200, XC6200 et XC9500 avec la mise sur le marché des premiers FPGA XILINX en 1985. L'apparition du premier FPGA du constructeur ALTERA, qui est le concurrent le plus important de XILINX, s'est fait en 1992 avec un type de circuits assez différent, le FLEX 8000 (15 000 portes max). L'exploitation de la technologie EEPROM un an plus tard en 1993 pour

le chargement du code des circuits FPGA. Le lancement du VIRTEX II de XILINX (jusqu'à 10 millions de portes) s'est fait en 2001. Des FPGA de capacités supérieures à 50 millions de portes fonctionnant à des fréquences dépassant les 500 MHz ont vu le jour en 2005.

Le principe des FPGA de XILINX est de stocker la configuration dans une mémoire vive statique SRAM. Aujourd'hui des blocs de fonctionnalités supplémentaires dans quelques versions évoluées sont ajoutés et dédiés à des applications spécifiques. Parmi ces fonctionnalités supplémentaires on trouve : Mémoire RAM, Multiplieurs, Blocs DSP, Cœurs de processeurs RISC.

2.2- Domaines d'application des circuits FPGA

Actuellement, les FPGA sont utilisés dans de nombreuses applications civiles et militaires, on en cite dans ce qui suit quelques-unes [31, 50] :

- Informatique : périphériques spécialisés.
- Machinerie industrielle : contrôleur pour machines, systèmes de commande en temps réel.
- Télécommunications : traitement d'images, filtrage.
- Instrumentation : équipement médical, prototypage, fabrication de composants spéciaux en petite série.
- Transport : contrôle d'avions et métros.
- Aérospatiale : satellites.
- Militaire : radar, communication protégée, la détection ou la surveillance.

2.3- Architecture des circuits FPGA

Pour réussir à implémenter un système dans un circuit FPGA de manière efficace, il est indispensable de bien connaître sa structure interne et ses limites du point de vue performances. 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 la ou les fonctions numériques désirées.

Les circuits FPGA sont constitués d'une matrice de blocs logiques programmables entourés de blocs d'entrées sorties programmables. L'ensemble est relié par un réseau d'interconnexions programmables (figure 4.2). Certains circuits FPGA intègrent également des mémoires RAM, des multiplieurs et même des noyaux de processeurs [50,53].

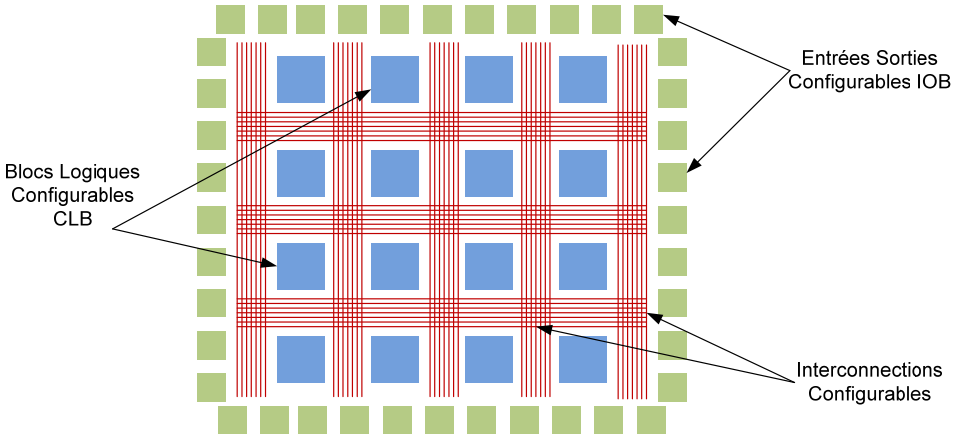


Figure 4.2 Architecture interne d'un circuit FPGA

L'architecture, retenue par Xilinx, se présente sous forme de deux couches :

- Une couche appelée circuit configurable.
- Une couche réseau mémoire SRAM

Dans les sections suivantes, on détaille l'architecture du circuit FPGA en se basant sur l'article publié par Stephen Trimberger [53].

2.3.1- Circuit configurable

La couche dite "circuit configurable" est constituée d'une matrice de blocs logiques configurables (CLB) permettant de réaliser des fonctions combinatoires et des fonctions séquentielles. Tout autour de ces blocs logiques configurables, nous trouvons des blocs d'entrées/sorties (IOB) dont le rôle est de gérer les entrées-sorties réalisant l'interface avec les modules extérieurs. La programmation du circuit FPGA consistera en l'application d'un potentiel adéquat sur la grille de certains transistors à effet de champ servant à interconnecter les éléments des CLB et des IOB, afin de réaliser les fonctions souhaitées et d'assurer la propagation des signaux. Ces potentiels sont tout simplement mémorisés dans le réseau mémoires SRAM.

a- Les CLB (Configurable Logic Bloc)

Les Blocs Logiques Configurables sont les éléments déterminants des performances du circuit FPGA. Chaque bloc est composé d'un bloc de logique combinatoire composé de deux générateurs de fonctions à quatre entrées et d'un bloc de mémorisation synchronisation composé de deux bascules D. Quatre autres entrées permettent d'effectuer les connexions internes entre les différents éléments du CLB. La LUT (Look Up Table) est un élément qui dispose de quatre entrées, il existe donc 2^4 = 16 combinaisons différentes de ces entrées. L'idée consiste à mémoriser la sortie correspondant à chaque combinaison d'entrée dans une petite table de 16

bits, la LUT devient ainsi un petit bloc générateur de fonctions. La figure 4.3 ci-dessous montre le schéma simplifié d'un CLB de la famille XC4000 de Xilinx.

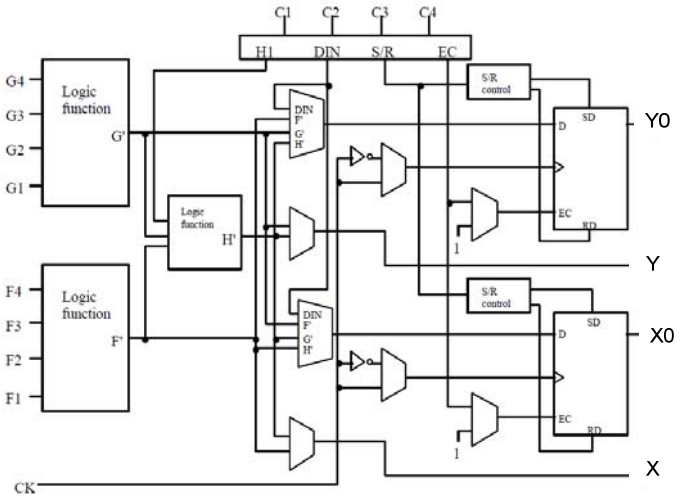


Figure 4.3 Cellules logiques (CLB) pour XC4000 de Xilinx [53]

Dans cette famille, la cellule de base contient deux LUT à 4 entrées qui peuvent réaliser deux fonctions quelconques à 4 entrées. Une troisième LUT peut réaliser une fonction quelconque à 3 entrées à partir des sorties des deux premières LUT (F' et G' qui deviennent H2 et H3) et d'une troisième variable d'entrée H1 sortant du bloc « sélecteur ». Le bloc sélecteur contient 4 signaux de contrôle : 3 signaux dédiées pour les registres : une donnée "Din", un signal de validation "Ec" et une remise à un ou à zéro asynchrone "S/R", et le 4ème signal représente l'entrée H1 de la LUT à 3 entrées.

Le plus large circuit de cette famille, le circuit XC40250, dispose d'un réseau de 92x92 cellules de base qui est équivalent à environ 250.000 portes logiques. Xilinx propose également des composants "haute densité" avec les familles Virtex (4 millions de portes) et Virtex II (6 millions de portes).

b- Les IOB (input output bloc)

Les blocs entrée/sortie IOB permettent l'interface entre les broches du composant FPGA et la logique interne développée à l'intérieur du composant. Ils sont présents sur toute la périphérie du circuit FPGA. Chaque bloc IOB contrôle une broche du composant et il peut être défini en entrée, en sortie, en signaux bidirectionnels ou être inutilisé (haute impédance). La figure 4.4 présente la structure de ces blocs.

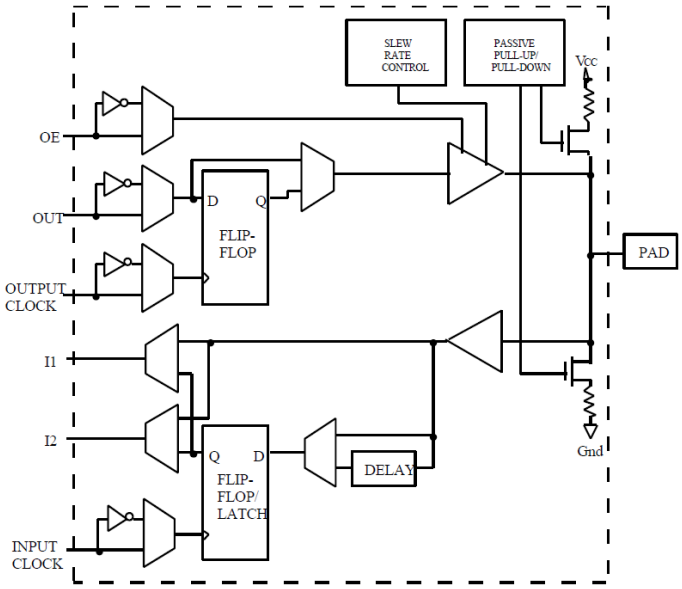


Figure 4.4: Les blocs entrée/sortie IOB pour XC4000 de Xilinx [53]

2.3.2- Réseau mémoires SRAM (Static Random Access Memory)

La programmation d'un circuit FPGA est volatile, la configuration du circuit est mémorisée sur la couche réseau SRAM et stockée dans une ROM externe. Un dispositif interne permet à chaque mise sous tension de charger la SRAM interne à partir de la ROM. Ainsi on conçoit aisément qu'un même circuit puisse être exploité successivement avec des ROM différentes puisque sa programmation interne n'est jamais définitive. Chaque cellule SRAM, figure 4.5, permet de stocker un unique bit de configuration, un FPGA comporte donc jusqu'à plusieurs millions de cellules mémoires de configuration. C'est ce qui les distingue des autres familles de circuits intégrés.

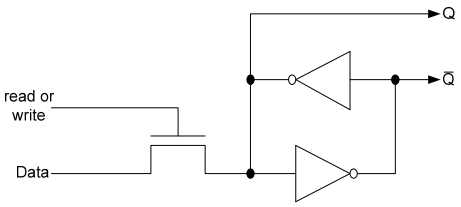


Figure 4.5 Structure d'une cellule SRAM

2.3.3- Les interconnexions

Les connexions internes dans les circuits FPGA sont composées de segments métallisés. Parallèlement à ces lignes, nous trouvons des matrices programmables réparties sur la totalité du circuit, horizontalement et verticalement entre les divers CLB. Elles permettent les connexions entre les diverses lignes, celles-ci sont assurées par des transistors MOS dont l'état est contrôlé par des cellules de mémoire vive ou RAM. Le rôle de ces interconnexions est de relier avec un maximum d'efficacité les blocs logiques et les entrées/sorties afin que le taux

d'utilisation dans un circuit donné soit le plus élevé possible. Pour parvenir à cet objectif, Xilinx propose trois sortes d'interconnexions selon la longueur et la destination des liaisons, figure 4.6.

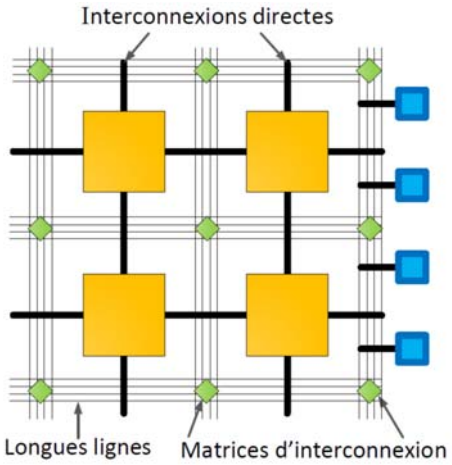


Figure 4.6: Interconnexion interne d'un FPGA.

a- Les interconnexions à usage général : Ce système fonctionne en une grille de cinq segments métalliques verticaux et quatre segments horizontaux positionnés entre les rangées et les colonnes de CLB et de l'IOB. Des aiguilleurs appelés aussi matrices de commutation sont situés à chaque intersection. Leur rôle est de raccorder les segments entre eux selon diverses configurations, ils assurent ainsi la communication des signaux d'une voie sur l'autre. Ces interconnexions sont utilisées pour relier un CLB à n'importe quel autre. Pour éviter que les signaux traversant les grandes lignes ne soient affaiblis, nous trouvons généralement des buffers implantés en haut et à droite de chaque matrice de commutation.

b- Les interconnexions directes : Ces interconnexions permettent l'établissement de liaisons entre les CLB et les IOB avec un maximum d'efficacité en terme de vitesse et d'occupation du circuit. De plus, il est possible de connecter directement certaines entrées d'un CLB aux sorties d'un autre.

c- Les longues lignes : Les longues lignes sont de longs segments métallisés parcourant toute la longueur et la largeur du composant, elles permettent éventuellement de transmettre avec un minimum de retard les signaux entre les différents éléments dans le but d'assurer un synchronisme aussi parfait que possible. De plus, ces longues lignes permettent d'éviter la multiplicité des points d'interconnexion.

3- La carte de développement Spartan 3E

La carte de développement Spartan 3E de Digilent, qu'on a utilisé pour implémenter l'algorithme proposé, fournit une solution complète de développement d'applications sur la famille Spartan 3E de Xilinx. Elle utilise le circuit « FPGA XC3S500E-4FG320C » qui appartient à la famille Spartan 3E de Xilinx [55]. La haute densité d'intégration des portes ainsi

que le nombre important d'entrées/sorties disponibles à l'utilisateur permettent d'implémenter des systèmes complets de solutions sur la plateforme FPGA. Cette carte offre un environnement de conception très adapté pour le prototypage d'applications varié dont celles des systèmes numériques à usage général et des systèmes embarqués. Ainsi, elle fournit une solution complète de développement d'applications sur la famille Spartan 3E et elle est de plus idéale pour les applications de contrôle, de traitement vidéo et de traitement de signal en général.

La carte Spartan 3E regroupe entre autre un FPGA XC3S500E-4FG320C Spartan 3E, un CPLD Coolrunner II, une mémoire PROM de 4 Mbit, une mémoire Flash sérielle de 16 Mbit, une mémoire Flash parallèle de 128 Mbit et une mémoire SDRAM DDR de 512 Mbit. De plus on trouve sur cette carte un écran LCD, un port PS/2, un port VGA, une prise Ethernet 10/100, deux ports RS-232 et un port de configuration USB. Ainsi, cette carte possède deux convertisseurs de données AN et NA, un oscillateur à 50 MHz, un connecteur d'expansion Hirose FX2 permettant 40 entrées/sorties génériques, des DEL, des boutons poussoirs et des commutateurs.

La carte supporte trois modes de configuration à la mise sous tension, soit le mode Master-Slave utilisant le PROM 4 Mbit, un mode SPI utilisant la mémoire Flash sériel et un mode BPI utilisant la mémoire Flash parallèle. La carte peut être programmée aussi directement à l'aide d'un port JTAG lorsqu'elle est sous tension. La Figure 4.7 montre une vue d'ensemble de la carte Spartan 3E et ses principaux modules.

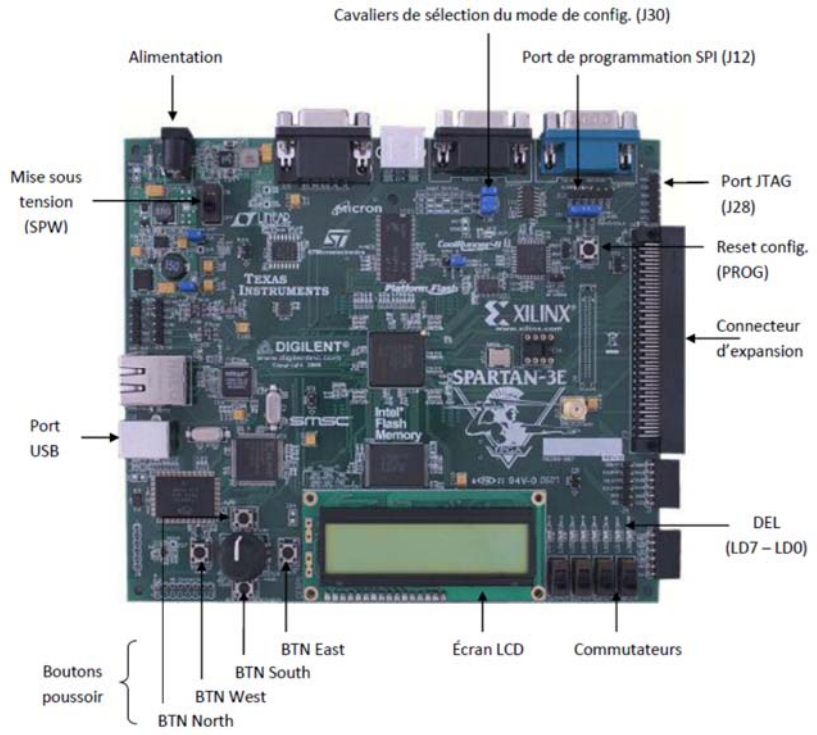


Figure 4.7 : La carte Spartan 3E et ses principaux modules embarqués [55].

3.1- Architecture interne d'un circuit Spartan 3E

La figure 4.8 montre l'architecture interne du circuit Spartan 3E publié par Xilinx [54]. D'après cette figure, l'architecture de la famille Spartan-3E se compose de cinq éléments configurables fondamentaux :

1. Blocs logiques configurables CLB
2. Blocs entrée/sortie IOB
3. Bloc mémoire RAM
4. Bloc des Multiplicateurs 18x18
5. Bloc Digital Clock Manager (DCM)

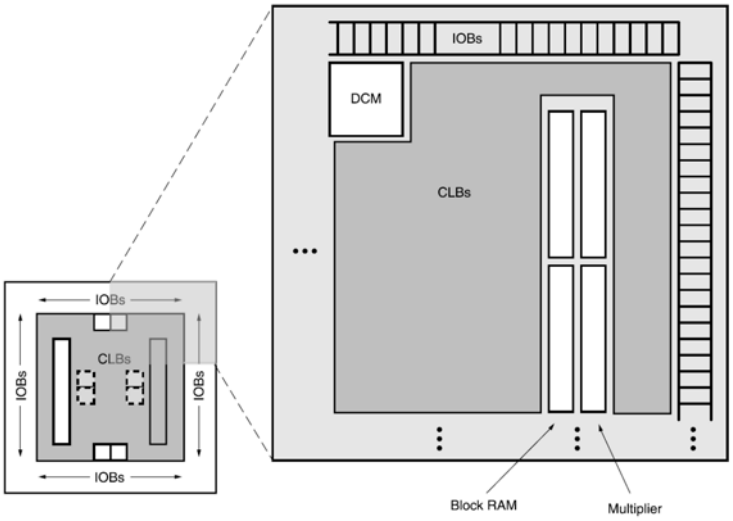


Figure 4.8 : L'architecture interne du circuit Spartan 3E [54]

4- Langage de description VHDL

4.1 Bref sur le VHDL

Le VHDL est un langage de description de matériel. Il décrit le comportement d'un circuit électronique, à partir de laquelle le circuit physique peut ensuite être atteint (implémenté).

Le VHDL signifie VHSIC Hardware Description Language. VHSIC est elle-même une abréviation de Very High Speed Integrated Circuits. Une initiative financée par le Département Américain de la Défense dans les années 1980 qui a conduit à la création de VHDL. Son objectif était de décrire les circuits complexes, de manière à établir un langage commun avec ses fournisseurs.

Sa première version était le VHDL 87, plus tard une mise à niveau vers le soi-disant VHDL 93. Le VHDL a été le premier langage de description de matériel normalisé par l'Institute of Electrical and Electronics Engineers IEEE, grâce à la norme IEEE 1076. Une norme supplémentaire, l'IEEE 1164, a été ensuite ajoutée pour introduire les systèmes logiques multi-valeurs [69].

