REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DE M'SILA
FACULTE DES SCIENCES ET SCIENCES DE L'INGENIEUR

DEPARTEMENT D'ELECTROTECHNIQUE

MEMOIRE DE FIN D'ETUDES EN VUE DE L'OBTENTION DU DIPLÔME

D'INGENIEUR D'ETAT EN GENIE ELECTROTECHNIQUE

OPTION: COMMANDE ELECTRIQUE

THEME

IMPLEMENTATION DES RESEAUX DE NEURONES ARTIFICIELS (RNA) SUR "FPGA" POUR LE DIAGNOSTIC DES DEFAILLANCES DE LA MACHINE ASYNCHRONE

Proposé et dirigé par :
Dr. KHODJA Djalal Eddine

Présenté par :
BENGUETTAF Brahim
ZAOUI Redha

Année Universitaire: 2007/2008

Dédicace

Avant tous, je remercie dieu le tout puissant
de m'avoir donner le courage et la patience pour
réaliser ce travail malgré toutes les difficultés rencontrées.
Je dédie ce modeste travail :
A mes très chers parents, que dieu les garde et les protége
pour leurs soutien moral et financier, pour leurs
encouragements et les sacrifices qu'ils ont endurés.

A mon petit frère ***Hamza***
A mes sœurs surtout la petite ***Sihem***
A mes nièces ***Afaf*** *&* ***Aissa***
A mon enseignant durant les années du primaire
Merzougui Mokhtare
A mon encadreur ***Dr.Khodja Djalel Eddine***
A mon collègue ***Rédha Zaoui****, qui m'a accompagné*
pendant le long de cette période pour réaliser
ce modeste travail.
A tous mes amis d'études
A toute ma grande famille.

Brahim

Dédicace

Au nom de dieu Je dédie ce modeste mémoire à ***mes parents*** *qui ont su me soutenir tout long de mes études, parfois me réconforter dans les moments difficiles, et qui sans eux je n'aurai pu effectuer ce travail.*

Je dédie ce modeste travail

A ma très chère sœur ***Amina***

A mes très chers frères : ***Taha, Salah Eddine*** *&* ***Abdelhalim***

A toute ma famille

A mon collègue ***Brahim BenGuettaf,*** *qui m'a accompagné pendant le long de cette période pour réaliser ce modeste travail.*

A mon encadreur ***Dr.Khodja Djalel Eddine***

A tout mes profs et surtout Mr : ***Benjaima.Bachir*** *qu'il m'ont appris durant toutes mes années d'étude dans la spécialité d'électrotechnique.*

A tout les enseignants qui m'ont aidé de proche ou de loin pour être un jour un ingénieur d'état en électrotechnique.

A mon compagnons de route ***Younes Bakache***

A tous les étudiants de l'université et surtout ***d'Ouled Sidi Brahim.***

Notre dédicace va également à ceux qui ont participé de prés ou de loin à l'aboutissement de nos efforts.

Rédha

Sommaire

Chapitre III : supports physiques d'implémentation

Chapitre IV : Implémentation des RNA sur FPGA pour la détection des défauts de la MAS asynchrones

Introduction générale

Introduction générale

Le moteur asynchrone à rotor bobiné a été utilisé jusqu'à un passé récent pour les entraînements à vitesse peu variable. Mais il ne présentait pas une sensible amélioration par rapport au moteur à courant continu. La machine à rotor à cage d'écureuil était pour sa part réservé aux entraînements à vitesse constante à cause de la difficulté de sa commande et de la difficulté du suivi de ses paramètres rotoriques. Cependant, Ce moteur présente de nombreux atouts : sa puissance massique, sa robustesse, son coût de fabrication relativement faible et un entretien minimum.

Toutes ces qualités justifient le regain d'intérêt de l'industrie vis-à-vis de ce type de machine. De plus, les développements récents de l'électronique de puissance et de commande permettent aux moteurs asynchrones à cage d'avoir les mêmes performances que celles des machines à courant continu. Ceci explique son développement dans l'industrie et le remplacement progressif des machines à courant continu.

La maintenance et le diagnostic de ce type de machine asynchrone deviennent donc un enjeu économique. Il est important de détecter de manière précoce les défauts qui peuvent apparaître dans ces machines et donc de développer des méthodes de surveillance de fonctionnement ou de maintenance préventive.

La nécessité d'une rapidité pour détecter et localiser une défaillance à cause des besoins de l'industrie et la complexité des systèmes fait appel à plusieurs techniques de diagnostics qui possèdent des caractéristiques différentes et qui permettant de résoudre ces problèmes.

Parmi les techniques les plus utilisées on trouve la technique d'intelligence artificiel a base de Réseaux de Neurones Artificiel (RNA) qu'ils sont plus facile à implémenter sur les circuits électroniques à savoir : les DSP (Digital Signal Processing), les ASIC (Application Specific Integrated Circuits) ou sur les FPGA (Field programmable gate array).

Le but de notre travail porte sur l'élaboration d'un système de diagnostic en temps réel des défaillances de la machine asynchrone en utilisant les réseaux de neurones artificiels implémentés sur des circuits électroniques tels que les FPGA.

Ce mémoire est organisé comme suite :

Le premier chapitre sera consacré à l'étude de diagnostic des défaillances des machines asynchrones. Nous commençons dans un premier lieu par une introduction sur le diagnostic, ensuite nous présentons les différentes défaillances qui peuvent surgir sur les moteurs asynchrones et nous terminons ce chapitre par une analyse des différentes techniques de diagnostic existantes en présentant ses avantages et ses inconvénients.

Dans le deuxième chapitre de ce travail nous détectons les défauts de l'association moteur onduleur et commande par les RNA. Commençant par la modélisation de la machine dans le repère de Park puis la commande vectorielle de l'association de la MAS-Onduleur de tension et leurs résultats de simulation à l'aide de logiciel Matlab/Simulink et nous terminons ce chapitre par l'affectation du diagnostic par technique de RNA.

Dans Le troisième chapitre nous parlons sur les différents supports physiques d'implémentation et précisément sur le circuit FPGA qui est le meilleur pour notre étude.

Le dernier chapitre de notre étude présentera l'implémentation du RNA sur FPGA pour la détection des défauts de la machine asynchrone.

REMERCIMENTS

Nous tenons à remercier tout premièrement ***allah*** *le tout puissant pour la volonté, la santé et la patience, qu'il nous a donné durant toutes ces longues années.*

Nos parents, pour leurs encouragements, leur grande compréhension et leur inconditionnel soutien très souvent indispensable...

Ainsi, Nous tenons à remercier notre Promoteur :
Dr khodja .DJalel Eddine *qui a bien voulu nous encadrer et pour ses conseils précieux, sa disponibilité et sa patience*

Ainsi que tous le personnel du Département de L'électrotechnique de M'sila, sous la direction de ***Mr BENJAIMA .B***

En fin, nous adressons nos vifs remerciements à Tous les enseignants qui ont participés le long de ces années à notre formation ainsi qu'à notre promotion et à tous ceux qui ont participés de prêt ou de loin à l'élaboration de ce travail.

Brahim & Rédha

Chapitre I :

Etat de l'Art du diagnostic de défaillance des Machines asynchrones

I.1 Introduction

Les arrêts forcés de production suite à des pannes de machines peuvent être coûteux et représenter une part importante de l'investissement initial. Cependant, l'absence d'une stratégie de maintenance peut causer la perte des éléments d'un processus et une diminution de la sécurité, d'où son intérêt stratégique.

La détection automatique des défaillances devient de plus en plus indispensable à cause de la faiblesse de l'opérateur humain ; qui est une conséquence de la fatigue, de l'oubli, et parfois de la pression de l'environnement (bruits, chaleurs, etc.).

De plus, les réparations avant incident sont plus faciles à exécuter et moins coûteuses (en temps d'intervention et en matériel). Il est aussi important de savoir qu'un défaut non traité peut entraîner des dégradations encore plus importantes. Ce qui provoque des conséquences non souhaitables (autant pour la production que pour le personnel), à savoir:

- Arrêts fréquents du processus de production (d'où le manque à gagner)
- Augmentation des pertes d'énergie ;
- Augmentation des coûts de maintenance ;
- Augmentation des coûts de production ;
- Risque de mise hors service des équipements ;
- Danger pour la sécurité du personnel.

Par conséquent, on s'intéresse obligatoirement au diagnostic automatique qui permet de détecter de façon précoce les anomalies, chose qui représente un des moyens sûrs pour contribuer à améliorer la productivité des différents secteurs.

I.2 Les différentes défaillances qui peuvent surgir sur les moteurs asynchrones

L'historique du diagnostic des défaillances est ancien comme les machines électriques elles mêmes. Les utilisateurs de la machine électrique se sont initialement penchés sur les simples protections comme les surtensions, les surintensités et les défauts à la terre. Cependant, comme la complexité de la machine se développe en croissance, des améliorations sont aussi recherchées dans le domaine du diagnostic des défaillances.

Les défaillances de la machine électrique, dans leur majorité, peuvent être classifiées comme suit **[1]** :

- Les défauts du stator résultant de la coupure ou du court-circuit d'une ou de plusieurs phases de l'enroulement statorique;

- La connexion anormale de l'enroulement statorique;
- La cassure d'une barre du rotor ou le craquement de l'arbre de la machine;
- Les irrégularités de l'entrefer;
- L'inclinaison de l'axe du rotor, pouvant résulter d'un frottement entre le rotor et le stator;
- Le court-circuit dans l'enroulement rotorique ;
- Les défauts sur les roulements.

Le fonctionnement à vitesse variable de la machine asynchrone nécessite un contrôle de l'énergie par un convertisseur statique. En effet, on peut envisager d'autres types de défauts potentiels qui peuvent surgir sur la partie puissance et commande, à savoir :

- La défaillance d'un bras d'onduleur: c'est à dire l'un des composants électroniques (Transistor ou thyristor) est maintenu hors tension (ouvert) ;
- L'un des composants est maintenu fermé ;
- La défaillance du capteur de vitesse : il n'y a pas de retour de la valeur de la vitesse sur la commande (défaut sur la commande);
- La défaillance partielle du capteur de vitesse (offset sur le capteur) ;

Par ailleurs, le diagnostic d'une défaillance fait appel le plus souvent à l'utilisation de Signatures élaborées à partir de signaux qui contiennent les informations jugées pertinentes par les spécialistes du domaine. La complexité des signaux dépend de la nature des systèmes et des matériels à diagnostiquer et varie en fonction de l'anomalie recherchée.

En ce qui concerne la machine asynchrone et pour identifier les défaillances citées ci-dessus, on peut citer les techniques de validation des signaux :

- la mesure des températures ;
- le contrôle des émissions de fréquences radio ;
- le contrôle du champ électromagnétique ;
- le contrôle des vibrations ;
- l'analyse chimique;
- les mesures des bruits acoustiques ;
- l'analyse de la signature du courant du moteur;
- la mesure de la vitesse.

La technique retenue dans cette étude est l'analyse des courants du moteur et de sa vitesse de rotation ainsi que de la tension d'alimentation. Les défaillances considérées

seront celles du stator. Par conséquent, le courant, la vitesse du moteur et la tension peuvent être considérés comme étant les valeurs indicatrices les plus représentatives des défaillances considérées.

I.3 Analyse de la procédure du diagnostic

Lorsqu'un défaut apparaît dans un équipement industriel, le système de diagnostic lié à ce dernier doit d'abord détecter l'anomalie du fonctionnement puis y identifier la (ou les) cause (s) de défaillance à l'aide d'un raisonnement logique pour qu'il puisse être isolé **[2]**.

En effet, l'organisation générale de la procédure de diagnostic s'articule autour des étapes suivantes :

- A partir des moyens de mesure ou d'observation appropriés, nous effectuons l'extraction des informations nécessaires à la mise en forme des caractéristiques associées aux fonctionnements normaux et anormaux ;
- l'élaboration des signatures associées à des symptômes de défaillance en vue de la détection d'un dysfonctionnement ;
- la détection d'un dysfonctionnement par comparaison avec des signatures associées à des états de fonctionnements normaux ;
- la mise en œuvre d'une méthode de diagnostic de défaillance à partir de l'utilisation des connaissances sur les relations de causalité (catalogue cause-effets) ;
- une phase d'interprétation des données de diagnostic (identification du type et de la nature du défaut);
- la prise de décision en fonction des conséquences et de l'importance des défauts.

La figure (I.1), représente l'ensemble des tâches à réaliser pour assurer un fonctionnement satisfaisant d'un processus industriel :

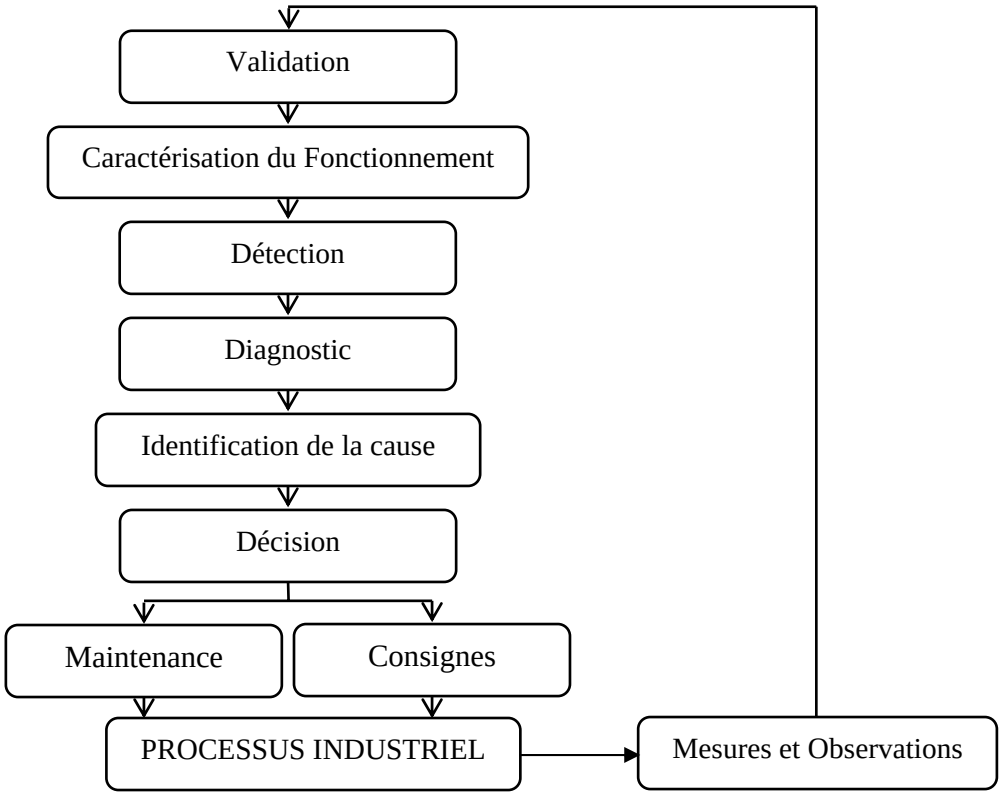


Figure (I.1) : Les différentes étapes du diagnostic industriel **[2]**

I.4 Les méthodes de diagnostic

Le diagnostic qu'il soit médical ou bien industriel est toujours basé sur la comparaison entre le comportement du procédé défaillant et les connaissances du comportement sain ou de son modèle.

La comparaison nécessite des indicateurs , des symptômes révélateurs qui, une fois analysés permettent d'abord de détecter le comportement défaillant, d'en déduire la fonction ou l'élément en dysfonctionnement (localiser), puis d'en déterminer la cause et enfin , si possible d'y remédier .

Le graphe suivant présente les différentes méthodes de diagnostic **[3]**.

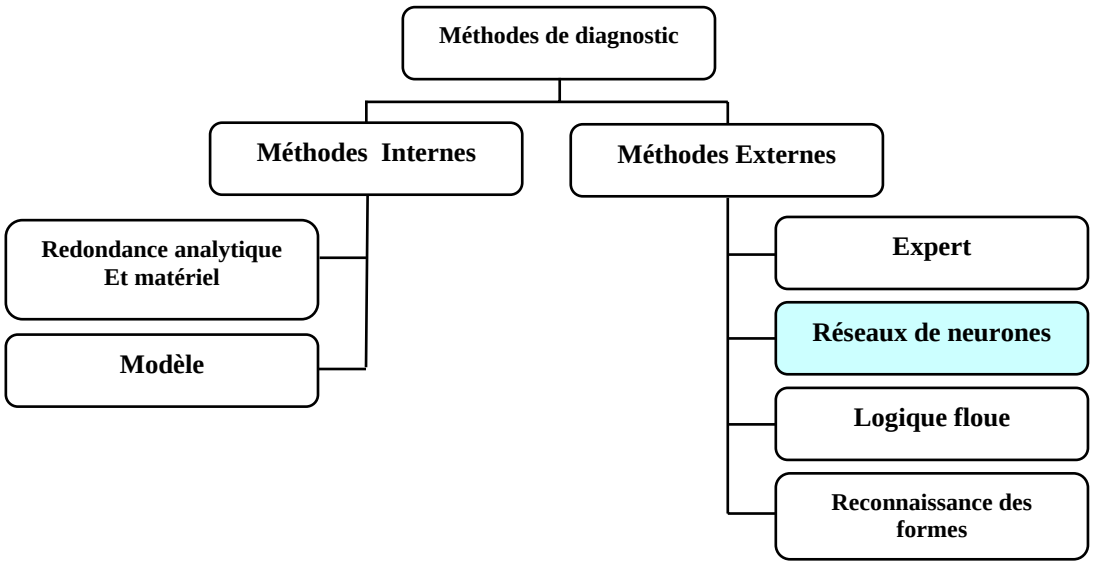


Figure (I.2) : Synoptique de classement des méthodes de diagnostic

I.4.1 Les méthodes internes

Cette famille de méthodes est principalement dérivée des techniques utilisées par les automaticiens. A partir de modèles physiques ou de comportement validé par les techniques d'identification de paramètres, il devient possible de mettre en œuvre la méthode du problème inverse .Le diagnostic de défaillance est possible en suivant en temps réel l'évolution des paramètres physiques ou bien en utilisant l'inversion de modèles de type « boite noire» .Les méthodes de diagnostic internes se regroupent en deux grandes familles :

- la méthode du modèle .
- la méthode de redondance analytique et matérielle **[2]**.

I.4.1.1 La méthode du modèle

La méthode du diagnostic basée sur le modèle **[3]** consiste à comparer les grandeurs déduites d'un modèle représentatif du fonctionnement des différentes entités du processus avec les mesures directement observées sur le processus industriel. La figure (I.3) représente le fonctionnement d'un système de détection de défaillances utilisant l'approche basée sur le modèle.

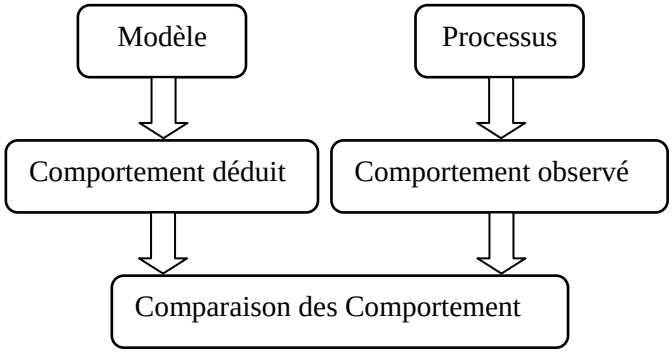


Figure (I.3): Principe de fonctionnement de la méthode du modèle

En général, pour des processus continus dotés de systèmes de régulation, il est judicieux d'utiliser un modèle prenant séparément en compte leschaînes de régulation, les actionneurs et les sous ensembles du processus industriel. Comme il a été utilisé en **[4]** pour un système d'entraînements électriques, le modèle possède une structure identique à Celle du processus. La comparaison des comportements sert à la détection des écarts entre les signaux d'état normaux et anormaux.

I.4.1.2 La redondance analytique et matérielle

La redondance analytique se base sur la connaissance d'un modèle mathématique du système à surveiller, ce modèle placé dans les mêmes conditions que le système est calculé en temps réel, fournit les relations de redondances nécessaires à la comparaison du procédé avec le modèle **[4]**.

D'un côté, il existe une méthode par la redondance matérielle qui consiste à doubler ou à tripler l'équipement et plusieurs actionneurs plusieurs capteurs et plusieurs systèmes de commande **[4]**. L'utilisation de cette méthode est essentiellement réservée aux cas où la continuité de service est obligatoire (avionique). En effet, elle apporte l'avantage une fois.

La défaillance détectée, d'utiliser la partie de l'équipement encore saine c'est une méthode chère et complexe.

I.4.2 Les méthodes externes

Les méthodes de diagnostic externes s'appliquent dans les situations où la modélisation des mécanismes reliant les causes des défaillances et des dégradations ne sont pas techniquement modélisables, ou bien ne présentent pas d'intérêt économique compte tenu des enjeux recherches.

Dans cette catégorie on retrouve toutes les méthodes basées sur l'intelligence artificielle. L'intelligence artificielle est presque vieille que l'informatique, les premiers travaux et réalisation datent de début des années 50, l'expression "intelligence artificielle"a été proposée par "J.Me carthy"en 1956. parmi ces méthodes on peut citer : les systèmes experts,la logique floue ,les reconnaissances des formes et les RNA.

I.4.2.1 Diagnostic par systèmes experts

Un système expert doit fournir des réponses associées à une situation donnée sachant que la complexité du domaine étudié ne permet pas d'établir a priori toutes les configurations possibles des situations ; ainsi un spécialiste du dépannage réalise son diagnostic à l'aide d'une description partielle de la situation.

A partir de l'ensemble de symptômes dont il dispose, il peut déduire toutes les conclusions possibles, élaborer de nouvelles hypothèses et approfondir son diagnostic en exploitant des informations supplémentaires collectées sur le système à diagnostiquer **[2]**.

Un système expert est constitué de plusieurs éléments principaux dont on peut citer **[5]**.

a. la base de connaissances :

La base de connaissances est souvent la partie stable du système, elle est immédiatement exécutée dès l'appel du système et c'est elle qui rassemble les connaissances nécessaires à la résolution des problèmes du domaine.

b. la base de faits :

Les faits sont les objets élémentaires de l'application ; la base de faits que au départ contiennent les informations initiales relatives au domaine s'enrichit progressivement de conclusions ou de nouveaux faits déduits par le moteur d'inférence.

c. le moteur d'inférence :

Le moteur d'inférence exploite de façon indépendante les informations contenues dans la base de faits et la base de connaissances pour fournir une solution au problème donné.

Par ailleurs, le moteur d'inférence produit dans son mécanisme de raisonnement de nouveaux faits qui sont mémorisés dans la base de faits.

I.4.2.2 Diagnostic par logique floue

Cette approche, contrairement à la logique du tout ou rien de G. Boole, a pour but la prise en compte de la logique de la pensée humaine qui est approximative.

Le concept de sous-ensemble flou a été introduit pour prendre en compte l'imprécision. La fonction d'appartenance à un ensemble flou prend des valeurs sur l'intervalle **[0, 1]**.

Celles-ci expriment le degré d'appartenance d'un élément à cet ensemble : 0 pour la non appartenance stricte et 1 pour l'appartenance stricte.

Le principe de fonctionnement du diagnostic basé sur la logique floue consiste à extraire et à calculer les signaux d'entrée, en utilisant les règles linguistiques représentées sous forme de fonctions d'appartenances ; ces règles peuvent contenir tous les modèles possibles qui correspondent au mode de défaut considéré.

En général, la procédure de fonctionnement d'un système flou est effectuée en trois étapes :

- La fuzzification ;
- L'inférence ;
- La défuzzification.

Le bloc fuzzification contient en général un traitement des données préliminaire ; ces données sont alors traitées par des règles linguistiques, ce qui nécessite leur définition par des fonctions d'appartenance. La fuzzification fournit une série de variables floues, réunies par un vecteur, qui va être introduit au bloc d'inférence.

Dans le bloc inférence, les valeurs des variables linguistiques sont liées par plusieurs règles qui doivent tenir compte du comportement statique et dynamique du système (causes des effets du système à diagnostiquer).

