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: ELECTROMECANIQUE

THEME

IMPLEMENTATION DES "RNA" SUR "FPGA" POUR LE DIAGNOSTIC DES DEFAILLANCES DE LA MACHINE ASYNCHRONE EN UTILISANT LA CO-SIMULATION

Proposé et dirigé par :
Dr. KHODJA Djalal Eddine

Présenté par :
MEZAACHE Farouq
MENASRI Abderrachid

Année Universitaire: 2008/2009

Remerciements

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

Nous tenons de remercier Monsieur Khodja Djalal Eddine, pour avoir encadré nos travaux de fin d'étude.

A travers ce mémoire, nous adressons nos reconnaissances à tous nos enseignants qui ont contribué à notre formation depuis la première classe du primaire jusqu'à aujourd'hui.

Nous remercions tous les membres du jury qui ont accepte de juger notre travail et pour l'intérêt qu'il porte à ce dernier

Nous tenons à remercier vivement toutes les personnes qui nous ont aidés à élaborer et réaliser ce mémoire.

Et enfin Nous tenons à remercier également tous nos collègues de la promotion 2008-2009pour leur l'aide inestimable.

Farouq et Abderrachid

Dédicace

Je dédie ce modeste travail :

A mes très chers parents et ma grande famille.

A mes frères et mes sœurs.

A tous mes amis.

A tous mes collèges de la promotion ELM 2009 (Ammar, Farhet, Touati, B.Khaled, Abdelhak, Sekkai, Ali, Elakhdar, Amal, Abdelmalek, Dj. Mohamed, H. Nourredine, L.Mohamed, L. Zine Laabidine, Rachid, Mahammed Taher, Walid, M. Khaled, Abderrachid, Souad, Saïd, Slimane, Ahmed, Ryadh, Naserredine, S. Zine Laabidine, Dahmane, Abderrazak, Youssouf, Antara, Moustapha, Kamel, Rida, Chaabane, Z. Nourredine).

A tous mes enseignats.

A tous ceux qui m'aiment et que j'aime.

Kouhyn

Farouq

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 protégé pour leurs soutien moral et financier, pour leurs encouragements et les sacrifices qu'ils ont endurés.

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

A mes très chers parents et ma grande famille.

A mes frères et mes sœurs.

A tous mes amis.

A tous ceux qui m'aiment et que j'aime.

A vous.

Abderrachid

Nomenclature

Indices

(s), (r), (e)	indice respectifs du stator, du rotor et d'entrefer
s	opérateur de Laplace ($=d/dt$).
d	axe d du repère tournant (d, q)
q	axe q du repère tournant (d, q)
α	axe du repère statorique (α, β)
β	axe du repère statorique (α, β)
n	grandeur nominale

Principaux paramètres moteur asynchrone

R_s, l_s	résistance et inductance propre d'une phase statorique
R_r,l_r	résistance et inductance propre d'une phase rotorique
M_s	coefficient de mutuelle inductance entre deux phases de stator
M_r	coefficient de mutuelle inductance entre deux phases de rotor
M_{sr}	maximum de l'inductance mutuelle entre d'une phase de stator et une phase du rotor.
L_s	inductance cyclique statorique
L_r	inductance cyclique rotorique
T_r	constante de temps rotorique
M	inductance mutuelle cyclique entre stator et rotor
p	nombre de paires de pôles
J	moment d'inertie ramené sur l'axe moteur
f_r	coefficient de frottement visqueux

Principales grandeurs

$x_{s\alpha}$, $x_{s\beta}$	grandeurs statoriques liées au repère fixe au stator
x_{rd} , x_{rq}	grandeurs rotoriques liées au repère tournant (d, q)
x_{sd} , x_{sq}	grandeurs statoriques liées au repère tournant (d, q)
v_{sa}, v_{sb}, v_{sc}	tensions statoriques simples
U_c	tension d'entrée de l'onduleur
α	position du rotor
θ_s	angle électrique entre l'axe d du référentiel tournant et le référentiel fixe lié au stator

θ_r	angle électrique entre l'axe d du référentiel tournant et le référentiel fixe lié au rotor
Ω	vitesse angulaire mécanique du rotor
ω	vitesse angulaire électrique du rotor,($\omega = p.\Omega$)
ω_s	vitesse des axes « *d, q* » dans le repère statorique,
ω_r	vitesse des axes « *d, q* » dans le repère rotorique.
C_{em}	couple électromagnétique délivré par le moteur
C_r	couple résistant, ou de charge
x_{ref} (ou x^*)	consigne de *x*
$\dot{x}$	dérivé de la grandeur *x* (=*dx/dt*).
[*P*]	matrice de transformation de Park
[*C*]	matrice de transformation de Concordia
σ	facteur de dispersion
m	indice de modulation.
r	coefficient de réglage en tension

Notations liées au réseau de neurones

y	signal d'un neurone de sortie
η	pas d'apprentissage.
θ_i	seuil du neurone *i*
φ	fonction d'activation
x_i	entrée du neurone *i*
w_{ij}	poids de la connexion du neurone *j* vers le neurone *i*
m	nombre d'exemple
E_k	erreur sur l'exemple *k*
θ_i	signal d'erreur du neurone *i*
d_o^k	signal désiré pour l'exemple *k* sur le neurone de sortie *o*
O	nombre de neurones de la couche cachée
H	nombre de neurones sur la couche de sortie
b	biais

Liste des abréviations

ASIC	Application Specific Integrated Circuit
BLE	Basic Logic Element
CLB	Configurable Logic Block
CPLD	Complex Programmable Logic Device
DSP	Digital Signal Processing
EPROM	Erasable Programmable Read Only Memory
EEPROM	Electrically Erasable Programmable Read Only Memory
FPGA	Field Programmable Gate Array
I/O (ou E/S)	Input Output (ou Entrée/Sortie)
IA	Intelligence Artificiel
IEEE	Institute Electrical and Electronics Engineers
LUK	Look-Up Table
LE	Logic Element
MAS	Machine ASynchrone.
MLP	MultiLayer Perceptron
MLI	Modulation de Largeur d'Impulsions
PAL	Programmable Array Logic
PLAs	Programmable Logic Array
PROM	Programmable Read Only Memory
PLD	Programmable Logic Device
ROM	Read Only Memory
RNA	Réseau de Neurones Artificiels
SIF	Systèmes d'Inférence Floue
SPLD	Simple Programmable Logic Device
SRAM	Static Random Access Memory
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VHDL	VHSIC Hardware Description Language
VHSIC	Very-High-Speed Integrated Circuit

Table des matières

Table des figures

Chapitre I

Chapitre II

Chapitre III

Chapitre IV

Liste des tableaux

Introduction générale

Introduction générale

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 par les réseaux de neurones artificiels implémentés sur des circuits électroniques tels que les FPGA en utilisant la Co-simulation

Ce mémoire est organisé comme suite :

Le premier chapitre sera consacré à l'étude de diagnostic des défaillances des machines asynchrones. Nous allons dans un premier temps de décrire les différents constituants de la machine asynchrone. Ensuite, les défaillances les plus courantes pouvant apparaître aussi bien dans la machine asynchrone seront développés. La terminologie

propre au diagnostic utilisée dans la littérature scientifique sera rappelée. Les principales approches de détection seront énoncées. Nous évoquerons par la suite le problème à résoudre en terme de choix de la méthode du diagnostic adéquate et son implémentation sur des circuits reconfigurables.

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 à l'état sain et en présence des défauts puis leurs résultats de simulation à l'aide de logiciel Matlab/Simulink.

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 un circuit FPGA en utilisant la Co-simulation pour la détection des défauts de la machine asynchrone.

Chapitre I

Etat de l'art de diagnostic de la machine asynchrone

I.1 Introduction

La surveillance et le diagnostic des machines électriques ont été sous le foyer pendant au moins de vingt années avec un intérêt spécial pour les machines d'induction triphasées à cage d'écureuil. Ces derniers présentent de nombreux avantages dus à leur robustesse et à leur rapport de puissance-poids. Ainsi, ils sont employés couramment dans l'industrie. Par conséquent, il y a une demande considérable pour réduire des coûts de maintenance et empêcher des temps de panne imprévus pour les systèmes électriques d'entraînement **[2]**.

Durant maintenant plus d'une vingtaine d'années, des études et des recherches ont été menées sur la façon dont on pourrait détecter une panne, une défaillance et d'y comprendre la relation cause à effet. Ainsi, on pourrait améliorer la fiabilité du moteur asynchrone, donc augmenter sa durée de vie **[7, 24]**.

Dans ce chapitre introductif, Nous allons dans un premier temps de décrire les différents constituants de la machine asynchrone. Ensuite, les défaillances les plus courantes pouvant apparaître aussi bien dans la machine asynchrone seront développés.

La terminologie propre au diagnostic utilisée dans la littérature scientifique sera rappelée. Ainsi que les principales approches de détection et du diagnostic seront énoncées.

Enfin, on posera le problème à résoudre pour le choix de la méthode adéquate de détection des défauts de la machine asynchrone puis son implémentation sur des circuits reconfigurables afin d'aboutir a un diagnostic en temps réel.

.

I.2 Eléments de Constitution de la machine asynchrone à cage

La machine asynchrone est constituée des principaux éléments suivants:

- le stator (partie fixe) constitué de disques en tôles magnétiques portant les enroulements chargés de magnétiser l'entrefer.
- le rotor (partie tournante) constitué de disques en tôles magnétiques empilés sur l'arbre de la machine portant un enroulement injecté.
- les organes mécaniques permettant la rotation du rotor et le maintien des différents sous-ensembles [**9**].