4.2 Utilité du VHDL :

Le VHDL est un langage de spécification, de simulation et également de conception [70]. Contrairement à d'autres langages (CUPL, ABEL) qui se trouvaient être en premier lieu des langages de conception, le VHDL est d'abord un langage de spécification. La normalisation a d'abord eu lieu pour la spécification et la simulation (1987) et ensuite pour la synthèse (1993).

a- Spécification : c'est dans ce domaine que la norme est actuellement la mieux établie. Il est tout-à-fait possible de décrire un circuit par un code VHDL standard pour qu'il soit lisible partout. Cette possibilité de décrire des circuits dans un langage universel est aussi très pratique pour éviter les problèmes de langue.

b- Simulation : Le VHDL est également un langage de simulation. Pour ce faire, la notion de temps, sous différentes formes, y a été introduite. Des modules, destinés uniquement à la simulation, peuvent ainsi être créés et utilisés pour valider un fonctionnement logique ou temporel du code VHDL. La possibilité de simuler avec des programmes VHDL devrait considérablement faciliter l'écriture de tests avant la programmation du circuit et éviter ainsi de nombreux essais sur un prototype qui sont beaucoup plus coûteux et dont les erreurs sont plus difficiles à trouver.

c- Conception : Le VHDL permet aussi la conception des circuits. Les deux principales applications immédiates de VHDL sont dans le domaine des circuits Logiques programmables, y compris CPLD et FPGA, et dans le domaine des circuits ASIC.

Le VHDL est destiné à la synthèse de circuit ainsi que la simulation de circuit. Cependant, ce langage étant conçu en premier lieu pour de la spécification et ensuite pour la simulation, de ce fait certaines variantes du langage ne sont pour l'instant pas utilisables pour la conception.

4.3- Structure d'une description VHDL simple

Comme illustré sur la figure 4.9, La structure typique d'une description VHDL est composée au moins de trois parties fondamentales :

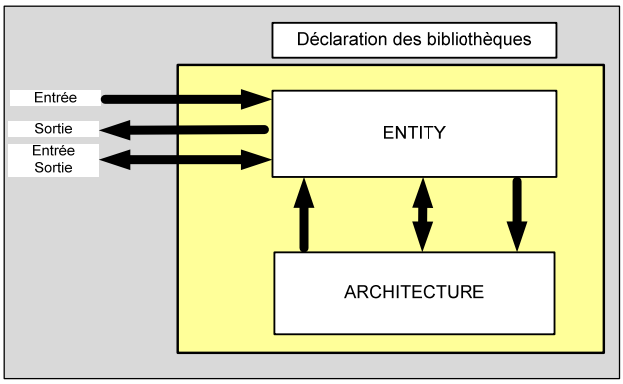


Figure 4.9 : Structure de base d'une description VHDL

a- Déclaration des bibliothèques Toute description utilisé dans le code VHDL doit être définit dans une bibliothèque. Les bibliothèques principales sont normalisées par IEEE. Elles contiennent les définitions des types de signaux électroniques, des fonctions et sous programmes permettant de réaliser des opérations arithmétiques et logiques, etc. La directive "**use**" permet de sélectionner les bibliothèques à utiliser.

Exemple:

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.numeric_std.all;
Use ieee.std_logic_unsigned.all;
```

b- L'entité : elle représente une vue externe de la description. La déclaration de l'entité permet de définir le nom de la description VHDL ainsi que les entrées et les sorties utilisées, l'instruction qui les définit est "**port**".

Exemple :

```
entity ana is
port (DEC :in std_logic_vector(3 downto 0);
SEG:out std_logic_vector(6 downto 0));
end ana ;
```

c- L'architecture : contient le code VHDL appropriée, qui décrit comment le circuit doit se comporter pour réaliser le fonctionnement attendu. Elle représente la structure interne de la description. L'exemple suivant montre l'architecture d'un diviseur de fréquence.

Exemple :