Dans la dernière étape, il faut transformer la valeur floue (linguistique) en une valeur déterminée, ceci eu faisant dans le bloc défuzzification ; en on rencontre les applications de cette méthode pour le diagnostic des machines synchrones **[6].**

I.4.2.3 Les méthodes de reconnaissance des formes

Parmi toutes les approches du diagnostic, la reconnaissance des formes est une technique de définition d'algorithmes permettant de classer des objets dont l'aspect a varié par rapport à un objet type. Il s'agit de définir à quelle forme–type une forme observée ressemble le plus **[3]**.

Dans ce cas, la forme est définie par un ensemble de 'd' paramètres appelés caractères qui sont les composants du vecteur forme x ; les prototypes sont les points représentatifs du vecteur forme dans l'espace à 'd' dimensions ;

Dans un problème de reconnaissance des formes on suppose généralement que les formes x à classer appartiennent à M catégories bien déterminées appelées classes et notées w1, w2,…..wm.

Compte tenu du bruit de mesure les vecteurs appartenant à une classe donnée wi forment une zone particulière dans l'espace de dimension "d".

Le problème de la reconnaissance des formes consiste à déterminer les formes séparant les M classes dans le but de pouvoir ultérieurement classer une nouvelle forme "x" lors de l'opération de classification ou de discrimination **[4]**.La procédure de diagnostic basée sur les techniques de reconnaissance des formes met en oeuvre quatre étapes principal suivants :

- Choix de la signature initiale
- Construction des vecteurs formes
- Réalisation de la base d'apprentissage de représentation des classes
- Acquisition de nouvelles données, (Représentation, classification et détection)

I.4.2.4 Le diagnostic externe avec les réseaux de neurones artificiels

Cette méthode faite partie des méthodes de diagnostic externes, elle utilise la classification automatique des signaux et des formes **[2]**.

Le principe de fonctionnement des réseaux de neurones artificiels est inspiré des mécanismes de fonctionnement du cerveau humain. De l'extérieur, le réseau de neurones artificiels se présente comme une ' boite noire ' qui reçoit des signaux d'entrée et qui fournit les réponses appropriées.

Dans le cas du diagnostic, les différents états de fonctionnement normaux et anormaux sont le plus souvent caractérisés par des signatures. L'opérateur humain, après avoir mémorisé et appris les différentes formes de signatures associées à un état déterminé, est capable à la lecture d'une nouvelle signature d'identifier très rapidement l'état de la machine. Si la nouvelle signature est déjà apprise, il devra l'interpréter et la mémoriser. En général, le RNA permet de reconstituer le processus de raisonnement humain décrit ci-dessus.

Par ailleurs, un réseau de neurones artificiels est défini par **[2]** :

- la nature des cellules élémentaires qui le constituent (binaires ou continues) ;
- l'architecture et le nombre des couches du réseau ;
- la nature des connexions ;
- les méthodes d'apprentissage ;
- les performances de classification ;
- les mécanismes de mémorisation .

I.5 Position du problème à résoudre

Le problème à résoudre en termes de diagnostic est le choix des méthodes de diagnostic et leurs implémentations dans les processus industriels.

La grande diversité des technologies des systèmes industriels ne permet pas d'utiliser une méthode universelle qui posséderait tous les avantages et aucun inconvénient. Une méthode universelle de diagnostic industrielle n'existe pas.

D'autres méthodes de diagnostic utilisent les techniques de l'automatique pour suivre en temps réel ou en temps différé des éléments qui ont un sens physique. Ces méthodes de diagnostic utilisent schématiquement des "boites noires " entre les signatures associées aux causes et à leurs effets.

L'analyse des implémentations des systèmes de diagnostic dans les différents secteurs industriels fait ressortir que la majorité d'entre eux font appel au système expert suivi par ceux basés sur la reconnaissance des formes. Toutefois, il est à remarquer que le développement d'un système de diagnostic basé sur la technique des systèmes experts, nécessite un grand effort (pour sa conception et sa réalisation). En outre, son installation coûte très cher.

Partant de la considération que le caractère principal du système de diagnostic est de constituer un catalogue défauts–symptômes, les approximateurs universels (réseaux de neurones artificiels) paraissent très intéressants pour la mise en place de la procédure du diagnostic .

Par ailleurs, les réseaux de neurones artificiels possèdent des caractéristiques permettant la résolution de problèmes complexes, à savoir :

- La capacité de classification des signatures et des formes ;
- Le RNA peut apprendre des règles à partir des exemples (défauts) ;
- La capacité de mémorisation des exemples ;
- Les RNA sont plus facile à implémenter sur les circuits électroniques à savoir : les DSP, les ASIC ou sur les FPGA.

Par ailleurs, le circuit numérique reconfigurable connu actuellement sur le marché sous l'appellation de FPGA (Field Programmable Gate Arrays) est fabriqué à un niveau d'intégration d'autour de 90 nanomètres. Il comprend habituellement plusieurs millions de portes logiques sur la même puce. Les FPGA offrent des avantages considérables pour accélérer le temps d'application, réduis le temps de développement et le coût de la production. En effet, les circuits reconfigurables réduisent considérablement le risque lié à

l'implémentation (une erreur de conception peut être corrigée en reprogrammant le FPGA), certains circuits FPGA modernes permettent des corrections en cours de fonctionnement.

Il est clair qu'un FPGA peut répondre efficacement aux défis actuels et futurs parmi lesquels :

1. Les améliorations de performances du contrôle. Par exemple, le temps d'exécution peut être réduit considérablement en adoptant une architecture parallèle. En plus, les dernières versions des FPGA peuvent être adaptés et reconfigurés durant leurs fonctionnement;

2. La baisse du coût pour au moins trois raisons: l'usage d'une architecture basé seulement sur les besoins spécifiques de l'algorithme d'implémenter l'application de méthodologies très avancées et la spécification du temps de la mise en application, et l'intégration de tout le système de contrôle avec son interface analogique dans une seule puce;

En plus les FPGA peuvent être programmés facilement en utilisant des langages de description évolués très simples tels que le VHDL ou le Verilog. Par ailleurs, la firme Xilinx propose une bibliothèque sur Matlab/simulink permettant le développement et la simulation comportementale sur Matlab. Un logiciel de simulation dit ModelSim offre des simulations reflétant le fonctionnement réel de l'architecture sur FPGA, permettant par conséquent d'éliminer le risque d'erreur.

Par conséquent, le présent travail sera consacré à l'élaboration d'un système de diagnostic en temps réel des défaillances du moteur asynchrone en utilisant les réseaux de neurones artificiels. Ensuite, on s'intéressera à l'implémentation des réseaux de neurones artificiels sur les circuits reconfigurables tels que les FPGA.

I.7 Conclusion

Dans ce chapitre, les différentes méthodes du diagnostic et leur implémentation ont été présentés.

Ensuite, après une synthèse de ces méthodes, il a été constaté que les réseaux de Neurones sont les mieux adaptés pour le diagnostic des différents défauts de la machine asynchrone. Pour leur implémentation, les circuits FPGA ont été choisis à cause de leurs performances qui sont les meilleurs. Avant l'implémentation des RNA sur FPGA on doit passer dans le deuxième chapitre par l'étude du comportement de la MAS et commande et leur diagnostic par RNA.

Chapitre II :

Détection des défauts de l'association moteur-onduleur et commande par les RNA

II.1 Introduction

Dans ce chapitre on s'intéressera à l'étude de l'ensemble convertisseur-moteur et leur commande en vue de leur diagnostic.

Ensuite, nous allons appliques les RNA pour la détection des différentes défaillances qui peuvent surgir sur cet ensemble.

II.2 Modélisation de la machine asynchrone en vue de sa commande et son diagnostic

II.2.1 Hypothèses simplificatrices

L'étude de cette machine traduit les lois de l'électromagnétisme dans le contexte habituel d'hypothèses simplificatrices suivantes, **[7]**:

- Entrefer constant.
- Effet des encoches négligé.
- Circuit magnétique non saturé.
- Pertes ferromagnétiques négligeables.
- L'influence de l'effet de peau et de l'échauffement sur les caractéristiques n'est pas prise en compte.

Parmi les conséquences de ces hypothèses on peut citer :

- L'additivité des flux.
- La constance des inductances propres.
- La loi de variation sinusoïdale des inductances mutuelles entre les enroulements statorique et rotorique en fonction de l'angle entre leurs axes magnétiques.

La machine est représentée à la figure (II.1) par ces six enroulements dans l'espace électrique, l'angle (θ) repère l'axe fixe de la phase rotorique de référence (Ra) par rapport à la phase statorique de références (Sa), les flux sont comptés positivement selon les axes des phases **[8].**

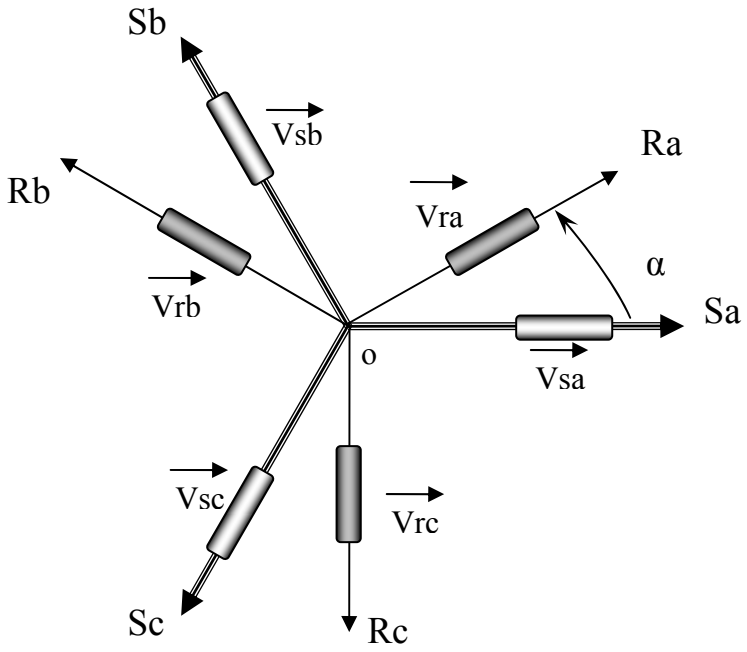


Figure (II.1): Représentation schématique d'une machine asynchrone triphasée.

II.2.2 Equations générales

Le comportement de la machine asynchrone est entièrement défini par trois types d'équations à savoir :

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

II.2.2.1 Equations électriques

Nous pouvons à présent écrire le système matriciel électrique suivant dans le repère a, b, c :

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

Ou de manière plus raccourcie :

$$[V_{sabc}] = [R_s][i_{sabc}] + (d/dt)[\Phi_{sabc}] \qquad \text{(II.02)}$$

$$[V_{rabc}] = [R_r][i_{rabc}] + (d/dt)[\Phi_{rabc}] \qquad \text{(II.03)}$$

II.2.2.2 Équations magnétiques

Maintenant, nous devons exprimer les grandeurs magnétiques au stator et au rotor, toujours dans le repère a, b, c :

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

Pour l'écriture condensée, on met :

$$[L_{ss}]=\begin{bmatrix} l_s & M_s & M_s\\ M_s & l_s & M_s\\ M_s & M_s & l_s\end{bmatrix} \qquad \text{(II.05)} \qquad [L_{rr}]=\begin{bmatrix} l_r & M_r & M_r\\ M_r & l_r & M_r\\ M_r & M_r & l_r\end{bmatrix} \qquad \text{(II.06)}$$

On aura :

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

On aura finalement :

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

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

II.2.2.3 Équations mécaniques

Pour étudier les phénomènes transitoires électromécaniques avec une vitesse rotorique variable (par exemple le démarrage, le freinage, la variation de la charge à l'arbre, etc.…), il faut ajouter l'équation de mouvement au système d'équations différentielles **[9]**.

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

Notons que la vitesse électrique du rotor est donnée par l'expression suivante :

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

II.2.3 Modélisation dans le repère de Park

La transformation de park consiste à transformer un système triphasé (a b c) en un système biphasé équivalent (d q), comme il est montré sur la figure (II.2)

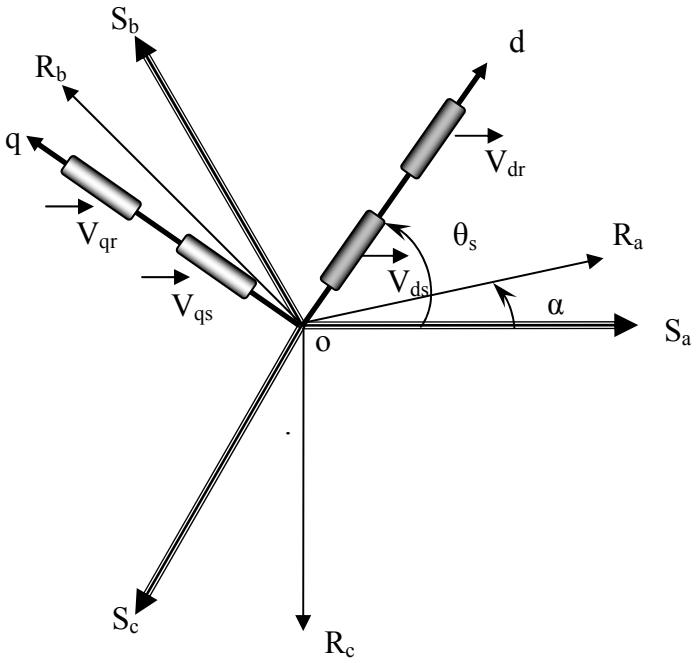


Figure (II.2): Représentation du passage d'un système triphasé à celui biphasé

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

II.2.3.1 Équations électriques

$$V_{ds}=R_s.i_{ds}+\frac{d\Phi_{ds}}{dt}-\omega_s.\Phi_{qs} \quad \text{(II.13)}$$

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

$$V_{dr}=R_r.i_{dr}+\frac{d\Phi_{dr}}{dt}-(\omega_s-\omega_r).\Phi_{qr}=0. \quad \text{(II.15)}$$

$$V_{qr}=R_r.i_{qr}+\frac{d\Phi_{qr}}{dt}+(\omega_s-\omega_r).\Phi_{dr}=0 \quad \text{(II.16)}$$

II.2.3.2 Equations magnétiques

$$\Phi_{ds} = L_s.i_{ds} + M.i_{dr}. \quad (II.17) \qquad \Phi_{dr} = L_r.i_{dr} + M.i_{ds}. \quad (II.18)$$

$$\Phi_{qs} = L_s.i_{qs} + M.i_{qr}. \quad (II.19) \qquad \Phi_{qr} = L_r.i_{qr} + M.i_{qs}. \quad (II.20)$$

II.2.3.3 Équations mécaniques

$$\begin{aligned} &J.\frac{d\Omega_r}{dt} = C_e - C_r - f.\Omega_r \\ &C_e = (3/2).p.M.(i_{rd}.i_{sq} - .i_{sd}.i_{rq}). \\ &\omega_r = p.\Omega_r. \end{aligned} \quad (II.21)$$

II.3 Représentation d'état

$$[X'] = [A][X] + [B][U] \quad (II.22)$$

$[X]$: Vecteur d'état, avec $[X] = [i_{ds} \quad i_{qs} \quad \Phi_{dr} \quad \Phi_{qr}]^t$. (II.23)

$[A]$: Matrice d'évolution d'état du système.

$[B]$: Matrice de la commande.

$[U]$: Vecteur du système de commande.

$$[A] = \begin{bmatrix} -R_t & \omega_s & \frac{M}{\sigma.L_s L_r.T_r} & \frac{M.\omega}{\sigma.L_r L_s} \\ -\omega_s & -R_t & -\frac{M.\omega}{\sigma.L_r L_s} & \frac{M}{\sigma.L_s L_r.T_r} \\ \frac{M}{T_r} & 0 & -\frac{1}{T_r} & \omega_r \\ 0 & \frac{M}{T_r} & -\omega_r & -\frac{1}{T_r} \end{bmatrix} \quad (II.24)$$

Avec $R_t = \frac{1}{\sigma.L_s}\left[R_s + \frac{M^2}{L_r.T_r}\right]$; $T_r = \frac{L_r}{R_r}$; $T_s = \frac{L_s}{R_s}$; $\sigma = 1 - \frac{M^2}{L_s L_r}$

Pour notre étude, nous avons choisi le référentiel lié au stator parcequ'il est mieux adapté (pour l'étude du comportement de la machine asynchrone dans plusieurs régimes de fonctionnement). $\omega_s = 0$, $\omega_r = -\omega$

$$[A]=\begin{bmatrix} -R_t & 0 & \frac{M}{\sigma.L_s L_r.T_r} & \frac{M.\omega}{\sigma.L_r L_s} \\ 0 & -R_t & -\frac{M.\omega}{\sigma.L_r L_s} & \frac{M}{\sigma.L_s L_r.T_r} \\ \frac{M}{T_r} & 0 & -\frac{1}{T_r} & -\omega \\ 0 & \frac{M}{T_r} & \omega & -\frac{1}{T_r} \end{bmatrix} \qquad \text{(II.25)}$$

$$[B]=\begin{bmatrix} \frac{1}{\sigma.L_s} & 0 \\ 0 & \frac{1}{\sigma.L_s} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \text{(II.26)} \qquad [U]=\begin{bmatrix} V_{ds} & V_{qs} \end{bmatrix}^t \qquad \text{(II.27)}$$

II.4 Simulation et interprétation

II.4.1 Démarrage à vide

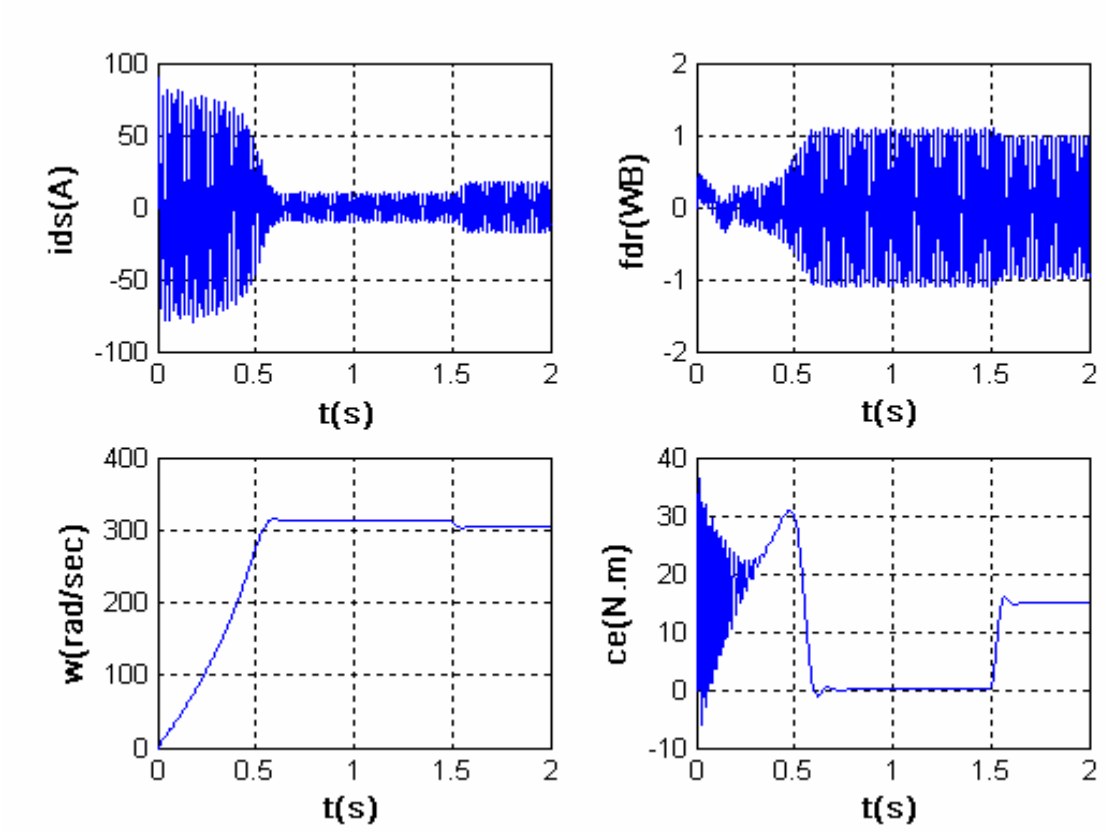


Figure (II.3): Résultats de la simulation du démarrage à vide du moteur asynchrone triphasé alimenté par une source de tension équilibrée, suivie d'une application d'un couple résistant (Cr=15N.m) à l'instant t=1.5s.

II.4.2 Interprétation des résultats

a.La vitesse

Au démarrage et dans un temps étroit, la vitesse est en accroissement presque linéaire jusqu'à la valeur de synchronisme, l'orsqu'on applique un couple résistant (perturbation) on remarque que la vitesse est décroissante ce qui signifie qu'il y a un glissement.

b.Le courant statorique

Au démarrage, le courant prend sa valeur maximale, au moment où le moteur joue un rôle de transformateur. Le courant oscille autour de zéro jusqu'à l'instant où il prend sa valeur permanente (sinusoïdal).

Lorsqu'on applique une charge (perturbation), on remarque naturellement une demande (augmentation) du courant ce qui implique un échauffement des enroulements du moteur surtout lorsque le couple résistant devient plus grand que celui du nominal.

c.Le couple électromagnétique

La croissance de la vitesse au démarrage indique la présence du fort couple électromagnétique (ou plus un couple résistant), ce dernier oscille de manière décroissante jusqu'à la valeur 0.

Lorsqu'on applique un couple résistant après certain temps, on remarque une augmentation du couple électromagnétique jusqu'à la valeur de perturbation associée.

d.Le flux rotorique

Au démarrage, le flux presque nul, dans le régime permanent prend sa valeur maximale, Lorsqu'on applique un couple résistant, on remarque une deménution du flux rotorique.

II.5 Modélisation de l'onduleur en utilisant la technique MLI

Le réglage de la vitesse du rotor de la MAS se réalise logiquement par action simultanée sur la fréquence et la tension statorique. Par conséquent, pour se donner les moyens de cette action, il faut disposer d'une source d'alimentation capable de délivrer une tension d'amplitude et de fréquence réglable en valeur instantanée.

L'onduleur donne une réponse à ce besoin. Les trois cellules (bras) de commutation, formant un onduleur triphasé, sont bidirectionnelles en courant, et composées des (demi-bras) commandées à l'ouverture et à la fermeture et chaque demi-bras possède son complémentaire. L'emploi de la technique MLI pour déterminer les intervalles de

conduction des interrupteurs permet de régler de manière indépendante les valeurs moyennes de chacune des tensions Vas, Vbs etVcs sur chaque période de commutation.

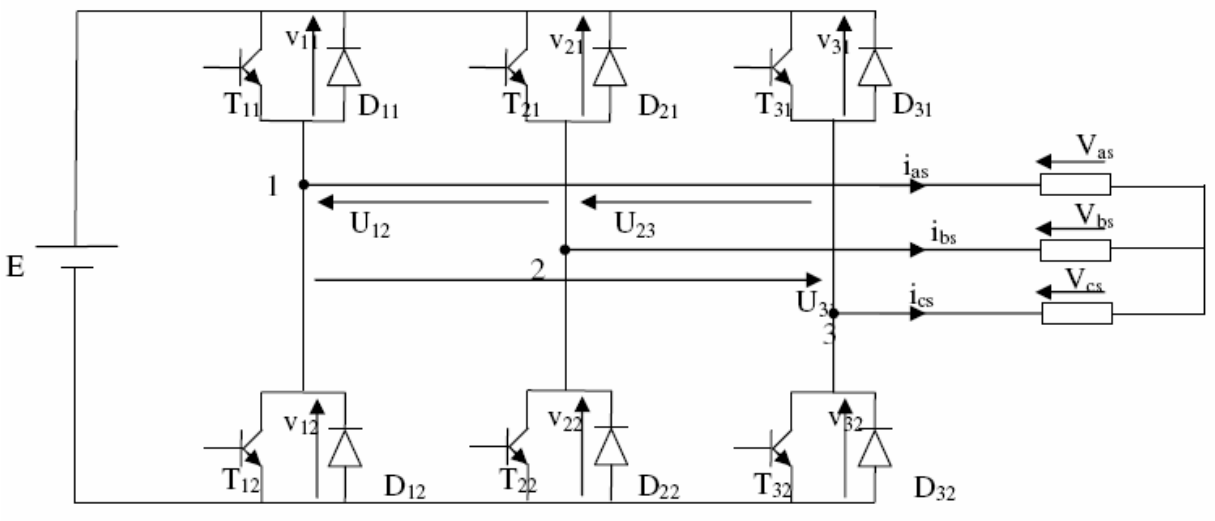


Figure (II.4): Représentation de l'onduleur de tension MLI**.**

II.5.1 Application de la technique MLI Sinus -Triangle

La MLI sinus triangle (MLI ST) utilise le principe d'intersection entre une référence Sinusoïdale de fréquence fr, appelée modulante, et un signal triangulaire de haute fréquence fp appelée porteuse p.