I.2.1 Le stator

Il est constitué d'un enroulement bobiné réparti dans les encoches du circuit magnétique. Ce circuit magnétique est constitué d'un empilage de tôles dans lesquelles sont découpées des encoches parallèles à l'axe de la machine. Le bobinage statorique peut se décomposer en deux parties : les conducteurs d'encoches et les têtes de bobines. Les conducteurs d'encoches permettent de créer dans l'entrefer le champ magnétique à l'origine de la conversion électromagnétique. Les têtes de bobines permettent, quant à elles, la fermeture des courants en organisant la circulation judicieuse des courants d'un conducteur d'encoche à l'autre. L'objectif est d'obtenir à la surface de l'entrefer une distribution de courant la plus sinusoïdale possible, afin de limiter les ondulations du couple électromagnétique [**9**].

I.2.2 Le rotor

Dans le rotor à cage, les anneaux de court-circuit permettent la circulation des courants d'un conducteur d'encoche (barre rotorique) à l'autre. Ces barres conductrices sont régulièrement réparties, et constituent le circuit du rotor. Cette cage est insérée à l'intérieur d'un circuit magnétique constitué de disques en tôles empilés sur l'arbre de la machine analogue à celui du moteur à rotor bobiné. Dans le cas de rotors à cage d'écureuil, les conducteurs sont réalisés par coulage d'un alliage d'aluminium, ou par des barres massives de cuivre préformées et frettés dans les tôles du rotor. Il n'y a généralement pas, ou très peu, d'isolation entre les barres rotoriques et les tôles magnétiques, mais leur résistance est suffisamment faible pour que les courants de fuite dans les tôles soient négligeables, sauf lorsqu'il y a une rupture de barre [**9**].

I.2.3 Les organes mécaniques

La carcasse sert de support, elle joue le rôle d'enveloppe et assure la protection contre l'environnement extérieur. L'arbre est un organe de transmission. Il comprend une partie centrale qui sert de support au corps du rotor et un bout d'arbre sur lequel est fixé un demi-accouplement. Il est généralement constitué en acier moulé ou forgé. Il est supporté par un ou plusieurs paliers. Ces paliers soutiennent le rotor et assurent la libre rotation. Le second palier est libre pour assurer les dilatations thermiques de l'arbre. Une isolation électrique de l'un des paliers assure l'élimination des courants dans l'arbre dû aux dissymétries des réluctances du circuit magnétique [**9**].

I.3 Les défaillances de la machine asynchrone

De nombreuses défaillances peuvent apparaître sur les machines asynchrones. Elles peuvent être électriques, mécaniques ou magnétiques. Leurs causes, très variées, sont classées en trois groupes:

- les initiateurs de défauts : surchauffe du moteur, usures des éléments mécaniques (roulements à billes), rupture de fixations, problème d'isolation électrique, surtension transitoire.
- les contributeurs aux défauts : surcharge fréquente, température ambiante élevée, ventilation, défaillante, humidité, fortes vibrations, vieillissement.
- les défauts sous jacents et erreurs humaines : défauts de fabrication, composants défectueux, protections inadaptées, absence de maintenance.

Ces différentes causes peuvent provoquer une dégradation des performances ou des arrêts intempestifs du système, le diagnostic précoce des pannes est donc nécessaire **[11]**.

I.3.1 Type des défaillances

Les statistiques de défauts des machines à induction ont trouvé les mécanismes de défaillances les plus communs dans des machines d'induction (la figure I.1). Celles-ci ont été classées par catégorie selon les composants principaux d'une machine : les défaillances des circuits électriques statoriques et celles rotoriques, les défaillances des roulements et d'autres défauts **[4]**.

I.3.1.1 Défaillances des roulements

Les roulements à billes jouent un rôle très important dans le fonctionnement de tout type de machine électrique. Les problèmes de rotation au sein de la culasse du roulement, causés par un roulement abîmé, écaillé ou fissuré, peuvent être crée des perturbations au sein de la machine. Nous savons que des courants électriques circulent au niveau des roulements d'une machine asynchrone ce qui, pour des vitesses importantes, peut provoquer la détérioration de ces derniers. La graisse, qui permet la lubrification et la bonne rotation des roulements peut, dans certains applications, se rigidifier et causer une résistance à la rotation. L'analyse vibratoire de la machine ou l'analyse harmonique des courants statoriques permet de détecter ce type défaillances **[12]**.

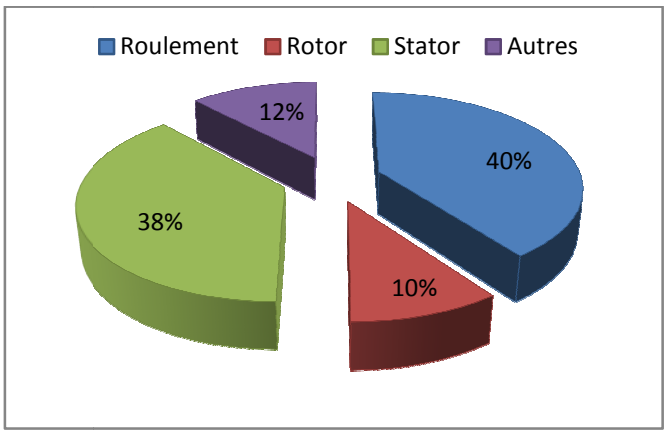


Figure (I.1) : *Types de défaillances de machine d'induction*

I.3.1.2 Défaillances des circuits électriques statoriques

Presque 40% de tous les rapports de défaillances de machine à induction se rangent dans cette catégorie. Des défauts d'enroulement de stator sont souvent provoqués par défaillance d'isolation entre deux spires adjacentes dans un enroulement, ceci s'appelle le court-circuit **[4]**.

Le court-circuit de spires est donc le défaut le plus nuisible et le plus fréquemment rencontré au stator, même si les risques d'ouverture de phase (conducteur coupé) demeurent physiquement réalisables **[24]**. Les courants induits résultants produisent la surchauffe et causent d'un déséquilibre dans le champ magnétique dans la machine. Si non détecté, la surchauffe locale endommagera l'isolation de stator jusqu'à ce qu'un défaut catastrophique se produise **[4]**. Donc le facteur principal de vieillissement est l'échauffement anormal des bobinages **[24]**. Le champ magnétique non équilibré peut

également avoir comme conséquence la vibration excessive qui peut causer des défauts prématurés de roulement **[4]**.

Pour le stator, les effets sont principalement dus à un problème **[6]** :

- thermique (surcharge,…),
- électrique (diélectrique,…),
- mécanique (bobinage,…),
- environnemental (agression,…).

I.3.1.3 Défaillances des circuits électriques rotoriques

Les défauts du rotor expliquent environ 10% de défaillance totale de machine à induction **[4]**.

Le problème des ruptures de barres dans les moteurs des stations de pompage offshore a été à l'origine des premiers travaux de recherche sur le diagnostic des machines tournantes. Les défauts rotoriques typiques des machines asynchrones sont dus à un défaut de fabrication, ou à un défaut d'utilisation. On peut en citer les principaux [**24]**:

- une rupture partielle ou totale d'une barre au rotor survenant généralement à cause de l'échauffement dû aux sollicitations,
- une excentricité prononcée du rotor,
- un mauvais alignement rotor-charge est aussi envisageable,
- une dégradation prématurée ou non des paliers (usure des roulements à billes) engendre souvent une modification des performances de la machine.

Pour le rotor, les effets sont essentiellement dus à un problème **[6]** :

- thermique (surcharge,…),
- électromagnétique (force en $B^2(t)$,…),
- résiduel (déformation,…),
- dynamique (arbre de transmission,…),
- mécanique (roulement,…),
- environnemental (agression,…).

I.3.1.4 D'autres défauts

L'excentricité se produit quand le rotor n'est pas centré dans le stator, produisant un entrefer non-uniforme entre eux. Ceci peut être provoqué par les roulements ou les défauts défectueux de fabrication. La variation de l'entrefer touche à la distribution de champ magnétique dans le moteur qui produit une force magnétique nette sur le rotor dans la direction du plus petit entrefer. Ce soi-disant « a déséquilibré la traction magnétique » peut causer la vibration mécanique **[4]**.

I.4 Terminologie propre au diagnostic

Il semble intéressant, de rappeler les principaux termes utilisés en diagnostic des systèmes. La terminologie suivante sera adoptée :

a) Système physique

Un système physique est un ensemble d'éléments (composants, constituants) interconnectés ou en interaction organisés pour réaliser une fonction **[1]**.

b) Composant

Un composant est une partie du système choisie selon des critères liés à la modélisation. En tout premier lieu, le comportement de référence de ce composant est bien adapté dans le sens où il peut être défaillant ou servir de support à la propagation des pannes dans le système. Un composant doit être simple à modéliser dans le sens où cela doit être naturel : il peut s'agir d'un composant (physique ou logique) complet du système ou d'une partie parfaitement délimitée de ce composant, d'un groupe de composants. Le comportement du composant élémentaire n'est pas décomposable ou alors cette décomposition n'est pas souhaitée, il constitue une « brique » du comportement du système **[1]**.

c) Modèle

Un modèle d'un système physique est une description de sa structure et une représentation comportementale ou fonctionnelle de chacun de ses composants. Une représentation comportementale est constituée de relations entre diverses variables du système, appelées classiquement relations de causes à effets. Une représentation fonctionnelle est plus abstraite puisqu'elle ne s'adresse qu'aux objectifs présumés que le

système physique doit remplir. Le niveau structurel, quant à lui, s'appuie sur la structure réelle du système physique et décrit les interconnections entre ses différents éléments ou constituants. Les niveaux comportemental et fonctionnel comprennent des relations entre des grandeurs physiques (variables) et permettent de mettre en évidence la présence d'un événement anormal ou anomalie. Le niveau structurel, quant à lui, permet de déterminer l'élément affecté par le défaut. L'intérêt de cette décomposition est de rappeler que, puisqu'un modèle contient toute l'information relative à un système physique, il est utilisable ensuite par la procédure de diagnostic **[1]**.

d) Défaut

On considère comme un défaut tout écart entre la caractéristique observée sur le dispositif et la caractéristique de référence, lorsque celui-ci est en dehors des spécifications **[10]**.