```
architecture Behavioral of div_fr is
begin
 PROCESS (clk_in)
 variable count1 : INTEGER range 0 to 25 :=0;
 variable clk1 : std_logic :='0';
 BEGIN
 IF (clk_in'EVENT AND clk_in='1') THEN
 count1 := count1 + 1;
 IF (count1 =25) THEN
 clk1 := not clk1;
 count1 := 0;
 END IF;
 clk_out <= clk1;
 END IF;
 END PROCESS;
end Behavioral;
```

4.4- Les deux modes de travail en VHDL

Le VHDL utilise deux modes de fonctionnement : le mode combinatoire (ou concurrent) et le mode séquentiel. Chacun de ces modes est utilisé dans des cas bien précis.

a- Le mode combinatoire : En mode combinatoire, toutes les instructions d'une description VHDL sont évaluées et affectent les signaux de sortie en même temps (en parallèle), l'ordre dans lequel les instructions sont écrites n'a donc aucune importance. En effet la description génère des structures électroniques, c'est la grande différence entre une description VHDL et un langage informatique classique. Ainsi avec VHDL il faut essayer de penser à la structure qui va être générée par le synthétiseur pour écrire une bonne description.

b- Le mode séquentiel : Le mode séquentiel utilise les "**process**" dans lesquels le temps est une variable essentielle. Un "**process**" est une partie de la description d'un circuit dans laquelle les instructions sont exécutées séquentiellement, c'est-à-dire les unes à la suite des autres. Il permet d'effectuer des opérations sur les signaux en utilisant les instructions standards de la programmation structurée comme dans les systèmes à microprocesseurs.

5- Développement d'un projet sur FPGA

Dans cette section on présente les étapes à suivre pour implémenter une conception sur un circuit FPGA de Xilinx en utilisant le langage de description VHDL. D'abord, le développement en VHDL nécessite l'utilisation de deux outils : le simulateur et le synthétiseur. Le premier va nous permettre de simuler notre description VHDL. L'objectif du synthétiseur est très différent : il doit traduire le comportement décrit en VHDL en fonctions logiques de bases, celles-ci dépendent de la technologie choisie ; cette étape est nommée « synthèse ». L'intégration finale dans le circuit cible est réalisée par l'outil de placement et routage. La figure 4.10 donne les différentes étapes nécessaires au développement d'un projet sur un circuit FPGA.

5.1- L'outil de conception ISE de Xilinx

La firme Xilinx a mis au point des logiciels performants de développement de circuits FPGA. Les séries de logiciels de Xilinx ont été conçues pour aider efficacement à l'étude des projets sur FPGA. Le résultat final de tels projets est un fichier de train binaire qui peut être téléchargé dans un dispositif FPGA. Le logiciel de développement de projets sur circuits FPGA mis sur le marché est l'ISE (Integrated Software Environement), figure 4.11. L'ISE contrôle tous les aspects de déroulement de la conception. A l'aide de l'interface de gestion de projets, nous pouvons accéder aux divers outils d'exécution de la conception, et également aux dossiers et aux documents liés au développement du projet.

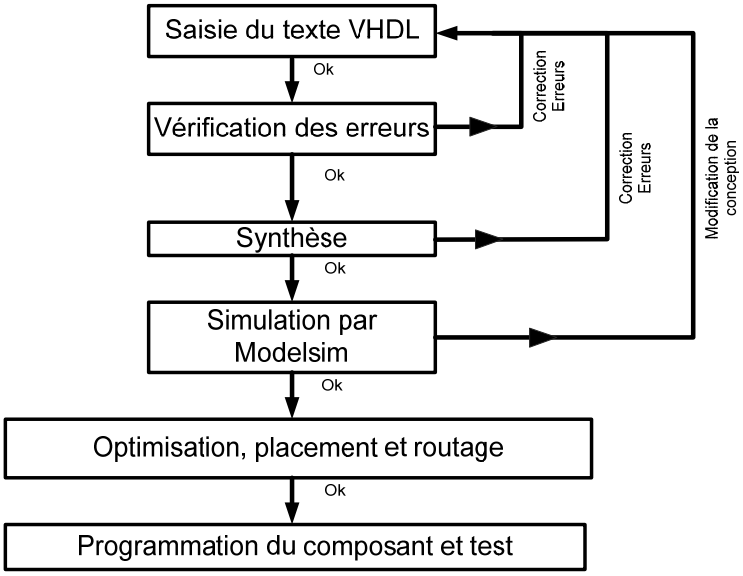


Figure 4.10 : Organigramme fonctionnelle de développement d'un projet sur circuit FPGA

5.2- Saisie du texte VHDL

La saisie du texte VHDL se fait sur le logiciel « ISE Xilinx Project Navigator ». Ce logiciel propose une palette d'outils permettant d'effectuer toutes les étapes nécessaires au développement d'un projet sur circuit FPGA. Il possède également des outils permettant de mettre au point une entrée schématique ou de créer des diagrammes d'état, qui peuvent être utilisés comme entrée au lieu du texte VHDL.

La figure 4.11 montre comment se présente le logiciel «ISE Xilinx Project Navigator». La saisie du texte se fait sur la partie droite de l'écran, on voit en haut à gauche la hiérarchie du projet, et en bas à gauche les nombreux outils nécessaires tout au long du développement du projet.

Il faut commencer par crée un projet, ensuite inclure des fichiers sources dans lesquels il faut saisir le texte VHDL désiré.

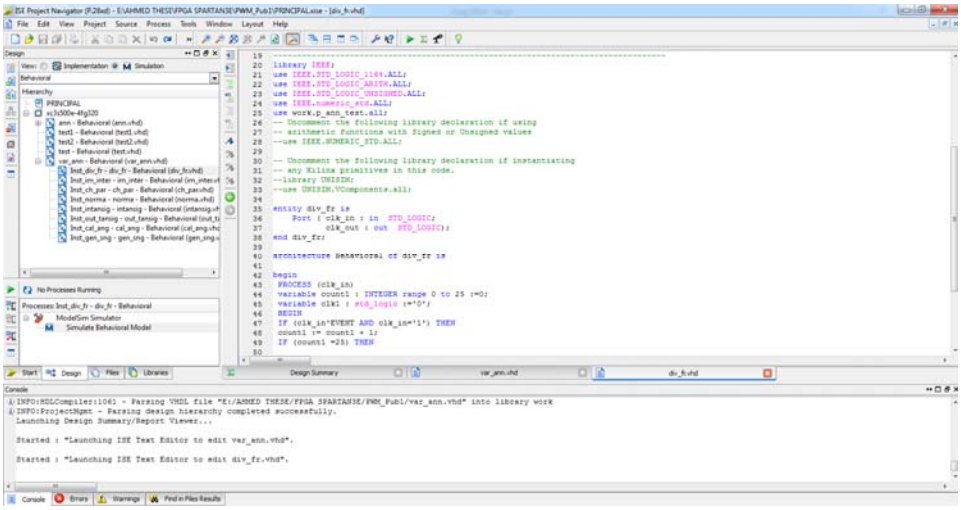


Figure 4.11 : Vue d'ensemble du logiciel "ISE Xilinx Project Navigator "

5.3- Vérification des erreurs

Cette étape est effectuée en appuyant sur le bouton « check syntax ». Elle permet de vérifier les erreurs de syntaxe du texte VHDL et d'afficher les différentes alarmes liées au programme, par exemple des signaux déclarés mais non utilisés dans le programme. S'il y'a des erreurs dans le programme, il ne peut pas être synthétisé, mais la présence d'alarmes n'empêche pas de poursuivre normalement les autres étapes du développement. Cette étape permet donc de valider la syntaxe du programme et de générer la « netlist », qui est un fichier contenant la description de l'application sous forme d'équations logiques.

5.4- Synthèse

La synthèse permet de réaliser l'implémentation physique d'un projet. Le synthétiseur a pour rôle de convertir le projet, en fonction du type du circuit FPGA cible utilisé, en portes logiques et bascules de base. L'outil « View RTL Schematic » permet de visualiser les schémas électroniques équivalents générés par le synthétiseur (figure 4.12).

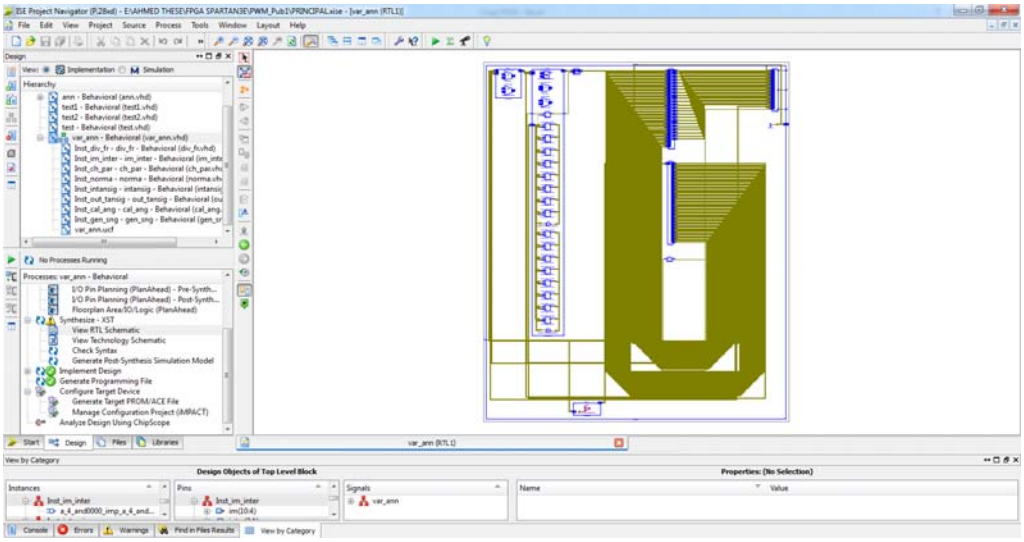


Figure 4.12 : Aperçu de l'outil « View RTL Schematic »

De plus, le synthétiseur permet à l'utilisateur d'imposer des contraintes de technologie (User constraints) : par exemple fixer la vitesse de fonctionnement (Create Timing Constraints), délimiter la zone du circuit FPGA dans laquelle le routage doit se faire (Create Area constraints) ou affecter les broches d'entrées/sorties (Assign Package Pins). La figure 4.13 montre un aperçu de l'outil d'assignation des broches d'entrées/sorties.

5.5- Simulation

La simulation permet de vérifier le comportement d'un design avant ou après le synthèse. Elle représente une étape essentielle qui nous fera gagner du temps lors de la mise au point sur la carte. Il faut juste noter qu'un projet peut être simulé même s'il n'est pas synthétisable. Le simulateur qu'on a utilisé pour simuler notre application est le « ModelSim DE 6.5e »

Figure 4.13 : Aperçu de l'outil d'affectation des broches d'entrées/sorties

5.6- Optimisation, placement et routage

Pendant l'étape d'optimisation, l'outil cherche à minimiser les temps de propagation et à occuper le moins d'espace possible sur le circuit FPGA cible. Le placement et routage permet de tracer les routes à suivre sur le circuit afin de réaliser le fonctionnement attendu. La figure 4.15 donne un aperçu de l'outil de placement et routage « FPGA Editor » qui permet de visualiser et d'éditer le circuit routé.

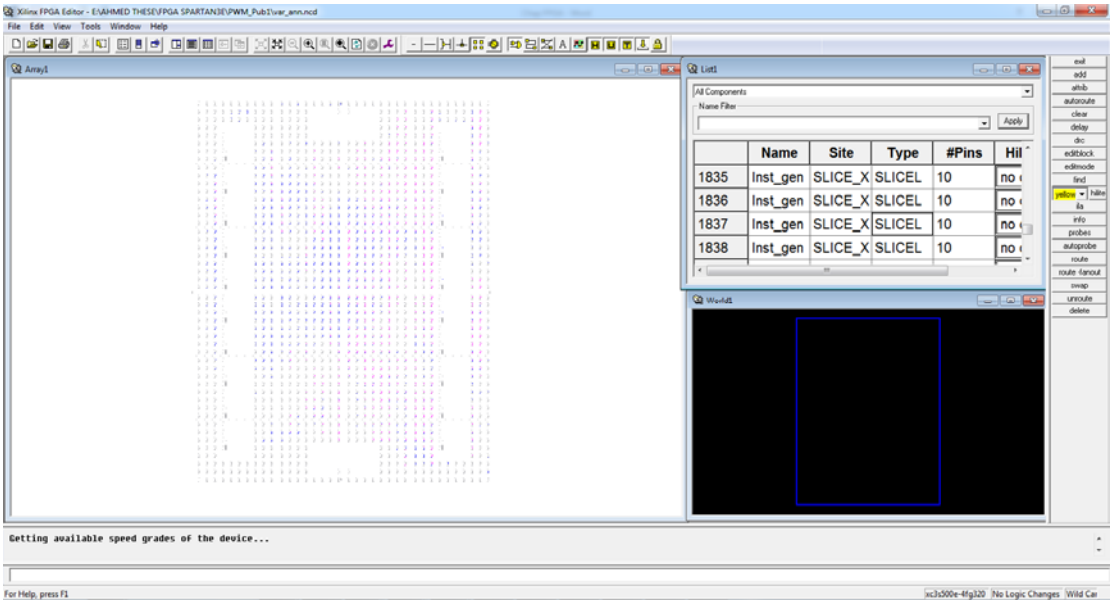


Figure 4.15 : Aperçu de l'outil « FPGA Editor »

5.7- Programmation du composant et test

Dans cette dernière étape (Generate Programming Files), on génère le fichier à charger sur le circuit FPGA. Une fois le programme chargé sur le circuit, on peut tester et visualiser les résultats directement sur la carte de développement à travers les nombreuses interfaces qu'elle offre.

6- Conclusion

Dans ce chapitre, on a présenté l'architecture des circuits FPGA et la méthode à suivre pour développer un projet sur ces circuits. D'après l'étude de l'architecture on retient que les circuits FPGAs sont des composants reconfigurables, ils sont constitués en générale d'une matrice de blocs logiques (CLB) programmables entourés de blocs d'entrée sortie (IOB) programmable. Ces deux blocs peuvent être reliés par un réseau d'interconnexions programmables. De plus le circuit Spartan 3E de Xilinx contient deux autres blocs qui sont le bloc des Multiplicateurs 18x18 et le bloc Digital Clock Manager (DCM). On a aussi présenté le langage de description VHDL qui est un langage de spécification, simulation et conception. Enfin, pour développer une application avec le VHDL il faut suivre six étapes, la saisie du texte VHDL, la vérification des erreurs, la synthèse, la simulation, l'optimisation, le placement et routage et la programmation du composant et test.

Dans le chapitre suivant, nous montrons les étapes qu'on a suivies pour implémenter l'algorithme ANNSHE PWM proposé sur un circuit FPGA Spartan 3E de Xilinx. Ainsi les détails de chaque partie du code VHDL seront présentés.

Chapitre V : Optimisation de l'implémentation de l'algorithme ANNSHE PWM sur un circuit FPGA

1- Introduction

La recherche dans le domaine des machines intelligentes ressemblant à des humains a mené à la naissance des réseaux de neurones artificiels (ANN). De nombreux travaux basés sur des simulations par ordinateur ont prouvé la capacité des ANNs à mapper, modeler et classer les systèmes non linéaires. Les caractéristiques particulières des ANNs telles que la capacité d'apprendre à partir d'exemples, le parallélisme, la robustesse au bruit et la tolérance aux défauts ont ouvert leurs applications dans divers domaines tels que l'ingénierie, la science et l'économie [37]. L'implémentation hardware des ANNs peut être réalisée en utilisant de l'analogique ou le numérique [41]. L'implémentation numérique est plus populaire car elle a beaucoup d'avantages : une plus grande précision, une faible sensibilité aux bruits et une plus grande flexibilité. D'autre part, les systèmes analogiques sont plus difficiles à concevoir et ne peuvent être réalisable pour les applications à grande échelle [41]. L'implémentation numérique des ANNs est également classée comme suit : 1) implémentations sur field-programmable gate array (FPGA), 2) implémentations surdigital signal processor (DSP) et 3) implémentations sur application specific integrated chip (ASIC) [41], [42].

L'implémentation sur des DSP est séquentielle et donc ne permet pas l'architecture parallèle des neurones dans une couche. D'autre part, l'implémentation ASIC n'offre pas de la reconfiguration. En conséquence l'implémentation FPGA est la plus appropriée pour les ANNs car elle préserve l'architecture parallèle des neurones et peut être reconfigurée par l'utilisateur [37].

Dans le chapitre III on a présenté les différentes étapes suivies pour développer l'algorithme ANNSHE PWM. On a aussi montré l'efficacité et la précision de cet algorithme dans l'asservissement de la tension fondamentale et dans l'élimination des harmoniques indésirables sélectionnés. Lors du développement de cet algorithme on a toujours pris en considération l'implémentation hardware. Par exemple, le choix de l'architecture des réseaux ANN-i (voir section 3.2) où on a choisi l'architecture la plus simple. De même le choix du nombre d'angles par intervalle (voir section 4.3) où on a varié le nombre d'angles pour

minimiser l'espace consommé lors de l'implémentation. On a testé également le fonctionnement de l'algorithme proposé sur un circuit FPGA mais la conception sous VHDL n'était pas optimisée. En conséquence, dans ce chapitre, on explique la façon dont on a procédé pour optimiser l'implémentation de l'algorithme proposé sur un circuit FPGA ainsi les détails de chaque partie du code VHDL seront présentés. Enfin, ce chapitre sera terminé par une validation expérimentale.

2- La structure détaillée de l'algorithme ANNSHE PWM

Dans la littérature plusieurs travaux de recherche se sont intéressés à l'implémentation des ANNs sur un circuit FPGA. La plupart de ces recherches sont citées dans les articles [33-35]. D'après ces recherches, on trouve des articles où le réseau et l'algorithme d'apprentissage, en générale l'algorithme Back-propagation, sont implémentés sur le circuit FPGA, c'est-à-dire les paramètres (poids et seuils) du réseau sont calculés par le circuit lui-même à la fin de l'étape d'apprentissage et seront utilisés par la suite lors du calcul de la sortie du réseau [36-38]. De plus on trouve des articles où le réseau seul est implémenté sur le circuit FPGA et les paramètres, poids et seuils, du réseau sont calculés off-line et stockés dans une mémoire [39-40]. Pour l'algorithme ANNSHE PWM proposé les paramètres, poids et seuils, sont calculés off-line donc on n'a pas besoin d'implémenter l'algorithme d'apprentissage. De plus pour développer la structure générale de l'algorithme ANNSHE PWM on se réfère à l'architecture de l'algorithme présentée sur la Figure 3.1 et au Tableau 3.1 ainsi qu'à l'organigramme de test présenté sur la Figure 3.6. Le but du code VHDL est la conception d'un circuit qui permet de générer les trois signaux PWM à partir de l'indice de modulation im et une horloge Clk_in. L'organigramme détaillé de l'algorithme ANNSHE PWM utilisé pour créer le code VHDL est présenté sur la Figure 5.1. Le rôle et le détail de chaque module de cette structure sont présentés dans les sections qui suivent.