Les trois signaux de références sont donnés par les équations suivantes :

$V_{refc} = V_m \sin(2\pi f.t - 2(c-1)\pi/3)$; c =1, 2, 3, (II.28)

L'équation de la porteuse est donnée par:

$$V_p(t) = \begin{cases} V_{pm}\left(4\frac{t}{T_p} - 1\right) & \text{si} \quad 0 \le t \le \frac{T_p}{2} \\ V_{pm}\left(-4\frac{t}{T_p} + 3\right) & \text{si} \quad \frac{T_p}{2} \le t \le T_p \end{cases} \qquad \text{(II.29)}$$

Les instants de commutation sont déterminés par comparaison de trois ondes de références avec une onde porteuse qui fixe la fréquence de commutation. Cette comparaison fournit trois signaux logiques f_1, f_2 et f_3 qui valent 1 quand les interrupteurs du coté haut sont en conduction et ceux de coté bas sont bloquées et valent 0 dans le cas contraire. A partir de ces signaux l'électronique de commande élabore les signaux de commande des interrupteurs.

$$\begin{aligned} &\text{si} \quad V_{ref\ 1} \geq V_p(t) \quad f_1 = 1 \quad \text{si .non} \quad f_1 = 0 \\ &\text{si} \quad V_{ref\ 2} \geq V_p(t) \quad f_2 = 1 \quad \text{si .non} \quad f_2 = 0 \\ &\text{si} \quad V_{ref\ 3} \geq V_p(t) \quad f_3 = 1 \quad \text{si .non} \quad f_3 = 0 \end{aligned} \tag{II.30}$$

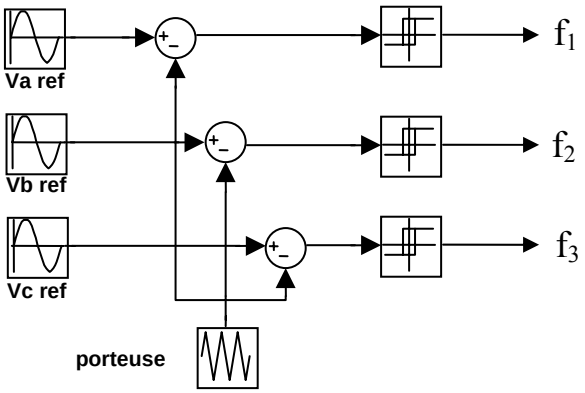


Figure (II .5): principe de la MLI sinus triangle

L'expression des tensions composées est donnée par :

$$\begin{aligned} U_{12} &= V_{as}-V_{bs} = V_{21}-V_{11} \\ U_{23} &= V_{bs}-V_{cs} = V_{31}-V_{21} \\ U_{31} &= V_{cs}-V_{as} = V_{11}-V_{31} \end{aligned} \tag{II.31}$$

En introduisant les fonctions de connexion relatives à chacun d'entre eux, il vient :

$$\begin{bmatrix} U_{12} \\ U_{23} \\ U_{31} \end{bmatrix} = E \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \tag{II.32}$$

Il, en découle :

$$\begin{bmatrix} V_{as} \\ V_{bs} \\ V_{cs} \end{bmatrix} = \frac{1}{3} E \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \tag{II.33}$$

La commande MLI est caractérisée par les deux paramètres :

- L'indice de modulation "m" égale au rapport de la fréquence de modulation sur la Fréquence de référence (m=fp/f).
- Le coefficient de réglage en tension "r" égale au rapport de l'amplitude de la tension de référence à la valeur crête de l'onde de modulation (r=Vm /Vpm).

II.6 Association de la MAS-Onduleur de Tension

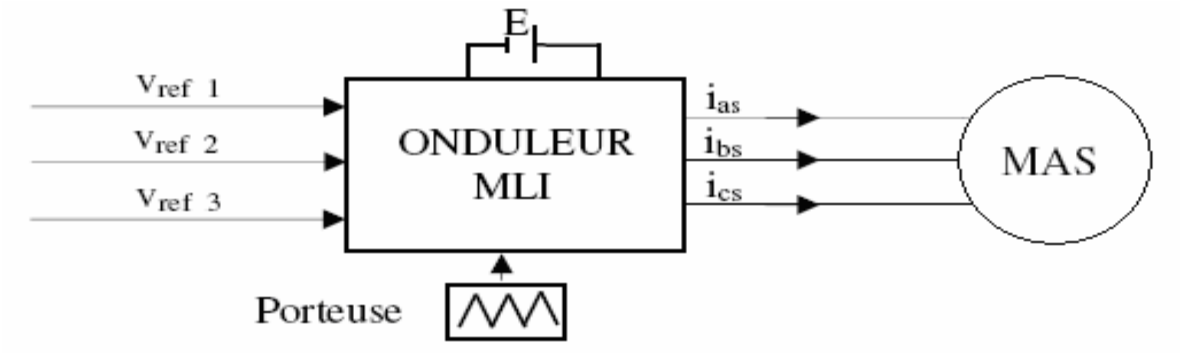


Figure (II.6): Association de la MAS avec onduleur MLI.

II.6.1 Résultats de Simulation

La simulation numérique est effectuée pour m= 21 et r= 0,8 ; les résultats de simulation de l'association de onduleur avec la MAS sont représentés dans la figure (II.7)

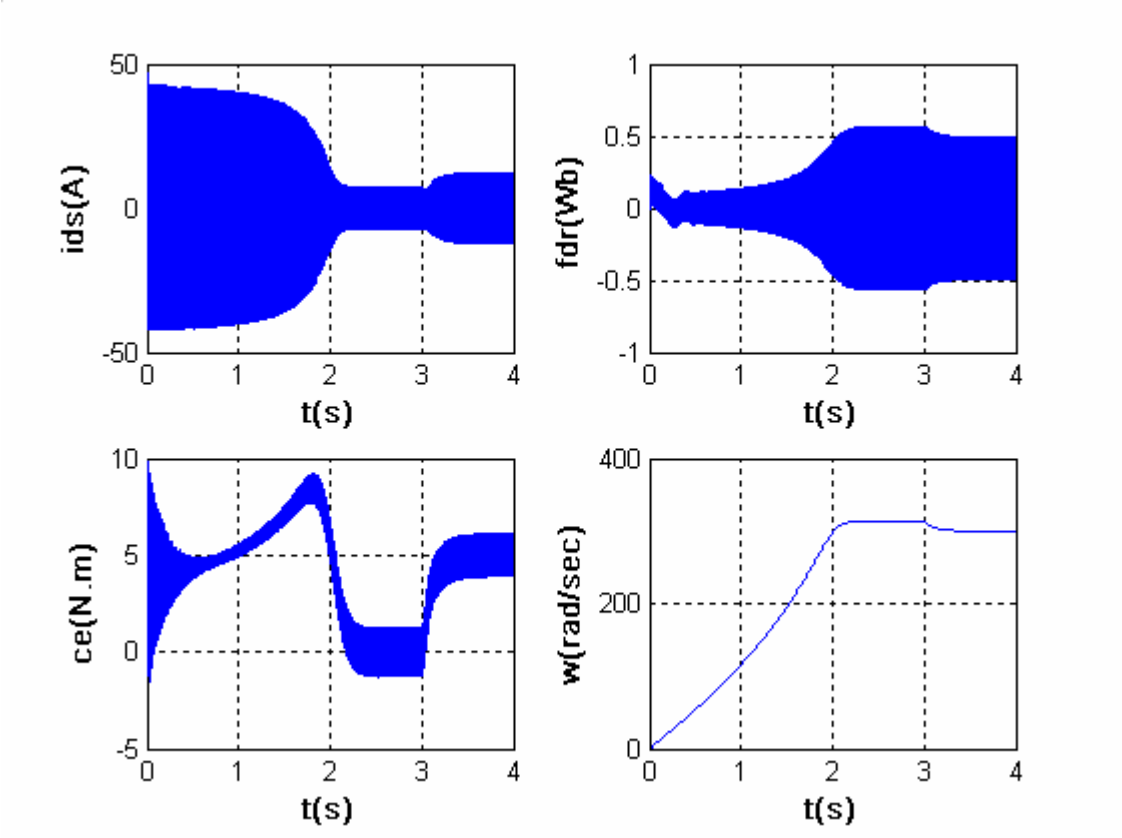


Figure (II.7): Réponses de la MAS alimentée par un onduleur de tension lors d'une application d'un couple résistant (Cr =5N.m) durant (3,4) s.

II.7 Commande vectorielle de la machine asynchrone

Le but de la commande vectorielle est d'arriver à commander la machine asynchrone comme une machine à courant continu à excitation indépendante où il y a un découplage naturel entre la grandeur commandant le flux (le courant d'excitation), et celle liée au couple (le courant d'induit). Ce découplage permet d'obtenir une réponse très rapide du couple. En parlant d'orientation du flux, c'est plutôt le système d'axe d-q que l'on oriente de manière à ce que l'axe d soit en phase avec le flux, c'est-à-dire:

$$\begin{cases} \varphi_d = \varphi \\ \varphi_q = 0 \end{cases} \qquad \text{(II.34)}$$

II.7.1 Equations des tensions

En imposant φ_{qr} Les équations de la machine dans un référentiel lié au champ tournant deviennent :

$$\begin{cases} \varphi_r = \varphi_{dr} \\ V_{ds} = R_s I_{ds} + \sigma L_s \dfrac{dI_{ds}}{dt} + \dfrac{M}{L_r}\dfrac{d\varphi_r}{dt} - \omega_s \sigma L_s I_{qs} \\ V_{qs} = R_s I_{qs} + \sigma L_s \dfrac{dI_{qs}}{dt} + \omega_s \dfrac{M}{L_r}\varphi_r + \omega_s \sigma L_s I_{ds} \\ \tau_r \dfrac{d\varphi_r}{dt} + \varphi_r = M I_{ds} \\ \omega_r = \dfrac{M}{\tau_r \varphi_r} I_{qs} \\ C_e = p\dfrac{M}{L_r}\varphi_r I_{qs} \end{cases} \qquad \text{(II.35)}$$

Après passage par une transformation de Laplace nous obtenons :

$$\begin{cases} V_{ds} = (R_s + p\sigma L_s) I_{ds} + p\dfrac{M}{L_r}\varphi_r - \omega_s \sigma L_s I_{qs} \\ V_{qs} = (R_s + p\sigma L_s) I_{qs} + \omega_s \dfrac{M}{L_r}\varphi_r - \omega_s \sigma L_s I_{ds} \end{cases} \qquad \text{(II.36)}$$

$\varphi r = \dfrac{M}{1+p\tau_r} I_{ds}$ Ainsi $\varphi_r = M.I_{ds}$

En régime permanent :

$$\omega r = \frac{M}{\varphi_r \tau_r} I_{qs} \qquad \text{(II.37)}$$

D'ou : $I_{qr} = \dfrac{-M}{L_r} I_{qs}$

Ce qui conduit a l'expression :

$$Ce = \frac{3}{2} p \frac{M}{L_r} \varphi_r I_{qs} \qquad (II.38)$$

II.7.2 Méthodes de commande vectorielle des moteurs asynchrones

a. Commande vectorielle directe

Pour la commande directe, l'angle de park. est calculé directement à l'aide des grandeurs mesurées ou estimées.

b. Commande vectorielle indirecte par orientation de flux rotorique

Dans ce type de commande, l'angle *θs* utilisé pour la transformation directe et inverse est alculé à partir de la formule suivante **[10]**.

$$\theta_s = \int \left(p\Omega + \frac{I^{\bullet}_{qs}}{\tau_r I^{\bullet}_{ds}} \right) dt \qquad \text{où} \qquad I^{\bullet}_{ds} = \frac{\varphi^{\bullet}_r}{M}$$

La figure (II.8). représente le schéma bloc d'une régulation de vitesse du moteur asynchrone commandé par orientation du flux rotorique.

Les principaux constituants dans ce type de commande sont la boucle de régulation de vitesse, celles des courants I_{ds} et I_{qs}, le bloc de calcul de θs et les transformations directe et inverse.

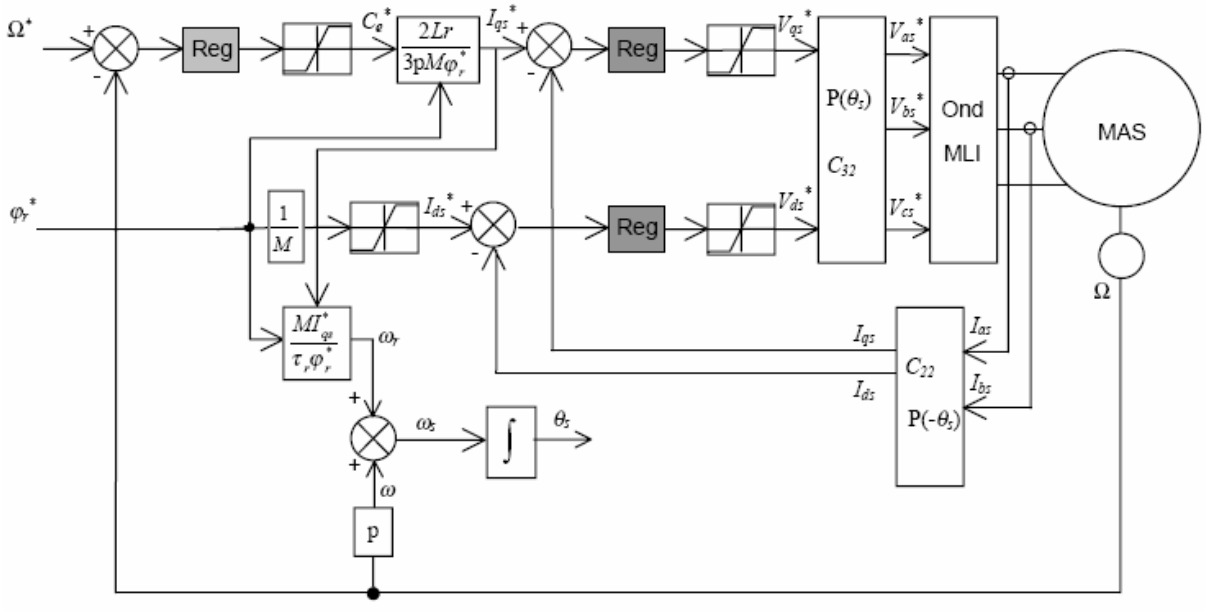


Figure (II.8): Régulation de vitesse par la commande vectorielle indirecte

II.7.3 Résultat de simulation à d'application de la commande vectorielle de la machine asynchrone avec onduleur de tension

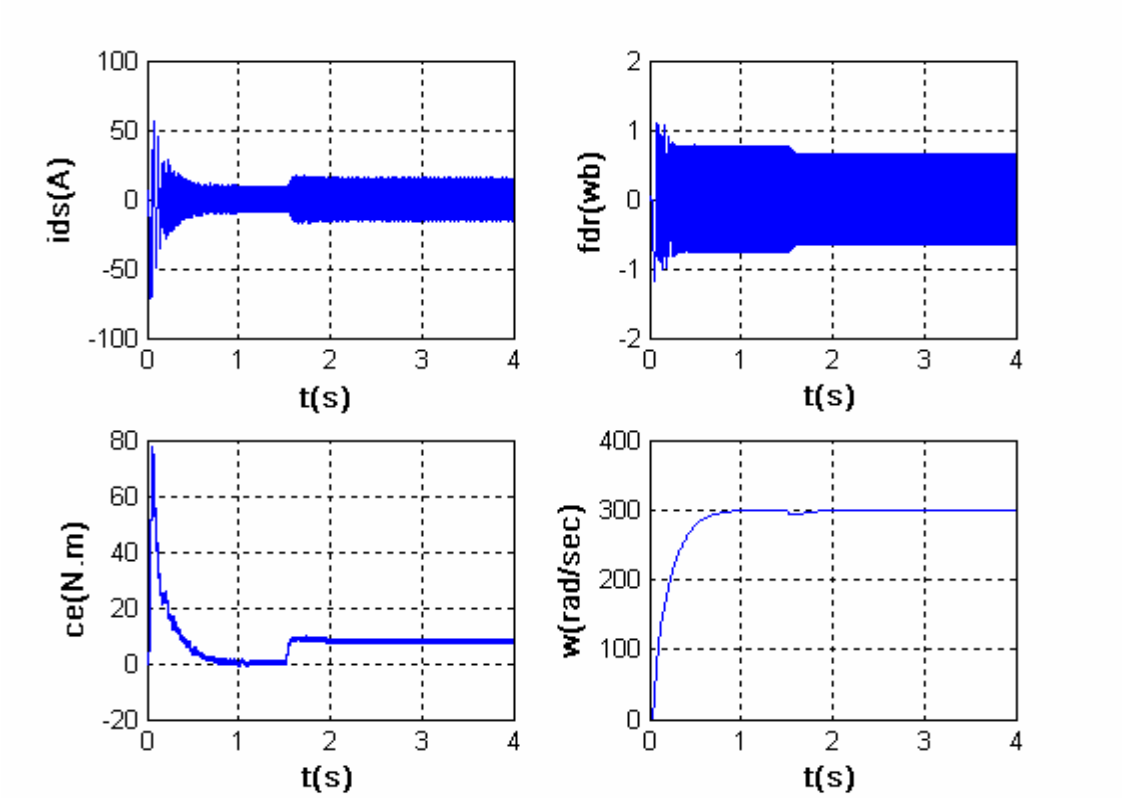


Figure (II.9): Simulation de la commande vectorielle de la MAS alimentée par un onduleur de tension lors d'une application d'un couple résistant (Cr =8N.m)durant (1.5,4) s.

II.7.3.1 Interprétation

Cette figure montre avant l'application de la charge, la vitesse possède une caractéristique presque linéaire stabilisée par la valeur de vitesse de référence .après l'application de charge (Cr =8N.m à t =1.5 s), la courbe présente une chute dans la valeur de la vitesse , ensuite elle se stabilise à sa valeur de référence (300 rad/s). Le courant statorique suit la variation de la charge. Le couple subit un pic au premier moment de démarrage, puis atteint la valeur du couple résistant après l'application de charge. Le flux rotorique subit une chute au moment de l'application de la charge puis de stabilise à sa valeur désirée.

II.8 Application des RNA pour la détection des défauts de l'association moteur convertisseur

Le modèle qu'on vient d'élaborer on va l'utiliser dans cette section pour le diagnostic de la machine asynchrone en utilisant les RNA

II.8.1 Introduction au RNA

Le terme "réseaux de neurones artificiels" regroupe un certain nombre de modèles dans l'intention d'imiter certaines fonctions du cerveau humain reproduisant quelques unes de ses structures de base.

Par ailleurs, les réseaux de neurones sont adaptés comme outil d'aide aux opérations de reconnaissance et de classification, entre autre, celles liées à la résolution des problèmes de diagnostic utilisant la classification automatique des signaux et des formes pour plus de détails voir l'annexe 2.

II.8.2 Construction des blocs RNA

Les réseaux de neurones que nous avons simulés sont tous des réseaux multicouches qui utilisent l'algorithme de rétropropagation pour leurs apprentissages.(voir annexe) Pour l'implémentation des blocs RNA dans le système automatique de diagnostic, on se propose d'étudier un seul réseau.

Les étapes de construction et de validation des réseaux de neurones sont réparties en trois phases:

a. Choix des entrées des réseaux

les entrées du RNA sont les valeurs efficaces (Ia, Ib, Ic, Va, Vb, Vc et w), ce qui signifie que le nombre d'entrées de ce réseau est égal à 7 (voir la figure (II.10)).

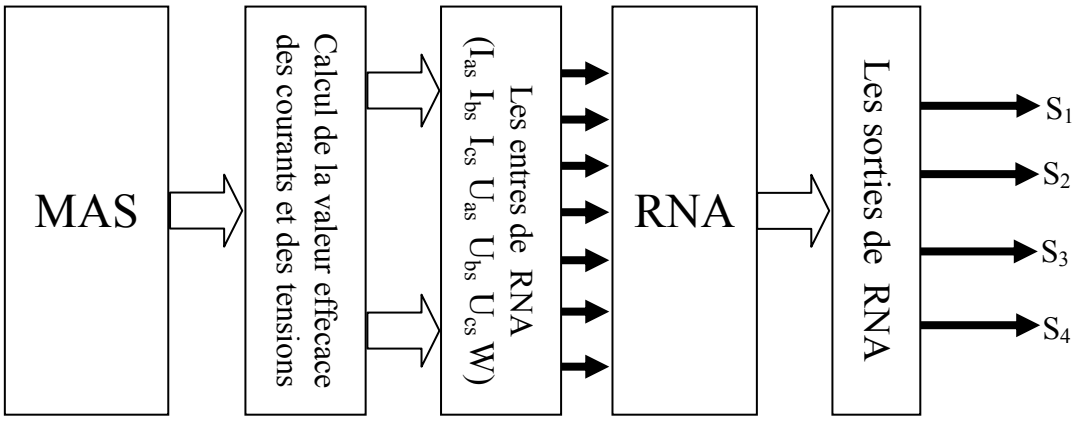


Figure (II.10): Structure du RNA réseau à étudier (les valeurs indicatrices sont les valeurs efficaces)

b. Choix des sorties des réseaux

Lors de la détection d'un défaut, le réseau doit indiquer un nombre binaire quelconque(par exemple 0110) à sa sortie, qui correspond à un type de défaut (défaut de coupuremonophasée).

- Le nombre de sorties de réseau est égal à 4.
- Les défauts sont représentés sur le tableau 1, avec leurs symboles et leurs codes associés.