- IEEE : N'importe quel état indésirable d'un composant ou d'un système. Un défaut n'implique pas nécessairement une défaillance **[1].**

e) Défaillance

Une défaillance définit une anomalie fonctionnelle au sein d'un système physique, c'est-à-dire caractérise son incapacité à accomplir certaines fonctions qui lui sont assignées **[10]**.

Les défauts incluent les défaillances mais la réciproque n'est pas vraie. Un système peut remplir sa fonction tout en présentant une anomalie de comportement. Par exemple, une machine électrotechnique peut produire un bruit anormal tout en entraînant correctement une charge, en supposant que telle soit sa fonction. Le bruit anormal est un défaut qui peut permettre de présager d'une défaillance à venir. La recherche de défauts est donc fondamentale en diagnostic **[1]**.

f) Cause de défaillance

La norme définit la cause de défaillance par « les circonstances liées à la conception, la fabrication ou l'emploi et qui ont entraîné la défaillance » **[5, 10]**.

g) Dégradation

Une dégradation est l'état d'un ensemble qui présente **[5, 10]** :

- une perte de performances d'une des fonctions assurées par l'ensemble.
- un sous-ensemble lui-même dégradé, voire défaillant (sans conséquence fonctionnelle sur l'ensemble).

h) Sûreté de Fonctionnement

La sûreté de fonctionnement est également appelée « Science des défaillances ». Cette discipline intervient non seulement au niveau du produit fini (système existant) mais aussi au niveau conceptuel pour la réalisation d'un système ou la connexion de plusieurs sous-systèmes (surtout s'ils sont de natures différentes). La sûreté de fonctionnement consiste à connaître, évaluer, prévoir, mesurer, et maîtriser les défaillances des systèmes **[10]**.

i) Panne

La panne est l'inaptitude d'un dispositif à accomplir une fonction requise. Il est clair que dès l'apparition d'une défaillance, caractérisée par la cessation du dispositif à accomplir sa fonction, on déclarera le dispositif en panne. Par conséquent, une panne résulte toujours d'une défaillance **[10]**.

j) Symptôme

Caractère distinctif d'un état fonctionnel anormal **[1]**.

k) Résidu

Un résidu est un signal potentiellement indicateur de défauts. Il reflète la cohérence des données vis-à-vis du modèle comportemental du système **[28]**.

l) Diagnostic

Un diagnostic est un état expliqué d'un système physique compatible avec les informations disponibles sur le comportement réel du système et avec le modèle de comportement de référence disponible. Habituellement, le diagnostic est exprimé par les états des composants ou les états des relations de description du comportement **[1]**.

m) Perturbation

Entrée du système physique qui n'est pas une commande. Autrement dit, c'est une entrée non contrôlée.

La figure (I.2) représente les anomalies suivant leur criticité. Il existe également une criticité croissante entre défaillance et panne. De la non conformité (ou anomalie) dans le cas d'une défaillance, on passe à une inaptitude à accomplir une fonction dans le cas d'une panne **[1]**.

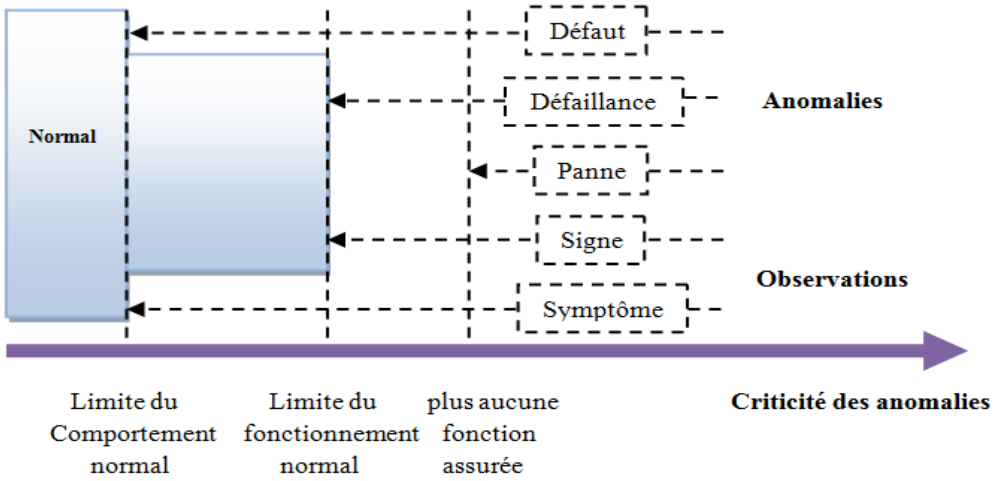


Figure (I.2) : *Anomalies et Observations classées par criticité croissante.*

Ces notions sont illustrées à partir de l'exemple d'un moteur devant assurer une fonction de ventilation (tableau 1.1).

Tableau (1.1) : *Illustration des définitions à l'aide d'un moteur de ventilateur.*

Définition illustrée	**Événement (exemple)**	**Ecart au Comportement nominal (courant, vitesse)**	**Aptitude à Remplir la Fonction de ventilation**
Perturbation	Variation de température extérieure (normal)	Petit	Totale
défaut	Fort échauffement (anormal)	Moyen	Totale
Défaillance	Déclenchement intermittent d'un relais thermique stoppant le ventilateur jusqu'à ce qu'une réparation soit effectué	Grand	Partielle
Panne	Suite aux forts échauffements répétitifs, les isolants sont progressivement endommagés: un court-circuit apparaît; le moteur ne peut plus tourner jusqu'à ce qu'une réparation soit effectuée.	Grand	Nulle

I.5 Les approches usuelles de détection

Dans ce paragraphe, on va présenter les principales méthodes utilisées en diagnostic de systèmes physiques, à savoir **[1]**:

- Méthodes basées sur les modèles analytiques,
- Méthodes sans modèle analytique.

Les principales méthodes de diagnostics sont organisées sur la figure (I.3) :

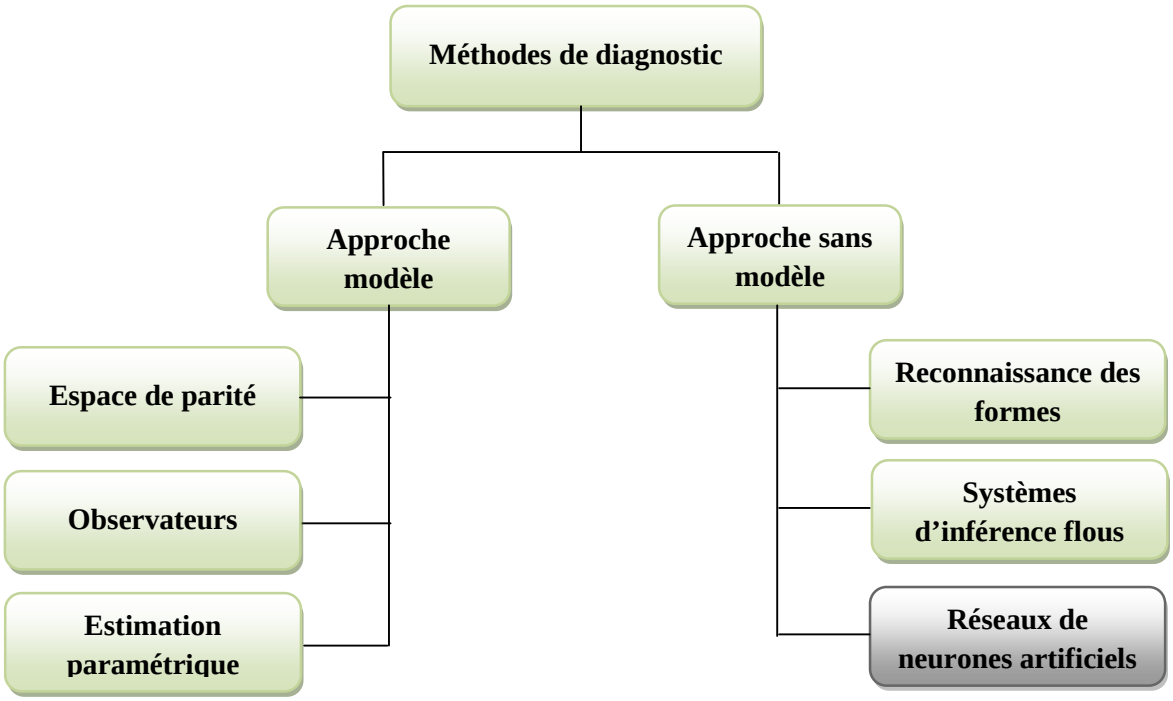


Figure (I.3) : *principales méthodes utilisées en diagnostic des systèmes physiques*

I.5.1 Approche avec modèle

Ces méthodes reposent sur l'utilisation de modèles du processus à surveiller incluant ou non l'influence des défauts et des perturbations sur l'état et la sortie. La sortie du modèle est comparée aux données accessibles pour former un résidu exploité pour alimenter un mécanisme dédié à la détection des défauts. Selon ce mécanisme de diagnostic, on distingue [**24]**:

- la surveillance par les observateurs,
- la surveillance par redondance analytique (espace de parité),
- la surveillance par estimation paramétrique.