3- Sélection du Réseau

D'après le tableau 4.1 on a divisé l'intervalle de variation de im en six intervalles. Pour chaque intervalle on a construit un réseau ANN-i. En conséquence, le but du module "Selection du Réseau" est la sélection du réseau ANN-i qui convient à l'entrée im Figure 5.2. Dans l'étape d'apprentissage on a exprimé im en pourcentage c'est-à-dire im est compri entre 0 et 100 %. Dans la conception on a dimensionné l'entrée im sur dix bits, sept bits pour la partie entière et trois bits pour la partie fractionnelle, donc le pas de variation de im est 2^{-3}=0.125%. Ce choix peut être changé selon le choix du pas de variation de im. De plus, pour simplifier le choix de chaque intervalle en fonction de im on a défini les limites de chaque intervalle comme indiqué sur le tableau 5.1. Ce choix permet donc de minimiser l'espace

consommé sur le circuit FPGA. Les paramètres de l'apprentissage ainsi que les caractéristiques de chaque réseau ANN-i sont résumés dans le Tableau 5.2.

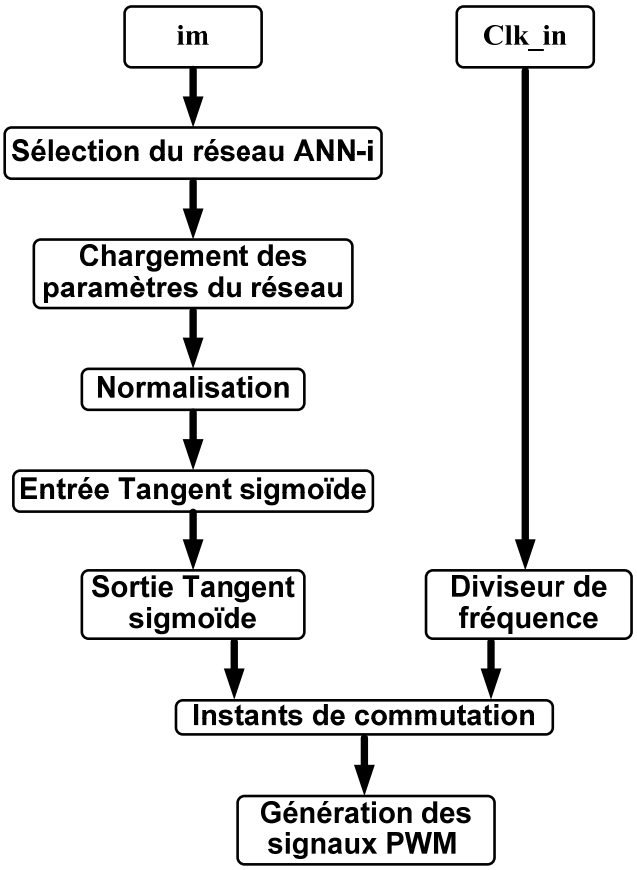


Figure 5.1 : Organigramme détaillé de l'algorithme ANNSHE PWM

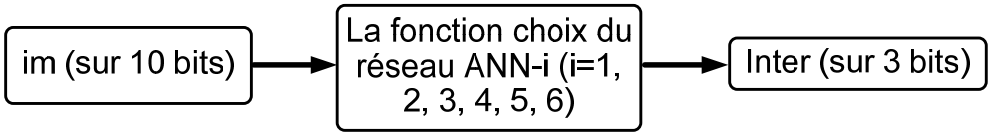


Figure 5.2 : Schéma synoptique du module "Sélection du Réseau"

Tableau 5.1 : Intervalle de variation de im pour chaque réseau ANN-i

ANN	Intervalle de variation de im en pourcentage im_{min}<=im< im_{max}	La limite inferieure en binaire	Inter
ANN-1	01<= im <16%	0000000000	001
ANN-2	16<= im <32%	0010000000	010
ANN-3	32<=im <56%	0100000000	011
ANN-4	56<=im <76%	0111000000	100
ANN-5	76<=im <92%	1001100000	101
ANN-6	92<=im <100%	1011100000	110

Tableau 5.2 : Caractéristiques des réseaux ANN-i

ANN	Nombre de neurones dans la couche cachée	Nombre de neurones dans la couche de sortie	L'algorithme d'apprentissage	Les fonctions d'activation	Nombre d'éléments dans la base de données d'apprentissage	Nombre d'éléments dans la base de données de test	Paramètres d'apprentissage	
							performance	Nombre d'itérations
ANN-1	1	23	Back-propagation Algorithme	Hyperbolique Tangent Sigmoïde + linéaire	23x10	23x20	$2.2\ 10^{-5}$ max 10^{-5}	54693 max 10^5
ANN-2	1	19			19x10	19x20	$2.6\ 10^{-5}$ max 10^{-5}	52745 max 10^5
ANN-3	1	15			15x20	15x40	$3.1\ 10^{-5}$ max 10^{-5}	53267 max 10^5
ANN-4	1	7			7x20	7x40	$8.9\ 10^{-5}$ max 10^{-5}	54735 max 10^5
ANN-5	1	5			5x20	5x40	0.0003 max 10^{-5}	6390 max 10^5
ANN-6	1	3			3x20	3x40	0.0008 max 10^{-5}	8191 max 10^5

D'après le Tableau 5.1 on trouve :

$$
\begin{aligned}
&ANN_1 = \text{not im(10) and not im(9) and not im(8)}\\
&ANN_2 = \text{not im(10) and not im(9) and im(8);}\\
&ANN_3 = \text{not im(10) and im(9) and (not im(8) or not im(7));}\\
&ANN_4 = \text{(not im(10) and im(9) and im(8) and im(7)) or (im(10)a nd not im(9) and not im(8) and (not im(7) or not im(6)));}\\
&ANN_5 = \text{im(10) and not im(9) and ((not im(8) and im(7) and im(6)) or (im(8) and (not im(7) or not im(6))));}\\
&ANN_6 = \text{im(10) and (((not im(9) and im(8) and im(7) and im(6))) or ((im(9) and not im(8) and not im(7)) and (not im(6) or}\\
&\text{(im(6) and not im(5) and not im(4)))));}
\end{aligned}
\tag{5.1}
$$

$$
\begin{aligned}
&\text{Inter}(1) = ANN_1 \text{ or } ANN_3 \text{ or } ANN_5\\
&\text{Inter}(2) = ANN_2 \text{ or } ANN_3 \text{ or } ANN_6\\
&\text{Inter}(3) = ANN_4 \text{ or } ANN_5 \text{ or } ANN_6
\end{aligned}
\tag{5.2}
$$

Où *"Inter"* représente l'indice du réseau en binaire, voir Tableau 5.1.

En se basant sur les deux équations (5.1) et (5.2) une conception basée sur la logique combinatoire du module *"Sélection du Réseau"* est crée sous VHDL afin de sélectionner le réseau adéquat à l'entrée im. La Figure 5.3 montre la simulation sous Modelsim du code VHDL pour différentes valeurs de im.

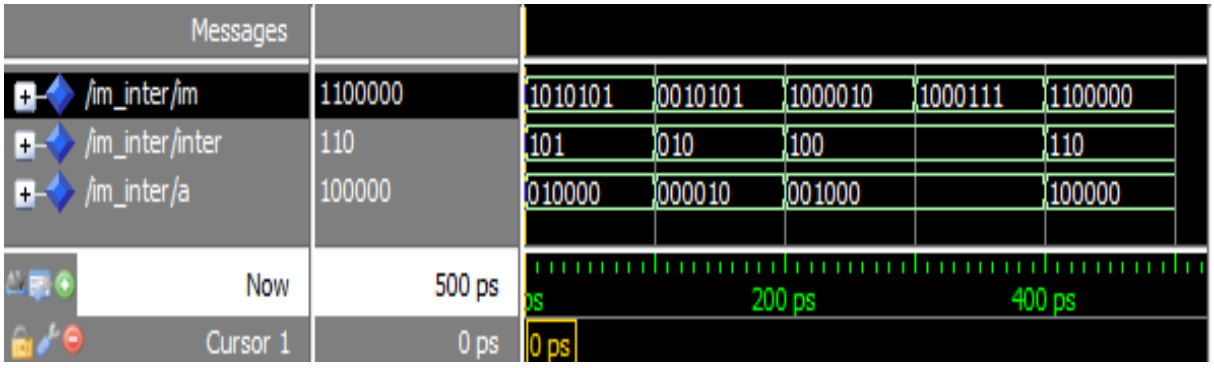


Figure 5.3 : Simulation du module "Sélection du Réseau" sous Modelsim

4- Normalisation

Avant de faire l'apprentissage, on a normalisé toutes les données de la base d'apprentissage, afin qu'elles soient actives, en moyenne sur la partie linéaire de la fonction d'activation. Par conséquent le but de cette fonction est de calculer la valeur normalisée de im qui sera l'entrée du réseau ANN-i Figure 5.4. La fonction de normalisation est donnée par l'équation (5.3).

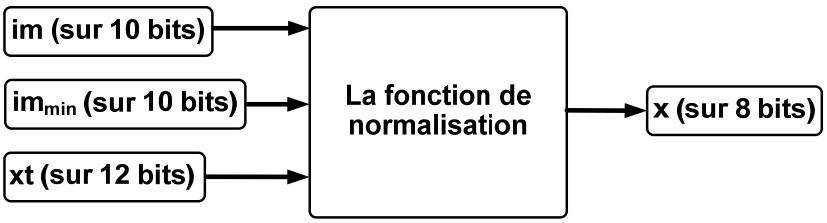


Figure 5.4 : Schéma synoptique du module "Normalisation"

$$f_{norm}(im) = x = \frac{x_{max} - x_{min}}{im_{max} - im_{min}}(im - im_{min}) + x_{min} \quad (5.3)$$

Dans l'étape d'apprentissage on a utilisé $x_{max}=1$ et $x_{min}=0$, en remplaçant dans l'équation (5.3) on trouve :

$$f_{norm}(im) = x = xt(im - im_{min}) \quad (5.4)$$

$$\text{Où : } xt = \frac{1}{im_{max} - im_{min}} \quad (5.5)$$

La dimension de la valeur normalisée "X" dépend de la précision dans le calcul de la fonction tangent-sigmoïde. Dans le programme on a dimensionné "X" sur huit bits, un bit pour la partie entière et sept bits pour la partie fractionnelle. Ainsi, pour les six réseaux on a : im - im_{min}<32=25 (voir Tableau 5.1 et 5.3). De ce fait "xt" sera dimensionné sur 7+5=12 bits fractionnels. Les valeurs de im_{min} et "xt" sont illustrées dans le Tableau 5.3 pour chaque réseau ANN-i. Ces valeurs seront stockées dans le circuit FPGA. Le choix des valeurs de im_{min} et "xt" qui convient à l'entrée im est assuré par le module "Chargement de Paramètres".

Tableau 5.3 : Les valeurs de im_{min} et xt pour chaque réseau ANN-i

ANN	im_{min} (%)	xt	im_{min} en binaire	xt en binaire
ANN-1	01	0,07142857	0000001000	000100100100
ANN-2	16	0,06666667	0010000000	000100010001
ANN-3	32	0,04347826	0100000000	000010110010
ANN-4	56	0,05263158	0111000000	000011010111
ANN-5	76	0,06666667	1001100000	000100010001
ANN-6	92	0,12500000	1011100000	001000000000

Une conception du module "Normalisation" sous VHDL est réalisée en se basant sur l'équation (5.4). D'après cette équation, la conception contient un soustracteur de 10bits et un multiplicateur (12bits x 10bits). La figure 5.5 montre la simulation du module "Normalisation" sous Modelsim pour différentes valeurs de im, im_{min} et xt.

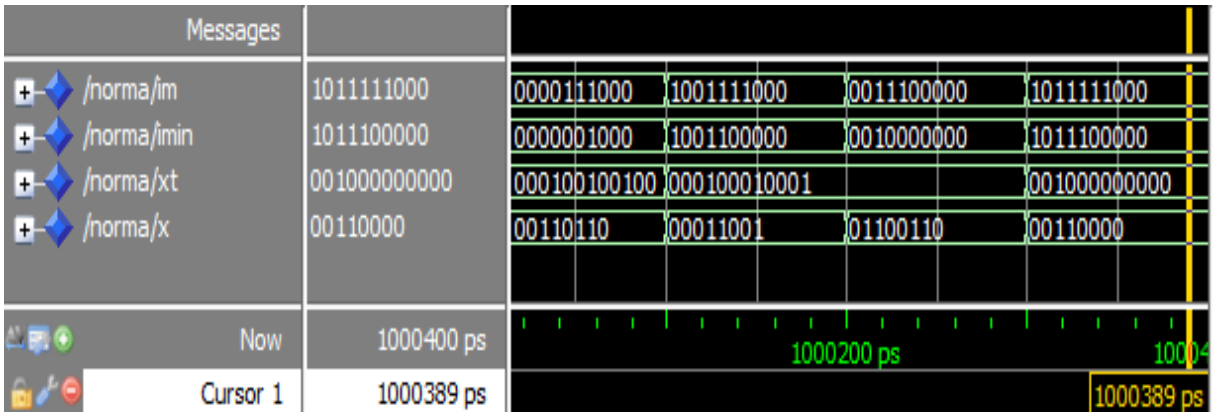


Figure 5.5 : Simulation du module "Normalisation" sous Modelsim

5- Entrée Tangente-Sigmoïde

Dans la section précédente on a calculé la valeur normalisée "x" de l'indice de modulation im qui représente l'entrée unique du réseau ANN-i. De plus, d'après l'architecture choisie, Figure 4.2, on a utilisé, comme fonction d'activation, la fonction Tangente-Sigmoïde entre l'entrée et la couche cachée. En conséquence, le but de ce module est de calculer la valeur "x_out" donnée par l'équation (5.6) qui sera l'entrée de la fonction Tangente-Sigmoïde.

$$x\_out = \mathrm{x} \times \mathrm{w}_1 + b_1 \qquad (5.6)$$

Où w_1 et b_1 représentent le poids et le seuil respectivement entre l'entrée et la couche cachée.

Les valeurs de w_1 et b_1 pour chaque réseau sont montrés dans le tableau 5.4.

Tableau 5.4 : Les valeurs de w_1 et b_1 pour chaque réseau ANN-i

ANN	w_1	b_1	w_1 en binaire	b_1 en binaire	s_in
ANN-1	1,64820136	-0,831685308	0000001000	000100100100	0
ANN-2	-1,501175243	0,757855017	0010000000	000100010001	1
ANN-3	-1,08827136	0,55806052	0100000000	000010110010	1
ANN-4	0,98634884	-0,52255451	0111000000	000011010111	0
ANN-5	-1,22971402	0,67017374	1001100000	000100010001	1
ANN-6	-1,52228874	0,86779769	1011100000	001000000000	1

D'après le tableau 5.4 on a remarqué que les signes de "w1" et "b1" sont différents. Donc on a utilisé "s_in" pour représenter cette différence où s_in =0 lorsque w1 est positif et s_in =1 lorsque w1 est négatif, de plus on a utilisé s_out pour représenter le signe de x_out.

Le choix des valeurs de w1, b1 et s_in qui convient au réseau ANN-i est assuré par le module "Chargement de Paramètres". La figure 5.6 montre une simulation du module entrée Tangente-Sigmoïde pour différentes valeurs de w1 et b1.

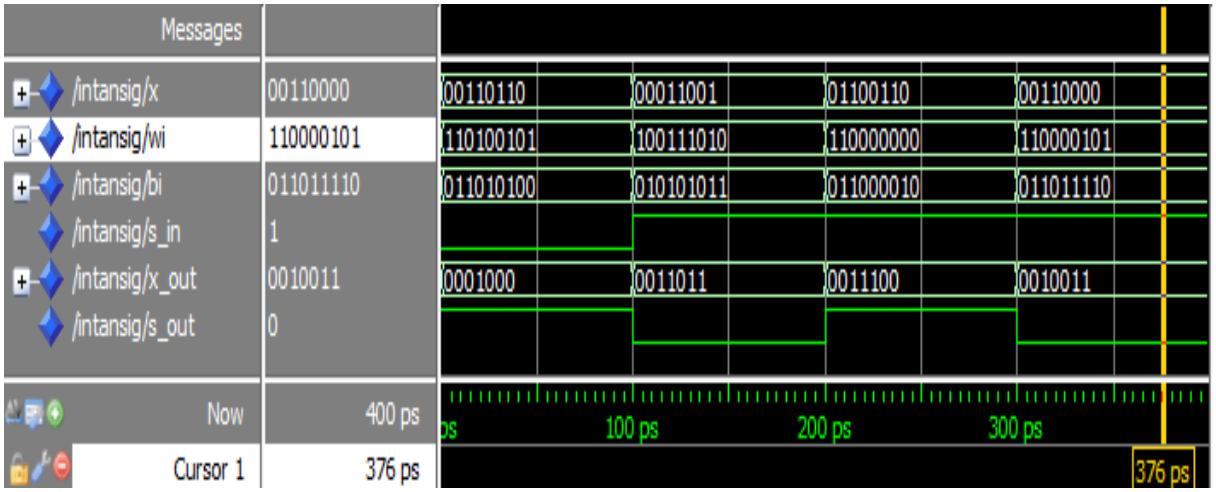


Figure 5.6 : Simulation du module " Entrée Tangente-Sigmoïde" sous Modelsim

6- Sortie Tangente-Sigmoïde

6.1- Introduction

Les principales composantes de base nécessaires pour l'implémentation hardware des ANNs sont les multiplicateurs, les additionneurs, et la fonction d'activation non linéaire. Cette dernière est l'élément le plus important et le plus complexe, qui est utilisé à la sortie de chaque neurone. Plusieurs fonctions d'activation sont disponibles aujourd'hui, y compris la fonction sigmoïd, la fonction tangente hyperbolique (tanh) et la fonction échelon [43-46]. La fonction tangente hyperbolique est parmi les fonctions d'activation les plus utilisées dans les ANNs [44]. Cette fonction produit une courbe sigmoïde ayant la forme en S figure 5.7. En raison des termes exponentiels et de division présents dans cette fonction, il est difficile de l'implémenter directement dans un circuit électronique. Pour résoudre ce problème, des méthodes d'approximations sont généralement appliquées [43]. Ces méthodes sont basées sur l'approximation linéaire par morceaux (piecewise linear approximation PWL), l'approximation non linéaire par morceaux (piecewise nonlinear approximation PWNL), la table de consultation (lookup table LUT) et les méthodes hybrides. Généralement, dans des méthodes d'approximation PWL ou PWNL, une série de segments linéaires (PWL) ou non linéaire (PWNL) sont utilisés pour approximer la fonction [47]. Le nombre et l'emplacement de ces segments sont choisis de telle sorte que l'erreur, le temps de traitement et la surface d'implémentation sont minimisés. Cette approche nécessite généralement plusieurs cycles d'horloge et l'utilisation de multiplicateurs, qui sont coûteux en termes de surface et vitesse [43-44]. Dans les méthodes à base de LUT, la fonction est approximée par un nombre limité

de points. Ces points sont répartis sur tout l'intervalle d'entrée et stockés dans une LUT [44,46]. Les Méthodes hybrides utilisent une combinaison des méthodes mentionnées [45,48]. La méthode à base de LUT est la plus rapide de ces techniques mais la taille de la LUT augmente avec la précision. Sur la base de l'étude comparative entre plusieurs méthodes d'implémentation faite par Babak [43], on a opté pour la méthode à base de LUT optimisée par Maher [46]. L'architecture proposée est basée sur l'approximation linéaire en combinaison avec une LUT. Elle nécessite 7 ou 15 valeurs à stocker dans une LUT pour une erreur de 0,04 ou 0,02 respectivement. La réduction du nombre de points mémorisés réduit la surface nécessaire pour l'implémentation.

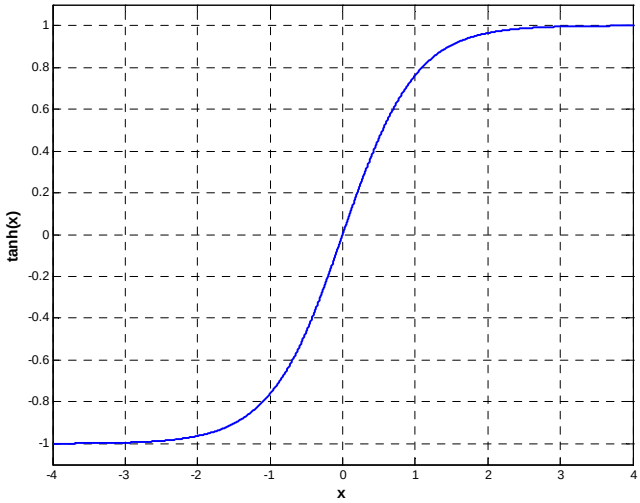


Figure 5.7 : La fonction tangente hyperbolique

6.2- Les caractéristiques de la fonction tangente hyperbolique

a- Propriété 1

La fonction tangente hyperbolique est définie par l'équation suivante :

$$\tanh(x) = \left(e^{x} - e^{-x}\right)/\left(e^{x} + e^{-x}\right) \tag{5.7}$$

D'après cette équation, on voit bien que cette fonction est impaire c'est-à-dire

$$\tanh(-x) = -\tanh(x) \tag{5.8}$$

En utilisant cette propriété, on doit stocker seulement les valeurs de tanh(x) pour la moitié droite de la courbe de la figure 5.7, c'est-à-dire pour x>0.

b- Propriété 2

Le développement en série de Taylor de la fonction tanh(x) est le suivant :

$$\tanh(x) = x - \frac{x^3}{3} + \frac{2x^5}{15} - \frac{17x^7}{315} + \cdots \tag{5.9}$$

Pour les petites valeurs de x (Figure 5.7), les termes d'ordre supérieur deviennent tris petits et peuvent être ignorés. Par conséquent, la tangente hyperbolique transmet les petites valeurs d'entrée à la sortie (5.10).

$$\lim \tanh(x) = x \qquad x \to 0 \tag{5.10}$$

De plus, pour les petites valeurs de x on peut approximer l'expression (5.9) par l'expression suivante :

$$\tanh(x) \approx x - \frac{x^3}{3} + \frac{2x^5}{15} \tag{5.11}$$

Si ε est l'erreur maximale autorisée, ainsi on peut trouver δ la valeur maximale de x pour laquelle on peut supposer tanh (x) = x en utilisant l'équation :