Tableau (II.1) : classification des defaux

catégorie	Type de defaut	symbole	code			
			S1	S2	S3	S4
01	Etat sain	ES	0	0	0	0
02	Coupure monophasée	MC	1	0	0	0
03	Coupure biphasée	BC	0	1	0	0

c. Choix de fonction d'activation

Comme les sorties sont binaires et les entrées réelles, la fonction d'activation une fonction sigmoïde, présentée sur Figure (II.11).

$$F(x)=\frac{1}{1+e^{-x}}$$

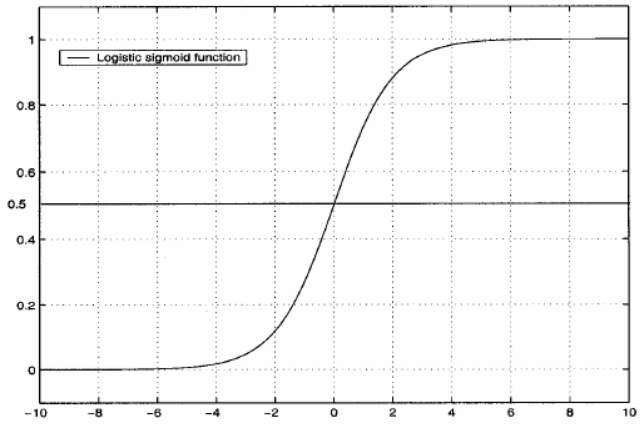


Figure (II.11): fonction segmoide

II.8.3 Résultats d'essais de réseau

On a effectué un apprentissage automatique à l'aide du logiciel MATLAB jusqu'à où on a obtenu une erreur quadratique la plus petite après 16 étérations, (voir figure (II.12))

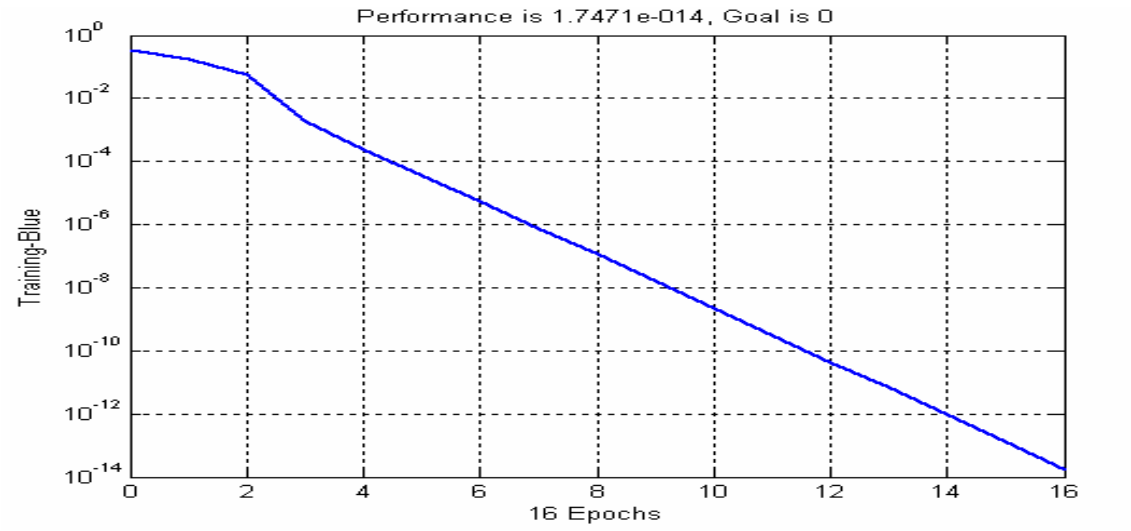


Figure (II.12): Evolution de l'erreur quadratique moyenne du RNA

II.8.4 Tests de RNA

Une fois que le RNA construit et que sont apprentissage a atteint des performances satisfaisantes, En fait, cet exemple de tests leurs résultats sont présentés dans le tableau ci-dessous.

Tableau (II.2) : Résultats de simulation (test) du RNA pour différent cas

Sortie du RNA	Etat sain	Coupure monophasée	Coupure biphasée
S1	2.2489 e-007	1.0000 e+000	7.6137 e-007
S2	2.1249 e-009	1.3985 e-007	1.0000 e+000
S3	1.9817 e-009	4.2075 e-015	5.7261 e-011
S4	1.4025 e-008	5.7641 e-011	4.7708 e-009

II.8.5 Interprétation des résultats

D'après les résultats obtenus dans la phase du test, on constate que les sorties de Réseau suivent presque conformément (avec erreur de e-008) les sorties désirées préétablies auparavant.

II.9 Conclusion

Dans ce chapitre Nous avons commencé par la modélisation de la MAS puis l'association de cette dernière avec onduleur de tension commandée par la technique de modulation de largeur d'impulsion MLI qui montre le fort couplage entre le flux et le couple électromagnétique. Ceci nous a conduit à introduire une commande découplant, Il s'agit de la commande par orientation du flux rotorique (commande vectorielle), qui permet aussi de régulier la chute de la vitesse lors d'application d'une charge.

Ensuit, nous avons appliqués un réseau de neurone artificiel pour la Détection des defaux de l'association moteur-onduleur et commande. cela a été réalisé grâce à une implémentation logicielle utilisant MATLAB comme logiciel de simulation, et la règle de rétro propagation comme règle d'apprentissage des RNA .

Enfin, pour l'implémentation physique (sur des circuits électroniques) des RNA, on vas étudier dans la chapitre trois les différents supports physiques existants dans le marche,en vers d'en choisir celui le plus performent pour le diagnostic.

Chapitre III :

Supports physiques d'implémentation

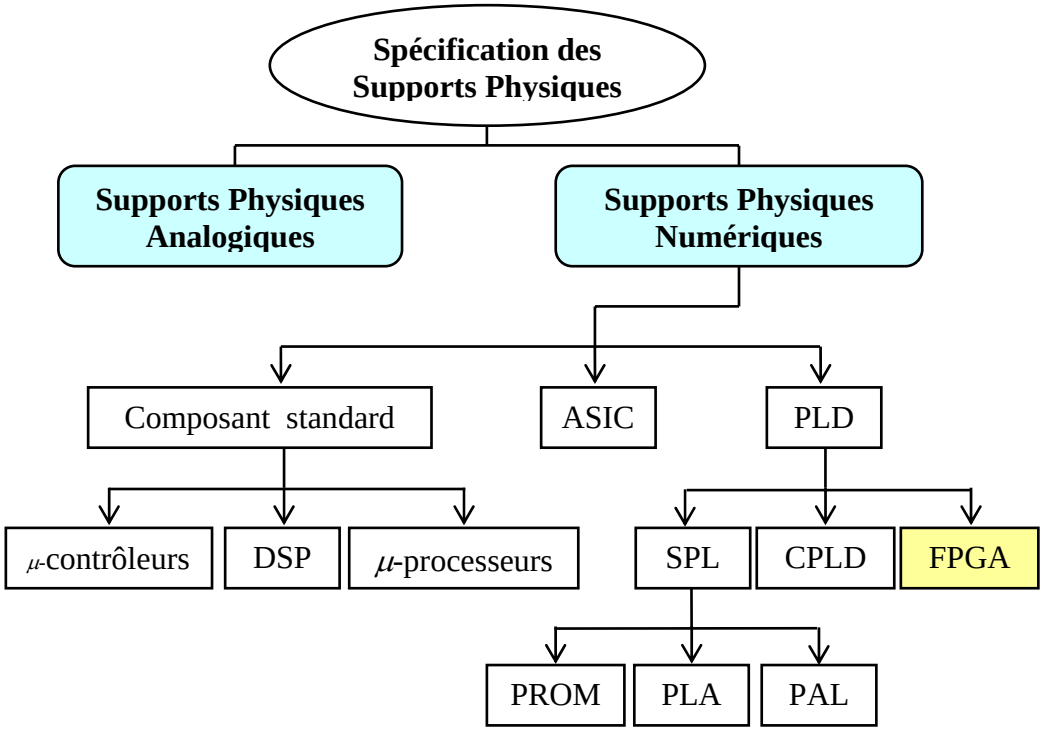


Figure (III.1) : Différentes solutions technologiques

III.2.1 Support physique analogique

L'ensemble des circuits analogiques donne une bande passante large avec réponse rapide à haute résolution; néanmoins, ils souffrent de plusieurs inconvénients, (principalement, ils souffrent des inconvénients de bruit, ensemble de circuits complexes et difficulté dans la modification du circuit).

En outre, la fiabilité du système électronique décroît avec l'augmentation du nombre de composants. Aussi ces circuit sont d'une grande complexité et très difficiles à modifier. En plus, nous enregistrons une difficulté de modification du circuit.

Enfin, les problèmes décrits ci-dessus et le coût de la maintenance pour un système totalement câblé deviennent vite un handicap **[12]**.

Toutes ces contraintes nous incitent à s'orienter vers de nouveaux supports technologiques d'implémentation basés sur l'électronique numérique.

III.2.2 Support physique numérique

Les supports physiques numériques peuvent être subdivisés en trois grandes technologies selon les différentes fonctions numériques qu'ils sont en mesure de réaliser (Figure III.1) **[11]**. Nous distinguons :

Les composants standards : cette famille englobe toutes les anciennes technologies des composants, qui continuent à se développer, tels que les microprocesseurs, les microcontrôleurs et toutes les logiques TTL ou CMOS ;

Les ASIC's : ce sont des circuits intégrés dédiés à une application particulière ;

Les composants logiques programmables : ce sont des circuits intégré qui n'ont pas de fonctionnalités figées. Chaque fonctionnalité doit être programmé à l'aide de logicielle spécifique à la technologie du circuit à programmer. Nous distinguons trois types principaux de circuits programmables, à savoir : les SPLD, les CPLD et les FPGA.

III.3 Les composants Standards

III.3.1 Microprocesseur

Un microprocesseur est un circuit intégré, dont la fonction principale est d'implémenter n'importe quelle fonction (instruction) ou suite d'instructions constituant un programme et les exécuter de manière séquentielle **[13]**.

Le grand avantage du microprocesseur est sa généricité, puisque n'importe quel programme peut y être exécuté. De plus, pour le programmer il suffit de transcrire dans le langage du microprocesseur un problème particulier.

Le fonctionnement de base d'un microprocesseur est divisé en trois unités : la mémoire, l'unité de traitement et l'unité de commande, Le microprocesseur considéré comme l'unité centrale de traitement, est chargé de traiter toutes les informations entrée/sortie. Il gère des mémoires centrales, **RAM (** Random Access Memory)**, ROM (**Read Only Memory **)** et **EPROM (** Erasable Programmable ROM), des « entrées-sorties » ainsi que des circuits d'interface qui assurent la communication avec l'utilisateur. Il assure aussi l'exécution et le traitement des programmes.

Malgré les avantages que présente le microprocesseur et sa constante évolution, il est sanctionné par sont fonctionnement en série (instruction après instruction). En effet, l'exécution d'un programme est constitué de plusieurs étapes,ce qui rend le temps d'exécution d'un calcul très considérable, surtout lors du traitement des algorithmes complexes tels que : ceux du la diagnostic des défaillances et du controle.

Une architecture de microprocesseur peut alors être plus adaptée pour une tâche que pour une autre. A cet effet, les microprocesseurs ne peuvent pas, sous des contraintes temporelles trop fortes, répondre à tous les besoins fonctionnels.

III.3.2 Les DSP (Digital Signal Processing)

Les DSP sont des microprocesseurs dédiés au traitement du signal et destinés aux applications nécessitant de nombreux calculs (transformée de Fourier, produits de convolutions, etc.). La particularité de ces puces est de regrouper un certain nombre d'éléments architecturaux qui les rend adaptées aux calculs intensifs **[14]**.

En effet, ces processeurs sont fortement adaptés à des calculs liés au traitement du signal. Ils sont donc conçus pour traiter des flots de calculs, et les tests de conditions et les branchements sont très pénalisants puisqu'ils tendent à vider le pipeline **[14]**; ce qui ralentit le fonctionnement et peut faire perdre l'intérêt de l'utilisation de tels processeurs.

On voit cependant des DSP avec des architectures de plus en plus complexes qui font apparaître une forte potentialité dans le calcul parallèle.

III.4 Les ASIC's (Application Specific Integrated Circuits)

Si les composants numériques précédents peuvent être développés avec un simple ordinateur, les supports numériques ASIC's nécessitent l'intervention d'un fondeur qui produira le circuit demandé à partir des masques fournis par le demandeur.

Cette cible matérielle représente de nombreux avantages lors de son application dans un environnement aussi contraignant que celui des systèmes électrotechniques avec leurs commandes et régulations **[15]** , Ces circuits présentent d'énormes atouts, à savoir :

une consommation faible : dû à une conception ciblée de circuit .

un faible volume et de meilleures performances : L'échelle d'intégration permet la concentration de nombreuses fonctions (équivalentes à plusieurs centaines de circuits standard) sur une seule puce. Cette considérable réduction de la taille fait diminuer le nombre de composants, de cartes et de connecteurs utilisés dans le système de commande.

L'immunité au bruit, la vitesse d'exécution, la gamme de température et la résistance mécanique s'en trouvent alors améliorées.

III.5 PLD (Programmable Logic Device)

L'évolution de la microélectronique à permis de promouvoir d'une manière spectaculaire les circuits logiques programmables, en effet, nous comptons à ce jours plusieurs types, bien différents par leurs structures que par leurs puissances d'intégration et par leurs fréquences de travail. Les circuits logiques programmables sont classés principalement en trois classes **[11]** ; les SPLD's, les CPLD's et les FPGA (voir Figure III.1).

III.5.1 SPLD (Simple Programmable Logic Device)

Les SPLDs **[16]**, (Figure III.2), sont composés d'une grille de portes ET et d'une grille de portes OU, les deux étant reliées. Les entrées du système peuvent être connectées aux portes ET, et le résultat des portes OU correspond à la sortie du système.

Dans ces circuits, les connexions sont préexistantes, les différentes lignes étant reliées par des fusibles, des transistors EPROM, ou des transistors EEPROM (Electrically EPROM). En brûlant certains de ces fusibles, ou en programmant les transistors, il est alors possible de réaliser différentes fonctions logiques.

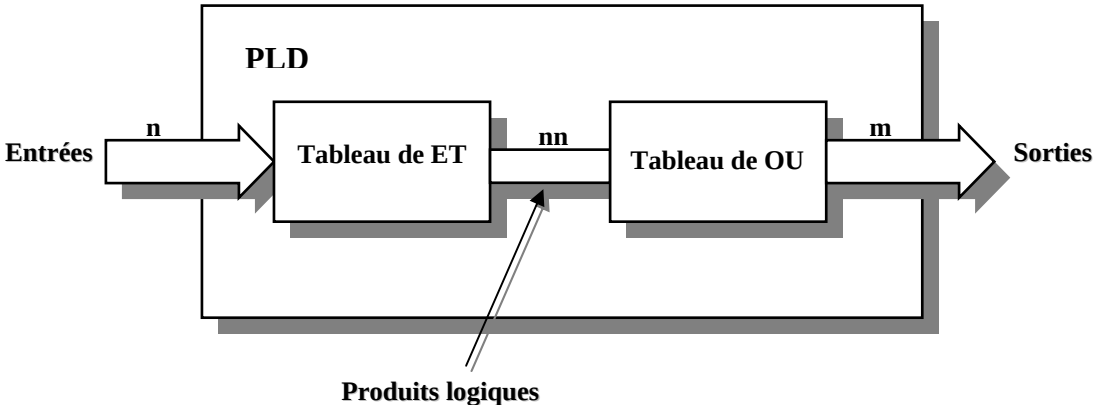


Figure (III.2) : Architecture d'un SPLD

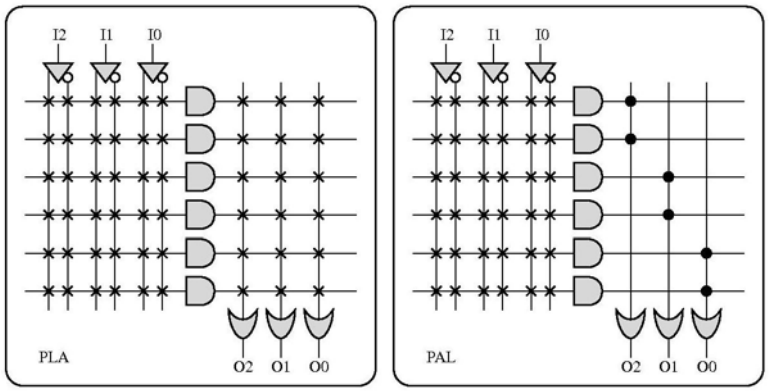
III.5.1.1 PLA (Programmable Logic Array)

Les PLAs **[17]** apparurent aux environs de l'année 1975 dans le but de pallier les limitations des PROMs. En effet, dans un PLA (Figure III.3.a), contrairement à une PROM, toutes les interconnexions peuvent être programmées, ce qui en fait le PLD le plus général. Il est alors possible de définir les produits et les sommes, les rendant particulièrement efficaces lorsque plusieurs sorties utilisent les mêmes produits. Une amélioration s'accompagnant souvent de désagréments, citons un des inconvénients du PLA comparé à la PROM ; à savoir : étant donné que les deux tableaux (ET/OU) sont programmés, le temps de propagation d'un signal de l'entrée à la sortie est nettement plus important dans le cas de programmation d'un seul tableau.

III.5.1.2 PAL (Programmable Array Logic)

Vers la fin des années 1970, les PALs **[18]** furent introduites afin de contrer le problème de vitesse de propagation des PLAs. Dans un PAL (Figure III.3.b), les connexions entres les portes ET et OU sont fixes, et les connexions entre les entrées et les

portes ET peuvent être programmées. L'avantage des PALs sur les PLAs est donc leur rapidité, mais elles présentent l'inconvénient de n'avoir qu'un nombre limité de produits pour chaque porte OU.



(a) *L'architecture d'un PLA* (b) *L'architecture d'un PAL*

Figure (III.3) : Architectures d'un PLA et d'un PAL

Les SPLDs présentent deux limitations majeures, à savoir l'impossibilité de réaliser des fonctions à plusieurs niveaux et celle de ne pouvoir partager les produits de différentes fonctions. A cet effet une nouvelle génération de circuits programmables est développée.

III.5.2 CPLD (Complex Programmable Logic Device)

Les CPLDs (ou dit EPLD par la société Altera), apparus au début des années 80, sont l'évolution des SPLDs. Ils permettent l'implémentation de systèmes nettement plus complexes, et sont composés d'éléments de base programmables, connectés entre eux par un réseau d'interconnexions programmable relativement simple. Ces éléments de base sont du type SPLD.

Notons qu'un des avantages des CPLDs sur les FPGAs, que nous présenterons dans le paragraphe suivant, est la rapidité. En effet, le réseau d'interconnexions, en étant nettement plus simple, est plus rapide que celui d'un FPGA. De plus, les connexions se font toujours avec une destination pour une source, et le temps de propagation est donc toujours le même. Le placement d'un désigne dans un CPLD n'est donc pas critique, et le routage peut être systématisé, sans avoir besoin de tenir compte de contraintes de temps.

Le routage des CPLDs commence à être intéressant, puisqu'en plus de définir la fonctionnalité de blocs simples comme les SPLDs, il devient possible de les interconnecter.

III.5.3 FPGA (Field programmable gate array)

Les FPGA sont des circuits intégrés constitués d'une matrice de cellules logiques identiques interconnectées par programmation (reliées par des bus de communication configurables). Le principe de base est simple : afin d'implanter une fonction logique dans un FPGA, il suffit de configurer les cellules logiques et de les relier correctement en utilisant les bus internes. En effet, les FPGA sont les descendants de CPLD, ils sont simplement beaucoup plus complexes et puissants. Il y en a toutes les grosseurs, quelques milliers de portes logiques à quelques millions, ce qui permet d'implanter des circuits complexes sans avoir recours à concevoir un ASIC. En plus des cellules configurables, plusieurs FPGA possèdent des mémoires et des modules plus complexes comme les multiplicateurs. Certains fabricants vont même jusqu'à implanter des noyaux de microprocesseurs **[18]**.

Le coût unitaire d'un FPGA est plus élevé que celui d'un circuit dédié de la même densité. Cette différence de coût est largement compensée, pour un volume de production restreint, par la réduction du coût de conception pour réaliser le composant.

En plus d'une grande flexibilité qui résulte en un style de conception à faible risque où les conséquences des erreurs logiques sont faibles à la fois en coût et en délais de projet.

Les FPGAs sont donc, pratiques pour le développement rapide de produits et de prototypes. Ils permettent des cycles de conception très courts. En plus, ils sont totalement testés après la production, donc les designs ne demandent pas la génération de programmes de test du composant, ni la génération automatique des vecteurs de tests (ou le design pour la testabilité). Ils nécessitent seulement des tests de type fonctionnel.

En ce qui concerne la vitesse, les FPGAs offrent des unités qui opèrent à des vitesses dépassant les 400 MHz dans plusieurs applications. Évidemment, les vitesses sont plus élevées que celles dans les systèmes réalisés en utilisant des circuits discrets, mais elles restent plus faibles que celles obtenues par les circuits dédiés. La raison principale provient de la programmabilité. Les interconnections programmables rajoutent des résistances aux chemins internes. La vitesse des FPGAs est adéquate pour la plupart des applications. Dans les cas critiques, les applications peuvent être accélérées par simple utilisation d'unités plus rapides, souvent sans modifier la conception du circuit. Avec les circuits dédiés, la situation est totalement différente. En effet, les nouveaux procédés de fabrication

nécessitent la réalisation de masques et augmentent le coût global et le délai de la production.

Le développement des FPGAs s'accompagne par une évolution constante des outils de conception. Ces outils de haut niveau restent abordables même par des petites entreprises de conception. Le temps de développement est composé uniquement du temps de simulation et de réalisation du prototype, tandis que les temps des autres phases nécessaires pour les circuits dédiés : génération des vecteurs de test / production des masques, fabrication des gaufres, mise en boîtier et le test final en manufacture, sont évités. Ce qui mène à un temps de développement pour les FPGAs mesuré en journées ou en semaines. Alors que pour les circuits dédiés les durées sont calculées en mois.

III.5.3.1 Historique

Les Premiers FPGA ont été introduits en 1984 par Ross Freeman, Bernie Vonderschmitt, et Jim Barnett fondateurs de la compagnie Xilinx. En 1985, Xilinx introduit sur le marché le premier FPGA, le XC2064. Depuis ce temps, plusieurs compagnies, à savoir Altera et Actel sont rentréés en concurrence avec Xilinx dans le domaine de fabrication des FPGA. A cet effet, nous constatons un fort développement et une nette amélioration des produits, Cependant la compagnie Xilinx reste le premier producteur dans la matière. La technologie des FPGA pour cette compagnie évolue en deux axes différents : les modèles VIRTEX et les modèles SPARTAN.

Aujourd'hui, on peut compter plusieurs versions de ces modèles, évoluant de quelques portes logiques jusqu'au modèle VIRTEX-5 qui peut en compter plus de 20 millions avec des fréquences d'horloge dépassant les 400 MHz et le modèle Spartan-2^E qui compte plus de 5 millions de portes logiques avec une fréquence d'horloge de 50 MHz.

III.5.3.2 Avantages et inconvénients des FPGAs

Les FPGA ont plusieurs avantages comparés à d'autres technologies, principalement **[11]**:

- Vu leur structure, toutes les fonctions peuvent être implémentées, même les plus complexe, y compris des microprocesseurs ou des DSP
- Peuvent être reconfigurés indéfiniment ;
- Fonctionnement parallèle ;

- Les pattes d'un FPGA sont généralement configurables et peuvent s'adapter à plusieurs protocoles de communication (TTL, PC, etc.) ;
- Leur programmation se fait par des langages évolués, tels que le VHDL, Verilog, System generator.

Le FPGA possède quelque inconvénients (qui ne sont pas très importants) tels que :

- La programmation de ces circuits nécessite un apprentissage du langage de programmation, bien que plusieurs études ont contribuées à générer le langage VHDL par des langages plus évolués, à savoir : le langage C ou Matlab ;
- Etant donné que les FPGA ne sont pas des circuits dédiés, les fonctions implémentées sont plus lentes et prennent plus d'espaces comparé à un circuit dédié, tel que, le DSP ;
- La représentation en point flottant est presque impossible.

III.5.3.3 Architecture Interne d'un FPGA

L'architecture typique d'un FPGA est représentée sur la Figure (III.4), l'unité logique de base d'un FPGA est le CLB(Configurable Logic Blocks). Ces unités sont arrangées en matrice, et chaque CLB est divisée en plusieurs sections identiques nommées tranches (Slices). Ces éléments logiques se basent sur des multiplexeurs (MUX), comme dans le cas du XC6200 **[19]**, ou sur des tables de conversion LUT(Look-Up Tables), comme dans le cas du XC35500E **[20]**. A cet effet, ces unités logiques sont typiquement capables de réaliser de la logique combinatoire ou séquentielle de différentes complexités.

En outre, des ressources d'interconnexion permettent de créer (d'assurer) des liaisons programmables entre les entrées sorties des CLBs et les blocs d'entrées/sorties IOB(Input Output Blocks).