Le principe général du diagnostic à base de modèles analytiques appliqué à la machine asynchrone est illustré à la figure (I.4) **[24]**.

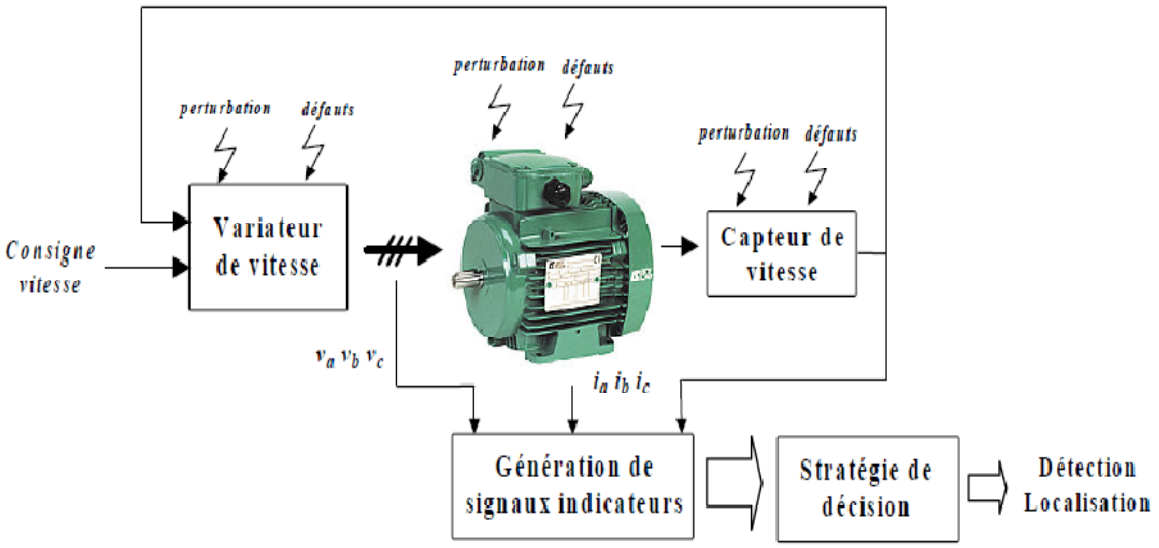


Figure (I.4) : *Principe général du diagnostic à base de modèles analytiques*

I.5.1.1 Diagnostic de défauts par observateurs

Les observateurs sont généralement utilisés dans la synthèse des lois de commande des machines à induction. Ils sont sensés pour fonctionner dans le contexte d'un système sans défaut : ils sont donc choisis et calculés en fonction de critères de stabilité et de performance. Un défaut se traduisant par la rupture des hypothèses d'équilibre du modèle, il peut se révéler dans la rupture des grandeurs observées par rapport aux grandeurs mesurées. La comparaison de la sortie réelle avec la sortie observée peut donc fournir des informations exploitables pour la détection et la localisation des défauts. On utilise généralement les observateurs de *Luenberger*, les observateurs à entrées inconnues ou les observateurs à grand gain...etc. **[24]**.

I.5.1.2 Diagnostic de défauts par redondance analytique

Comme pour les observateurs, la redondance analytique et la projection dans l'espace de parité sont basées sur l'exploitation des résidus issus du modèle de bon fonctionnement. La philosophie de cette approche est d'exploiter la redondance analytique existant dans les équations d'état lorsque celles-ci sont écrites sur un horizon d'observation. Le vecteur de résidus est alors génère en projetant les mesures observées sur cet horizon dans un espace appelé espace de parité **[24]**.

I.5.1.3 Diagnostic de défauts par estimation paramétrique

Dans la plupart des cas pratiques les paramètres de processus ne sont pas partiellement connus ou ne sont pas connus du tout. Puis, ils peuvent être déterminés avec des méthodes d'estimation paramétrique en mesurant l'entrée et les signaux de sortie si la structure modèle de base est connue **[3]**.

Cette approche possède l'avantage d'apporter de l'information sur la taille des déviations. Toutefois, un des inconvénients majeurs de la méthode réside dans la nécessité d'avoir un système physique excité en permanence. Ceci pose des problèmes pratiques dans le cas de procédés dangereux ou fonctionnant en mode stationnaire. De plus, les relations entre les paramètres mathématiques et physiques ne sont pas toujours inversibles de façon unitaire, ce qui complique la tâche du diagnostic basé sur les résidus **[24]**.

I.5.2 Approche sans modèle

I.5.2.1 Diagnostic de défauts par reconnaissance des formes

La Reconnaissance des Formes est un des nombreux aspects de l'intelligence artificielle ou plus exactement de l'intelligence calculatoire. A partir d'un ensemble de données ou d'informations apprises, elle offre la possibilité d'interpréter toute nouvelle observation (ou forme) **[11]**.

Les méthodes de diagnostic qui utilisent la reconnaissance des formes sont peut nombreux à ce jour. Un vecteur de paramètres, appelé vecteur de forme, est extrait à partir de plusieurs mesures. Les règles de décisions adoptées permettent de classer les observations, décrites par le vecteur de forme, par rapport aux différents modes de fonctionnement connus avec et sans défaut [**12**].

Pour classer ces observations, il faut obligatoirement être en mesure de fournir les données pour tel ou tel mode de fonctionnement (fonctionnement avec rotor sain à 0% de charge ou alors fonctionnent avec barre cassée à 100% de charge par exemple). Pour cela, il faut disposer d'une base de données, ce qui permettra ensuite de construire la classe correspondante au défaut crée (possibles pour les machines de petites et moyennes puissances). Une autre voie consisterait à calculer les paramètres du vecteur de forme en effectuant des simulations numériques de la machine étudiée (indispensable pour les moteurs de fortes puissances). Dans la dernière configuration, il faut un model

comportemental de la machine relativement précis pour obtenir des paramètres les plus proches possibles de la réalité [**12**].

I.5.2.2 Diagnostic de défauts par systèmes d'inférence flous

Pendant les vingt dernières années, les systèmes d'inférence floue (SIF) dont les bases relèvent de la théorie des ensembles flous de « *Zadeh* » sont devenus très populaires. Les applications dans le traitement du signal, la modélisation, la commande, la supervision de procédés et la prise de décision sont en effet autant d'applications qui démontrent la capacité des SIF à traiter des problèmes non linéaires grâce à l'utilisation de connaissances expertes.

La structure de base d'un SIF est constituée de **[1]** :

- un univers de discours qui contient les fonctions d'appartenance des variables d'entrée et de sortie à des classes. Ces fonctions peuvent avoir différentes formes, les plus usuelles étant les formes triangulaires, trapézoïdales, et gaussiennes,
- une base de connaissance qui regroupe les règles liant les variables d'entrées et de sorties sous la forme « Si…Alors… »,
- un mécanisme de raisonnement qui base son fonctionnement sur la logique du *modus ponens* généralisé. Les « SIF » peuvent être qualifiées de méthode « boîte grise »

I.5.2.3 Diagnostic de défauts par réseaux de neurones artificiels

Ces dernières années, la surveillance et la détection de défauts des machines électriques se sont déplacées des techniques traditionnelles aux techniques d'intelligence artificielle (IA) Les tendances de recherches prouvent que les techniques (IA) auront un plus grand rôle dans le système diagnostique de moteurs électriques avec la praticabilité, la sensibilité, la fiabilité, et l'automation [**2]**.

Quand la connaissance sur le procédé à surveiller n'est pas suffisante et que le développement d'un modèle de connaissance du procédé est impossible, l'utilisation de modèle dit « boîte noire » peut être envisagée. Pour cela des réseaux de neurones artificiels (RNA) ont été largement utilisés **[1]**. Dernièrement, les réseaux de neurones ont connu une utilisation large en ce qui concerne la modélisation, la commande et la surveillance des systèmes industriels **[26]**.

Un réseau de neurone est en fait un modèle de calcul dont la conception est très schématiquement inspirée du fonctionnement de vrais neurones humains. Cette technique est placée dans la famille des méthodes de l'intelligence artificielle qu'ils enrichissent en permettant de prendre des décisions s'appuyant davantage sur la perception que sur le raisonnement logique formel **[9]**.

Dans les années 1940, les neurologues Warren Sturgis McCulloch et Walter Pitts menèrent les premiers travaux sur les réseaux de neurones. Ils constituèrent un modèle simplifié de neurone biologique communément appelé neurone formel. Ils montrèrent également théoriquement que des réseaux de neurones formels simples peuvent réaliser des fonctions logiques, arithmétiques complexes **[9]**.

De manière générale, l'utilisation des RNA se fait en deux phases. Tout d'abord, la synthèse du réseau est réalisée et comprend plusieurs étapes : le choix du type de réseau, du type de neurones, du nombre de couches, des méthodes d'apprentissage **[1]**.