$$\frac{\delta^3}{3} - \frac{2\delta^5}{15} \leq \varepsilon \tag{5.12}$$

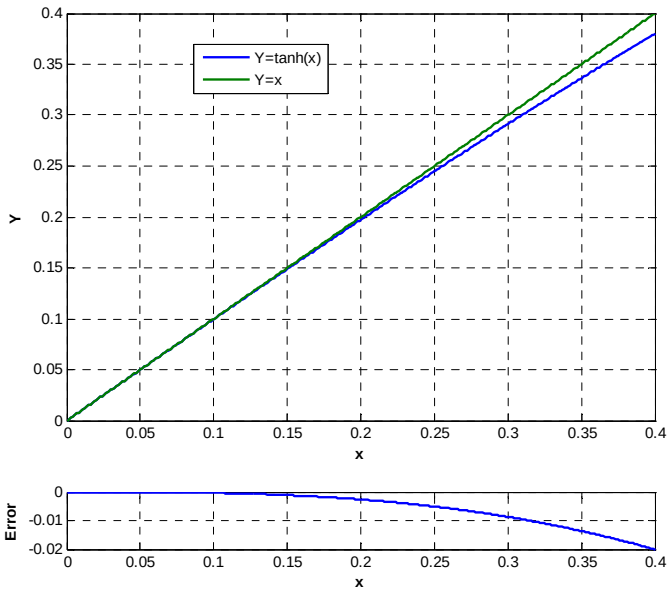


Figure 5.8 : Comparaison entre la fonction Y=tanh(x) et la fonction Y= x pour les petites valeurs de x

c- Propriété 3

La variation de la sortie pour les grandes valeurs de l'entrée est faible (5.13).

$$\frac{d\tanh(x)}{dx} = 0 \qquad x \to \infty$$
$$\tanh(x) = 1 \tag{5.13}$$

D'après cette propriété, la variation de tanh(x) est significative uniquement lorsque x varie de -3 à +3. Sa variation en dehors de cet intervalle de x est inférieure à 0,00015, qui n'est pas

significative pour la majorité des applications ANN [46]. Par conséquent, on peut stocker +1 pour toutes les valeurs de x> 3. On constate d'après la figure 5.7 que pour x = 2, tanh (x)> 0.96, pour x=2.4, tanh (x)> 0.98 et pour x = 2.7, tanh (x)> 0.99. Ainsi, on peut approximer tanh (x) = 1 pour x> 2.0, 2.4 ou 2.7 si l'erreur maximale admissible ε est de 0.04, 0.02, ou 0.01, respectivement (Figure 5.9).

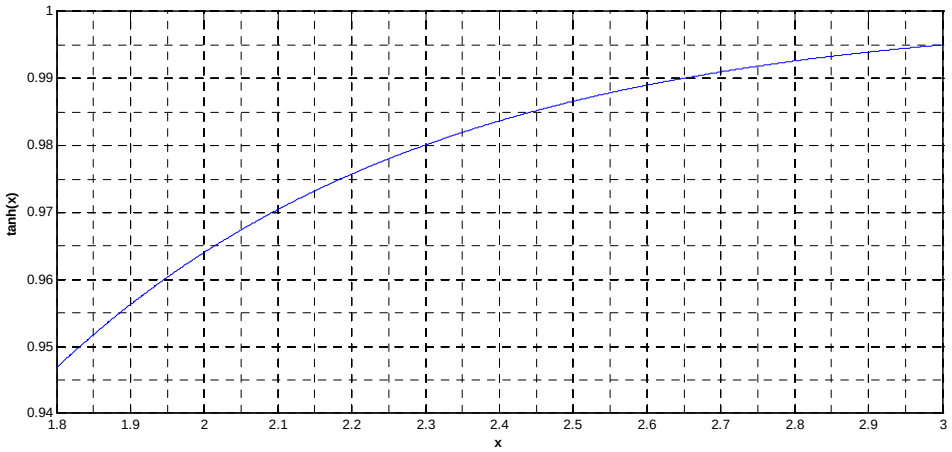


Figure 5.9 : La fonction Tangent Hyperbolique dans la région de saturation

6.3- LUT optimisée par Maher pour une erreur maximale admissible $\varepsilon = 0.02$ ***[46]***

Dans ce projet, on a opté pour une erreur maximale admissible $\varepsilon = 0.02$ dans le calcul de la fonction tanh. Pour construire une LUT optimisée, Maher a proposé de suivre les étapes suivantes :

Etape 1 : Détermination des limites inférieure et supérieure de l'entrée de la LUT

La limite inférieure δ de l'entrée de la LUT est calculée en utilisant l'équation (5.12). Pour $\varepsilon = 0.02$ on trouve $\delta = 0.390625$. Ainsi de la figure 5.9, on trouve que la limite supérieure de l'entrée $\alpha = 2.4$ pour $\varepsilon = 0.02$.

Etape 2 : Sélection du nombre de bits de l'entrée (largeur de l'adresse d'entrée)

En se basant sur des simulations sous Matlab, on trouve que le nombre de bits adéquates pour que la différence $\left|\tanh(x_2) - \tanh(x_1)\right| < 0.02$, où x_1 et x_2 sont deux adresses (entrées) successives, est n=8.

Etape 3 : Détermination des limites de chaque bande d'adresses

Dans cette étape on divise l'intervalle de variation de l'entrée $x, 0.390625 < x \le 2.4$, en N sous-domaines (bandes) $R_i(x_{i1}, x_{i2})$ et on détermine les limites x_{i1} et $x_{i2}, 1 \le i < N$, de chaque bande en se basant sur l'équation (5.13) et en commençant par $x_{11} = \delta = 0.390625$.

$$\tanh(x_{i2}) - \tanh(x_{i1}) < 2\varepsilon$$

Où N est le plus petit entier pour lequel $x_{N2} \geq 2.4$

Etape 3 : affectation de la sortie

Pour chaque intervalle $[x_{i1}, x_{i2}], 1 \leq i < N$, la valeur moyenne de tanh(x) c'est-à-dire $[\tanh(x_{i2}) + \tanh(x_{i1})]/2$ est stockée dans la LUT.

En se basant sur les étapes mentionnées ci-dessus, le tableau 5.5 montre la LUT de la fonction tanh(x) pour une erreur maximale admissible $\varepsilon = 0.02$.

Tableau 5.5 : La LUT de la fonction tangente hyperbolique pour une erreur maximale admissible $\varepsilon = 0.02$.

x1	x2	tanh(x)	x1 en binaire	a=1024tanh(x)
…	0,390625	x		
0,390625	0,453125	0,4049	0011001	414
0,453125	0,515625	0,4558	0011101	466
0,515625	0,578125	0,5038	0100001	515
0,578125	640625	0,549	0100101	562
0,640625	0,703125	0,5911	0101001	605
0,703125	0,78125	0,6348	0101101	650
0,78125	0,859375	0,6791	0110010	695
0,859375	0,9375	0,719	0110111	736
0,9375	1,048675	0,7609	0111100	779
1,046875	1,171875	0,8057	1000011	825
1,171875	1,328125	0,8493	1001011	869
1,328125	1,53125	0,8916	1010101	912
1,53125	1,859375	0,9329	1100010	955
1,859375	2,90625	0,974	1110111	997
2,90625	…	1	10111010	

6.4- La conception du module "Sortie Tangente-Sigmoïde"

D'après l'étude précédente pour calculer la sortie de la fonction tangente sigmoïde, avec une erreur maximale admissible $\varepsilon = 0.02$, on a besoin de connaitre le signe et la valeur absolue de l'entrée sur 8 bits, 2 bits pour la partie entiere et 6 bits pour la partie fractionnelle. D'abord la valeur absolue de l'entrée sera comparée avec les valeurs x1 du tableau 5.5 pour déterminer la bande Ri qui permet de déduire la valeur absolue de tanh(x) en utilisant toujours le tableau 5.5. Ensuite et d'après la propriété 1 de la fonction tangente hyperbolique, le signe de la valeur de sortie est le même que celui de l'entrée. Dans notre application, l'entrée est toujours inférieure à 2, de ce fait on a besoin de 7 bits, un bit pour la partie entière et 6 bits pour la partie fractionnelle, pour coder la valeur absolue de l'entrée. De plus, on va voir dans la section suivante que la sortie de la fonction tangente hyperbolique doit être multipliée par 2^{10}=1024, en conséquence dans la LUT implémentée sur le circuit FPGA on stocke la valeur a=1024tanh(x), voir tableau 5.5.

La figure 5.10 montre une simulation du module "Sortie Tangente-Sigmoïde" pour différentes valeurs de l'entrée.

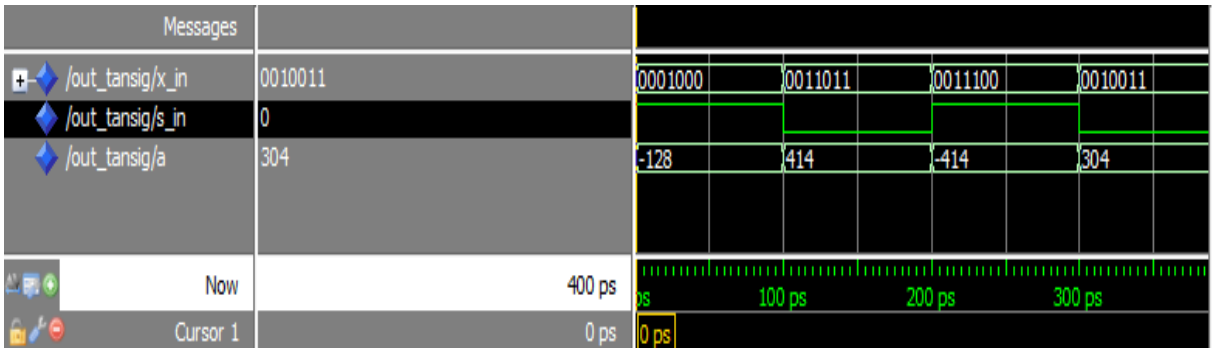


Figure 5.10 : Simulation du module " Sortie Tangente-Sigmoïde" sous Modelsim

7- Les instants de commutation

D'après la figure 4.2, pour calculer les angles de commutation α_i, la sortie tangente sigmoïde "a=tanh(x)" doit passer d'abord dans la couche de sortie et ensuite dans une fonction de normalisation inverse puisque la base des données a été normalisée avant de faire l'apprentissage, Figure 5.11.

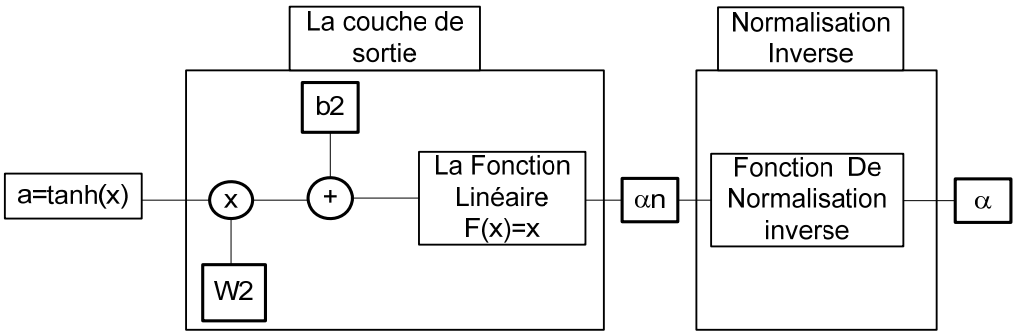


Figure 5.11 : Calcul des angles de commutation

D'après la Figure 5.11, les angles de commutation sont calculés en utilisant les équations suivantes :

$$\alpha n_i = w2_i \times a + b2_i \tag{5.14}$$

$$\alpha_i = \frac{\alpha_{i\max} - \alpha_{i\min}}{\alpha n_{i\max} - \alpha n_{i\min}} (\alpha n_i - \alpha n_{i\min}) + \alpha_{i\min} \tag{5.15}$$

Où :

$w2_i$ et $b2_i$ sont respectivement le poids et le seuil de la couche de sortie correspond à l'angle α_i (voir tableau 5.6).

$\alpha_{i\max}$ et $\alpha_{i\min}$ sont respectivement l'angle maximum et l'angle minimum de la base de données correspond à l'angle α_i (voir tableau 5.7) .

Tableau 5.6 : Les poids et les seuils correspondant aux angles α_i

ANN / i	ANN-6		ANN-5		ANN-4		ANN-3		ANN-2		ANN-1	
	$w2_i$	$b2_i$	$w2_i$	$b2_i$	$w2_i$	$b2_i$	$w2_i$	$b2_i$	$w2_i$	$b2_i$	$w2_i$	$b2_i$
1	0,7470	0,4415	0,8798	0,4623	-1,066	0,4747	0,9747	0,4895	0,7429	0,4961	-0,690	0,4960
2	0,4119	0,5534	-0,841	0,6640	1,0609	0,5620	-0,951	0,5261	-0,672	0,5071	0,6054	0,5068
3	0,7467	0,4567	0,8790	0,4709	-1,065	0,4791	0,9746	0,4917	0,7429	0,4969	-0,690	0,4965
4	---	---	-0,882	0,5688	1,0677	0,5490	-0,969	0,5268	-0,725	0,5087	0,6634	0,5061
5	---	---	0,8781	0,4714	-1,065	0,4805	0,9745	0,4932	0,7430	0,4976	-0,690	0,4970
6	---	---	---	---	1,0671	0,5339	-0,973	0,5251	-0,735	0,5089	0,6775	0,5067
7	---	---	---	---	-1,065	0,4768	0,9744	0,4938	0,7431	0,4980	-0,690	0,4974
8	---	---	---	---	---	---	-0,974	0,5225	-0,738	0,5086	0,6833	0,5069
9	---	---	---	---	---	---	0,9744	0,4937	0,7431	0,4981	-0,690	0,4976
10	---	---	---	---	---	---	-0,974	0,5194	-0,740	0,5080	0,6860	0,5068
11	---	---	---	---	---	---	0,9744	0,4929	0,7431	0,4980	-0,690	0,4977
12	---	---	---	---	---	---	-0,974	0,5162	-0,741	0,5073	0,6875	0,5066
13	---	---	---	---	---	---	0,9745	0,4915	0,7431	0,4977	-0,690	0,4976
14	---	---	---	---	---	---	-0,974	0,5131	-0,742	0,5064	0,6885	0,5062
15	---	---	---	---	---	---	0,9747	0,4894	0,7431	0,4973	-0,690	0,4975
16	---	---	---	---	---	---	---	---	-0,742	0,5055	0,6891	0,5058
17	---	---	---	---	---	---	---	---	0,7430	0,4967	-0,690	0,4972
18	---	---	---	---	---	---	---	---	-0,742	0,5047	0,6896	0,5053
19	---	---	---	---	---	---	---	---	0,7429	0,4960	-0,690	0,4969
20	---	---	---	---	---	---	---	---	---	---	0,6899	0,5048
21	---	---	---	---	---	---	---	---	---	---	-0,690	0,4965
22	---	---	---	---	---	---	---	---	---	---	0,6901	0,5044
23	---	---	---	---	---	---	---	---	---	---	-0,690	0,4960

Tableau 5.7 : Les angles maximums et les angles minimums correspondant aux angles α_i .

ANN / i	ANN-6		ANN-5		ANN-4		ANN-3		ANN-2		ANN-1	
	α_{imax}	α_{imin}	α_{imax}	α_{imin}	α_{imax}	α_{imin}	α_{imax}	α_{imin}	α_{imax}	α_{imin}	α_{imax}	α_{imin}
1	16,311	14,852	12,945	11,377	11,254	9,8997	6,4676	5,7098	5,5893	5,2017	4,9786	4,6792
2	37,685	37,604	23,310	23,086	16,778	16,431	7,8889	7,7371	6,1510	6,0789	5,0523	5,0034
3	46,090	44,081	32,410	30,478	26,020	24,488	13,944	13,150	11,585	11,190	9,9786	9,6778
4	---	---	46,181	45,353	33,088	32,423	15,683	15,419	12,271	12,142	10,095	10,006
5	---	---	52,966	51,249	40,991	39,436	21,416	20,586	17,578	17,173	14,978	14,673
6	---	---	---	---	49,234	48,200	23,435	23,072	18,374	18,197	15,133	15,009
7	---	---	---	---	56,254	54,881	28,894	28,041	23,570	23,156	19,978	19,668
8	---	---	---	---	---	---	31,160	30,703	24,464	24,244	20,167	20,011
9	---	---	---	---	---	---	36,384	35,525	29,564	29,143	24,977	24,664
10	---	---	---	---	---	---	38,861	38,31	30,544	30,286	25,196	25,013
11	---	---	---	---	---	---	43,889	43,044	35,560	35,136	29,977	29,660
12	---	---	---	---	---	---	46,538	45,910	36,614	36,322	30,223	30,015
13	---	---	---	---	---	---	51,412	50,598	41,560	41,137	34,977	34,657
14	---	---	---	---	---	---	54,190	53,487	42,676	42,352	35,247	35,016
15	---	---	---	---	---	---	58,953	58,187	47,563	47,146	39,977	39,656
16	---	---	---	---	---	---	---	---	48,729	48,378	40,268	40,018
17	---	---	---	---	---	---	---	---	53,571	53,163	44,977	44,657
18	---	---	---	---	---	---	---	---	54,774	54,400	45,286	45,019
19	---	---	---	---	---	---	---	---	59,583	59,190	49,977	49,661
20	---	---	---	---	---	---	---	---	---	---	50,301	50,020
21	---	---	---	---	---	---	---	---	---	---	54,977	54,666
22	---	---	---	---	---	---	---	---	---	---	55,314	55,021
23	---	---	---	---	---	---	---	---	---	---	59,978	59,674

Dans l'étape d'apprentissage on a utilisé $\alpha n_{i\max} = 1$ et $\alpha n_{i\min} = 0$, en remplaçant dans (5.15) on a :

$$\alpha_i = (\alpha_{i\max} - \alpha_{i\min})\alpha n_i + \alpha_{i\min} \tag{5.16}$$

En remplaçant (5.14) dans (5.16) on trouve :

$$\alpha_i = (\alpha_{i\max} - \alpha_{i\min}) \times w2_i \times a + (\alpha_{i\max} - \alpha_{i\min}) \times b2_i + \alpha_{i\min}$$

Posant

$$wk_i = (\alpha_{i\max} - \alpha_{i\min}) \times w2_i \tag{5.17}$$

$$bk_i = (\alpha_{i\max} - \alpha_{i\min}) \times b2_i + \alpha_{i\min} \tag{5.18}$$

En remplaçant dans (5.16) on a :

$$\alpha_i = wk_i \times a + bk_i \tag{5.19}$$

Pour générer les signaux de commande PWM on a besoin de transformer les angles de commutation en instants de commutation. Dans notre application on a opté pour la commande v/f=cte. En utilisant cette propriété on trouve :

$$v / f = v_0 / f_0 \tag{5.20}$$

Ce qui implique

$$v / v_0 = f / f_0 = im \tag{5.21}$$

D'où

$$f = f_0 im \tag{5.23}$$

Où im est l'indice de modulation et $f_0 = 50Hz$ la fréquence pour im=1.

La relation entre l'angle de commutation et l'instant de commutation est donnée par la relation suivante :

$$t_i = \frac{\alpha_i}{360} \times \frac{1}{f} \tag{5.24}$$

En utilisant les équations (5.23) et (5.24) on trouve :

$$t_i = \frac{10^{-3}}{18} \times \frac{\alpha_i}{im} \tag{5.25}$$

Et comme on a exprimé im en pourcentage, on trouve :

$$t_i = \frac{\alpha_i}{180} \times \frac{1}{im} \tag{5.26}$$

De plus, pour générer les signaux de commande à partir des instants de commutation on a utilisé une horloge de 1MHz, donc les instants de commutation doivent être exprimés en μs , équation (5.27).

$$t_i(\mu s) = \frac{10^5}{18} \times \alpha_i \times \frac{1}{im} \tag{5.27}$$

En combinant les deux équations (5.27) et (5.19) on trouve :

$$t_i(\mu s) = \left(wt_i \times a + bt_i\right) \times \frac{1}{im}$$

(5.28)

Où :

$$wt_i = \frac{10^5}{18} \times wk_i \tag{5.29}$$

$$bt_i = \frac{10^5}{18} \times bk_i \tag{5.30}$$

D'après l'équation (5.28) on a une division sur im qui pose un problème d'implémentation sur un circuit FPGA. Pour éviter cette division, dans cette étape on calcule les valeurs de θ_i données par l'équation (5.31). Ensuite, dans l'étape de génération de signaux de commande, à chaque front montant de l'horloge (1 μs), au lieu d'ajouter 1 à l'accumulateur de temps "cm" (si on a calculé les instants t_i), on ajoute im et on compare "cm" avec θ_i au lieu de t_i (voir la section suivante).

$$\theta_i = \left(wt_i \times a + bt_i\right) \tag{5.31}$$