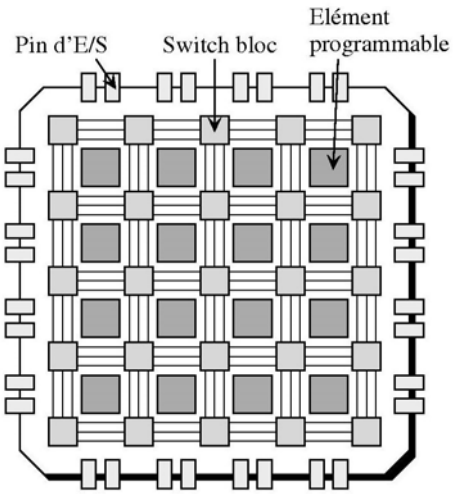


Figure (III.4) : Architecture générale d'un FPGA

III.5.4 Comparaison entre les composants programmables

Nous présentons sur la Figure III.5 une comparaison entre les PAL, les CPLD et les FPGA. Nous prenons en compte seulement les deux caractéristiques essentielles ; à savoir : la capacité d'intégration de chaque circuit et sa fréquence (sans faire allusion au coût de production et d'intégration, à la consommation du circuit, au volume et la fiabilité).

Les PAL sont des circuits très rapides et à capacité d'intégration très faible ce qui limite leurs utilisations à des applications simples, contrairement aux FPGA qui sont des composants moins rapides à grande capacité d'intégration ce qui les place dans la meilleure position pour l'implémentation des algorithmes complexes. Tandis que, les CPLD se placent entre les deux. En effet, pour les systèmes peu complexes, l'utilisation des CPLD est plus intéressante.

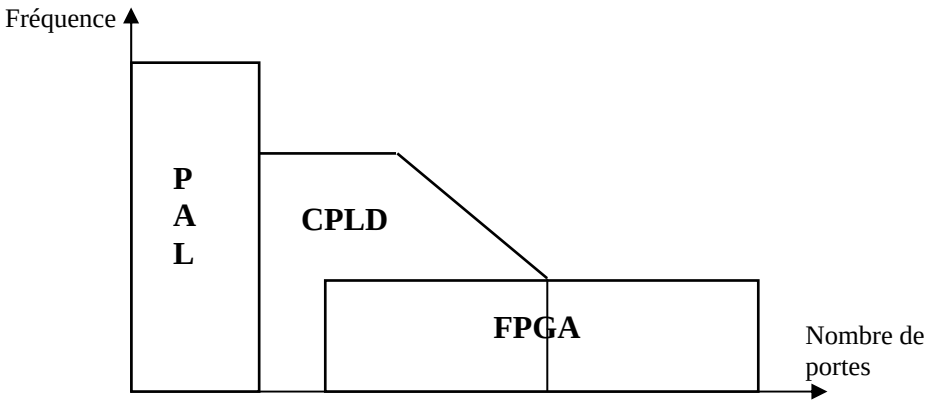


Figure (III.5) : Fréquence utile par nombre de portes

III.6 Flot de conception sur FPGA

Afin de programmer un FPGA d'une façon optimale, le concepteur doit respecter certaines étapes (voir Figure III.6)

1.Introduction de l'algorithme à implémenter : pour programmer les FPGA, le langage de description matérielle, telles que : le VHDL le VERILOG ou ABEL sont généralement utilisés.

a. Le VHDL : ce langage a été développé au début des années 80 par le Département USA de la défense. Il a été standardisé en 1987 par IEEE. Actuellement, il est le plus utilisé par les développeurs. Son principal rival est le VERILOG. Afin d'écrire un programme en VHDL, des logiciels, tels que : Xilinx de la société Xilinx ou Quartus de la société Altera sont généralement utilisés. Le code VHDL est très modulaire ce qui permet de réutiliser les fonctions déjà écrites et d'insérer des modules standard. Cependant, il demeure lourd et complexe pour quiconque n'est pas familier au VHDL. On note aussi que la simulation d'un code VHDL est trop longue. A cet effet, Xilinx commercialise un outil de conception de haut niveaux nommé System Generator (SG).

b. System Generator : le System Generator n'est pas un programme en soit, il s'agit plutôt d'une librairie pour le logiciel Matlab/Simulink. Sa force réside dans le fait, que n'importe quel utilisateur de Simulink peut facilement, en respectant quelques contraintes, implanter un algorithme sur un FPGA en utilisant les blocs fournis par Xilinx. En plus, il permet de faire une simulation fonctionnelle sur Matlab.

2. Simulation fonctionnelle n°1 : elle permet la vérification globale de l'algorithme programmé, dans ce cas les contraintes physiques des composants ne sont pas prises en compte. Les simulateurs de Xilinx ou Modelsim de la compagnie « montor graphic » peuvent être utilisés.

3. Simulation fonctionnelle n°2 : Le temps nécessaire pour fonctionnement de chaque opération dans l'algorithme est pris en compte. En cas de conflit, le concepteur doit revoir son programme.

4. Placement/routage : c'est une phase de synthèse logique, placement et routage. Cette phase peut être effectué d'une façon manuelle pour les très petits designs, ou par le logiciel « Xilinx Deseign Manager » pour les schémas les plus importants. Cependant, cette tache peut être très difficile (même pour le meilleur processeur existant) lorsque le design occupe presque la totalité du circuit. En effet, des modules interconnectés doivent être placés en

étant les plus proches possibles, afin de limiter les délais sur les fils, et le logiciel doit trouver le moyen de tous les connecter avec l'ensemble des fils de routage existants. Ou par le logiciel « Xilinx Deseign Manager ». C'est ce dernier qui est généralement utilisé.

5. Simulation temporelle : elle consiste à faire une simulation en prenant compte les contraintes de fonctionnement ainsi que des contraintes des composants à utiliser. Apres cette vérification le programme peut être implanter sur FPGA et tester.

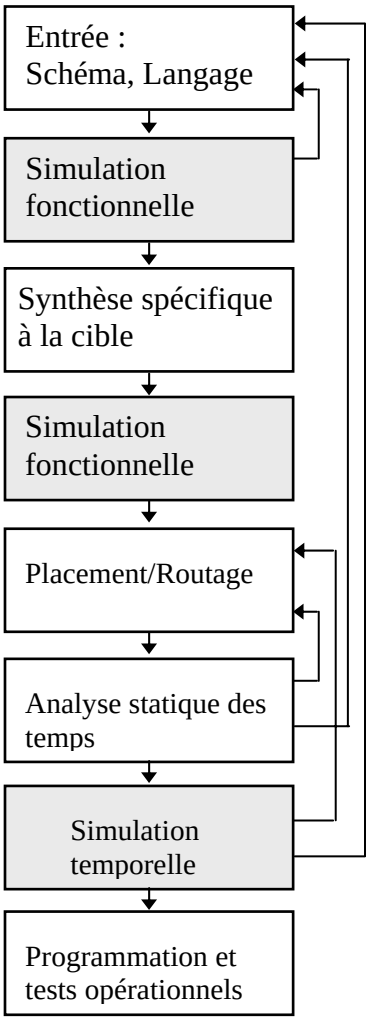


Figure (III.6) : Flot de conception

III.7 Choix de technologie

Sur le marché des FPGA, nous trouvons plusieurs types et plusieurs fournisseurs, entre autres : Xilinx, ALTERA, ACTEL, etc. Notre choix s'est orienté sur la compagnie Xilinx. Etant donné, la qualité et la disponibilité de ces composants. Par la suite, notre

choix à été guidé par des contraintes technico-économiques, en effet, nous avons choisi le circuit Xilinx Spatan-3E (le XC35500E implanté sur la carte SPARTAN-3^{E} STARTER KIT est de coût relativement faible comparé aux cartes Virtex). Sur le plan technique nous avons jugé que ce circuit est suffisant pour l'implémentation des RNA lorsque la conception est optimale.

III.7.1Architecture interne d'un XC3S500E

L'architecture générale d'un FPGA de la famille Spartan est représentée sur la Figure (III.7) Le circuit XC3S500E possède 1162 CLBs équivalente à 10476 unités logiques, et des IOBs ainsi que des blocs DCM (Digital Clock Manager Blocks) offrant un auto-calibrage, temporisation, multiplication, division et décalage de phase des signaux d'horloge **[20]**. Ce circuit contient également des blocs de RAM fournissant un stockage de données sous forme de blocs 18-Kbit avec un accès double et des blocs de multiplicateurs acceptant deux nombres binaires de 18 bit comme entrées et assurent le calcul du produit. Les FPGAs de la série SPARTAN peuvent aller jusqu'à une fréquence de 50 MHz.

Le Virtex-II offre la même architecture avec des technologies novelles, alors que la technologie de 65 nm est employée pour la fabrication des FPGA de la série Virtex-5 **[20]**. Ces dernières contiennent des blocs multiplieurs accumulateurs (MAC) de 25×18bits, des blocs de RAM de 36 Kbits et jusqu'à 640 "550 MHz DSP48E Slice".

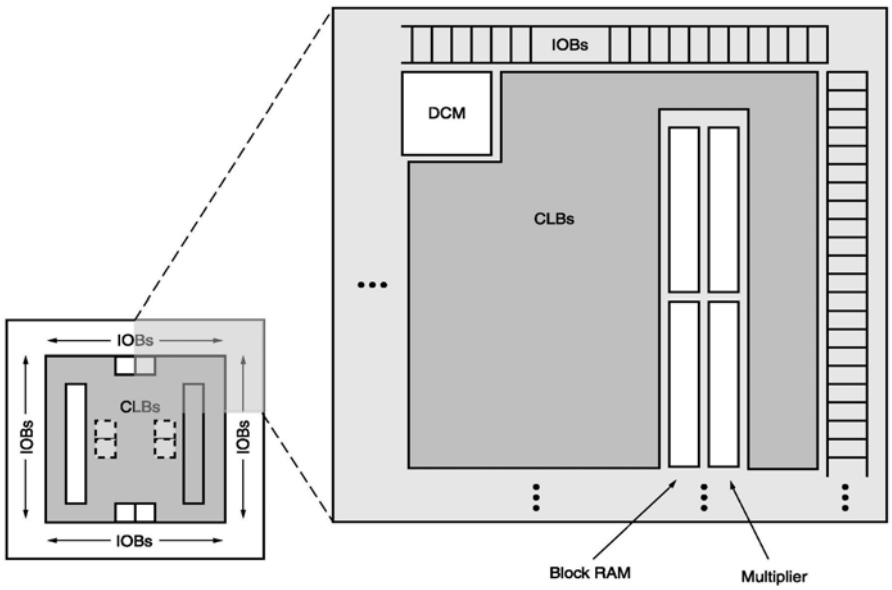


Figure (III.7) : Architecture générale de la série Spartan-3E

III.7.2 Les CLBs

Dans le XC3S500E Les CLBs constituent la principale ressource de la logique séquentielle et combinatoire. Chaque CLB contient quatre tranches; et chaque tranche en contient deux LUTs pour implémenter les fonctions logiques et deux éléments de stockage qui peuvent être utilisés comme flip-flop ou latches. Le LUTs peut être utilisé comme une mémoire 16x1 (RAM16) ou comme un registre du changement de 16 bit **[19]**.

Les tranches sont groupées dans des paires. Chaque paire est organisée comme une colonne. La paire gauche supporte la logique et la fonction mémoire et ses tranches sont appelées SLICEM. La paire droite, supporte seulement la logique et ses tranches sont appelées SLICEL. Par conséquent, un demi des LUTs supporte logique et mémoire (RAM16 et registres SRL16) tandis que l'autre demie supporte seulement la logique.

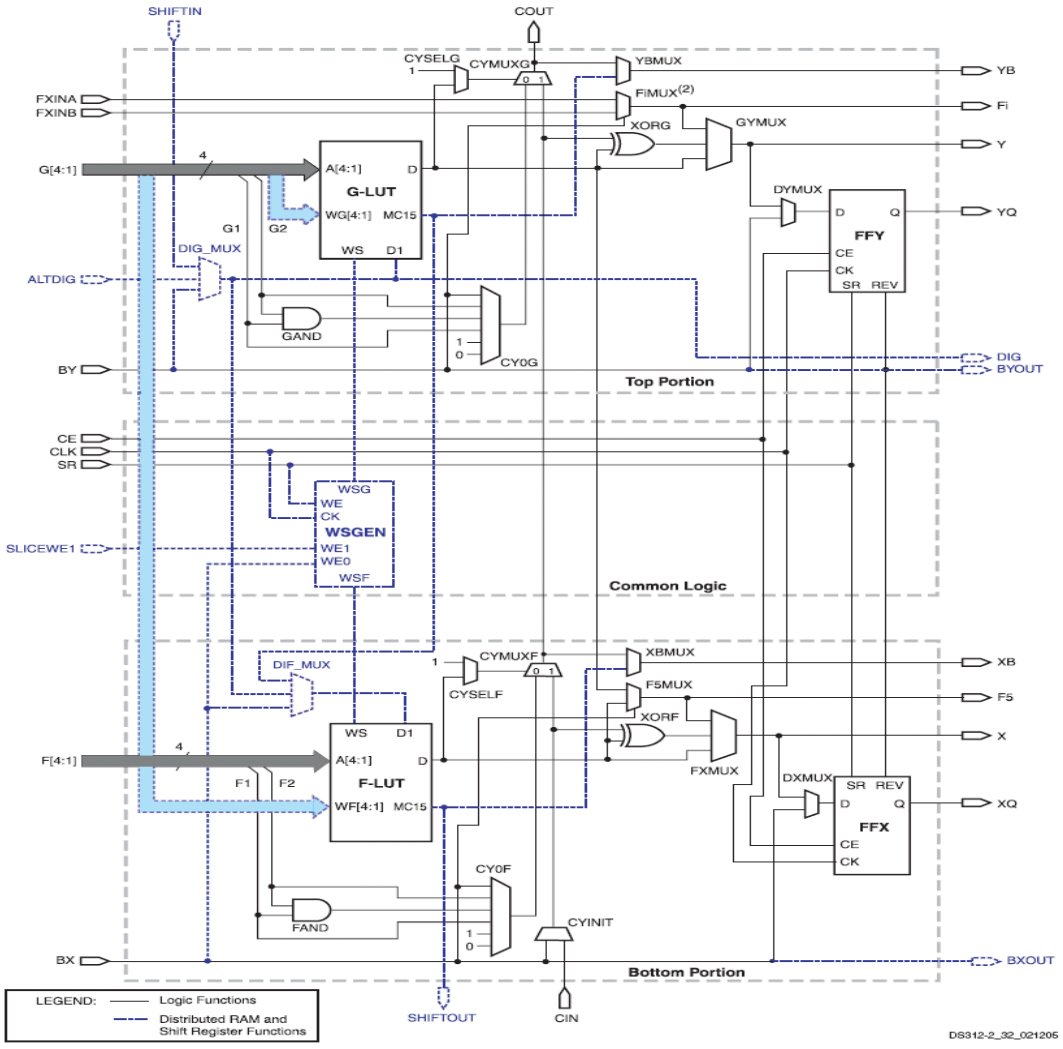


Figure (III.8) Partie gauche de la Tranche le SLICEM

Le SLICEL réduit la dimension du CLB et baisse le coût de l'appareil et peut fournir aussi des performances supérieures par rapport au SLICEM. Figure (III.8) présente le SLISEM **[19]**.

La technologie du Virtex- II XSV300 ou celle du XCV600 est très semblable à celle décrite, cependant, le nombre des CLBs dans ce cas est plus important.

III.7.3 Les IOB

Les blocs IOB sont programmables, leurs niveaux peuvent être programmés pour être compatible avec treize standards différents. Tous le E/S sont protégées contre les décharges électrostatiques et les surtensions transitoires Figure (III.9). Chacun des signaux d'entrées et de sorties est associé à une bascule D, un amplificateur pouvant positionner la sortie et munie d'un signal de contrôle direct ou pouvant passer par une bascule.

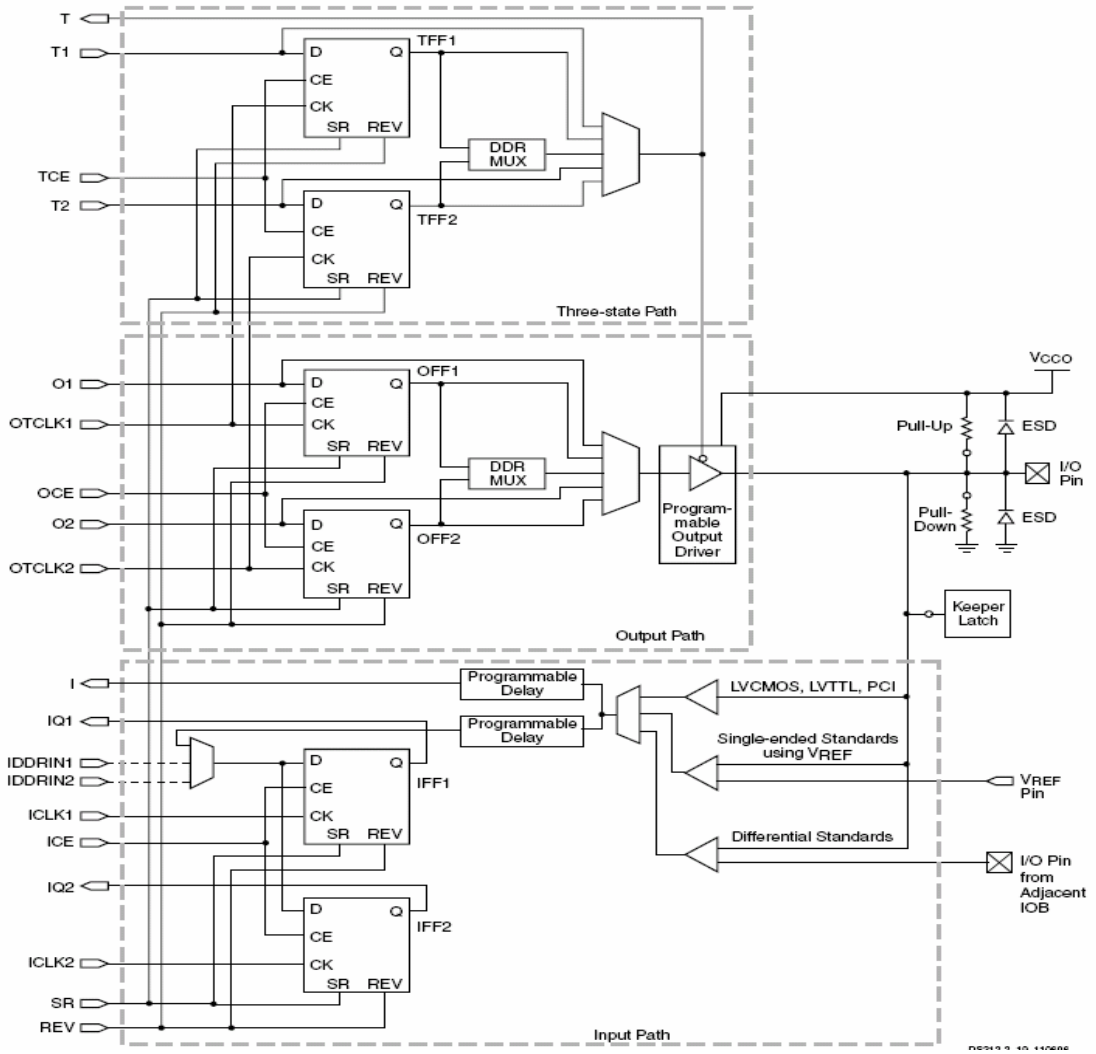


Figure (III.9) : Schéma simplifier d'un IOB du Spartan

III.7.4 Système d'interconnexion

Le routage est basé sur une architecture hiérarchique très élaborée. Cette structure comporte plusieurs niveaux. On retient les quatre principaux :

- Routage local : Il permet I'interconnexion entre les LUTs d'un même CLB, entre ces LUTs et les matrices de routage global appelées GRM (Global Routing Matrix), et, également, entre les CLBs horizontalement adjacents ;

- Routage à usage général : Représenté par les canaux de routage horizontaux et verticaux qui relient les lignes et les colonnes de CLBs. La majorité des interconnexions est située à ce niveau de la hiérarchie ;

- Routage dédié : Chaque rangée de CLBs dispose de quatre longues lignes horizontales pouvant se relier pour réaliser les bus à trois états. Chaque CLB contient deux lignes verticales dédiées pour la propagation de la retenue ;

- Routage global : Permet la distribution des signaux d'horloge ainsi que les signaux alimentant des grosses charges.

III.8 Conclusion

Dans ce chapitre, nous avons présenté les différents supports physiques existants sur le marché actuel. Pour ce faire, on a effectué un survol des circuits à fonctions programmables, suivi d'une analyse sur les circuits à configuration programmable. Nous avons prêté une attention particulière aux FPGAs, qui sont les circuits reconfigurables les plus flexibles et offrant le plus de fonctionnalités.

A partir de l'analyse des caractéristiques et des performances des différentes structures, nous constatons que les FPGA sont les meilleurs candidats pour la réalisation rapide d'un système de diagnostic des défaux, de par leur capacité à être reconfigurés, un très grand nombre de fois et de par leurs fonctionnement imitant le fonctionnement de l'électronique analogique ainsi de par les fonctions dont ils disposent (bloc RAM et Multiplieurs).

Par ailleurs, nous constatons que l'exploitation rationnelle des FPGA passe par la connaissance de leurs architectures internes et la maîtrise des moyens de programmation de ces circuits. En effet, les FPGAs sont des circuits flexibles à vitesse relativement grande et à grande densité d'intégration avec un coût qui décroît de plus en plus. Leur flot de

conception est aussi simple surtout lors de l'utilisation du System Generator. Les avantages suscités font des FPGA le support physique programmable le plus adéquat pour la réalisation de nos objectifs.

A cet effet, dans le chapitre 4, l'implémentation des RNA sur un FPGA de type Spartan sera considérée. La conception de l'algorithme à implémenter sera effectuée en utilisant le System Generator.

III.1 Introduction

L'évolution technologique de la microélectronique a permis de voir naître un nouveau type de composants électroniques, dits circuit programmable. Cette évolution, aujourd'hui en pleine expansion et progression offre de plus en plus des circuits programmables puissants avec une grande flexibilité et une grande rapidité de fonctionnement. Cette grande puissance fait de ces circuits une solution technologique (support physique) incontournable pour l'implémentation des différentes techniques de diagnostic.

De plus les circuits programmables qui sont de grande simplicité de conception, bénéficient des avantages de l'électronique analogique et de ceux des microprocesseurs. En effet, bien qu'ils aient les mêmes caractéristiques de l'électronique analogique, ils permettent de palier à plusieurs cas de contraintes temporelles.

Nous traitons dans ce chapitre les différents supports physiques permettant l'implémentation des RNA, cependant notre étude sera aiguillée vers les circuits programmables du fait que les FPGA (Field programmable gate array) sont choisi comme technologie cible pour la réalisation du présent travail. Aussi, les différentes méthodes de programmation de ces circuits seront présentées.

III.2 Différents supports physiques [11]

Les support physique d'implémentation se concrétise sous forme d'un ou de plusieurs composants, ces composant peuvent être regroupés sous deux grandes catégories : Analogique et Numérique (voir Figure III.1). La forme numérique est très riche et diversifiée vu l'évolution croissante de la microélectronique, cette forme peut être repartie en trois groupes essentiels ; les composant standards, tels que les microprocesseurs et les DSP (Digital Signal Processing), les composants PLD (Programmable Logic Devices) regroupant les différents circuits programmables tels que les FPGA et les CPLD (Complex Programmable Logic Device) et les composants d'ASIC (Application Specific Integrated Circuits numérique).

Chapitre IV :

Implémentation des RNA sur FPGA pour la détection des défauts de la MAS

IV.1 Introduction

L'objectif de ce chapitre est l'implantation de réseau de neurone artificiel sur une circuit FPGA. Cette implémentation à pour but d'étudier l'apport de solutions d'intégration matérielle (FPGA) dans le diagnostic des défaillances du moteur asynchrone.

Dans cette étude, nous commençons par l'adaptation de RNA afin de permettre une implémentation optimale. Cette implémentation doit assurer une efficacité, une rapidité d'exécution et un minimum possible d'espace sur le circuit FPGA.