Pour identifier des défauts dans un système, le diagnostic réalisé par réseaux de neurones doit disposer d'un nombre suffisant d'exemples de bon fonctionnement et de défauts pour pouvoir les apprendre. Pendant la phase d'apprentissage, les exemples sont présentés au réseau en entrée avec les diagnostics correspondants à la sortie. Le réseau s'auto-organise, apprenant à relier les exemples montrés aux diagnostics. Après l'apprentissage, le réseau ne reconnaît pas seulement les exemples appris mais également des paradigmes leur ressemblant, ce qui correspond à une certaine robustesse par rapport aux déformations de signaux par le bruit.
Cependant, l'inconvénient majeur est d'arriver à déterminer une méthodologie pour maîtriser les problèmes inhérents, qui sont principalement le choix de la structure, de la taille du réseau et des algorithmes d'apprentissage pour un problème précis **[9]**.

Par contre, la principale raison de leur intérêt en diagnostic industriel est leur faculté et de mémorisation d'un grand volume d'information **[9]**. Leur faible sensibilité aux bruits de mesure, leur capacité à résoudre des problèmes non linéaires et multivariables, à stocker la connaissance de manière compacte, à apprendre en ligne et en temps réel, sont de propriétés qui les rendent attrayants pour cette utilisation **[1]**.

Leur emploi peut alors se faire à trois niveaux **[1]**:

- comme modèle du système à surveiller en état normal et générer un résidu d'erreur entre les observations et les prédictions,

- comme système d'évaluation de résidus pour le diagnostic,
- ou comme système de détection en une étape (en tant que classificateurs).

I.6. Position du problème à résoudre

Le problème à résoudre en terme 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) paraissent très intéressants pour la mise en place de la procédure du diagnostic ;

Par ailleurs, les réseaux de neurones possèdent des caractéristiques permettant la résolution de problèmes complexes, à savoir **[10, 11]** :

- 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, réduit 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. **[37, 38, 39, 40, 41]**

Il est clair qu'un FPGA peut répondre efficacement aux défis actuels et futurs parmi lesquels **[31, 40, 41, 43, 44]** :

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 fonctionnements,
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'implémentation, l'application de méthodologies très avancées et la spécification du temps de la mise en application, et le l'intégration de tous 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 FPGA en utilisant la Co-simulation. Cette dernière est proche au cas réel par rapport à la simulation numérique. On peut l'appeler aussi Hardware in the loop car dans cet environnement, on fait la simulation et l'implémentation en même temps du système étudié qui est dans notre cas le module du réseau de neurones artificiels.

I.7 Conclusion

Dans ce chapitre nous avons présenté les différents outils et les différentes méthodes du diagnostic des défaillances de la machine asynchrone. Ainsi nous avons évoqué le problème à résoudre en termes de choix et implémentation de la méthode du diagnostic adéquate pour la détection des défaillances de l'association machine asynchrone-convertisseur et commande.

A cet effet, notre choix a été opte pour la méthode des réseaux de neurones artificiels, en suite ils seront implémentés sur un circuit FPGA en utilisant la Co-simulation.

Enfin, avant de passer a l'implémentation des réseaux de neurones le deuxième chapitre a la modélisation et la simulation de l'association machine-convertisseur et commande en présence des défaillances.

Chapitre II

Etude du comportement de la MAS en présence des défauts

II.1 Introduction

Entrainer des charges à vitesse variable, tout en contrôlant la vitesse ou le couple, est devenu incontournable dans les industries modernes. De la montre au concasseur, en passant par l'électroménager, les systèmes de traction, de levage, les applications sont innombrables et variés, de même que les solutions techniques. Toutefois, depuis une vingtaine d'années, les entrainements par moteurs électriques connaissent un essor important. Ceci est du en grande partie aux progrès réalisés dans le domaine de la commande des machines a courant alternatif, et est grâce au développement de la technologie des composants de l'électronique de puissance, et l'apparition des processus numériques a fréquence élevée et a forte puissance de calcul **[15, 20]**.

Dans ce chapitre, on présentera, dans une première partie, la modélisation d'une machine asynchrone associée à un convertisseur statique (un onduleur à MLI).

La simulation de la machine asynchrone en état sein, contrôlé par commande vectorielle à orientation de flux rotorique sera donnée dans ce chapitre ainsi que La simulation de la machine asynchrone en cas de défaillance à la fin de ce chapitre.

II.2 Modélisation de la MAS

Les modèles des machines électriques les plus utilisés sont basés sur la théorie unifiée des machines électriques classiques, dite encore théorie généralisé. Cette théorie est basée sur la transformation de *Park*, qui rapporte les équations électriques statoriques et rotoriques à un système cartésien d'axes, « *d*, *q* » **[23]**.

II.2.1 Hypothèses simplificatrices

Les phénomènes physiques inhérents au fonctionnement du système peuvent être partiellement ou totalement pris en compte dans un modèle. Ils découlent plusieurs niveaux de modélisation liés aux hypothèses simplificatrices associées.

Plus le nombre d'hypothèses est grand, plus simple sera le système. Cela permet une étude et une exploitation plus aisées. Ces simplifications proviennent des propriétés des machines à courant alternatifs **[15]**.

Pour cette raison, on adopte les hypothèses suivantes **[19]** :

- entrefer constant,
- effet des encoches négligé,
- distribution spatiale sinusoïdale des forces magnétomotrices d'entrefer,
- circuit magnétique non saturé et à perméabilité constante,
- pertes ferromagnétiques négligeables,
- l'influence de l'effet de peau et de l'échauffement sur les caractéristiques n'est pas prise en compte.
- l'additivité des flux,
- la constance des inductances propres,
- la loi de variation sinusoïdale des inductances mutuelles entre les enroulements statoriques et rotoriques en fonction de l'angle électrique de leurs axes magnétiques.

La figure (II.1) tiré du **[19]** représente la distribution spatiale des enroulements statoriques et rotoriques d'une telle machine asynchrone.

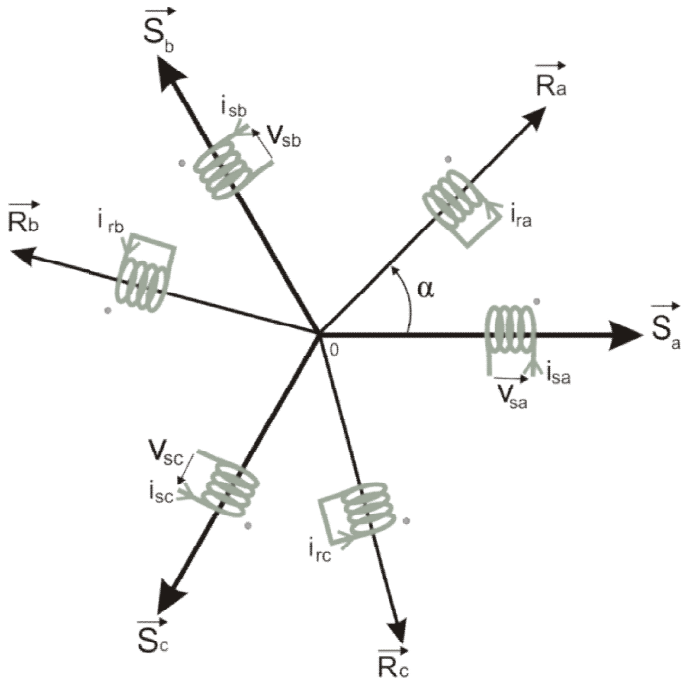


Figure (II.1) : *Représentation des enroulements du MAS triphasé dans l'espace électrique*

II.2.2 Equations générales de la machine

II.2.2.1 Equations des tensions

Par application de la loi de *Faraday* à l'un des six enroulements statoriques et rotoriques de la machine, la loi des mailles s'exprime par la relation **[19]** :

$$v = Ri + \frac{d\phi}{dt} \qquad (II.1)$$

On déduit pour l'ensemble des phases,

Statoriques :

$$\begin{bmatrix} v_{sa} \\ v_{sb} \\ v_{sc} \end{bmatrix} = \begin{bmatrix} R_s & 0 & 0 \\ 0 & R_s & 0 \\ 0 & 0 & R_s \end{bmatrix} \begin{bmatrix} i_{sa} \\ i_{sb} \\ i_{sc} \end{bmatrix} + (d/dt) \begin{bmatrix} \phi_{sa} \\ \phi_{sb} \\ \phi_{sc} \end{bmatrix} \qquad (II.2)$$

Ou :

$$[v_{sabc}] = [R_s][i_{sabc}] + d/dt[\phi_{sabc}] \quad (II.3)$$

et rotoriques :

$$\begin{bmatrix} v_{ra} \\ v_{rb} \\ v_{rc} \end{bmatrix} = \begin{bmatrix} R_r & 0 & 0 \\ 0 & R_r & 0 \\ 0 & 0 & R_r \end{bmatrix} \begin{bmatrix} i_{ra} \\ i_{rb} \\ i_{rc} \end{bmatrix} + (d/dt) \begin{bmatrix} \phi_{ra} \\ \phi_{rb} \\ \phi_{rc} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (II.4)$$

Ou :

$$[v_{rabc}] = [R_r][i_{rabc}] + d/dt[\phi_{rabc}] = [0] \quad (II.5)$$

Le rotor étant en court-circuit ses tensions sont nulles.