De plus, on a dimensionné im sur 10 bit dont 3 bit représente la partie fractionnelle (voir section 3). En conséquence, les valeurs de θ_i doivent être multipliées par 2^3=8, c'est-à-dire les valeurs de wt_i et bt_i stockées dans le circuit FPGA seront multipliées par 8 (voir tableau 5.8).

Concernant la sortie de la fonction tangente sigmoïde $'a'$, on l'a dimensionnée sur 10 bits fractionnels, ce choix peut être changé en fonction de la précision dans le calcul de la fonction tangente sigmoïde. Dans le circuit FPGA on a stocké la partie fractionnelle de $'a'$, c'est-à-dire on l'a multiplié par 2^{10}=1024 (voir Tableau 5.5). Ensuite, après le calcul du produit $wt_i \times a$ on a fait un décalage de 10 bits vers la droite et on l'a ajouté à bt_i pour obtenir la valeur de θ_i .

Tableau 5.8 : Les valeurs de wt_i et bt_i stockées dans le circuit FPGA.

ANN-6		ANN-5		ANN-4		ANN-3		ANN-2		ANN-1	
bt	wt	bt	wt	bt	wt	bt	wt	bt	wt	bt	wt
688750	48466	537873	61309	468580	-64219	270261	32829	239735	12799	214567	-9186
1673303	1491	1032692	-8378	738949	16336	347424	-6423	271800	-2157	223476	1313
1999960	66697	1395049	75473	1121002	-72564	601820	34424	506089	13041	436764	-9228
---	---	2036641	-32472	1457269	31574	691492	-11368	542589	-4140	446744	2637
---	---	2313724	67033	1785949	-73664	933152	35939	772225	13374	658899	-9346
---	---	---	---	2166767	49030	1033920	-15728	812770	-5769	669877	3754
---	---	---	---	2468286	-65034	1265011	36922	1038354	13682	881010	-9489
---	---	---	---	---	---	1375206	-19806	1082507	-7203	892904	4733
---	---	---	---	---	---	1597764	37177	1304596	13905	1103123	-9628
---	---	---	---	---	---	1715516	-23660	1351875	-8488	1115841	5599
---	---	---	---	---	---	1931601	36628	1571025	14009	1325253	-9742
---	---	---	---	---	---	2054877	-27247	1620913	-9643	1338699	6370
---	---	---	---	---	---	2266604	35276	1837689	13976	1547413	-9816
---	---	---	---	---	---	2393246	-30466	1889638	-10674	1561485	7056
---	---	---	---	---	---	2602779	33185	2104619	13794	1769611	-9841
---	---	---	---	---	---	---	---	2158058	-11576	1784202	7660
---	---	---	---	---	---	---	---	2371834	13460	1991855	-9808
---	---	---	---	---	---	---	---	2426174	-12346	2006854	8187
---	---	---	---	---	---	---	---	2639345	12976	2214149	-9713
---	---	---	---	---	---	---	---	---	---	2229441	8638
---	---	---	---	---	---	---	---	---	---	2436497	-9550
---	---	---	---	---	---	---	---	---	---	2451967	9014
---	---	---	---	---	---	---	---	---	---	2658902	-9319

D'après l'équation (5.31) on a besoin d'un multiplicateur pour calculer chaque valeur θ_i, où i peut varier de 1 à 23, donc si on calcule tous les θ_i parallèlement on a besoin de 23 multiplicateurs. Mais d'après le module de génération des signaux, section suivante, dans le premier front d'horloge on a besoin de calculer θ_1 et θ_m (m est le nombre d'angles de commutation), dans le dixième front on calcule θ_2 et θ_{m-1}, et ainsi de suite jusqu'au calcul de tous les instants de commutation θ_i. En conséquence, dans la conception de ce module on utilise seulement deux multiplicateurs.

La figure 5.12 montre une simulation du module "Instants de commutation" pour im=1001111000 et m=5. On voit que dans le premier front on calcule θ_1 et θ_5, ensuite on calcule θ_2 et θ_4 et enfin on calcule θ_3.

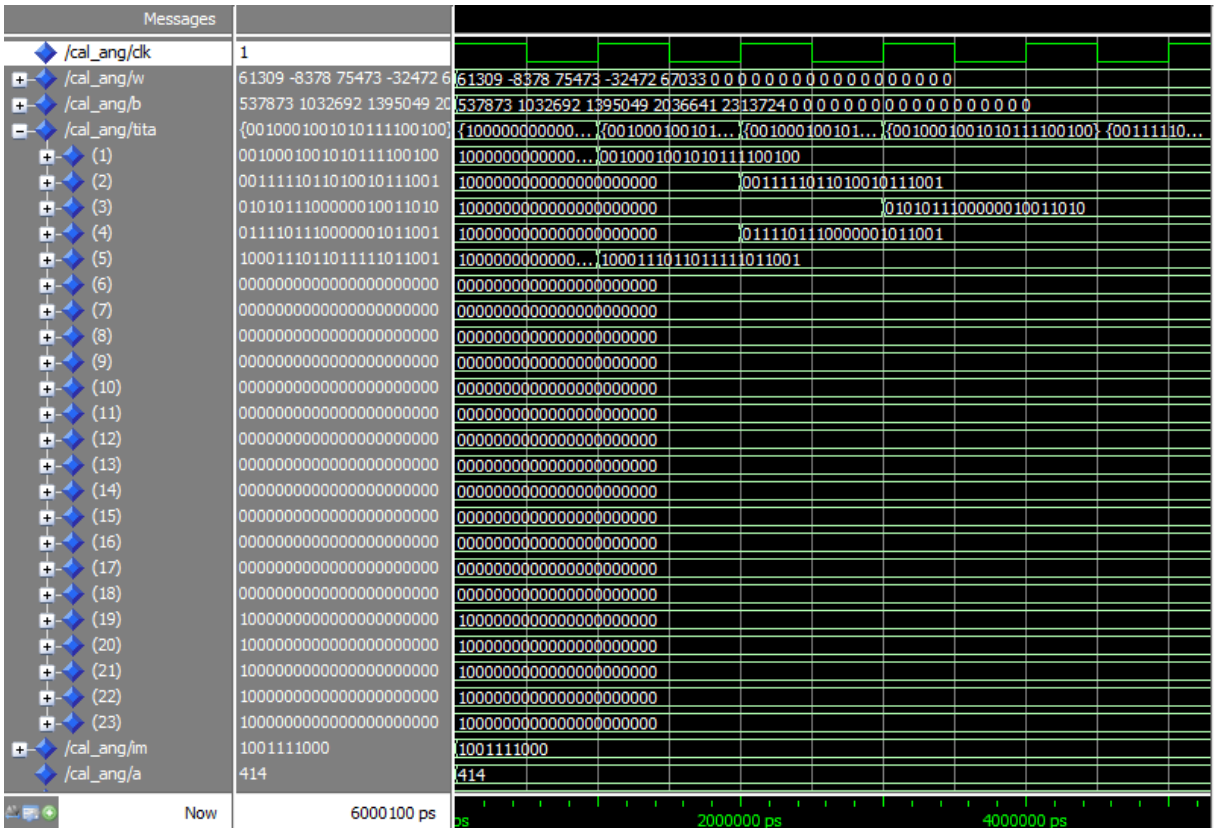


Figure 5.12 : Simulation du module " instants de commutation " sous Modelsim por im=1001111000 et m=5

8- Génération des signaux PWM

Le but de ce module est de générer les signaux PWM s1, s2 et s3 à partir des instants θ_i calculés dans l'étape précédente. Au début, le signal s1 est à 1 et son accumulateur de temps "cm1" commence à 0, puis à chaque front on l'incrémente par im et on le compare avec θ_1. Lorsque "cm1" devient supérieur à θ_1, on inverse s1 et on commence à comparer "cm1" avec θ_2 et ainsi de suite jusqu'à ce que "cm1" devient supérieur à θ_m. Après on commence à comparer "cm1" avec $\theta_{T/2}-\theta_m$, où $\theta_{T/2}=111010000100100000000$ la valeur de θ correspond à la demi période ($\alpha=180^\circ$). Lorsque "cm1" devient supérieur à $\theta_{T/2}-\theta_m$, on inverse s1 et on commence à comparer "cm1" avec $\theta_{T/2}-\theta_{m-1}$ et ainsi de suite jusqu'à que "cm1" devient supérieur à $\theta_{T/2}-\theta_1$. Après on compare "cm1" avec $\theta_{T/2}$. Lorsque "cm1" devient supérieur à $\theta_{T/2}$, on inverse s1 et on met "cm1" à 0 et on répète le processus, Figure 5.13.

D'après la base de données utilisée durant l'apprentissage pour tous les réseaux ANN-i, on a remarqué que $\theta_m<60°$. Aussi, on sait que s2 est déphasé de 120° par rapport à s1, et d'après la figure 3.5, on voit que après 120° le signal s1=0 et l'instant de commutation suivant se trouve

à $\theta_{T/2} - \theta_m$. En conséquence, au début le signal s2 démarre à 0, son accumulateur de temps "cm2" commence à $\theta_{T/3} = 1010001011\ 0000101010\ 101$ et "cm2" sera comparé avec l'instant de commutation $\theta_{T/2} - \theta_m$ et on répète le même processus de génération de s1, Figure 5.13.

Pour le signal s3, on sait qu'il est diphasé de 240° par rapport à s1, et d'après la figure 3.5 on voit que après 240° le signal s1=1 et l'instant de commutation suivant se trouve à $\theta_{T/2} - \theta_m$. En conséquence, au début le signal s3 démarre à 1, son accumulateur de temps "cm3" commence à $\theta_{T/6} = 0101000101\ 1000010101\ 011$ et "cm3" sera comparé avec l'instant de commutation $\theta_{T/2} - \theta_m$ et on répète le même processus de génération de s1, Figure 5.13.

D'après la Figure 5.13, les trois signaux sont générés parallèlement et sont indépendants l'un par rapport à l'autre ce qui confirme l'utilité d'utilisation d'un circuit FPGA pour générer un signal PWM triphasé.

9- Le diviseur de fréquence

Dans la conception proposée on a travaillé avec une horloge 1μs. Cependant, sur la carte de développement SPARTAN-3E, utilisée pour implémenter la commande proposée, on trouve une horloge de 50 μs. De ce fait, le but de ce module est la division de la fréquence d'entrée par 50.

La figure 5.14 montre une simulation du module " Le diviseur de fréquence ", on voit bien que la fréquence de sortie est de1μs.

10- Chargement des paramètres du réseau

Dans les sections précédentes on a défini tous les paramètres de l'algorithme proposé qui doivent être stockés dans le circuit FPGA. De ce fait, le but de ce module est le chargement des paramètres du réseau de neurone ANN-i sélectionné.

La figure 5.15 montre une simulation du module " Chargement des paramètres du réseau" pour les six réseaux ANN-i.

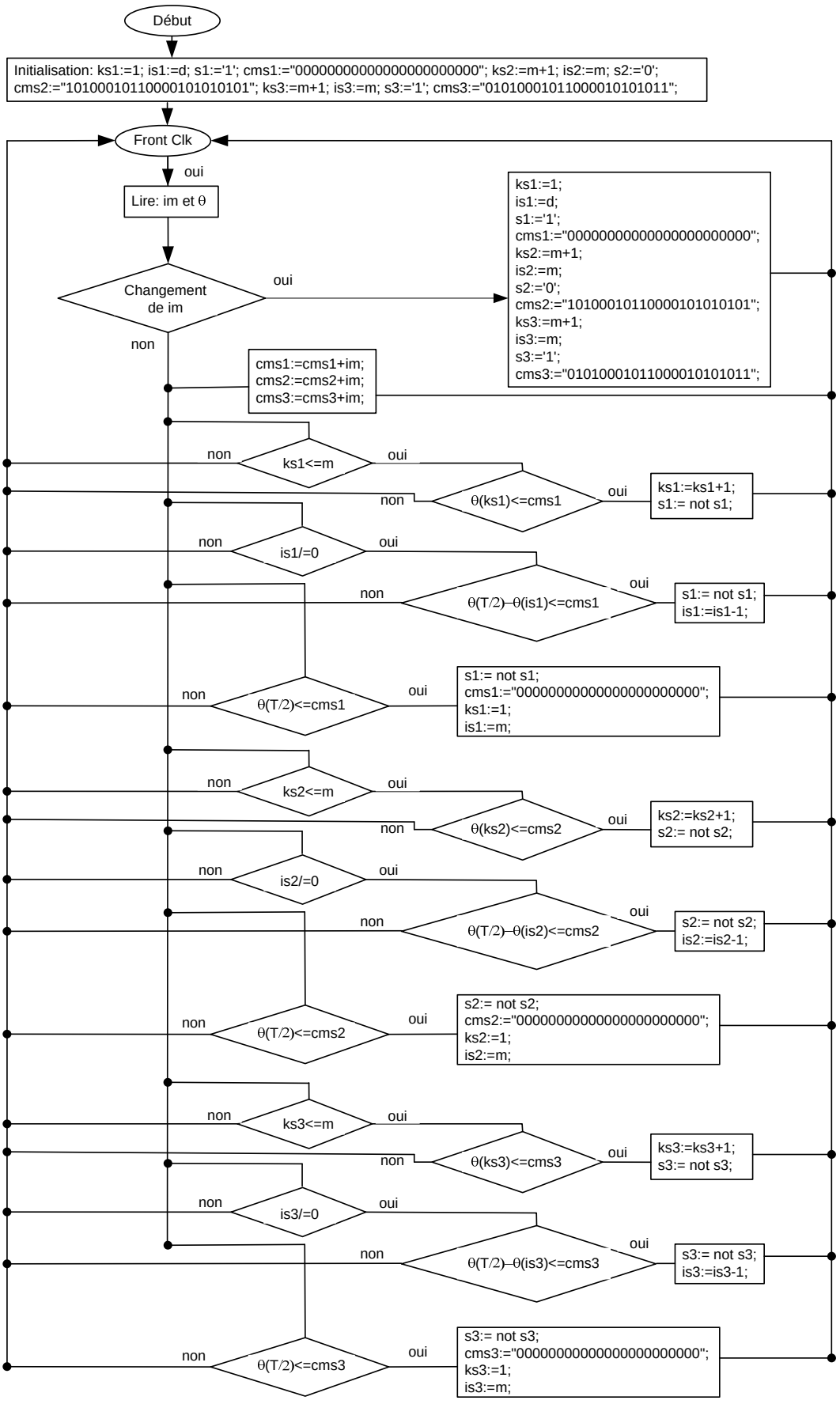


Figure 5.13 : Organigramme de la conception du module " Génération des signaux PWM"

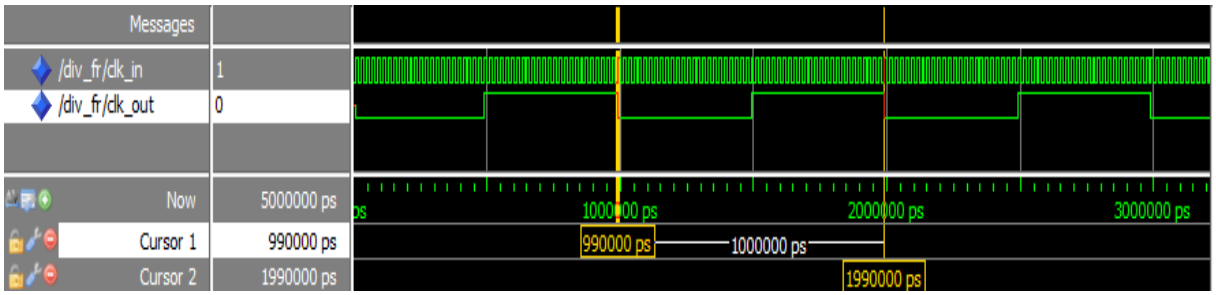


Figure 5.14 : Simulation du module " Le diviseur de fréquence " sous Modelsim

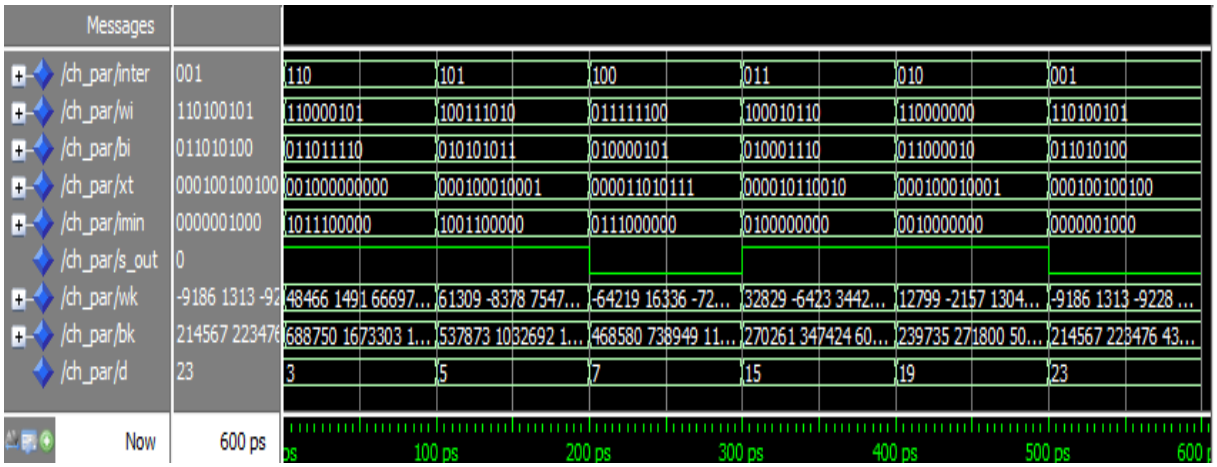


Figure 5.15 : Simulation du module " Chargement des paramètres du réseau" sous Modelsim

11- Simulation de l'implémentation de l'algorithme ANNSHE PWM sur FPGA

Dans les sections présidentes, on a expliqué le code VHDL de chaque partie du l'algorithme ANNSHE PWM. Ainsi dans cette section on montre la simulation de tout l'algorithme sur Modelsim.

D'après la figure 5.1, l'algorithme ANNSHE PWM possède deux entrées qui sont l'indice de modulation im et l'horloge clk_in, et trois sorties qui représentent les trois commandes PWM déphasées de 120°.

La figure 5.16 montre la simulation sous Modelsim de l'implémentation de l'algorithme ANNSHE PWM sur FPGA pour différentes valeurs de im. Ainsi, La figure 5.17 montre la simulation de l'algorithme ANNSHE PWM pour im=40%.

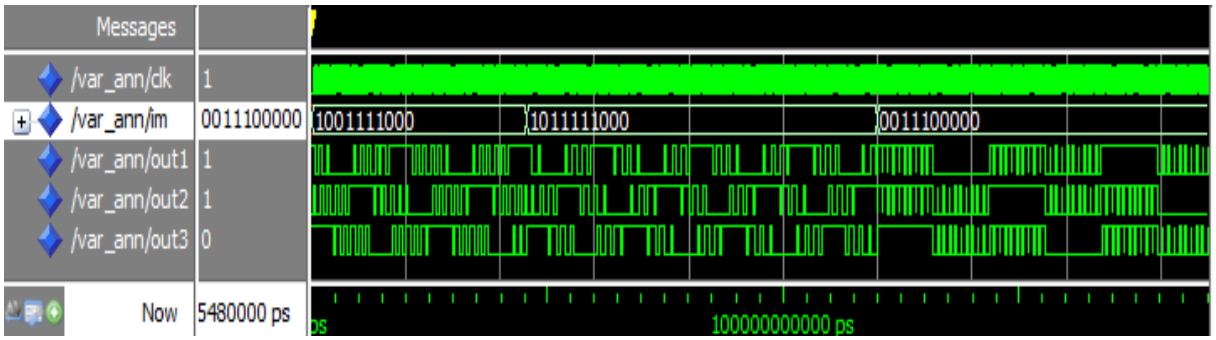


Figure 5.16 : Simulation de l'algorithme ANNSHE PWM pour différentes valeurs de im

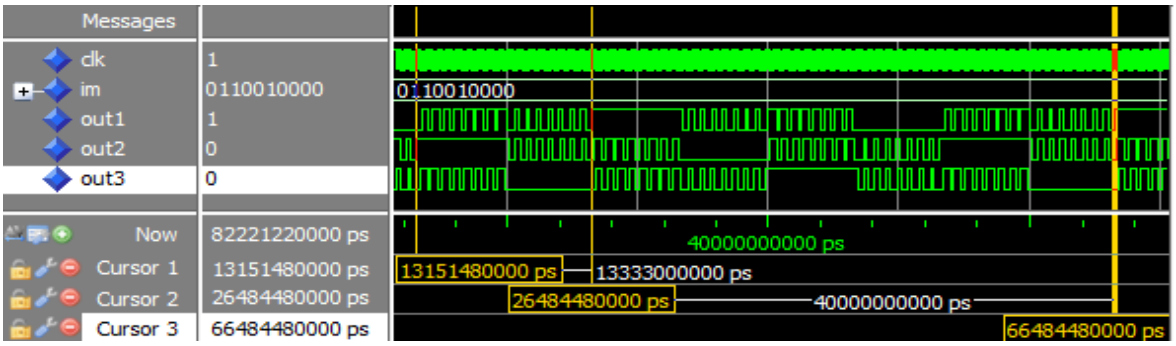


Figure 5.17 : Simulation de l'algorithme ANNSHE PWM pour im=50% (f=25Hz ou T=40ms).

D'après la figure 5.16, on voit bien que les trois signaux sont générés parallèlement et sont indépendants l'un par rapport à l'autre, de plus d'après la figure 5.17 on remarque que les signaux sont déphasé de 120°.

12- Implémentation sur FPGA

Pour implémenter l'algorithme proposé sur un circuit FPGA on a utilisé la carte de développement Spartan 3E de Digilent (voir chapitre IV section 3).

Sur cette carte, on a utilisé l'horloge de 50 MHz qui se trouve sur le pin C9 du circuit FPGA pour générer l'horloge d'entrée "clk". Ainsi, pour générer les quatre premiers bits de l'entrée im on a utilisé les quatre commutateurs "sw_i" pour les quatre premiers bits, le connecteur d'extension J2 pour générer les quatre bits suivants, un pin du connecteur d'extension J1 et un pin du connecteur d'extension J4 pour générer les deux bits qui restent. De plus on a utilisé les deux connecteurs d'extension J1 et J4 pour générer les trois signaux de commandes PWM et ses complémentaires.

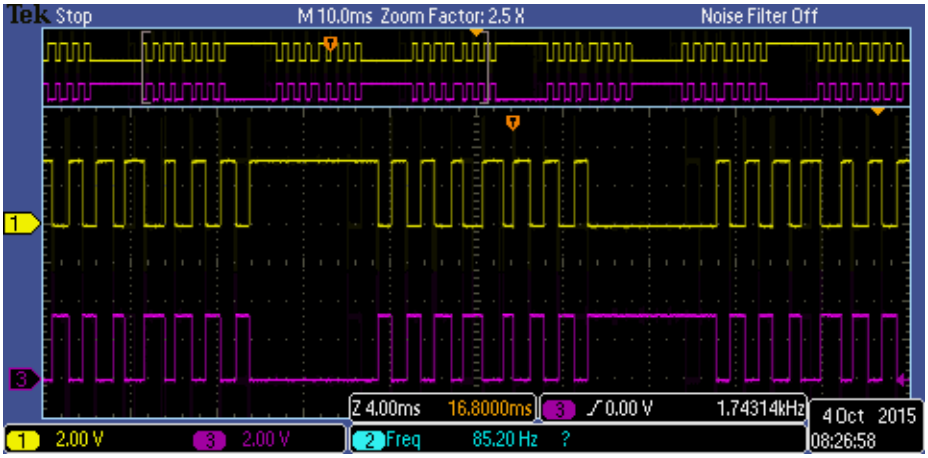
Dans le tableau 5.9, on montre pour chaque entrée et sorti de l'algorithme le pin utilisé du circuit FPGA.

Tableau 5.9 : Les pins du circuit FPGA utilisés pour implémenter l'algorithme

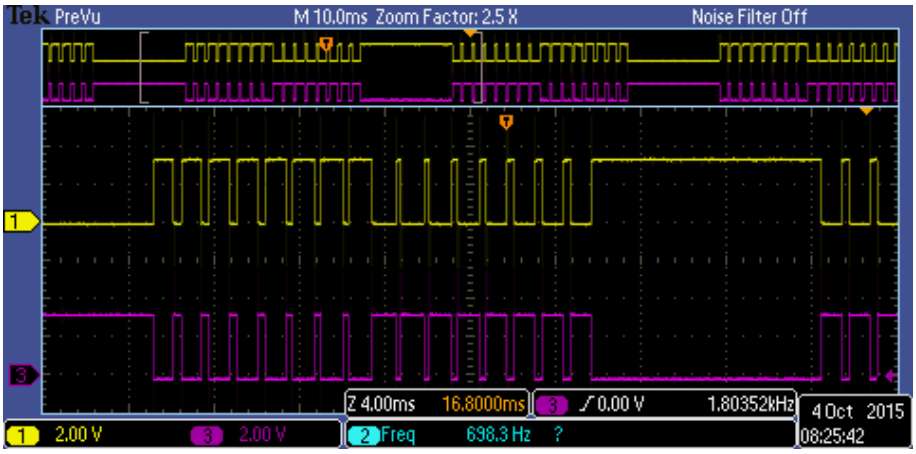
Entrées\ sorties	Pin
im(10)	N17
im(9)	H18
im(8)	L14
im(7)	L13
im(6)	F7
im(5)	E7
im(4)	B6
im(3)	A6
im(2)	C5
im(1)	E8
Out1	D7
Out2	C7
Out3	F8
Not out1	B4
Not out2	A4
Not out3	D5
clk	C9

12-1 Résultats

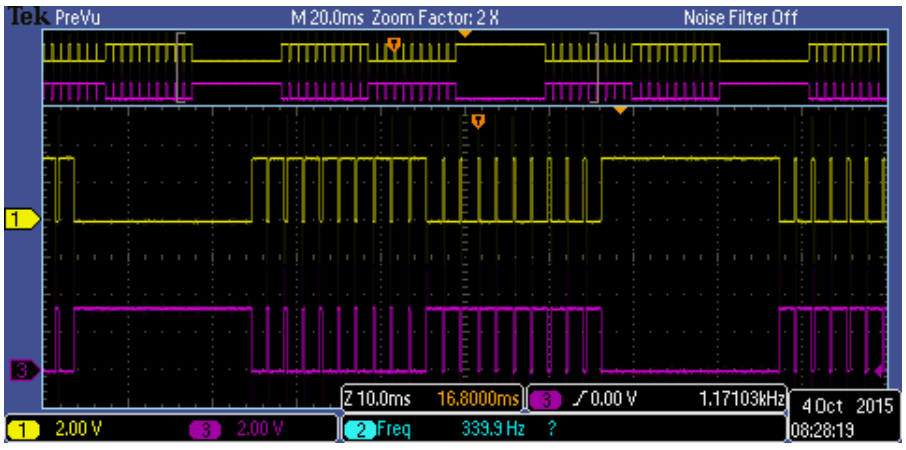
Pour visualiser les signaux de commande ANNSHE PWM, on a utilisé l'oscilloscope Tektronix MSO 2024B. Ainsi, les figures 5.18 et 5.19 montrent la visualisation de ces signaux pour différentes valeurs de im.



im=64% (T=31.25ms)

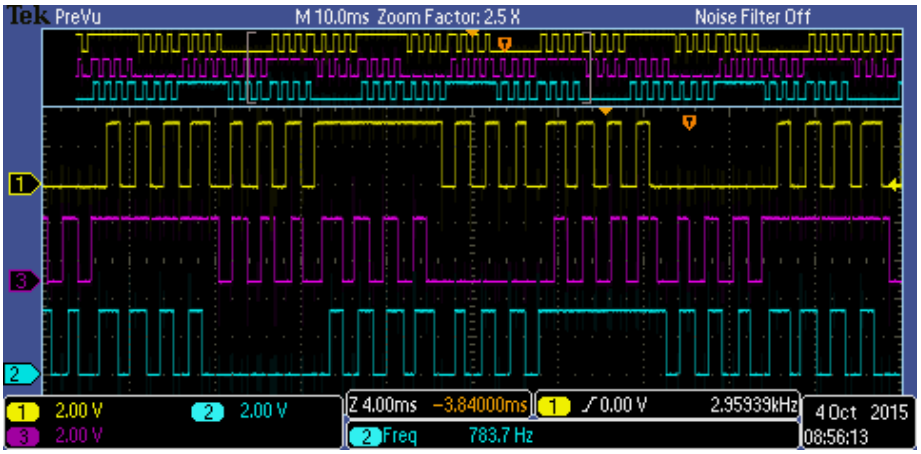


im=32% (T=62.5ms)

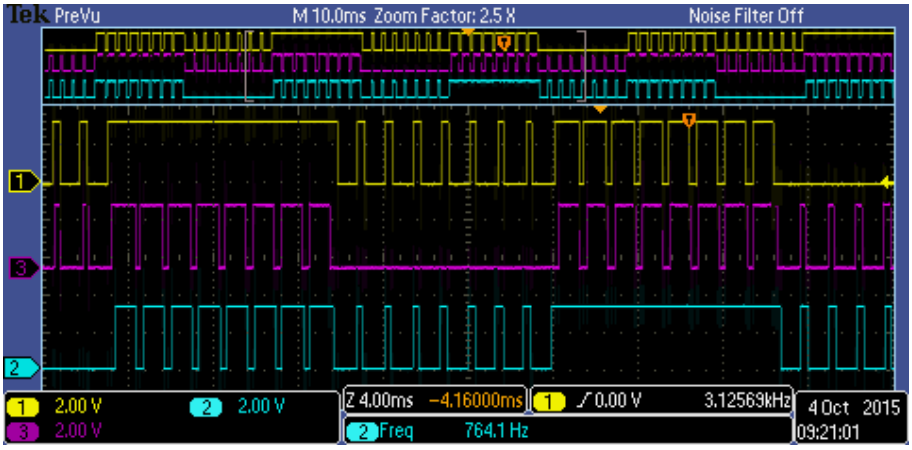


im=16% (T=125ms)

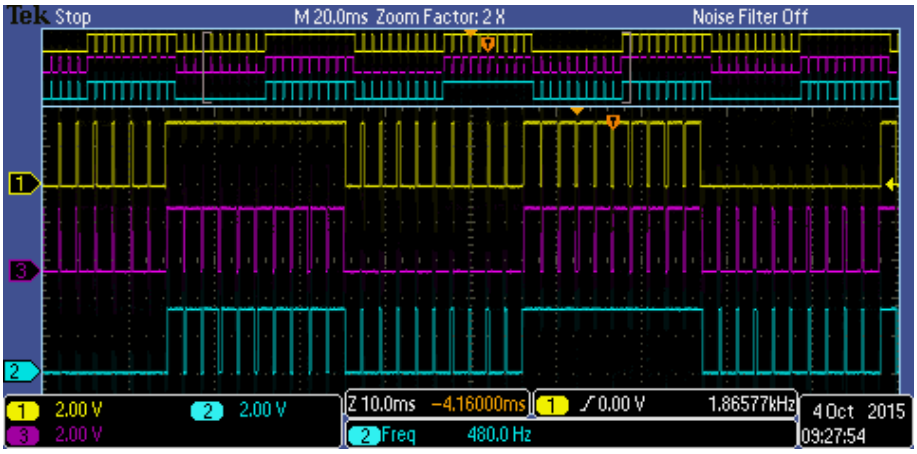
Figure 5.18 : Visualisation d'un signal ANNSHE PWM et son complémentaire sur l'oscilloscope pour différentes valeurs de im



im=64% (T=31.25ms)



im=32% (T=62.5ms)



im=16% (T=125ms)

Figure 5.19 : Visualisation des signaux ANNSHE PWM sur l'oscilloscope pour différentes valeurs de im

12-1 Interprétation

Des figures 5.18 et 5.19 on peut vérifier la fréquence des signaux ainsi que le nombre d'angles de commutation. De plus sur la figure 5.18, on voit bien que les deux signaux sont complémentaires. La figure 5.19 montre aussi que les trois signaux sont déphasés de 120°.

13- Validation expérimentale

Dans le but de valider les résultats de simulation et d'implémentation FPGA et de tester l'efficacité de l'algorithme proposé, on l'a appliqué à la commande de la vitesse d'une machine asynchrone triphasée pilotée par un onduleur de tension triphasé.

Le banc d'essais (figure 5.20) comporte les éléments suivants :

1. Une alimentation continue pour alimenter l'onduleur de tension.
2. Une alimentation stabilisée pour alimenter les différents circuits de l'étage de commande de l'onduleur (les optocoupleurs et les drivers).
3. Un oscilloscope pour visualiser les signaux.
4. Une carte de développement FPGA pour implémenter l'algorithme.
5. Un onduleur de tension triphasé pour piloter la machine asynchrone.
6. Une machine asynchrone triphasée.

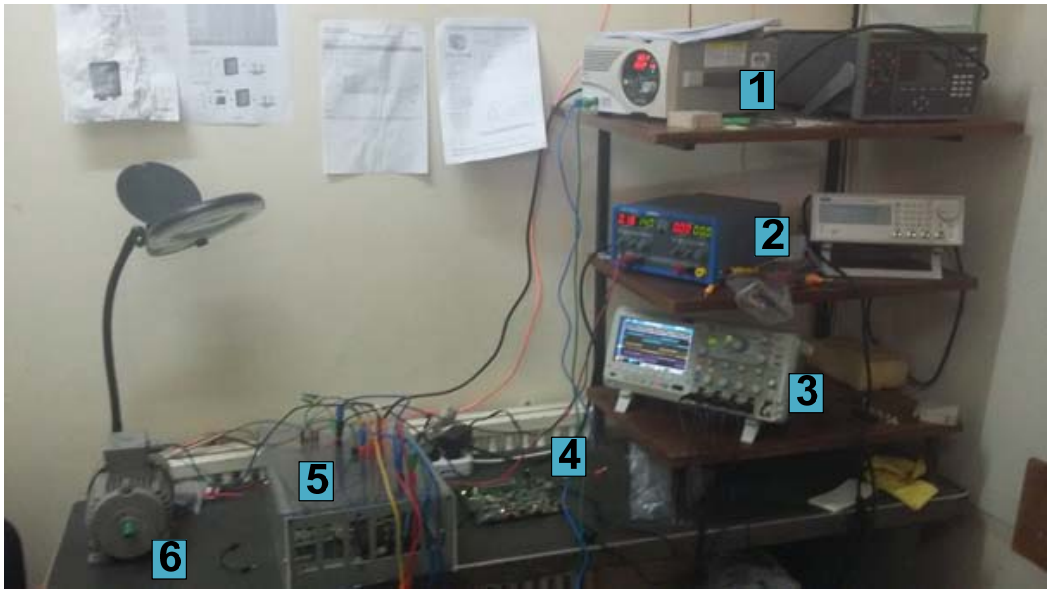
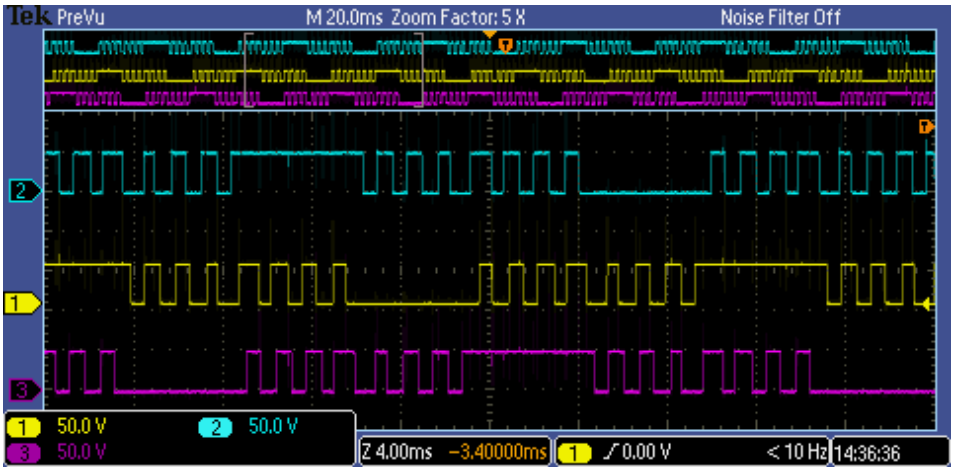


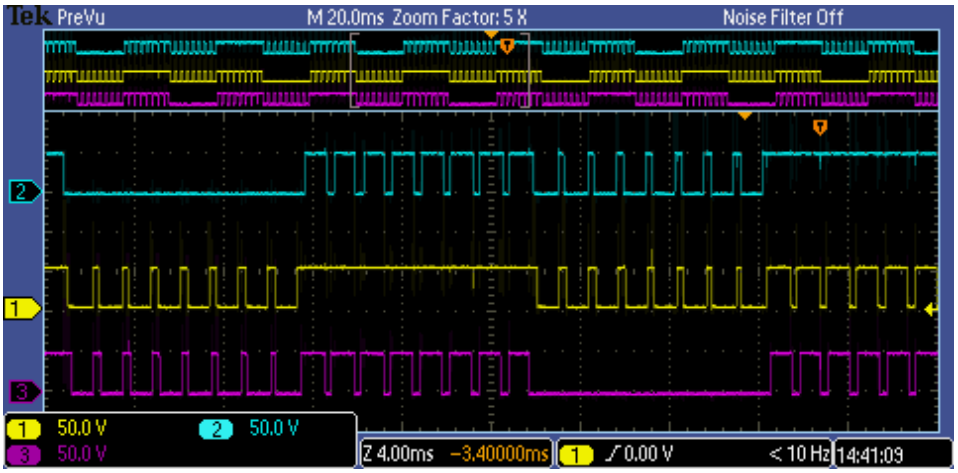
Figure 5.20 : Le banc d'essais

13-1 Résultats expérimentaux

La figure 5.21 montre la visualisation sur l'oscilloscope des tensions de phases de l'onduleur pour différentes valeur de im. La figure 5.22 montre la visualisation des tensions entre phases de l'onduleur pour différentes valeurs de la tension d'alimentation de l'onduleur. Ainsi, pour valider l'efficacité de l'algorithme proposé par rapport à l'élimination des harmoniques sélectionnés, on a enregistré les tensions de phases de l'onduleur pour différentes valeurs de im, ensuite on a fait une analyse spectrale de ces tensions (figure 5.23).

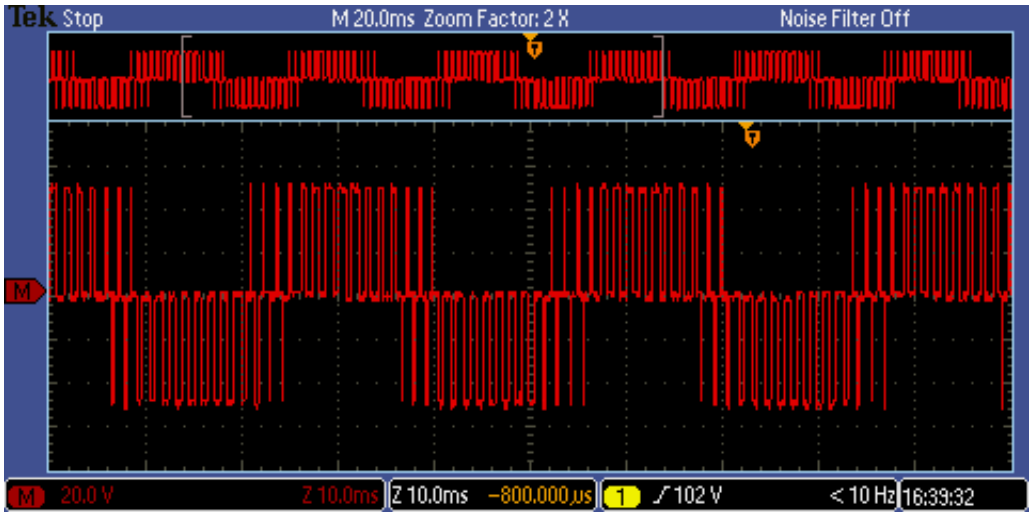


(a) im=64% (T=31.25ms)

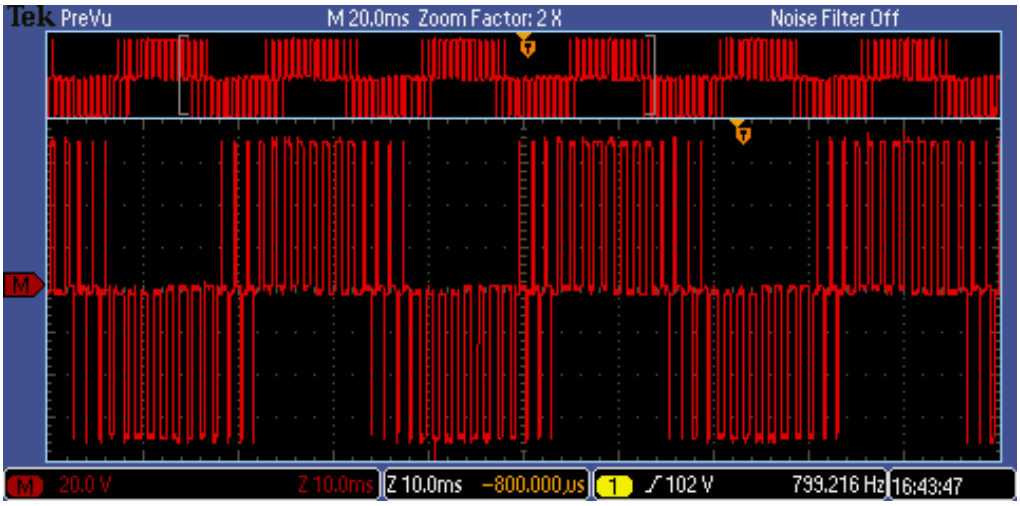


(b) im=32% (T=62.5ms)

Figure 5.21 : Visualisation des tensions de phases de l'onduleur pour (a) : im=64% (T=31.25ms) et (b) : im=32% (T=62.5ms)

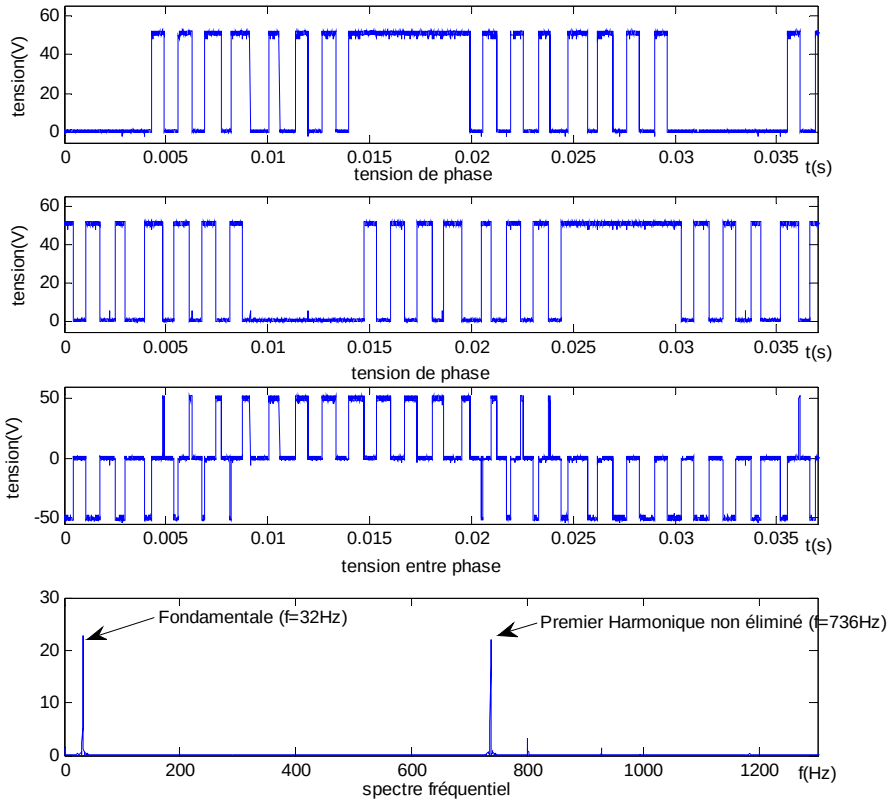


(a) La tension d'entrée =50VDC

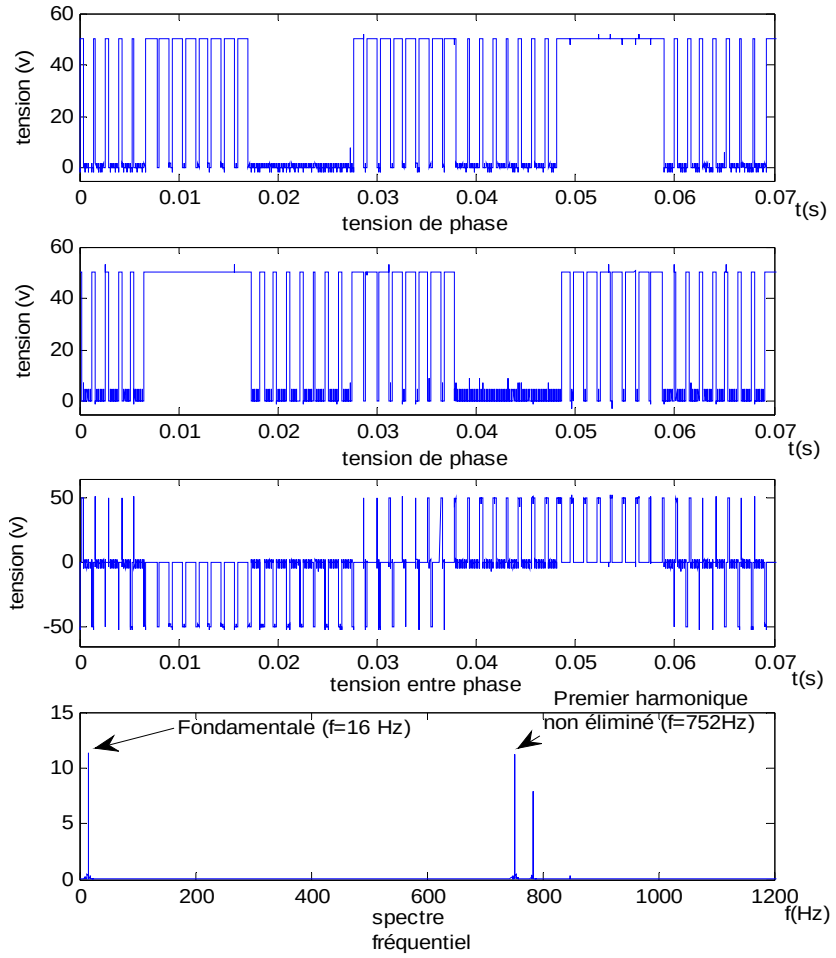


(b) La tension d'entrée =70 VDC