Ensuite nous programmons cet RNA sur le System Generator. Le system Generator permet de générer le code VHDL. Ce code est vérifié et implémenté sur un circuit FPGA de type Spartan par le logiciel ISE fundation de Xilinx (voir Figure (IV.1)) .

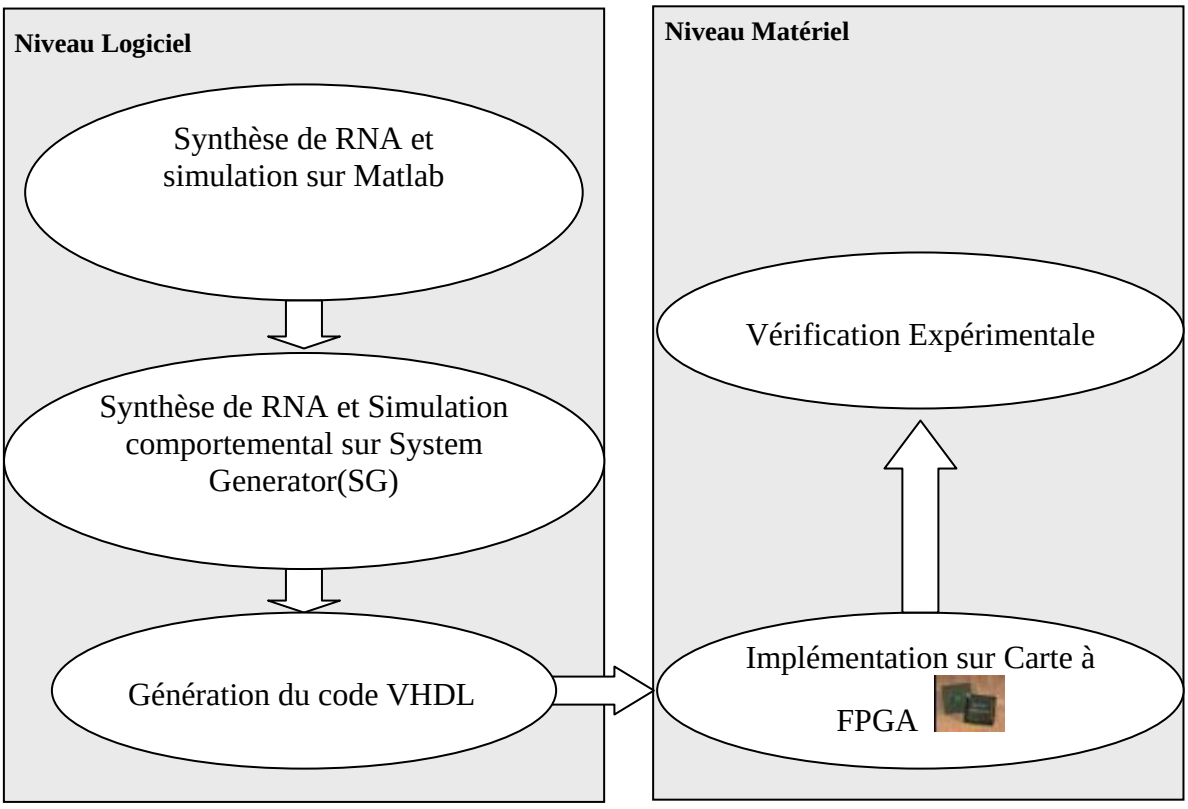


Figure (IV.1): Phases d'implémentation sur FPGA

IV.2 **Synthèse de RNA et simulation sur Matlab**

Pour assurer une efficacité, une rapidité d'exécution et un minimum possible d'espace sur le circuit FPGA, on se propose d'étudier un réseau qui contient deux couches cachées le première couche possède trois neurones et la seconde possède deux neurones, et une couche de sortie possède deux neurones, en plus nous choisissons les entrées et les sorties comme suit :

- trois entrées sont les valeurs efficaces (I_{as}, I_{bs}, I_{cs})
- deux sorties (S1,S2), Les défauts sont représentés sur le tableau (IV.1), avec leurs symboles et leurs codes associés.

Tableau (IV.1) : classification des défauts

catégorie	Type de défaut	symbole	code	
			S1	S2
01	Etat sain	ES	0	0
02	Coupure monophasée	CM	1	0
03	Coupure biphasée	CB	0	1

IV.2.1 Résultats d'essais de réseau

On a effectué un apprentissage automatique à l'aide du logiciel MATLAB jusqu'à où on obtient une erreur quadratique la plus petite (2.7401 e-015) après 202 étérations, (voir Figure IV.2).

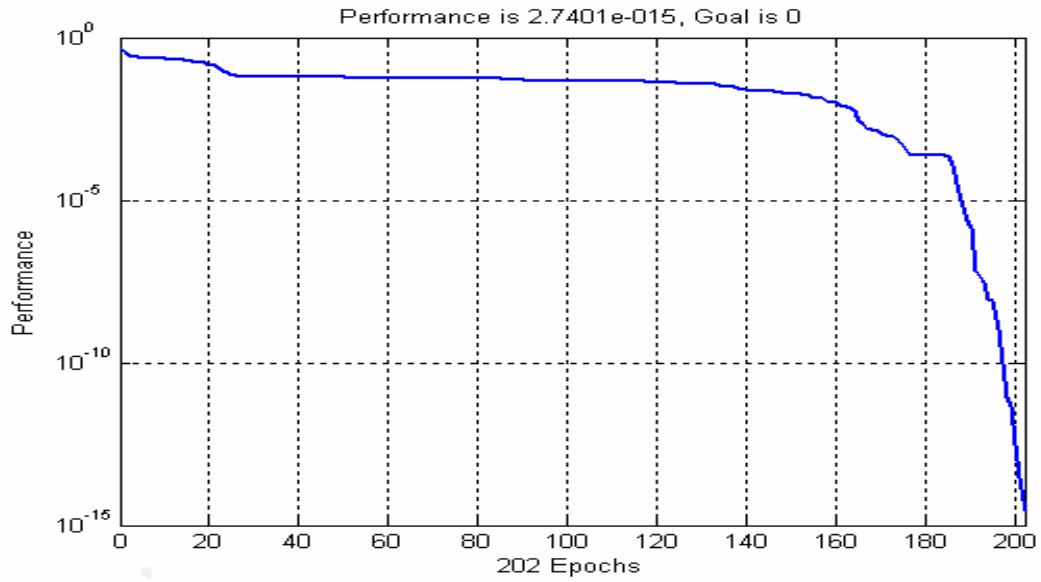


Figure (IV.2): Evolution de l'erreur quadratique moyenne du RNA

IV.2.2 la conception de RNA sous Simulink

La conception de RNA sous Simulink est représentée sur la figure (Figure IV-3)

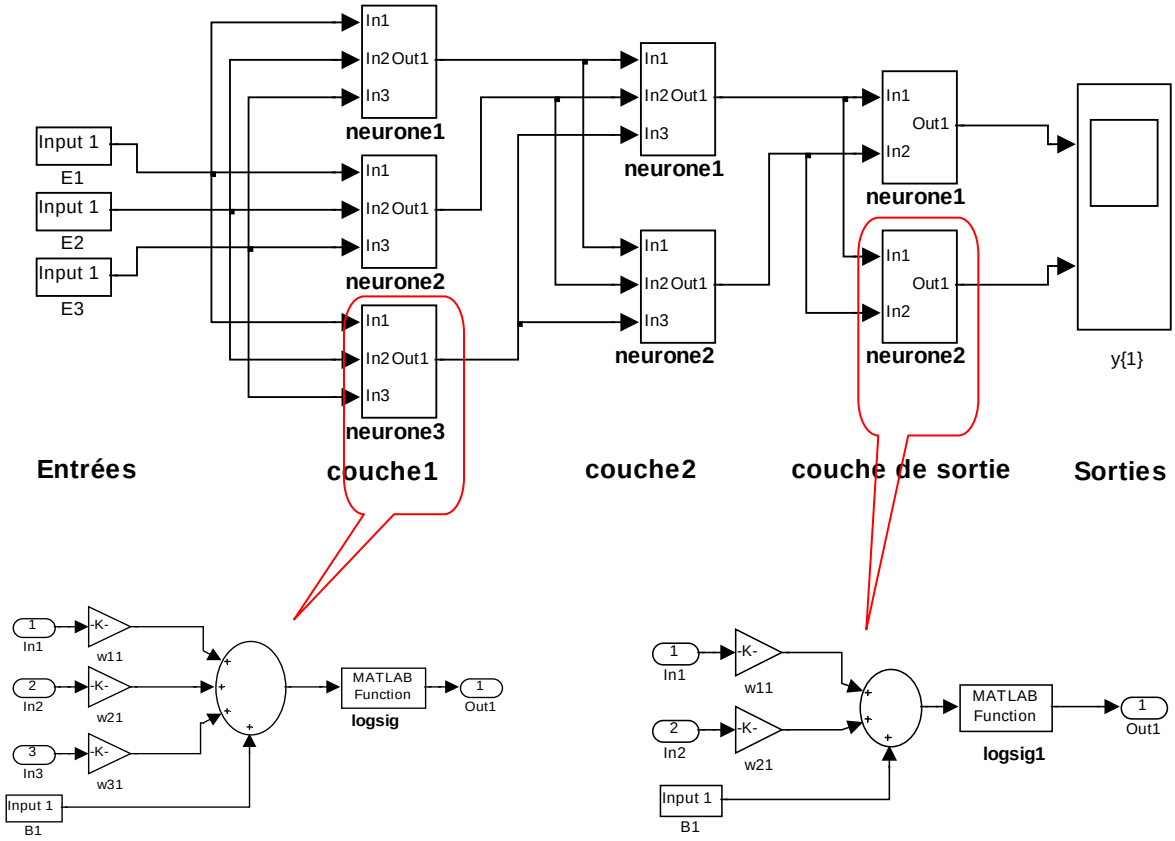


Figure (IV.3): Réseau de neurone artificiel fait par simulink

IV.2.3 Test de RNA

Une fois que le RNA construit et que sont apprentissage a atteint des performances Satisfaisantes, On fait des tests, leurs résultats sont présentés dans le tableau IV.2.

Tableau (IV.2): Résultats de simulation (test) du RNA pour différents cas

Sortie du RNA	Etat sain	Coupure monophasée	Coupure biphasée
S1	1.2222 e-018	1.0000 e+000	6.2836 e-016
S2	1.0000 e+000	1.6661 e-007	1.0000 e+000

D'après les résultats obtenus dans la phase du test, on constate que les sorties de RNA suivent presque conformément (avec erreur de e-008) les sorties désirées préétablies auparavant.

IV.3 Synthèse de RNA et Simulation sur le System Generator

IV.3.1 Aperçu sur le System Generator

Dans ce paragraphe nous allons présenter un bref aperçu sur le System Gernrator 8.2.02. Ce dernier est compatible avec le Xilinx fundation 8.1 et Malab R2006a.

Le System Generator de Xilinx est un logiciel qui s'intègre à Matlab. Il se compose de deux sections : une librairie de blocs pour Simulink et un logiciel qui transforme ces blocs en code utilisable pour programmer un FPGA.

La librairie Simulink, fournie par Xilinx, fonctionne selon le même principe que les autres éléments de Simulink. Elle contient une série de blocs représentant plusieurs fonctions et peuvant être reliés entre eux pour former des algorithmes **[21]**. Les blocs de cette librairie ne servent pas seulement à simuler, mais permettent également de générer les codes VHDL ou Verilog.

-Principales fonctions disponibles

Les blocs du System Genrator sont organisés en cinq groupes selon leurs fonctions. Ce sont des blocs simples qui imitent des fonctions déjà présentees dans Simulink.

Le tableau IV.3 présente les principaux blocs séparés par section. Notons que ces sections sont séparées en réalité sur Simulink en dix groupes, nous présentons seulement les sélections les plus essentielles.

Tableau (IV.3) : Principaux blocs du System Generator

Basic Element	Math	Matlab IO	Memory	**DSP**
System Generator	**Accumulator**	**Display**	**Addressable Shift**	**DAFIR**
Black Box	**AddSub**	**Enable**	**Rgister**	**DDS**
Assert	**Cmilt**	**Adapter**	**Dual Port Ram**	**FFT**
BitBasher	**Constant**	**Gateway In**	**FIFO**	**FIR**
Contact	**Convet**	**Gateway Out**	**LFSR**	**LFSR**
Clock Enable Probe	**Counter**	**Quantization**	**ROM**	**opmod**
Delay	**Expression**	**Error**	**Shared Memory**	
Down Sample	**Inverter**	**Sample Time**	**Single port Ram**	
Get Valid Bit	**Logical**	**Simulation**		
Mux	**MCode**	**Multiplesxer**		

Register	**Mult**			
Set Valid Bit	**Negate**	**WaveScop**		
Slice	**Relational**	**ModelSim**		
Sunc	**Scale**			
Time division	**Shift**			
Multiplexer	**SineCosine**			
Time division	**Treshod**			
demultiplex				
Up Sample				

Les éléments de base comportent des modules simples dont les noms décrivent très bien les fonctions. Le bloc « System genrator » est spécifique, il sert à transformer le modèle en code VHDL ou Verilog en utilisant le 'HDL Netlist', où le pas d'horloge ainsi que le type de circuit FPGA utilisé doivent être déclarés. Il permet aussi de faire une analyse approximative des temps consommés par chaque composant de l'algorithme construit.

IV.3.2 la conception de RNA sous Xilinx

IV.3.2.1 La modélisation de La fonction segmoide

La librairie Simulink fournie par Xilinx, dispose de tous les blocs qui sont nécessaires pour la conception de RNA sauf la fonction segmoide, qu'elle n'existe pas.

C'est pour cela et d'après **[22]**, Marco a proposé une meilleure exécution de la fonction segmoide par Séries de Taylor, La fonction résultante (IV.1) provoque une erreur de 0,51% en ce qui concerne le modèle continu.

$$g(x)=\begin{cases} 0.571859+(0.392773)x+(0.108706)x^2+ & \\ (0.014222)x^3+(0.000734)x^4 & -\infty < x \le -1.5 \\ \frac{1}{2}+\frac{1}{4}x-\frac{1}{48}x^3+\frac{1}{480}x^5 & -1.5 \le x < 1.5 \\ 0.428141+(0.392773)x-(0.108706)x^2+ & \\ (0.014222)x^3-(0.000734)x^4 & 1.5 \le x < \infty \end{cases}$$

(IV.1)

La modélisation de La fonction résultante sous Xilinx est donnée sur Figure (IV.4):

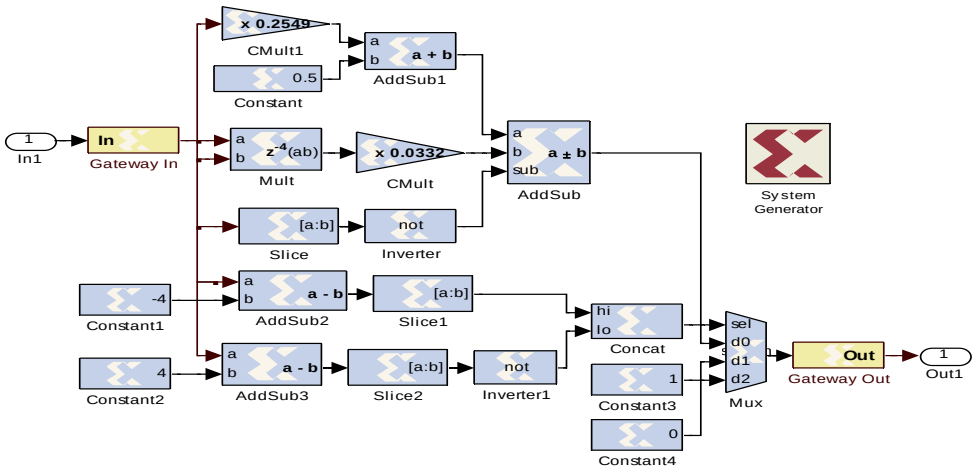


Figure (IV.4): modélisation de la fonction segmoide

Résultat de simulation représenté sur la figure (IV.5) :

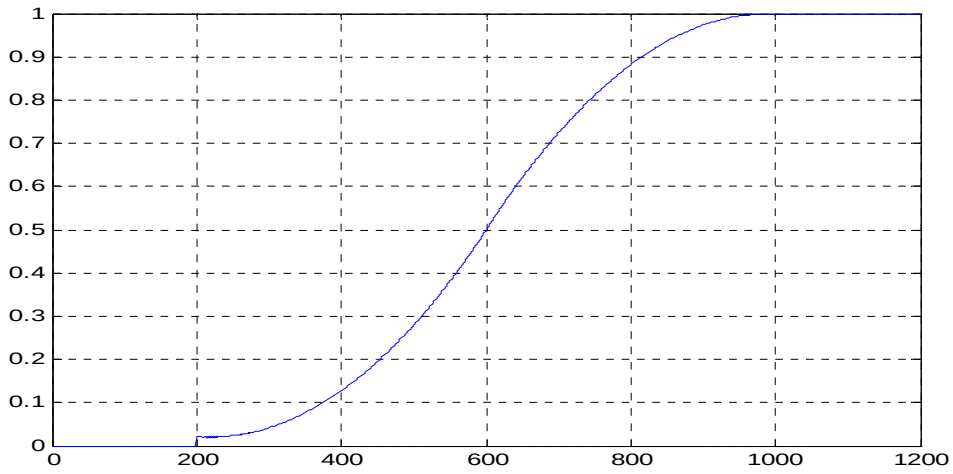
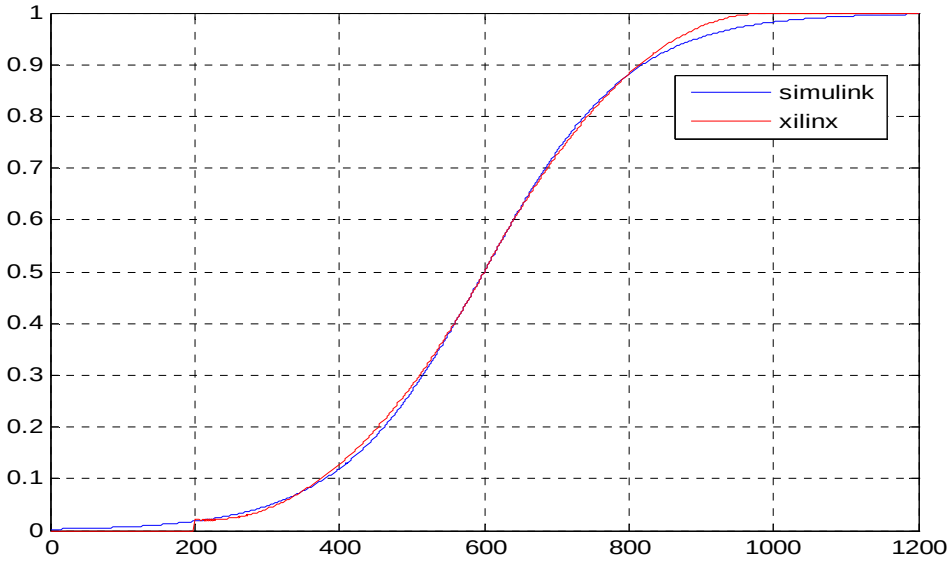


Figure (IV.5): approximation de fonction sigmoïde par série de Taylor

IV.3.2.2 Comparaison de la fonction sigmoide de simulink avec celle fait au xilinx

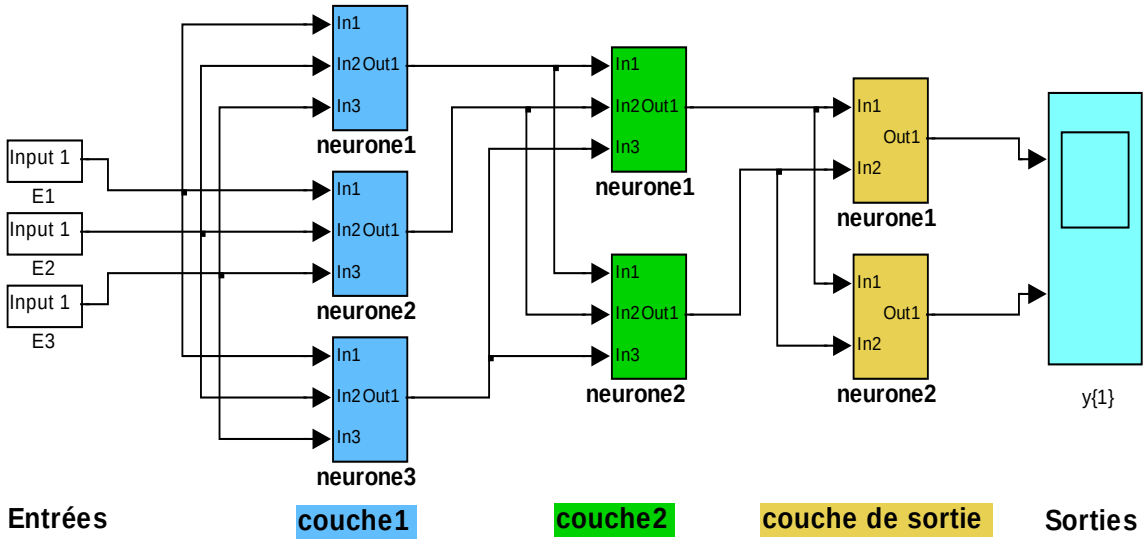
D'après la figure (IV.6) on remarque que la courbe de fonction sigmoide obtenue sous Simulink presque conformément avec la courbe de la fonction sigmoide obtenue sous xilinx



Figure(IV.6): Comparaison de la fonction sigmoide de simulink avec xilinx

IV.3.2.3 La modélisation de RNA sous Simulink

La conception de RNA sous Simulink est représenté sur la Figure(IV.7)



Figure(IV.7): Réseau de neurone artificiels sous xilinx

Le Contenu de chaque couche est donné comme suit :

- **couche cachée (couche 1) :** représenté sur la Figure (IV.8)

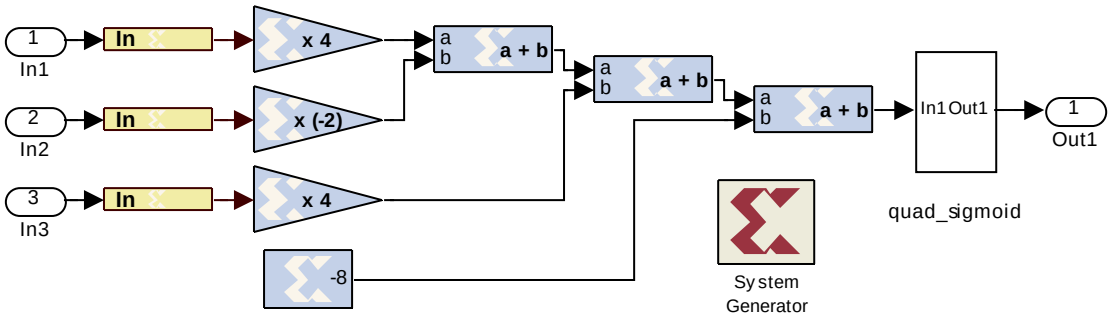


Figure (IV.8): couche cachée (couche 1) sous xilinx.

- **couche cachée (couche 2) :** représentée sur la Figure(IV.9)

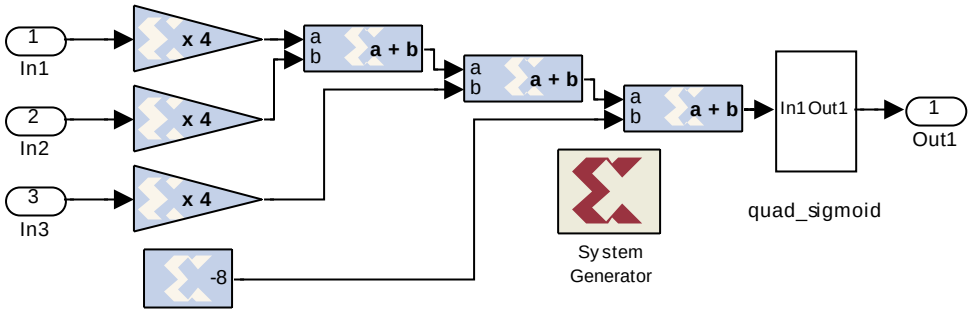


Figure (IV.9): couche cachée (couche 2) sous xilinx.