II.2.2.2 Equation des flux

$$\begin{bmatrix} \phi_{sa} \\ \phi_{sb} \\ \phi_{sc} \\ \phi_{ra} \\ \phi_{rb} \\ \phi_{rc} \end{bmatrix} = \begin{bmatrix} l_s & M_s & M_s & M_1 & M_3 & M_2 \\ M_s & l_s & M_s & M_2 & M_1 & M_3 \\ M_s & M_s & l_s & M_3 & M_2 & M_1 \\ M_1 & M_2 & M_3 & l_r & M_r & M_r \\ M_3 & M_1 & M_2 & M_r & l_r & M_r \\ M_2 & M_3 & M_1 & M_r & M_r & l_r \end{bmatrix} \begin{bmatrix} i_{sa} \\ i_{sb} \\ i_{sc} \\ i_{ra} \\ i_{rb} \\ i_{rc} \end{bmatrix} \quad (II.6)$$

Les coefficients instantanés de mutuelle inductance entre le rotor et le stator s'expriment en fonction de M_{sr} et de α :

$$\begin{cases} M_1 = M_{sr}\cos(\alpha) \\ M_2 = M_{sr}\cos(\alpha - 2\pi/3) \\ M_3 = M_{sr}\cos(\alpha + 2\pi/3) \end{cases} \quad (II.7)$$

La matrice des flux réels fait apparaitre quatre sous-matrices d'inductances **[19]** :

$$\begin{bmatrix} \phi_{sabc} \\ \phi_{rabc} \end{bmatrix} = \begin{bmatrix} [L_s] & [M_{sr}] \\ [M_{rs}] & [L_r] \end{bmatrix} \begin{bmatrix} i_{sabc} \\ i_{rabc} \end{bmatrix} \quad (II.8)$$

$$avec: [L_s] = \begin{bmatrix} l_s & M_s & M_s \\ M_s & l_s & M_s \\ M_s & M_s & l_s \end{bmatrix} \quad (II.9) \qquad et \quad [L_r] = \begin{bmatrix} l_r & M_r & M_r \\ M_r & l_r & M_r \\ M_r & M_r & l_r \end{bmatrix} \quad (II.10)$$

$$[M_{sr}] = [M_{rs}]^t = M_{sr}\begin{bmatrix} \cos(\alpha) & \cos(\alpha + 2\pi/3) & \cos(\alpha - 2\pi/3) \\ \cos(\alpha - 2\pi/3) & \cos(\alpha) & \cos(\alpha + 2\pi/3) \\ \cos(\alpha + 2\pi/3) & \cos(\alpha - 2\pi/3) & \cos(\alpha) \end{bmatrix} \quad (II.11)$$

Finalement :

$$[v_{sabc}] = [R_s][i_{sabc}] + (d/dt)\{[L_s][i_{sabc}] + [M_{sr}][i_{rabc}]\} \quad (II.12)$$

$$[v_{rabc}] = [R_r][i_{sabc}] + (d/dt)\{[M_{sr}]^t[i_{sabc}] + [L_r][i_{rabc}]\} \quad (II.13)$$

II.2.2.3 Equations mécaniques

L'équation fondamentale de la mécanique décrivant la dynamique du rotor de la machine **[15]** :

$$\begin{cases} J\dfrac{d\Omega}{dt} + f_r\Omega = C_{em} - C_r \\ \omega = p.\Omega \end{cases} \quad (II.14)$$

avec :

- J : moment d'inertie du rotor,
- Ω : vitesse angulaire mécanique du rotor,
- f_r : coefficient de frottement visqueux,
- C_{em} : couple électromagnétique délivré par le moteur,
- C_r : couple résistant, ou de charge,
- ω : vitesse angulaire électrique du rotor.

II.2.3 Modélisation de Park de la machine asynchrone

II.2.3.1 Principe de la transformation de Park

La transformation directe de *Park* est définie par la matrice [*P*]. Aux vecteurs originaux [v_{abc}], [i_{abc}] et [ϕ_{abc}], la transformation de *Park* fait correspondre les vecteurs [v_{dq0}], [i_{dq0}] et [ϕ_{dq0}]. La transformation de *Park* est appliquée de manière identique au vecteur de tensions, de courants, et de flux $[x_{dqo}] = [x_o\ x_d\ x_q]^t$. Le vecteur « x_o » représente la composante homopolaire, normale au plan formé par les vecteurs « x_a », « x_b », et

« x_c ». Les vecteurs « x_d » et « x_q » représentent les vecteurs diphasés qui correspondent aux vecteurs « x_a », « x_b », et « x_c » **[15, 20]**.

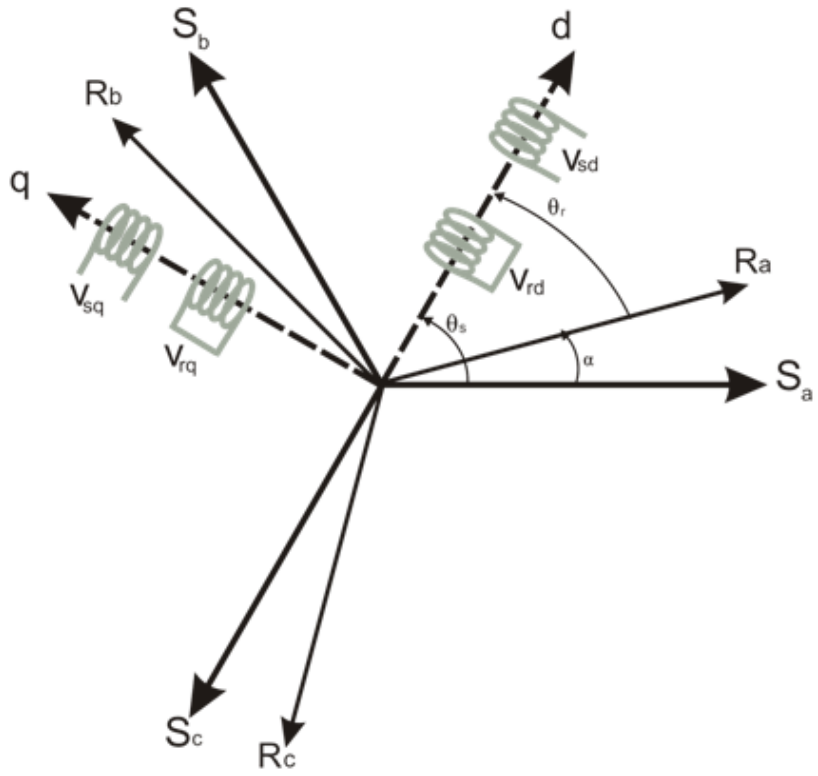


Figure (II.2) : *Représentation du passage d'un système triphasé à celui biphasé* **[19]**.

La transformation de *Park* est définie par :

$$[x_{abc}] = [P]^{-1}[x_{dq0}] \tag{II.15}$$

$$[x_{dq0}] = [P]\,[x_{abc}] \tag{II.16}$$

où $[P]$ et $[P]^{-1}$ sont les matrices de passage direct et inverse, elles sont données par :

$$[P] = c.\begin{bmatrix} \cos\psi & \cos(\psi - 2\pi/3) & \cos(\psi + 2\pi/3) \\ -\sin\psi & -\sin(\psi - 2\pi/3) & -\sin(\psi + 2\pi/3) \\ 1/\sqrt{2} & 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \tag{II.17}$$

$$[P]^{-1} = c.\begin{bmatrix} \cos\psi & -\sin\psi & 1/\sqrt{2} \\ \cos(\psi - 2\pi/3) & -\sin(\psi - 2\pi/3) & 1/\sqrt{2} \\ \cos(\psi + 2\pi/3) & -\sin(\psi + 2\pi/3) & 1/\sqrt{2} \end{bmatrix} \tag{II.18}$$

avec $(\psi = \theta_s)$ pour le stator, ou $(\psi = \theta_r)$ pour le rotor.

Où « *c* » est une constante qui peut prendre soit les valeurs (2/3) ou 1 pour la non conservation de puissance, soit la valeur $\sqrt{(2/3)}$ pour une conservation de puissance [**18]**. Dans notre cas, nous prendrons : c= $\sqrt{(2/3)}$

On remarque sur la figure (II.2) que « θ_s » et « θ_r » sont naturellement liés à « α » par la relation rigide :

$$(\theta_s - \theta_r) = \alpha \tag{II.19}$$

On déduit par dérivation :

$$(\omega_s - \omega_r) = \frac{d\alpha}{dt} = \omega = p\Omega \tag{II.20}$$

où:

- ω_s : vitesse des axes « *d, q* » dans le repère statorique,
- ω_r : vitesse des axes « *d, q* » dans le repère statorique.

II.2.3.2 Equations des tensions

Les équations de *Park* statoriques et rotoriques s'écrivent :

$$\begin{cases} v_{sd} = R_s i_{sd} + (d\phi_{sd}/dt) - (d\theta_s/dt)\phi_{sq} \\ v_{sq} = R_s i_{sq} + (d\phi_{sq}/dt) + (d\theta_s/dt)\phi_{sd} \\ v_{rd} = R_r i_{rd} + (d\phi_{rd}/dt) - (d\theta_r/dt)\phi_{rq} = 0 \\ v_{rq} = R_r i_{rq} + (d\phi_{rq}/dt) + (d\theta_r/dt)\phi_{rd} = 0 \end{cases} \tag{II.21}$$

avec :
$$\begin{cases} \omega_s = d\theta_s/dt \\ \omega_r = d\theta_r/dt \end{cases} \tag{II.22}$$

II.2.3.3Equations magnétiques

$$\begin{cases} \phi_{sd} = L_s i_{sd} + M i_{rd} \\ \phi_{sq} = L_s i_{sq} + M i_{rq} \\ \phi_{rd} = L_r i_{rd} + M i_{sd} \\ \phi_{rq} = L_r i_{rq} + M i_{sq} \end{cases} \tag{II.23}$$

II.2.3.4 Equation mécanique

$$J\frac{d\Omega}{dt} = C_{em} - C_r - f_r\Omega \qquad (II.24)$$

II.2.4 Expression du couple électromagnétique

Le couple électromagnétique peut prendre diverses formes, en fonction des variables que l'on élimine (où *p* est le nombre de paires de pôles) **[20]**:

$$C_{em} = \frac{pM}{L_r}(\phi_{rd} i_{sq} - \phi_{rq} i_{sd}) \qquad (II.25)$$

II.3 Choix du référentiel

Il y a trois transformations de référentiel qui sont couramment employées dans la simulation des machines électriques. Dans chaque cas il s'agit d'assigner une vitesse particulière au référentiel pour obtenir une transformation donnée. Ces transformations se font dans les référentiels suivants **[14]**:

- référentiel (*d, q*) fixé au stator ou stationnaire :

$$\frac{d\theta_s}{dt} = 0, \quad \frac{d\theta_r}{dt} = -\omega \qquad (II.26)$$

La transformation dans le référentiel (*d, q*) fixé au stator ou stationnaire porte aussi le nom de transformation de *Clarke* ou transformation (α, β). Ce référentiel étant fixe, la vitesse est nulle. Comme la vitesse est nulle, l'angle du référentiel demeurera constant et sa valeur est arbitraire. Nous avons choisi de travailler avec un angle nul **[14]**.