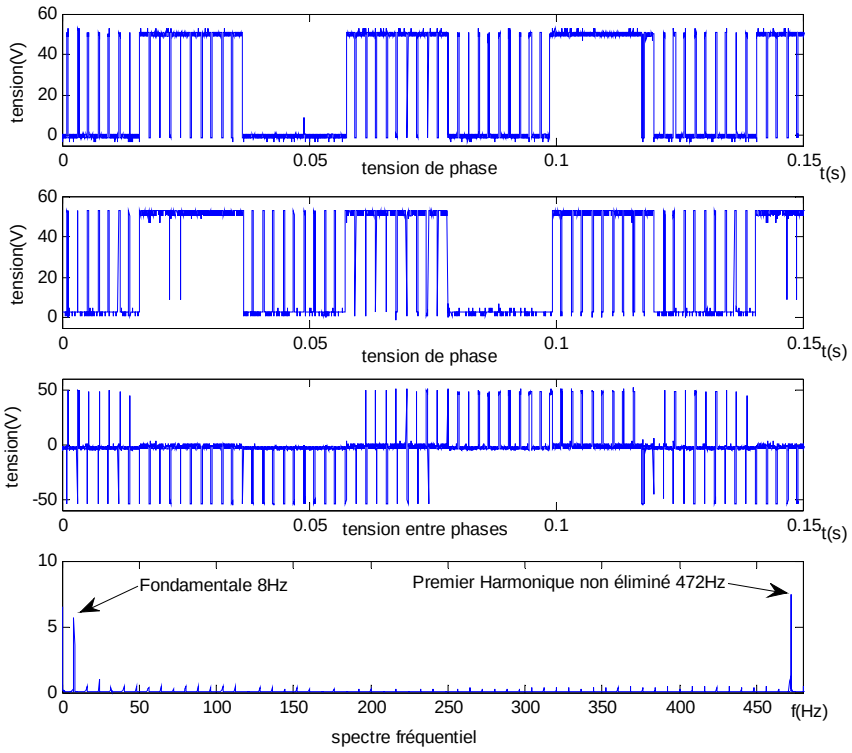
Figure 5.22 : Visualisation de la tension entre deux phases de l'onduleur pour im=64% (T=31.25ms), (a) : La tension d'entrée =50VDC et (b) : La tension d'entrée =70 VDC



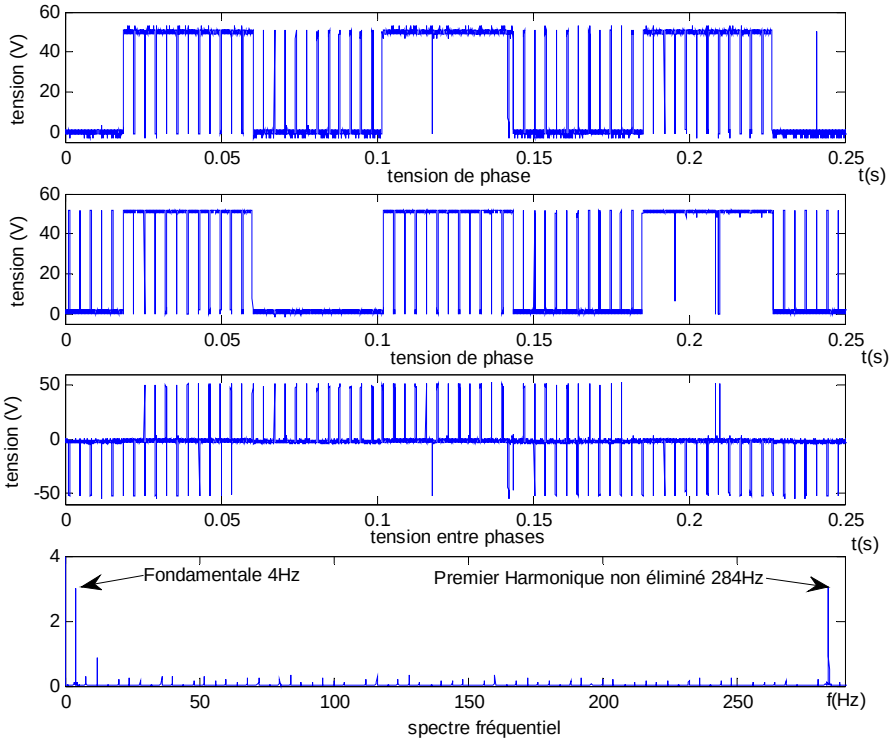
(a) : im=64% (T=31.25ms)



(b) : im=32% (T=62.5ms)



(c) : im=16% (T=125ms)



(d) : im=8% (T=250ms)

Figure 5.23 : Les tensions de phases de l'onduleur, la tension entre phase correspondante et le spectre fréquentiel de cette tension entre phases pour une tension d'entrée de 50 VDC, (a) : im=64% (T=31.25ms), (b) : im=32% (T=62.5ms), (c) : im=16% (T=125ms)et (d) : im=8% (T=250ms).

13-2 Interprétation

Les figures 5.21 et 5.23 montrent que les tensions de phases obtenues sont périodiques et de période 31.25ms pour im = 64%, 62.5ms pour im = 32%, 125ms pour im=16% et 250ms pour im=8%. Ainsi, on voit clairement l'existence de sept, quinze, dix-neuf et vingt-trois angles de commutation par quart de période pour im=64%, im = 16%, im = 32% et im = 8% respectivement. De plus, de ces figures on peut vérifier que les trois tensions de phases de l'onduleur sont déphasées de 120°.

De la figure 5.22 on peut vérifier que la tension entre phases varie entre " -E " et "+E", où "E" est l'alimentation continue de l'onduleur. Aussi, on constate que la période est égale à 31.25ms pour im=64%.

L'analyse spectrale de la tension entre phases dans la figure 5.23 (a) montre que le premier harmonique non éliminé pour im=64% (f=32 Hz et m=7) est le $23^{ème}$ (23*32=736 Hz), c'est-à-dire les six premiers harmoniques (5, 7, 11, 13, 17et 19) sont éliminés. Aussi, l'analyse spectrale de la tension entre phases dans la Figure 5.23 (b) montre que le premier harmonique non éliminé pour im=32% (f=16 Hz et m=15) est le $47^{ème}$ (47*16=752 Hz), c'est-à-dire les

quatorze premiers harmoniques (5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, 37, 41 et 43) sont éliminés. Ainsi, l'analyse spectrale de la tension entre phases dans la Figure 5.23 (c) montre que le premier harmonique non éliminé pour im=16% (f=8 Hz et m=19) est le $59^{ème}$ (59*8=472 Hz), c'est-à-dire les dix-huit premiers harmoniques (5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, 37, 41, 43, 47, 49, 53 et 55) sont éliminés. L'analyse spectrale de la tension entre phases dans la Figure 5.23 (d) montre aussi que le premier harmonique non éliminé pour im=8% (f=4 Hz et m=23) est le $71^{ème}$ (71*4=284 Hz), c'est-à-dire les vingt-deux premiers harmoniques (5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, 37, 41, 43, 47, 49, 53, 55, 59, 61, 65 et 67) sont éliminés . De plus, les figures 5.23 (a), (b), (c) et (d), montrent que les harmoniques multiples de trois sont éliminées automatiquement. De l'analyse spectrale on voit bien que l'amplitude de la tension fondamentale augmente avec im.

En conséquence, les résultats expérimentaux confirment l'efficacité et la précision de l'algorithme ANNSHE PWM proposé dans l'élimination des harmoniques sélectionnés et dans le contrôle de la tension fondamentale.

14- Conclusion

Dans ce chapitre, nous avons présenté les étapes d'implémentation de l'algorithme ANNSHE PWM sur un circuit FPGA. Nous avons insisté sur l'optimisation de cette implémentation vue l'importance de cette dernière et son impact sur les performances de notre système.

Le but de cette optimisation est de minimiser l'espace consommé sur le circuit FPGA et d'augmenter la précision de calcul des instants de commutation et la vitesse de génération des signaux PWM.

Pour atteindre cet objectif, nous avons procédé à des optimisations dans chaque étape de l'implémentation. D'abord, on a dimensionné l'indice de modulation im sur 10 bits afin d'augmenter la précision de la valeur de vitesse, c'est à dire le pas de variation est de 0.125%. De plus, pour générer les signaux PWM en temps réel on a travaillé avec une horloge de 1MHz.

Aussi, les limites de chaque intervalle de im pour chaque réseau sont choisies de telles sortes que la sélection du réseau en fonction de im soit simple. Également, Pour l'implémentation de la fonction tangente sigmoïde, on a opté pour la méthode à base de LUT optimisée par Maher qui nécessite 15 valeurs à stocker dans une LUT pour une erreur de 0,02. Aussi, lors du calcul des instants de commutation, on a trouvé une solution pour éviter l'opération de division et on a établi un procédé pour utiliser deux multiplicateurs au lieu de 23.

Lors de la génération des signaux, on a trouvé une manière pour comparer chaque accumulateur de temps avec un seul instant de commutation au lieu de le comparer avec tous

les instants de commutation en même temps. De plus les trois signaux sont générés parallèlement et sont indépendants l'un par rapport à l'autre.

Enfin, dans ce chapitre, on a validé l'efficacité et la précision de l'algorithme proposé par des essais expérimentaux qui confirment son efficacité et sa précision dans l'élimination des harmoniques sélectionnés et dans le contrôle de la tension fondamentale.

Conclusion générale

Dans cette thèse, la commande en tension et en fréquence variables d'un onduleur triphasé destiné pour commander une machine asynchrone dans un véhicule électrique a été réalisée par la proposition d'un nouvel algorithme ANNSHE PWM basé sur la théorie des réseaux de neurones artificiels et la commande SHE PWM. Cet algorithme a été validé par simulation, puis par une implémentation sur un circuit FPGA et enfin par des essais pratiques pour la commande de vitesse d'un moteur asynchrone.

En effet, la commande de la vitesse d'un moteur asynchrone nécessite l'utilisation d'un onduleur de tension avec une sortie sinusoïdale variable en tension et en fréquence. Dans la pratique, les stratégies de commande produisent des harmoniques indésirables dans la sortie de l'onduleur. Pour résoudre le problème des harmoniques indésirables, une solution bien connue est l'utilisation de la commande SHE PWM. Cependant l'utilisation de cette commande nécessite la résolution d'un système d'équations non-linéaire limitant cette stratégie de contrôle au calcul off-line. De ce fait, l'utilisation de cette technique dans une application EV qui nécessite des changements fréquents dans la fréquence et la tension est difficile et pas efficace.

Pour pallier cet inconvénient, nous avons proposé, dans cette thèse, une nouvelle technique basée sur la commande SHE PWM et le principe des réseaux de neurones artificiels.

Pour assurer la convergence dans le calcul des paramètres des réseaux de neurones artificiels et pour augmenter aussi la précision et l'efficacité de l'algorithme on a divisé l'intervalle de variation de l'indice de modulation en six intervalles. De plus le nombre d'harmoniques à éliminer dans chaque intervalle est choisi de telle sorte que celui-ci augmente lorsque la fréquence diminue.

Les résultats de simulation montrent que les angles de commutations calculés par l'algorithme ANNSHE PWM sont très proches de ceux calculés off-line par l'algorithme itératif de Newton-Raphson. De plus, ils montrent l'élimination effective des harmoniques sélectionnés et l'efficacité du contrôle de la tension fondamentale à la sortie de l'onduleur.

Les différentes étapes pour réaliser, optimiser et implémenter l'algorithme ANNSHE PWM sur un circuit FPGA ont été aussi détaillées.

Cette optimisation a permis de minimiser l'espace consommé sur le circuit FPGA et d'augmenter la précision de calcul des instants de commutation et la vitesse de génération des signaux PWM.

Lors de l'implémentation on a dimensionné l'indice de modulation sur 10 bit avec un pas de variation de 0.125% pour augmenter la précision sur la vitesse. De plus, les signaux PWM sont générés en temps réel avec une horloge de 1MHz.

Pour implémenter la fonction tangente sigmoïde, on a utilisé la méthode à base de LUT optimisée par Maher qui nécessite seulement 15 valeurs à stocker dans une LUT pour une erreur de 0,02. Ainsi, pour calculer les instants de commutation, on a utilisé seulement deux multiplicateurs et on a évité l'utilisation de l'opération de division qui nécessite beaucoup d'espace et même un temps de calcul élevé.

Enfin, les résultats d'implémentation finale sur un circuit FPGA montrent aussi que le calcul des instants de commutation et la génération des signaux PWM sont en temps réel et avec une très bonne précision. Les trois signaux de commande sont générés parallèlement et sont indépendants l'un par rapport à l'autre.

Les résultats des essais pratiques, montrent aussi l'élimination effective des harmoniques sélectionnés et la précision du contrôle de la tension fondamentale à la sortie de l'onduleur. La faisabilité de la variation de la vitesse du moteur en temps réel de zéro à sa vitesse nominale a ainsi été démontrée.

En perspective à ce travail nous envisageons de finaliser l'onduleur avec toutes les protections nécessaires afin de développer tout le système d'entraînement du véhicule électrique (moteur électrique, convertisseur de puissance, régulateur de vitesse, gestion de l'énergie, unité de commande électronique etc…)

Afin de comparer l'algorithme ANNSHE PWM proposé avec d'autres algorithmes utilisés dans l'industrie, nous prévoyons aussi d'appliquer ce dernier dans d'autres applications pilotées par un onduleur et nécessitant des variations soit dans la fréquence, soit dans l'amplitude du fondamentale, soit dans les deux en même temps.

Références

[1] J. Kiviluoma and P. Meibom, “Methodology for modelling plug-in electric vehicles in the power system and cost estimates for a system with either smart or dumb electric vehicles.” Energy 2011; 36: pp. 1758-1767.

[2] Z. Du, B. Ozpineci, L. M. Tolbert and J. N. Chiasson, “DC–AC Cascaded H-Bridge Multilevel Boost Inverter with no Inductors for Electric/Hybrid Electric Vehicle Applications.” IEEE Transactions on Industry Applications May/June 2009; 45 (3): pp. 936-970.

[3] C.E. Sandy Thomas, “How green are electric vehicles?” International Journal of Hydrogen Energy 2012; 37: pp. 6053-6062.

[4] Ali Emadi, Sheldon S. Williamson, and Alireza Khaligh, “Power Electronics Intensive Solutions for Advanced Electric, Hybrid Electric, and Fuel Cell Vehicular Power Systems” IEEE Transactions on Power Electronics, May 2006; 21(3): pp. 567-577.

[5] C. H. Rivetta, A. Emadi, G. A. Williamson, R. Jayabalan and B. Fahimi, “Analysis and Control of a Buck DC–DC Converter Operating With Constant Power Load in Sea and Undersea Vehicles” IEEE Transactions On Industry Applications March/April 2006; 42 (2): pp. 559-572.

[6] Adel Khedher and Mohamed Faouzi Mimouni, “Sensorless-adaptive DTC of double star induction motor.” Energy Conversion and Management 2010; 51: pp. 2878–2892.

[7] Ilhami Colak and Ersan Kabalci, “Developing a novel sinusoidal pulse width modulation (SPWM) technique to eliminate side band harmonics” International Journal of Electrical Power and Energy Systems 2013; 44: pp. 861–871.

[8] M. Valan Rajkumar, P.S. Manoharan and A. Ravi “Simulation and an experimental investigation of SVPWM technique on a multilevel voltage source inverter for photovoltaic systems” International Journal of Electrical Power and Energy Systems 2013; 52: pp. 116–131.

[9] A. Das, K. Sivakumar, C. Patel, K. Gopakumar and R. Ramchand, “A Pulse width Modulated Control of Induction Motor Drive Using Multilevel 12-Sided Polygonal Voltage Space Vectors” IEEE Transactions On Industrial Electronics, July 2009; 56 (7): pp. 2441-2449.

[10] Z. Du and L. M. Tolbert, “Active Harmonic Elimination for Multilevel Converters.” IEEE Transactions on Power Electronics March 2006; 21 (2): pp. 459-496.

[11] Zainal Salam, “An On-Line Harmonic Elimination Pulse Width Modulation Scheme for Voltage Source Inverter.” Journal of Power Electronics January 2010; 10(1): pp. 43-50.

[12] Hasmukh S. Patel AND Richard G. Hoft, “Generalized Techniques of Harmonic Elimination and Voltage Control in Thyristor Inverters: Part I-Harmonic Elimination.” IEEE Transactions on Industry Applications May/June 1973; IA-9 (3): pp. 310-317.

[13] Hasmukh S. Patel AND Richard G., “Generaiized Techniques of Harmonic Elimination and Voltage Control in Ihyristor Inverters: Part I1-Voltage Control Techniques.” IEEE Transactions on Industry Applications September/October 1974; IA-10 (5): pp. 666-673.