- **couche de sortie :** représentée sur la Figure(IV.10)

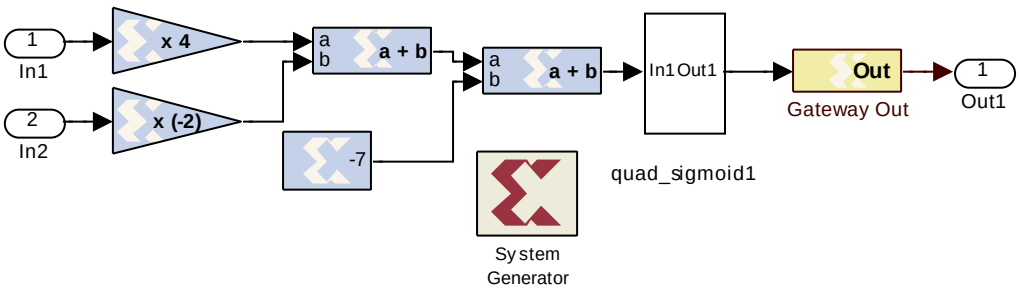


Figure (IV.10): couche de sortie sous xilinx.

IV.4 Génération du code VHDL de bloc RNA

D'après la construction de RNA sous Xilinx le System Generator permet également de générer le code VHDL.

- On place tout le contenu de chaque couche du bloc de RNA dans le même subsystem , avec un seul system generator **(**voir l'annexe 03), Ensuit on génère le code VHDL avec les caractéristiques suivantes :

- **Compilation: HDL Netlist**
- **Part: Spartan2 xc2s200e-6tq**
- **Synthesis Tool: XST**
- **Target Directory: Votre choix de répertoire**
- **Create Testbench:** Checked
- **FPGA System Clock Period (ns): 10**

- On Ouvrent le bloc **System Generator :**

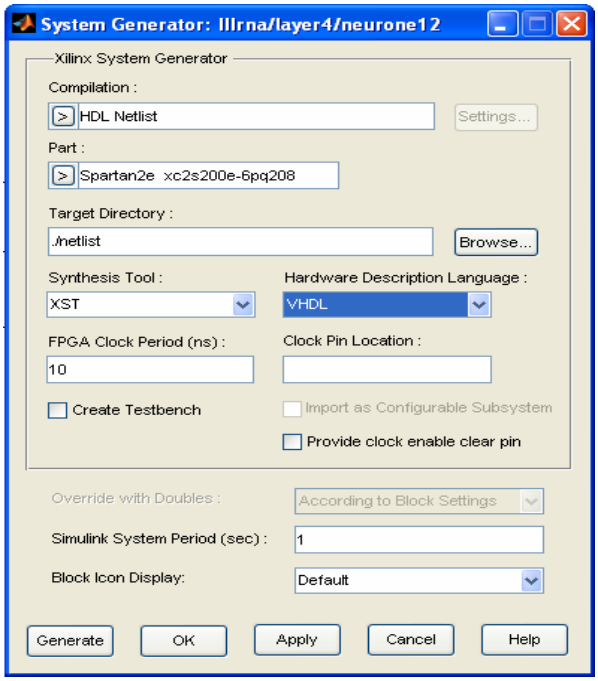


Figure (IV.11): la fenêtre de system Generator

- On choisi un répertoire de travail désire dans **Target Directory.** On met. /netlist qui va créer un sous répertoire dans le répertoire courant.
- On Choisi les paramètres suivant dans le reste de la fenêtre :
  - **Compilation: HDL Netlist**
  - **Part: Spartan2 xc2s200e-6tq**
  - **Synthesis Tool: XST**
- On met **10** dans **FPGA System Clock Period (ns).** Ceci sera la contrainte d'horloge pour le logiciel ISE.
- A la fin on Cliqué sur **Generate ,** qui donne le programme en **VHDL .**

La figure (IV.12) présente la résultat obtenu :

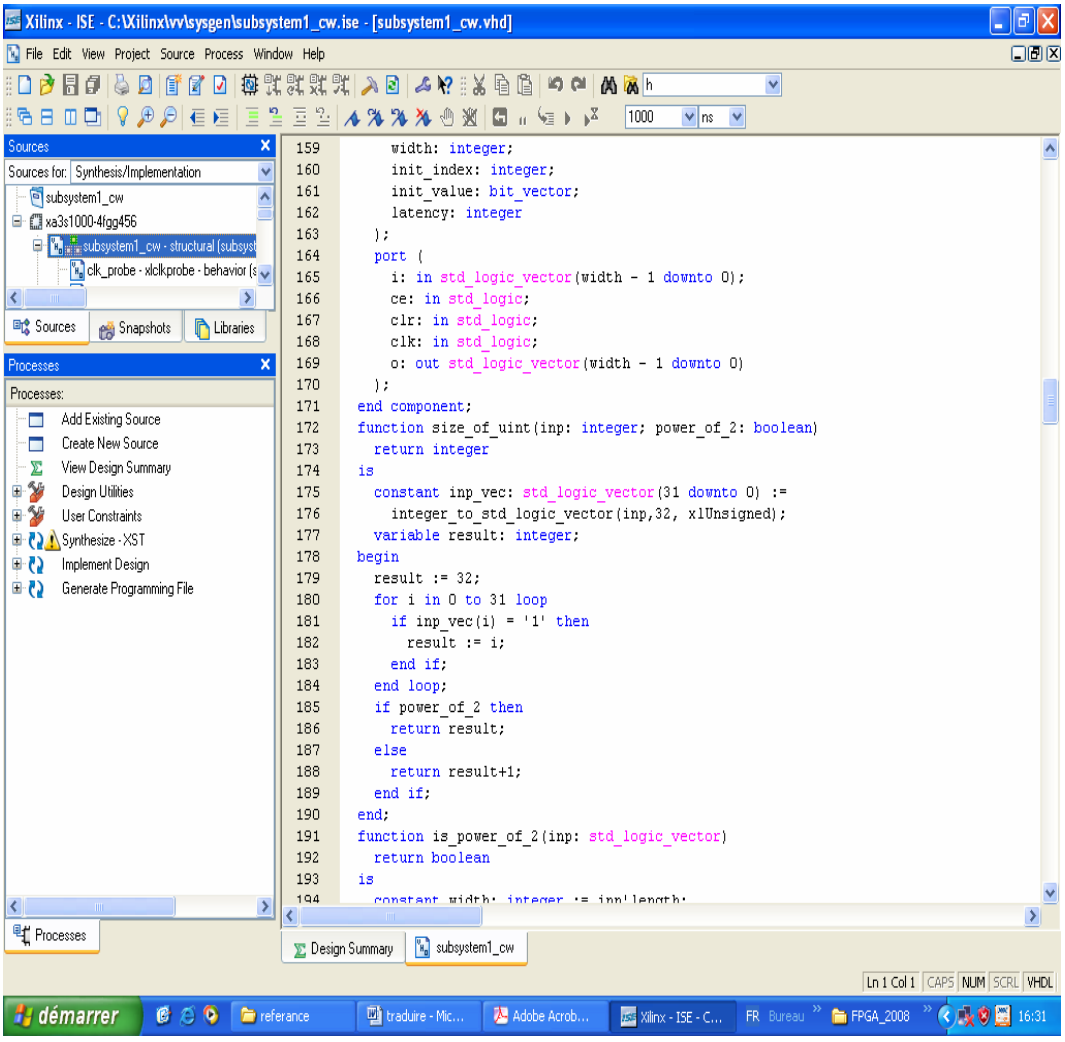


Figure (IV.12) : la résultat de code VHDL obtenu

IV.5 **Implémentation de RNA sur FPGA**

Cette partie est consacrée à la description de l'implantation de RNA décrit sur FPGA. Pour ce faire, nous utilisons le SystemGenrator de Xilinx.

Synthésize : on Cliqué sur **Synthésize** pour connaître l'espace **des ressources occupées par** le code VHDL voir la figure (IV.13)

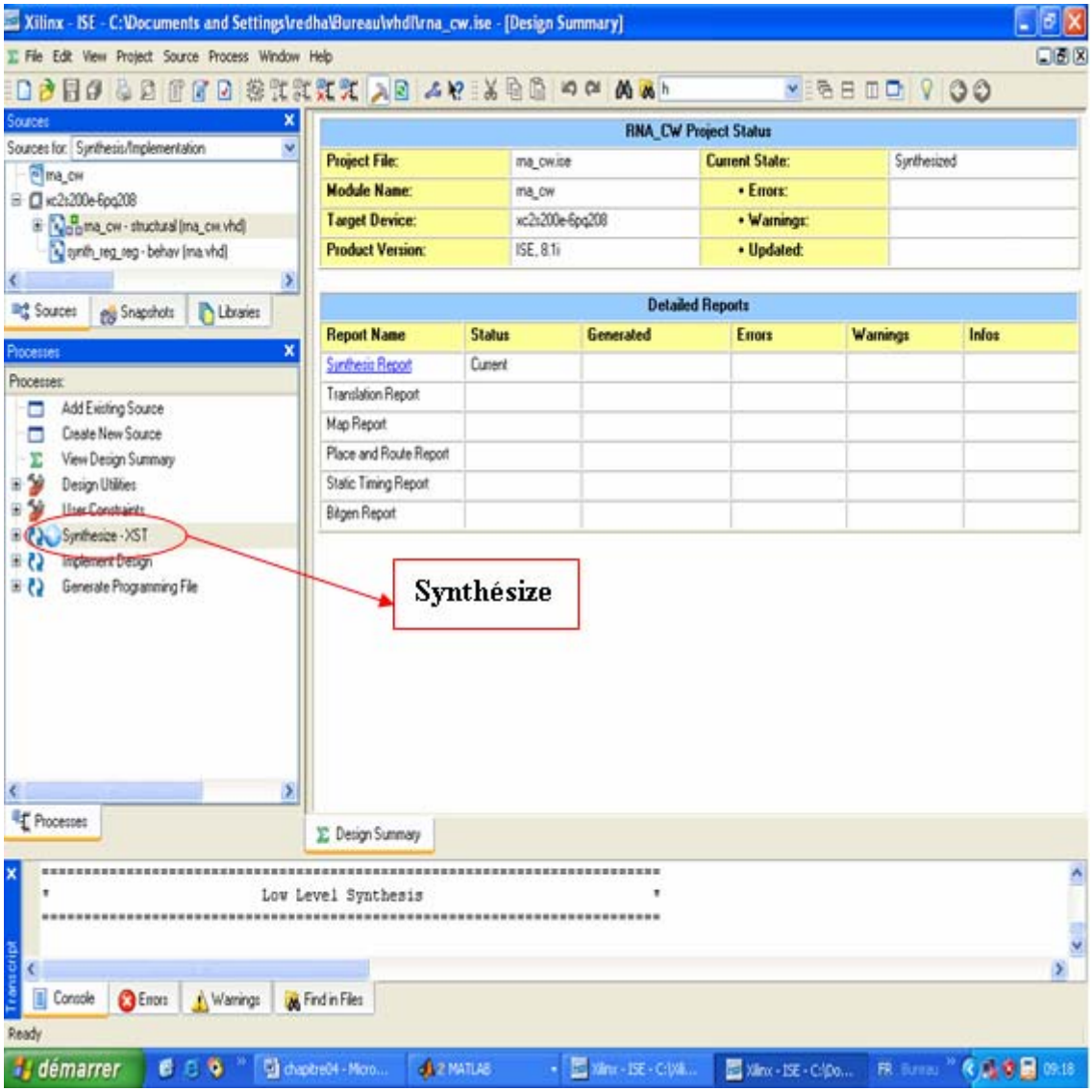


Figure (IV.13) la réalisation de la tàche Synthésize

Le résultat de Synthésize est affiché dans le tableau (IV.4).

Tableau (IV.4) : Résumé de l'espace utilisé sur le FPGA

RNA_CW Project Status			
Project File:	rna_cw.ise	**Current State:**	Synthesized
Module Name:	rna_cw	**• Errors:**	No Errors
Target Device:	xc2s200e-6pq208	**• Warnings:**	775 Warnings
Product Version:	ISE, 8.1i	**• Updated:**	

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	4895	2352	208%
Number of Slice Flip Flops	4777	4704	101%
Number of 4 input LUTs	8820	4704	187%
Number of bonded IOBs	167	146	114%
Number of GCLKs	1	4	25%

Detailed Reports					
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current		0	775 Warnings	15 Infos
Translation Report					
Map Report					
Place and Route Report					
Static Timing Report					
Bitgen Report					

D'après les résultats obtenus sur le tableau on remarque que le code VHDL occupe un espace de 208% sur FPGA , signifie que la circuit FPGA de type Spartan2 xc2s200e-6tq ne supporté pas cet code VHDL.

Par ailleurs, l'implémentation du même code obtenu sur un circuit Spartan3 donnera les résultas suivants

Tableau (IV.5) : Résumé de l'espace utilisé sur le FPGA

SUBSYSTEM1_CW Project Status			
Project File:	subsystem1_cw.ise	**Current State:**	Synthesized
Module Name:	subsystem1_cw	**• Errors:**	No Errors
Target Device:	xc2s200e-6pq208	**• Warnings:**	775 Warnings
Product Version:	ISE, 8.1i	**• Updated:**	

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	4895	7680	63%
Number of Slice Flip Flops	4777	15360	31%
Number of 4 input LUTs	8820	15360	57%
Number of bonded IOBs	167	333	50%
Number of GCLKs	1	8	12%

D'après ce tableau, on constaté le code implémenté sur Spartan3 a occupé uniquement 63% de l'espace mémoire sur FPGA.

IV.6 Conclusion

Dans ce chapitre, nous avons proposé un algorithme simple pour l'implémentation de la RNA. La synthèse matérielle de l'algorithme proposé, est effectué par le Systeme Generator. Le routage et l'implantation sur FPGA de type Spartan2E a été effectué par le ISE fundation.

L'utilisation de l'outil de conception haut niveau 'System Generator' est très bénéfique pour la vérification du comportement de l'algorithme sur Simulink. Les simulations effectuées pour la fonction segmoide montrent que les résultats obtenus sous Xilinx ont donné les mêmes performances que la fonction segmoide obtenus sous Simulink.

Par ailleur, l'implémentation du même code obtenu sur deux circuits differents Spartan2 et Spartan3 signifie que la circuit FPGA de type Spartan3 supporté cet code VHDL et donne des resultats adéquates.

Conclusion générale

Conclusion générale

Dans ce travail on a étudié une tache principale de la surveillance, qui est le diagnostic (détection et localisation), des défauts dans une MAS alimenté par un onduleur de tension.

Nous avons commencés par la représentation des différents défauts des machines asynchrones et leurs causes. La procédure de diagnostic a été expliquée avec la classification de ses méthodologies, après lesquelles on a montré que la technique des RNA permet de surveiller les chaînes de production beaucoup mielleux que l'automatisme ordinaire telles que les méthodes internes.

Ensuite, on a fait la modélisation de l'ensemble moteur-convertisseur statistique et leur commande par la suite l'application de RNA pour la détection des différentes défaillances qui peuvent surgir sur cet ensemble qui a donné des résultats efficaces et performants(en terme de rapidité et de précision).

Par ailleurs, dans l'objectif de donner suite à une implémentation des RNA pour avoir une plate forme expérimentale, Nous avons traité les différents supports physiques permettant l'implémentation de n'importe qu'elle fonction que ce soit simple ou complexe.

Cependant, il a était constaté que les FPGA sont les meilleurs candidats pour la réalisation rapide d'un système de diagnostic par RNA, de par leur capacité à être reconfigurés, leur fonctionnement est en parallèle, en plus leur programmation se fait par des langages évolués, tels que le VHDL.

En fin, on peut conclure que l'implémentation des fonctions sur FPGA est la plus simple car elle utilise non seulement un langage évolué tel que le VHDL, mais aussi le logiciel MATLAB nous offre la possibilité de simuler n'importe qu'elle fonction sur simulink, en utilisant la libraire de xilinx (systemGenerator) pour que cette dernière soit directement transformée en langage VHDL.

Notations et Symboles

Notations et symboles utilisés

.

Symbole	**Signification**	**Unité**
MAS	Moteur asynchrone	
s,r	Indices respectifs du stator et du rotor	
R_s	résistance propre d'une phase statorique	Ω
R_r	résistance propre d'une phase rotorique	Ω
L_r	Inductance propre d'une phase rotorique	H
L_s	inductance propre d'une phase statorique	H
M_s	inductance mutuelle entre deux phases du stator	H
M_r	Inductance mutuelle entre deux phases du rotor	H
M_{sr}	maximum de l'inductance mutuelle entre une phase du phase correspondante du rotor	H
L_{sr}	Inductance mutuelle entre rotor et stator	H
α	Ecart angulaire entre les axes des phases du stator et du	Degré
V_{sa},V_{sb},V_{sc}	tentions des phases S_a ,S_b,S_C	V
V_{ra},V_{rb},V_{rc}	tentions des phases R_a,R_b,R_c	V
i_{as},i_{bs} ,i_{cs}	courants statorique des phases S_a ,S_b,S_c	A
i_{ar} ,i_{br} ,i_{cr}	courants rotorique des phases R_a,R_b,R_c	A
Ω_r	vitesse angulaire de rotor	Tr/mn
Ω_s	vitesse angulaire de stator	Tr/mn
J	moment d'inertie	kg.m
f	coefficient de frottement visqueux	
Ce	couple électromagnétique	N.m
Cr	couple résistant	N.m
P	Nombre de paire de pole	
L_r ,L_s	Inductances cyclique rotorique et statorique	H
M	Inductance mutuelle cyclique entre stator et rotor	H
θ_s	Angle électrique entre(S_a,O_d).	Degre

θ_r	Angle électrique entre (R_a,O_d).	Degre
Φ	Le flux	Wb
V_d , V_q	Les tentions d'axe directe (d) et en quadrature (q)	V
i_d , i_q	Les courants d'axe directe (d) et en quadrature(q)	A
ω_s	vitesse electrique de stator	Rd/s
ω_r	vitesse electrique de rotor	Rd/s
τ_s	Constante de temp statorique	S
τ_r	Constante de temp rotorique	S
σ	Coefficient de dispersion	
RNA	Réseaux de Neurones artificiels	
F	Fonction s'activation	
η	Coefficient d'apprentissage	
W_{JI}	Poids synaptiques entre les neurones i et j	
η	Coefficient d'apprentissage	
FPGA	Field programmable gate array	
DSP	Digital Signal Processing	
PLD	Programmable Logic Devices	
SPLD	Simple Programmable Logic Device	
CPLD	Complex Programmable Logic Device	
ASIC	Application Specific Integrated Circuits numérique	
PLA	Programmable Logic Array	
PAL	Programmable Array Logic	
RAM	Random Access Memory	
ROM	Read Only Memory	
EPROM	Erasable Programmable ROM.	
EEPROM	Electrically Erasable Programmable ROM	
ISE	Integral of Square Error	
MLI	Modulation de Largeur d'Impulsion	

Annexe

Annexe 01

Paramètres de la machine asynchrone

Tension nominale	U =220 /380 V
Courant nominal	I_n = 21 /12 V
Vitesse nominale	Ω_n = 1420 tr/min
Puissance nominale	P_n = 5.5 KW
Résistance statorique	R_s = 2.25 Ω
Résistance rotorique	R_r = 0.70 Ω
Inductance statorique	L_s = 0.1232 H
Inductance rotorique	L_r = 0.1122 H .
Inductance mutuelle	M = 0.1118 H .
Constante de temp statorique	τ_s = 0.0546 s.
Constante de temp rotorique	τ_r = 0.1600 s.
Coefficient de dispersion	σ = 0.09.
Moment d'inertie	J = 0.038 kg.m.

Annexe 02

I.Les neurones artificiels

Le premier modèle d'un neurone artificiel a été présenté dans les années quarante par Mac Culloch et Pitts. Par analogie avec le modèle électrochimique décrit ci-dessus, ils ont proposé le modèle d'un neurone artificiel qui est établi conformément au modèle non linéaire représenté sur la Figure1.

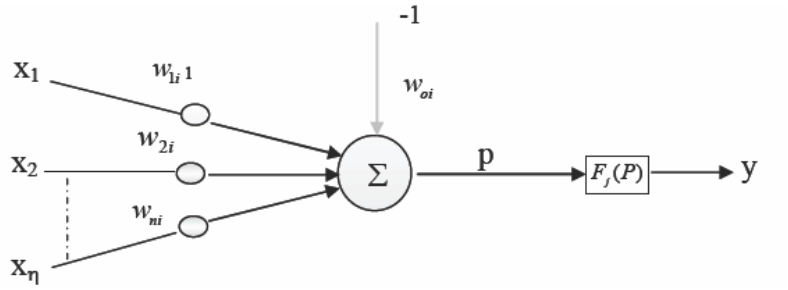


Figure1: modèle du neurone de Mc CULLOCH et PITTS

Chaque neurone artificiel transforme l'ensemble des signaux qu'il reçoit en un signal de sortie qui est communiqué à d'autres neurones. Cette transformation s'effectue en deux étapes:

Le neurone effectue une sommation pondérée des potentiels (principe de superposition), la valeur numérique obtenue représente l'état du neurone qui l'a émis, afin d'obtenir une stimulation résultante globale :

$$p_i = \sum_{j=1}^{j=N} w_{ij} x_j - w_{oi}$$

A l'aide d'une fonction de transfert, on teste le neurone. Si cette stimulation dépasse un certain seuil, le neurone est activé et transmet une réponse.

Dans ce cas :

$$s_i = f_i(P) \qquad f_i(P) = \begin{cases} 1 & \text{si} \quad P > \beta \\ 0 & \text{si} \quad P \le \beta \end{cases}$$

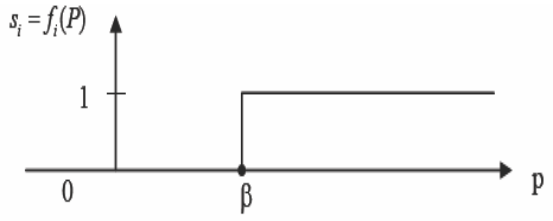


Figure2: fonction de seuillage avec un sommateur

II. Présentation de l'algorithme de rétropropagation

Toute information que possède un réseau de neurones est représentée par les poids d'interconnections, cette information est acquise durant la phase d'apprentissage.

Cette propriété d'apprendre a permis l'application de tels réseaux dans plusieurs domaines: contrôle, identification, diagnostic, etc.

Par ailleurs, cet algorithme que l'on désigne couramment par «Back propagation » est une généralisation de la règle de « WIDROW HOFF » pour un réseau multi couches.

Il a été mis au point simultanément par deux équipes indépendantes en France «FEGLAMM SAULIE, GALLINARI, LECUN » et aux Etats-Unis «RUMELHART, HITON, WILLIAMS ».

L'idée simple qui est à la base de cet algorithme et qui permet de lever la difficulté du « crédit assignent problème » est l'utilisation d'une fonction dérivable (fonction sigmoïde) en remplacement de la fonction de seuil utilisée dans le neurone linéaire à seuil.

Mathématiquement, cet algorithme utilise simplement les règles de dérivation composée et ne présente aucune difficulté particulière.

Le principe de cet algorithme est que, de même que l'on est capable de propager un signal provenant de cellules d'entrée vers la couche de sortie, on peut, en suivant le chemin inverse, rétro propager l'erreur commise en sortie vers les couches internes.

L'apprentissage supervisé consiste à ajuster les coefficients synaptiques pour que les sorties du réseau soient les plus proches possibles des sorties de l'ensemble d'entraînement. Donc il faut spécifier une règle d'apprentissage pour l'adaptation de ces paramètres.

Pour remédier à ce problème, on utilise la méthode de rétro propagation de l'erreur pour l'apprentissage des réseaux statiques multi couches (voir la figure 3) .