- référentiel (*d, q*) fixé au rotor :

$$\frac{d\theta_s}{dt} = \omega, \quad \frac{d\theta_r}{dt} = 0 \qquad (II.27)$$

La vitesse du référentiel ainsi que sa position angulaire deviennent celles du rotor.

- référentiel (*d, q*) synchrone :

$$\frac{d\theta_s}{dt} = \omega_s, \quad \frac{d\theta_r}{dt} = \omega_s - \omega = \omega_r \qquad (II.28)$$

Ce référentiel tourne à la vitesse du champ tournant du stator.

Le choix de transformation de référentiel est essentiellement dicté par les variables de phase que nous voulons observer ainsi que I' application.

Si les variables de phases au stator et au rotor ne nous intéressent pas, la machine peut être simulée dans le référentiel synchrone sans qu'aucune transformation ne soit nécessaire. Si l'on désire simuler cette machine dans des conditions équilibrées mais que cette fois, on désire observer les courants de phase statoriques, la transformation dans le référentiel stationnaire est celle qui requiert le moins de calculs. Si par contre on s'intéresse aux courants de phase rotoriques, la transformation dans le référentiel fixé au rotor sera la plus avantageuse **[14]**.

II.3.1 Référentiel lié au champ tournant

En substituant $(\text{II}.26)$ dans $(\text{II}.21)$ on obtient les équations de la machine asynchrone dans le repère (*d, q*) lié au champ tournant :

$$\begin{cases} v_{sd} = R_s i_{sd} + (d\phi_{sd}/dt) - \omega_s \phi_{sq} \\ v_{sq} = R_s i_{sq} + (d\phi_{sq}/dt) + \omega_s \phi_{sd} \\ v_{rd} = R_r i_{rd} + (d\phi_{rd}/dt) - (\omega_s - \omega)\phi_{rq} = 0 \\ v_{rq} = R_r i_{rq} + (d\phi_{rq}/dt) + (\omega_s - \omega)\phi_{rd} = 0 \end{cases} \qquad (\text{II}.29)$$

L'avantage d'utiliser ce référentiel, est d'avoir des grandeurs constantes en régime permanent. Il est alors plus aisé de faire la régulation [**29]**.

II.3.2 Référentiel lié au stator

Pour obtenir les équations de la machine asynchrone dans le référentiel (*d, q*) lié au stator, il suffit donc de substituer dans le système d'équations du référentiel (*d, q*) arbitraire (II.21) les valeurs (II.26), et en remplaçant « *d* » par « α » et « *q* » par « β ». Dans ces conditions, on obtient le système d'équations (II.30) :

$$\begin{cases} v_{s\alpha} = R_s i_{s\alpha} + (d\phi_{s\alpha}/dt) \\ v_{s\beta} = R_s i_{s\beta} + (d\phi_{s\beta}/dt) \\ v_{r\alpha} = R_r i_{r\alpha} + (d\phi_{r\alpha}/dt) + \omega_r \phi_{r\beta} = 0 \\ v_{r\beta} = R_r i_{r\beta} + (d\phi_{r\beta}/dt) - \omega_r \phi_{r\beta} = 0 \end{cases} \qquad (\text{II}.30)$$

C'est le repère le mieux adapté pour travailler avec les grandeurs instantanées, il possède des tensions et des courants réelles et peut être utilisé pour étudier les régimes de démarrage et de freinage des machines à courant alternatif

II.4 Simulation de la machine asynchrone alimentée en tension

II.4.1 Mise en équation d'état

Une caractéristique importante du modèle est la nature des variables d'état. Dans la plupart des modèles de machines électriques, les variables d'état sont généralement les courants circulant dans les divers enroulements de la machine ou les flux de ces mêmes enroulements. Le choix devrait être dicté par le système d'équations qui requiert le moins de calcul **[14]**.

Pour une machine asynchrone alimentée en tension, les tensions statoriques « $v_{s\alpha}$ » et « $v_{s\beta}$ » représentent les variables de commande, et nous considérons les courants statoriques $(i_{s\alpha}, i_{s\beta})$, les flux rotoriques $(\phi_{r\alpha}, \phi_{r\beta})$ et la pulsation mécanique «Ω» comme variables d'état, le couple résistant « C_r » étant comme perturbation. .

On cherche à obtenir un système d'équation écrit sous forme :

$$\begin{cases} \overset{\circ}{X} = AX + BU \\ Y = CX \end{cases} \qquad (\text{II}.31)$$

avec :

- X : vecteur d'état,
- Y : vecteur de sortie,
- A : matrice d'évolution d'état du système,
- B : matrice de commande (d'entrée),
- U : vecteur du système de commande,
- C : matrice d'observation.

Les équations d'état de la partie électrique de notre modèle s'obtiennent en substituant les flux de $(\text{II}.23)$ dans les équations des tensions $(\text{II}.30)$ puis en isolant les dérivées des courants et des flux. On obtient alors le système d'équations $(\text{II}.32)$ suivant :

$$\begin{cases} \dot{i}_{s\alpha} = -\gamma\, i_{s\alpha} + \dfrac{k}{T_r}\phi_{r\alpha} + k\omega\phi_{r\beta} + \dfrac{1}{\sigma L_s} v_{s\alpha} \\ \dot{i}_{s\beta} = -\gamma\, i_{s\beta} + k\omega\phi_{r\alpha} + \dfrac{k}{T_r}\phi_{r\beta} + \dfrac{1}{\sigma L_s} v_{s\beta} \\ \dot{\phi}_{r\alpha} = \dfrac{M}{T_r} i_{s\alpha} - \dfrac{1}{T_r}\phi_{r\alpha} - \omega\phi_{r\beta} \\ \dot{\phi}_{r\beta} = \dfrac{M}{T_r} i_{s\beta} + \omega\phi_{r\alpha} - \dfrac{1}{T_r}\phi_{r\beta} \end{cases} \qquad (\text{II}.32)$$

par identification :

$$A = \begin{bmatrix} -\gamma & 0 & \dfrac{k}{T_r} & k\omega \\ 0 & -\gamma & -k\omega & \dfrac{k}{T_r} \\ \dfrac{M}{T_r} & 0 & -\dfrac{1}{T_r} & -\omega \\ 0 & \dfrac{M}{T_r} & \omega & -\dfrac{1}{T_r} \end{bmatrix}, \quad B = \begin{bmatrix} \dfrac{1}{\sigma L_s} & 0 \\ 0 & \dfrac{1}{\sigma L_s} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \qquad (\text{II}.33)$$

$$X = \begin{bmatrix} i_{s\alpha} \\ i_{s\beta} \\ \phi_{r\alpha} \\ \phi_{r\beta} \end{bmatrix}, \qquad U = \begin{bmatrix} v_{s\alpha} \\ v_{s\beta} \end{bmatrix} \qquad (\text{II}.34)$$

avec :

- $k = \dfrac{M}{\sigma L_r L_s}$ *et* $\gamma = \dfrac{1}{\sigma L_s}(R_s + \dfrac{M^2}{L_r T_r})$,
- $\sigma = 1 - \dfrac{M^2}{L_s L_r}$: Facteur de dispersion,
- $T_r = \dfrac{L_r}{R_r}$: Constante du temps rotorique.

II.4.2 Equations mécaniques

De $(II.24)$, l'expression du couple électromagnétique exprimé dans notre référentiel (α, β), et celle du mouvement sont donnée par :

$$\begin{cases} C_{em} = \dfrac{pM}{L_r}(\phi_{r\alpha} i_{s\beta} - \phi_{r\beta} i_{s\alpha}) \\ J\dfrac{d\Omega}{dt} + C_r + f_r\Omega = C_{em} \end{cases} \qquad (II.35)$$

II.4.3 Modèle exprimé dans le repère (*d, q*) lié au champ tournant

Il est obtenu de la même manière que celui du référentiel (α, β), on obtient l'équation d'état de la machine asynchrone dans le repère (*d, q*) lié au champ tournant :

$$\begin{cases} \dfrac{di_{sd}}{dt} = -\gamma\ \ i_{sd} + \omega_s\ i_{sq} + \dfrac{k}{T_r}\phi_{rd} + k\omega\phi_{rq} + \dfrac{1}{\sigma.L_s}v_{sd} \\ \dfrac{di_{sq}}{dt} = -\omega_s\, i_{sd} - \gamma\ \ i_{sq} - k\omega\phi_{rd} + \dfrac{k}{T_r}\phi_{rq} + \dfrac{1}{\sigma.L_s}v_{sq} \\ \dfrac{d\phi_{rd}}{dt} = \dfrac{M}{T_r}i_{sd} - \dfrac{1}{T_r}\phi_{rd} + (\omega_s - \omega)\phi_{rq} \\ \dfrac{d\phi_{rq}}{dt} = \dfrac{M}{T_r}i_{sq}\ - (\omega_s - \omega)\phi_{rd} - \dfrac{1}{T_r}\phi_{rq} \end{cases} \qquad (II.36)$$

Ce modèle qui sera utilisé ci-après dans l'étude de la commande vectorielle de la MAS.