[14] Alinaghi Marzoughi, Hossein Imaneini and Amirhossein Moeini, “An optimal selective harmonic mitigation technique for high power converters” International Journal of Electrical Power and Energy Systems 2013; 49: pp. 34-39.

[15] Enjeti P.N., Ziogas P.D. and Lindsay, J.F., “Programmed PWM Techniques to Eliminate Harmonics: A Critical Evaluation.” IEEE Transaction on Industry Applications 1990; 26 (2): pp. 302-316.

[16] Dahidah, M.S.A.; Agelidis, V.G., “Selective Harmonic Elimination PWM Control for Cascaded Multilevel Voltage Source Converters: A Generalized Formula”, IEEE Transactions on Power Electronics Jul 2008; 23 (4): pp.1620-1630.

[17] Agelidis, V.G.; Balouktsis, A.I.; Dahidah, M.S.A., “A Five-Level Symmetrically Defined Selective Harmonic Elimination PWM Strategy: Analysis and Experimental Validation”, IEEE Transactions on Power Electronics Jan 2008; 23 (1): pp.19-26.

[18] Taufiq, J. A., Mellitt, B. and Goodman, C. J. “Novel Algorithm for Generating Near Optimal PWM Waveforms for AC Traction Drives” Electric Power Applications, IEE Proceedings B 1986; 133 (2): pp. 85-94.

[19] S. R. Bowes and P. R. Clark, “Simple Microprocessor Implementation of New Regular-Sampled Harmonic Elimination PWM Techniques”, IEEE Transaction on Industry Applications 1992; 28 No.1, pp. 89-95.

[20] Bowes, S.R., Grewal, S., “Novel space-vector-based harmonic elimination inverter control”, IEEE Transactions on Industry Applications Mar-Apr 2000; 36 (2): pp. 549-557.

[21] Agelidis V. G., Balouktsis A., and Balouktsis I., “On applying a minimization technique to the harmonic elimination PWM control: The bipolar waveform,” IEEE Power Electronics Letters Jun 2004; 2 (2): pp. 41-44.

[22] Liang, T.J., O’Connell, R.M.; Hoft, R.G., “Inverter harmonic reduction using Walsh function harmonic elimination method”, IEEE Transactions on Power Electronics Nov 1997; 12 (6): pp.971-982.

[23] Kato, T., “Sequential homotopy-based computation of multiple solutions for selected harmonic elimination in PWM inverters”, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications May 1999; 46 (5): pp.586-593.

[24] Chiasson, J.N.; Tolbert, L.M.; McKenzie, K.J.; Zhong Du.,“Elimination of harmonics in a multilevel converter using the theory of symmetric polynomials and resultants”, IEEE Transactions on Control Systems Technology Mar 2005; 13 (2): pp. 216-223.

[25] Tolbert, L.M.; Chiasson, J.N.; Zhong Du; McKenzie, K.J.,“Elimination of harmonics in a multilevel converter with no equal DC sources”, IEEE Transactions on Industry Applications Jan-Feb 2005; 41(1): pp. 75-82.

[26] Ozpineci, B.; Tolbert, L.M.; Chiasson, J.N., “Harmonic optimization of multilevel converters using genetic algorithms”, IEEE Power Electronics Letters Sep 2005; 3 (3): pp.92-95.

[27] Mohamed S.A. Dahidah, Vassilios G. Agelidis and Machavaram V. Rao, “Hybrid genetic algorithm approach for selective harmonic control”, Energy Conversion and Management 2008, 49: pp. 131-142.

[28] A.K. Al-Othman , Nabil A. Ahmed, M.E. AlSharidah, Hanan A. AlMekhaizim, “A hybrid real coded genetic algorithm – Pattern search approach for selective harmonic elimination of PWM AC/AC voltage controller” Electrical Power and Energy Systems 2013; 44:123-133.

[29] Khider M, "commande de vitesse en temps réel d’un moteur asynchrone triphasé " mémoire de Magister 2003, Ecole Nationale Polytechnique, Algiers, Algeria.

[30] BENDIB Douadi "Etude et réalisation d’une commande MLI on-line sur circuit FPGA" mémoire de Magister 2009, Ecole Nationale Polytechnique, Algiers, Algeria.

[31] Guellal Amar "Implémentation sur FPGA d’une commande MLI on-line basée sur le principe des réseaux de neurones" mémoire de Magister 2010, Ecole Nationale Polytechnique, Algiers, Algeria.

[32] Claude Touzet, "Les réseaux de neurones artificiels : introduction au connexionnisme". Laboratoire d'Etudes et Recherche en Informatique, ISBN 2 - 906 899 - 78X. Juillet 1992.

[33] G. Bosque a,n, I.delCampo b,nn, J.Echanobe "Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades" Engineering Applications of Artificial Intelligence32(2014)283–331

[34] N. Izeboudjen , C. Larbes, A. Farah "A new classification approach for neural networks hardware: from standards chips to embedded systems on chip" Artif Intell Rev (2014) 41:491–534

[35] Janardan Misra, IndranilSaha " Artificial neural networks in hardware: A survey of two decades of progress" Neurocomputing 74 (2010) 239–255

[36] S. L. Pinjare, Arun Kumar M. "Implementation of Neural Network Back Propagation Training Algorithm on FPGA" International Journal of Computer Applications (0975 – 8887) Volume 52– No.6, August 2012

[37] S. Himavathi, D. Anitha, and A. Muthuramalingam "Feedforward Neural Network Implementation in FPGA Using Layer Multiplexing for Effective Resource Utilization" IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 18, NO. 3, MAY 2007

[38] Ang Li, Qin Wang, Zhancai Li and Yong Wan "A Reconfigurable Approach to Implement Neural Networks for Engineering Application" Proceedings of the 6th World Congress on Intelligent Controland Automation, June 21 - 23, 2006, Dalian, China

[39] R.J.Aliaga, R.Gadea, R.J.Colom, J.M.Monzó, C.W.Lerche, J.D.Martínez, A.Sebastiá, F.Mateo "Multiprocessor SoC Implementation of Neural Network Training on FPGA" International Conference on Advances in Electronics and Micro-electronics IEEE 2008

[40] Nadia Nedjah, Rodrigo Martins da Silva a, Luiza de Macedo Mourelle "Compact yet efficient hardware implementation of artificial neural networks with customized topology" Expert Systems with Applications 39 (2012) 9191–9206

[41] L. M. Reyneri, "Implementation issues of neuro-fuzzy hardware: Going towards HW/SW codesign," IEEE Trans. Neural Netw., vol.14, no. 1, pp. 176–194, Jan. 2003.

[42] M. Cristea and A. Dinu, "A new neural network approach to induction motor speed control," in Proc. IEEE Power Electron. Specialist Conf., 2001, vol. 2, pp. 784–788.

[43] Babak Zamanlooy, and Mitra Mirhassani "Efficient VLSI Implementation of Neural Networks with Hyperbolic Tangent Activation Function", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 22, NO. 1, January 2014, pp. 39-48.

[44] K. Leboeuf, A. Namin, R. Muscedere, H. Wu, and M. Ahmadi, "High speed VLSI implementation of the hyperbolic tangent sigmoid function," in Proc. 3rd Int. Conf. Converg. Hybrid Inf. Technol. Nov. 2008, pp. 1070–1073.

[45] A. Namin, K. Leboeuf, R. Muscedere, H. Wu, and M. Ahmadi, "Efficient hardware implementation of the hyperbolic tangent sigmoid function," in Proc. IEEE Int. Symp. Circuits Syst., May 2009, pp. 2117–2120.

[46] P. Meher, "An optimized lookup-table for the evaluation of sigmoid function for artificial neural networks," in Proc. 18th IEEE/IFIP VLSI Syst. Chip Conf., Sep. 2010, pp. 91–95.

[47] C. W. Lin and J. S. Wang, "A digital circuit design of hyperbolic tangent sigmoid function for neural networks," in Proc. IEEE Int. Symp. Circuits Syst. Conf., May 2008, pp. 856–859.

[48] N. Abdelelah. "architectures pour la stereo-vision passive dense temps réel : application a la stereo-endoscopie." Thèse Doctorat, Microsystèmes et Intégration de Systèmes. Toulouse : Université PAUL SABATIER, 2006.

[49] Cherrad BENBOUCHAMA "Contribution à l'implémentation des réseaux de neurones artificiels sur hardware reconfigurable FPGA.", Thèse Doctorat, Laboratoire de Commande des Processus, Ecole Nationale Polytechnique, Algiers, Algeria Juin 2008.

[50] Eric Monmasson and Marcian N. Cirstea, "FPGA Design Methodology for Industrial Control Systems: A Review", IEEE Transactions On Industrial Electronics, VOL. 54, NO. 4, AUGUST 2007.
[51] S. Brown and J. Rose, “FPGA and CPLD Architectures: A Tutorial,” IEEE Design & Test of Computers, Vol. 13, No. 2, pp. 42-57 1996.
[52] S. Brown, “FPGA architectural research: A survey,” IEEE Des. Test. Comput., vol. 13, no. 4, pp. 9–15, Winter 1996.
[53] S. Trimberger, “A reprogrammable gate array and applications,” Proc. IEEE, vol. 81, no. 7, pp. 1030–1041, Jul. 1993.
[54] Spartan-3E FPGA Family, Data Sheet, www.xilinx.com/support/documentation/data_sheets /ds312 .pdf, july 2013.
[55] Spartan-3E FPGA Starter Kit Board User Guide, www.xilinx.com/support/documentation /boards_and_kits/ug230. pdf, 2011
[56] Eñaut Muxika Olasagasti, "Application des réseaux de neurones à l'identification d'un axe de machine-outil.", Thèse Doctorat, Engineering Sciences. Institut National Polytechnique de Grenoble - INPG, 2002. French.
[57] Izeboudjen Nouma, "Plateforme pour l’Implémentation des Réseaux de Neurones sur FPGA : Application à l’Algorithme de la Rétro Propagation du Gradient (RPG)", Thèse Doctorat, Laboratoire Signal et Communications, Ecole Nationale Polytechnique, Algiers, Algeria 2014.
[58] Samir Hamdani, "Modélisation, détection et classification des défauts rotoriques de la machine asynchrone à cage")", Thèse Doctorat, Laboratoire de Recherche en Electrotechnique Ecole Nationale Polytechnique, Algiers, Algeria 2012.
[59] W.S. McCulloch, W.Pitts, "A logical calculus of the ideas immanent in nervous activity", Bull Math Biophys, 1943 5(4): p115–133.
[60] Magali R. G. Meireles, Paulo E. M. Almeida, and Marcelo Godoy Simões, "A Comprehensive Review for Industrial Applicability of Artificial Neural Networks", IEEE Transactions on Industrial Electronics, June 2003, 50(3): pp. 585-601.
[61] Farid Khoucha, Abdelkader Khoudiri, Mohamed Benbouzid et Abdelaziz Kheloui "Commande DTC d'une propulsion moteur asynchrone /Onduleur multiniveaux asymétrique pour un véhicule électrique", European Journal of Electrical Engineering, 2011, 14 (2-3), pp.237-254.
[62] Zeraoulia M., Benbouzid M.E.H. and Diallo D, "Electric motor drive selection issues for HEV propulsion systems: A comparative study", IEEE Trans. Vehicular Technology, vol. 55, n° 6, November 2006, p. 1756-1764.
[63] Zhu Z.Q. and Howe D. "Electrical machines and drives for electric, hybrid, and fuel cell vehicles", Proceedings of the IEEE, vol. 95, n° 4, April 2007, p. 746-765.
[64] Abdul Moeed Amjad, Zainal Salam, Ahmed Majed et Ahmed Saif, "Application of differential evolution for cascaded multilevel VSI with harmonics elimination PWM switching", Electrical Power and Energy Systems 64 (2015) 447–456
[65] Abdul Moeed Amjad, Zainal Salam, " A review of soft computing methods for harmonics elimination PWM for inverters in renewable energy conversion systems", RenewableandSustainableEnergyReviews33(2014)141–153
[66] Mohamed S. A. Dahidah, Georgios Konstantinou et Vassilios G. Agelidis, " A Review of Multilevel Selective Harmonic Elimination PWM: Formulations, Solving Algorithms, Implementation and Applications", IEEE Transactions On Power Electronics, Vol. 30, NO. 8, August 2015 pp. 4091-4105
[67] Victor MESTER, " Conception Optimale Systémique des Composants des Chaînes de Traction Electrique", thèse de doctorat, Laboratoire L2EP, EA2697, Ecole Centrale de Lille, France, Mai 2007.

[68] MANSOUR Samir, "Commande par DSP TMS 320F2812 d'un onduleur MLI destiné à la traction des véhicules électriques", mémoire de Magister 2012, Ecole Nationale Polytechnique, Algiers, Algeria.

[69] Volnei A. Pedroni "Circuit Design with VHDL" MIT Press Cambridge, Massachusetts London, England, 2004

[70] Amar Guellal, Cherif Larbes, Douadi Bendib, Linda Hassaine et Ali Malek, " FPGA based on-line Artificial Neural Network Selective Harmonic Elimination PWM technique", Electrical Power and Energy Systems 68, 2015, pp. 33-43.

[71] A.Guellal, C.Larbes and L.Hassaine, "Une nouvelle approche basée sur le principe des réseaux de neurone destinée pour commander les onduleurs", Revue des Energies Renouvelables 2012 Vol. 15 N°1 : pp. 57 – 66, Bouzaréah - Algiers, Algeria.

[72] A.Guellal, C.Larbes and L.Hassaine, " An online PWM algorithm based on the HEVC PWM and the Aftificial Neural Network", The First International Conference on Power Electronics and their Applications ICPEA 2013, 06 et 07 November 2013 , University of Djelfa, Algeria.

[73] A.Guellal, C.Larbes and L.Hassaine, " A new on-line Pulse Width Modulation control based on the principle of neural networks", The First International Conference on Electrical Energy and Systems ICEES'13 22-24 October 2013, University of Badji Mokhtar–Annaba, Algeria.