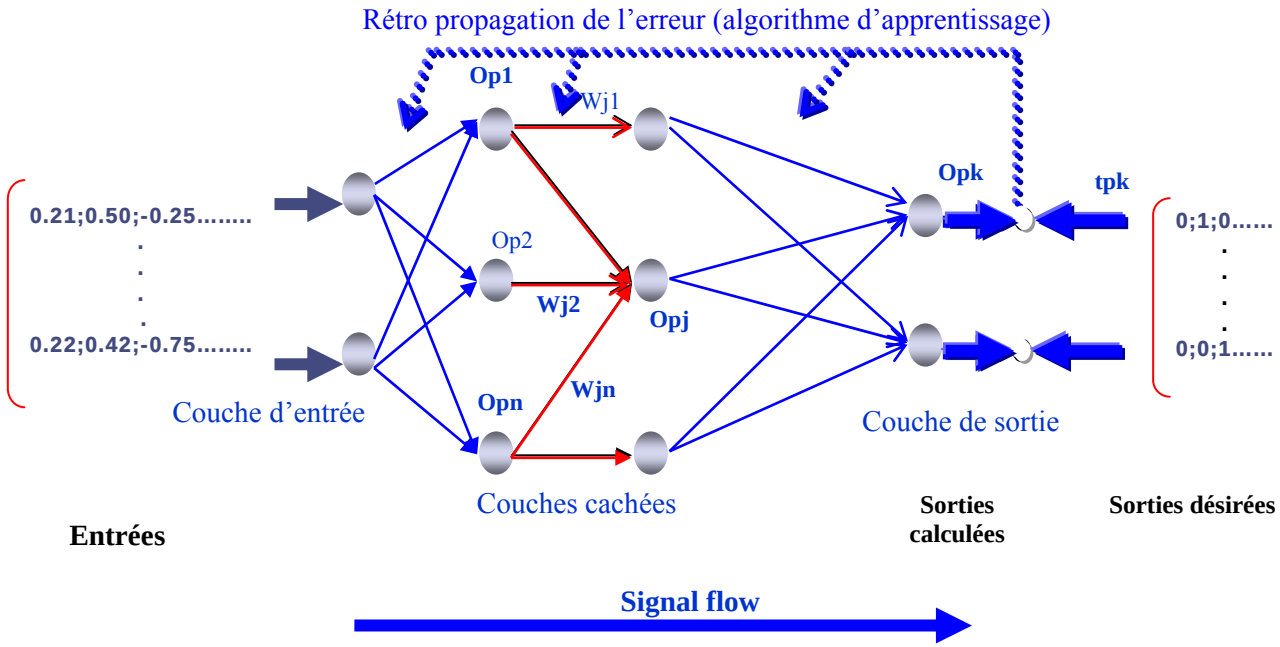


Figure 3 Mécanisme d'apprentissage des RNA multi-couches par l'algorithme de rétropropagation

II.1. Modèle et équation du réseau :

Le réseau utilisé est un réseau multi couches, comportant une couche d'entrée qui correspond à la rétine, une couche de sortie qui correspond à la décision et un certain nombre de couches dites cachées. Ces couches cachées constituent la véritable représentation du problème.

Le neurone utilisé dans le réseau est fondamentalement de même nature que le neurone linéaire à seuil du perceptron, il applique une fonction à la somme pondérée de ses entrées, cette fonction est une version lissée de la fonction à seuil, on utilise en général la fonction sigmoïde qui s'écrit: $f(S)=1/(1+e^{-s})$ (1)

Les états des différents neurones dans un réseau ayant « L» couches (couches cachées et couches de sortie) données ayant « N» entrées et « M» sorties, sont données par les équations suivantes: $O_{pi}=f(S_{pi})$ (2)

Avec: $S_{pi}=\sum Wij.O_{pi}$ (3)

Où:

O_{pi} : La sortie de neurone i pour l'exemple (entrée, sortie désirée) p;

S_{pi} : Le potentiel somatique du neurone i pour l'exemple (entrée, sortie désirée) p;

Wij : Coefficient synaptique (poids) de la j^{eme} entrée du neurone i.

II.2. **Principe de rétro propagation:**

La rétro propagation est basée sur l'adaptation des coefficients synaptiques dites encore des coefficients de pondération dans le but de minimiser une fonction de coût (performance) donnée par:

$$E(W)=\sum_{P=1}^{p} E_p(W) \qquad (4)$$

$$E_p(W)=\tfrac{1}{2}\sum_{i=1}^{n}(t_{pi}-O_{pi})^2 \qquad (5)$$

Où: t_{pi} et O_{pi} Représentant respectivement la sortie désirée (Target) et la sortie (output) du réseau;

P: Le nombre d'exemple ou de la longueur de l'ensemble d'entraînement.

La minimisation de l'erreur se fait par l'approximation d'une descente de gradient comme dans la règle de Widrow Hoff (règle delta). Toute la difficulté pour effectuer cette descente dans un réseau multi couches était de pouvoir calculer la dérivée de l'erreur quadratique par rapport à un poids donné.

II.3 Adaptation des poids

L'adaptation (ajustement, mise en forme) des coefficients synaptiques, se fait par la méthode du gradient basée sur la formule itérative suivante:

$$Wij(n+1)=Wij(n)+\Delta Wij^{(époque)} \qquad (6)$$

$$\Delta_p Wij=-\eta \,.\, \partial E_p / \partial Wij \qquad (7)$$

Où: n: Représente le numéro d'itération;

η: Représente le pas d'apprentissage. Ce facteur influe sur la vitesse de convergence du réseau.

La dérivée partielle de E (W) par rapport à chaque poids du réseau, pour cela et en utilisant la règle delta généralisée. Nous devons charger notre poids lié à:

$$\Delta_p Wij=\eta \,.\, \delta_{pi}.O_{pj} \qquad (8)$$

Avec:

O_{pj}: Sortie du neurone j pour un exemple p;

δ_{pi}: L'erreur commise à la sortie du neurone i pour l'exemple p, posant

maintenant:

$$\delta_{pi} = -\partial E_p / \partial S_{pi} = (t_{pi} - O_{pi}) \qquad (9)$$

L'astuce est de calculer δ_{pi} pour chaque neurone dans le réseau. Le résultat intéressant que nous avons obtenu maintenant, est qu'il y a un simple calcul récursif des δ qui peuvent être implémentés par la rétropropagation du signal d'erreur à travers le réseau.

Pour calculer:

$$\delta_{pi} = -\partial E_p / \partial S_{pi}$$

En applicant la décomposition en chaîne pour exprimer la dérivation partielle, nous obtenons:

$$\delta_{pi} = -\partial E_p / \partial S_{pi} = -\partial E_p / \partial O_{pi} \cdot \partial O_{pi} / \partial S_{pi}$$

Calculons le deuxième facteur de l'équation (9) nous constatons que:

$$\partial O_{pi} / \partial S_{pi} = f'(S_{pi}).$$

Où: $f'(S_{pi})$: Est la dérivée de la fonction d'activation du neurone 'i' évaluée à S_{pi}.

Pour calculer le premier facteur nous considérons deux cas:

Supposant que le neurone Ui est un neurone de sortie du réseau, dans se cas, il parait de la définition de E_p que:

$$\partial E_p / \partial O_{pi} = -(t_{pi} - O_{pi}).$$

Substitutions les deux facteurs dans l'équation (9) nous obtenons:

$$\delta_{pi} = (t_{pi} - O_{pi}) F'i(S_{pi}) \qquad (10)$$

Pour tout neurone de sortie Ui.

Si Ui n'est pas un neurone de sortie, nous utilisons la décomposition en chaîne pour écrire:

$$\sum \partial E_p / \partial S_{pk} * \partial S_{pk} / \partial O_{pi} = \sum \partial E_p / \partial S_{pk} * \partial / \partial O_{pi} \sum W_{ki} O_{pi} =$$

$$= \sum \partial E_p / \partial S_{pk} W_{ki} = \sum \delta_{pk} W_{ki.}$$

Dans ce cas, en substituant les deux facteurs dans l'équation (9) nous obtenons:

$$\delta_{pi} = f'(S_{pi}) \sum \delta_{pk} * W_{ki} \qquad (11)$$

Où: Ui est un neurone appartenant aux couches cachées.

Les équations (10) et (11) donnent une procédure récursive pour calculer les δ pour tout neurone dans le réseau lié à l'équation (7), cette procédure constitue «la règle delta généralisée» pour un réseau statique et aussi pour le neurone non linéaire.

Pour minimiser l'erreur totale sur l'ensemble des entraînements, les poids du réseau peuvent être ajustés par la présentation de l'ensemble d'apprentissage en entier.

La variation des poids $\Delta_p W_{ij}$ (n) peut alors s'écrire ainsi:

$$\Delta_p W_{ij}(n) = \eta \, \partial E_p(W) / \partial W_{ij}(n) \qquad (12)$$

II.4. Les étapes de l'algorithme de rétro propagation:

L'algorithme de rétro propagation est représenté comme suit :

Etape 01 : Initialiser les poids W_{ij} et les seuils internes des neurones à de petites valeurs aléatoires.

Etape 02 : Présenter l'ensemble des couches (entrée, sortie désirée).

Etape 03 : Présenter le première couché (entrée, sortie désirée) de l'ensemble.

Etape 04 : Calculer:

1- La somme des entrées des neurones de la couche cachée par l'équation (3);

2- Les sorties des neurones de la couche cachée par l'équation (2);

3- La somme des entrées de la couche de sortie par l'équation (4);

4- Les sorties de réseau par l'équation (2).

Etape 05 : Calculer:

1- Les termes de l'erreur pour les neurones de la couche de sortie par l'équation (10) ;

2- Les termes de l'erreur pou les neurones de la couche cachée par l'équation (11).

Etape 06 : Calculer la variation des poids par l'équation (8).

Etape 07 : Calculer l'erreur E_p par l'équation (5).

Etape 08 : Présenter un autre couple (entrée, sortie désirée) et aller à l'étape quatre.

Etape 09 : Si tout l'ensemble des couple est présenté, calculer la variation total des poids par époque, en utilisant l'expression :

$$\Delta \, Wij^{(\text{époque})} = \sum_{P=1}^{p} \Delta_p \, Wij.$$

Etape 10 : Ajuster les poids par l'équation (6).

Etape 11 : Calculer l'erreur totale par l'équation (4).

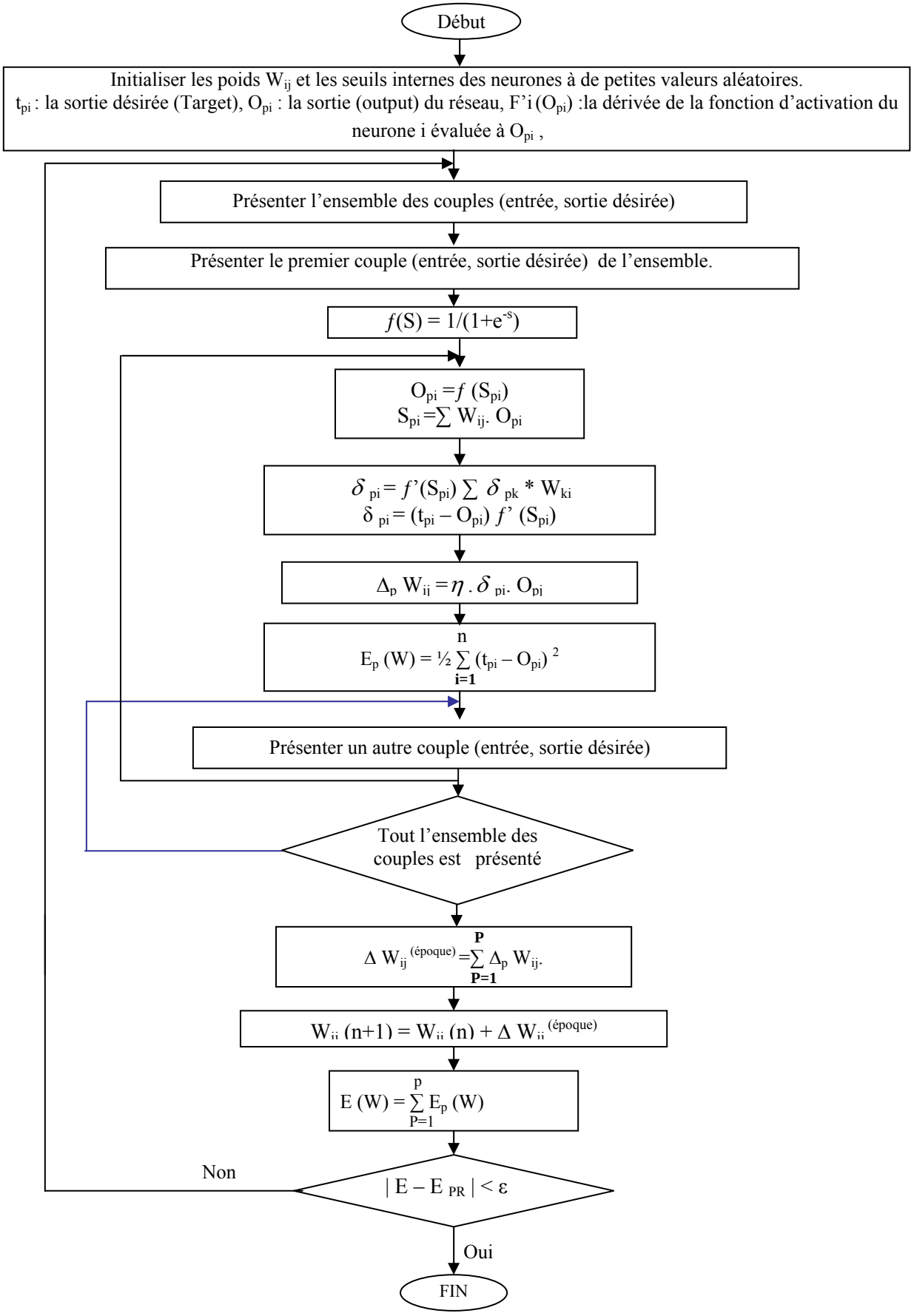
Etape 12 : Comparer l'erreur totale avec une erreur prédéterminée (E_{PR}). $[E - E_{PR} < \varepsilon]$.

Si la condition est vérifiée aller à l'étape 13.

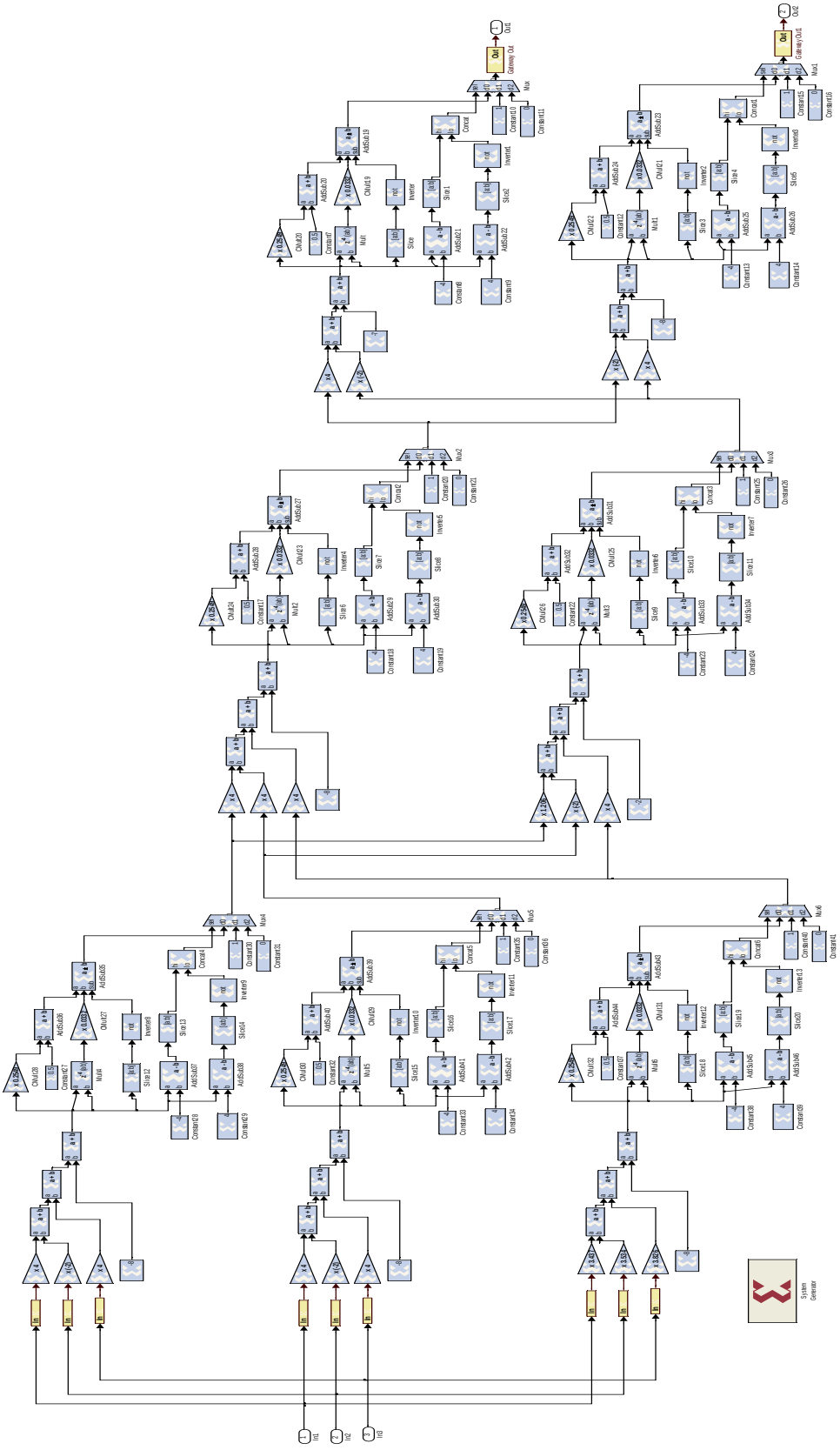
Si non aller à l'étape 03.

Etape 13 : Fin

II.5 Algorithme de rétro propagation



Annexe 03



Réseau de neurone artificiels fait par xilinx

Annexe 04

Photo de la réalisation

Spartan 2^E
starter Kit

Bibliographie

Bibliographie

[01] Roland CASIMIR, « diagnostique des défauts des machines asynchrone par Reconnaissance des forme », thèse de doctorat Lyon 2003.

[02] G. Zwinngelesten, « diagnostic des défaillances, théorie et pratique pour les systèmes industriels », Ed. Hermes paris. 1995.

[03] Mohammed Riyad Zemmouri, « Contribution à la surveillance des systèmes de production à l'aide des réseaux de neurones dynamiques», Thèse de Doctorat, Université de franche compté, Déc 1999.

[04] D.Hadiouche, H.Razik, A.Rezzoug, « Modelling of a double-star induction motor With an arbitrary shift angle between its three phase windings », EPEPEMC 2000, Kosice.

[05] B. Dubuisson, « détection et diagnostic des pannes sur processus », Technique de l'ingénieur.R7597, 1992.

[06] B. Fouad et M .Ilyase, « diagnostic en temps réel des défaillances d'un ensemble Moteur asynchrone –convertisseur électrique par application des techniques d'intelligence artificielles (réseaux de neurones) », PFE. Université de M'sila, 2004.

[07] H. Khaled, S.Ali, « Commande Vectorielle De La Machine Asynchrone », Mémoire d'Ingénieur, Encadré par M.Gaad, Université Med Boudiaf de M'Sila.

[08] Jean-pierre Caron, « modélisation et commande de la machine asynchrone », 1995.

[09] Abdessemed R. Kadjoudj M, « Modélisation des machines électriques », Université de Batna, 1997.

[10] Baghli, « Contribution à la commande de la machine asynchrone, utilisation de la Logique floue, des réseaux de neurons et des algorithms génétiques », thèse de doctorat, University Henri Poincaré, Nancy, Jan 1999.

[11] Y. Kebbati, « Développement d'une Méthodologie de Conception Matérielle à Base de Modules Génériques VHDL/VHDL-AMS en Vue d'une Intégration de Systèmes de Commande Electrique », Thèse de Doctorat, Université Louis Pasteur, Strasbourg1, 2002.

[12] C. Cecati, « Microprocessors for power electronics and electrical drives applications », IEEE Industrial Electronics Society Newsletter, vol. 46, no. 3, Sept. 1999, pp. 5-9.

[13] H.L. Huy, « Microprocessors and digital IC's for motion control », Proc.IEEE, Vol. 82, No.8, 1994, pp. 1140-1163.

[14]S.S. Deol, "VHDL simulation for reduction of state space representation of linear sequential machine", Wayne state university, Detroit, USA, 2004.

[15] F. Aubepart, C. Girerd, Y.A. Chapuis, P. Poure, F. Braun, «ASIC implementation of Direct Torque Control for induction machine: functional validation by power and control simulation » in Proc. PCIM'98 Conf, 1998, pp. 251-260.

[16] H. Cherraj, «Conception et Réalisation d'un Dispositif de Prétraitement de Signaux Radar en Utilisant le VHDL», Thèse de Maîtrise, Université du Québec, Canada, 2001.

[17] Y. Kebbati, « Développement d'une Méthodologie de Conception Matérielle à Base de Modules Génériques VHDL/VHDL-AMS en Vue d'une Intégration de Systèmes de Commande Electrique », Thèse de Doctorat, Université Louis Pasteur, Strasbourg1, 2002.

[18] E. Monmasson, Y. A. Chapuis, « Perspectives technologiques en commande de systèmes électriques, chapitre 5 : Apport des FPGA dans la commande des systèmes électriques », Traité EGEM, Collection HERMES Science, 2002.

[19] Xilinx Data Book, « Virtex FPGA Family: Complete Data Sheet », Xilinx, 2006.

[20] Xilinx Data Book, « Spartan-3E FPGA Family: Complete Data Sheet », Xilinx, 2006.

[21] C. L. Toh, N. R. N. Idris and A. H. M. Yatim FazIli Patkar, "Design and Implementation of a Direct Torque Control of Induction Machine utilizing a Digital Signal Processor and the Field Programmable Gate Arrays", IEEE PEDS, April 2005, pp 72-77.

[22] XIAOGUANG LI « A Hardware-Software co-design Approch for face Recogntion By Artificial Neural Networks » August, 2004.

MEMOIRE DE FIN D'ETUDES EN VUE DE L'OBTENTION DU DIPLÔME D'INGENIEUR D'ETAT EN GENIE ELECTROTECHNIQUE

OPTION: COMMANDE ELECTRIQUE

Proposé et dirigé par : Dr. KHODJA Djalal Eddine

Présenté par: BENGUETTAF Brahim

ZAOUI Redha

Thème :

IMPLEMENTATION DES RESEAUX DE NEURONES ARTIFICIELS (RNA) SUR "FPGA" POUR LE DIAGNOSTIC DES DEFAILLANCES DE LA MACHINE ASYNCHRONE

Résumé :

Dans ce travail nous proposons l'application des techniques de réseaux de neurones pour le diagnostic des défaillance de l'association moteur convertisseur et leur commande. D'autre part, l'utilisation des circuits modernes à architectures configurables permet de surmonter les contraintes liées à l'implémentation des éléments de diagnostic tels que les réseaux de neurones artificiels. Ces circuits conviennent parfaitement à une implémentation optimale du système de diagnostic ainsi que la commande des MAS, du fait qu'ils ont un coût réduit et qu'ils sont caractérisés par une grande densité d'intégration et une grande flexibilité avec une structure totalement reconfigurable. Par ailleurs, nous avons procédé à l'implémentation des RNA sur un circuit de type FPGA. L'algorithme d'implémentation proposé est basé sur la structure simple des RNA. L'implémentation de cet algorithme a été faite à l'aide d'un outil de haut niveau ; a savoir: le System Generator de Xilinx.

Mots Clés:

Machine asynchrone, Diagnostic, RNA, Défaillances, défauts commande vectorielle, Circuit reconfigurable, FPGA.