II.5 Simulation de démarrage à vide de la MAS triphasé

II.5.1 Montage de simulation

En ingénierie, la simulation est un moyen efficace et économique, couramment utilisé pour faire des études préliminaires et/ou comparatives, tant au stade du développement (conception), qu'au cours du fonctionnement normal des systèmes. Actuellement, plusieurs outils de simulation, parmi lesquels *Matlab/Simulink*, sont utilisés dans l'industrie et dans les milieux universitaires **[36]**.

Les tensions appliquées aux trois bobinages statoriques sont :

$$\begin{cases} v_1 = 310.\sin(\omega_s.t) \\ v_2 = 310.\sin(\omega_s.t - \frac{2\pi}{3}) \\ v_3 = 310.\sin(\omega_s.t + \frac{2\pi}{3}) \end{cases} \qquad (II.37)$$

Le schéma bloc de simulation du modèle de la machine asynchrone alimenté en tension est donné par la figure (II.4) :

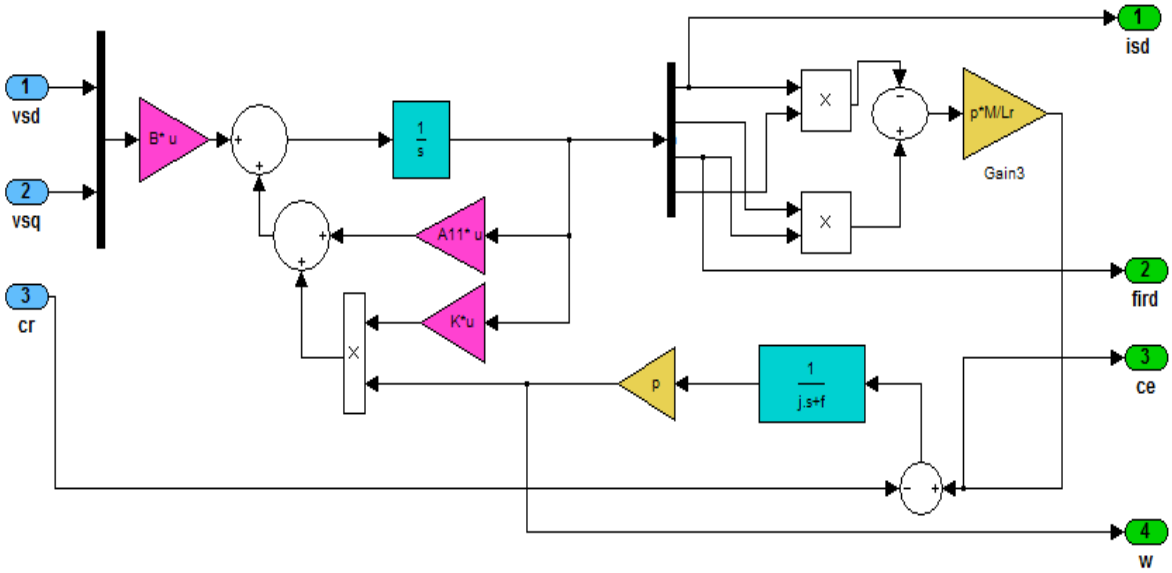


Figure (II.4) : *Le schéma bloc de simulation du modèle de la machine asynchrone*

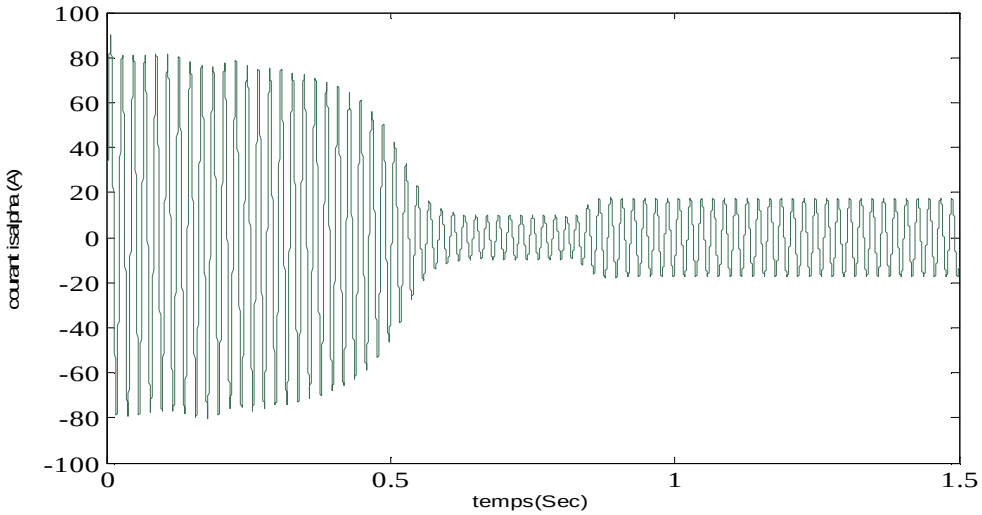
II.5.2 Résultats de simulation

La machine asynchrone est alimenté directement par le réseau triphasé 220/380V, de fréquence : 50Hz, cette machine démarre à vide, afin d'appliqué un couple résistant de (*Cr* = 15 N.m) à (t= 0.8*sec*).

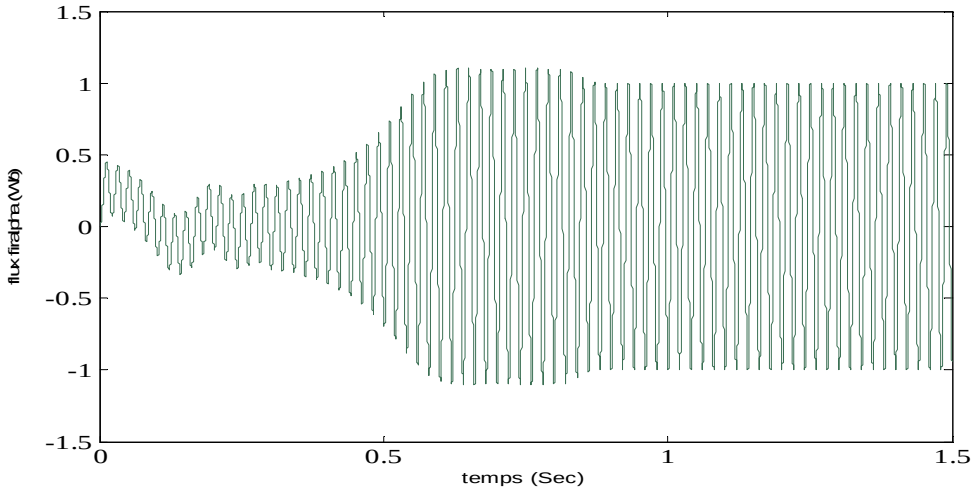
Nous illustrons la vitesse de rotation du moteur exprimé en (*rad/s*), le couple électromagnétique ainsi que le courant statorique et le flux rotorique.

Les paramètres de la machine asynchrone sont indiqués en Annexe (A).

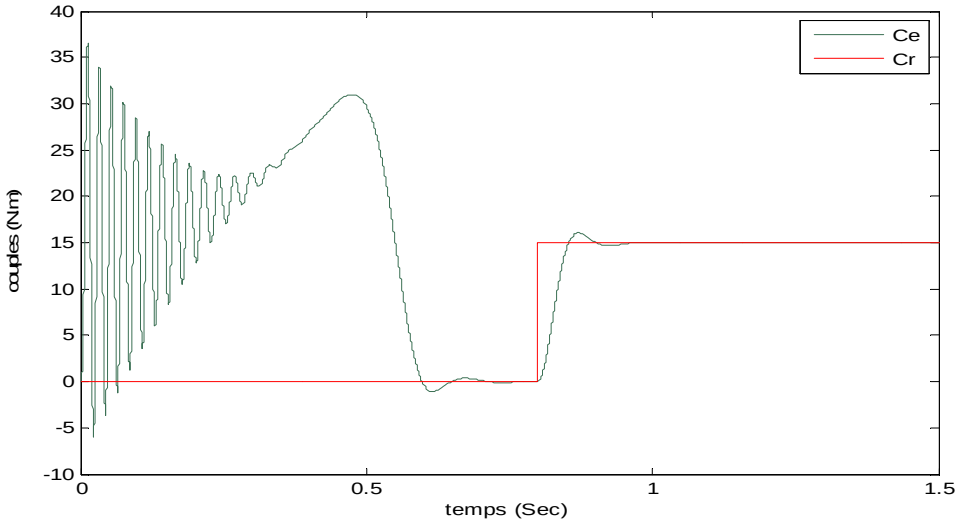
Les résultats de simulation sont représentés par la figure (II.5).


courant isalpha (A)
temps(Sec)

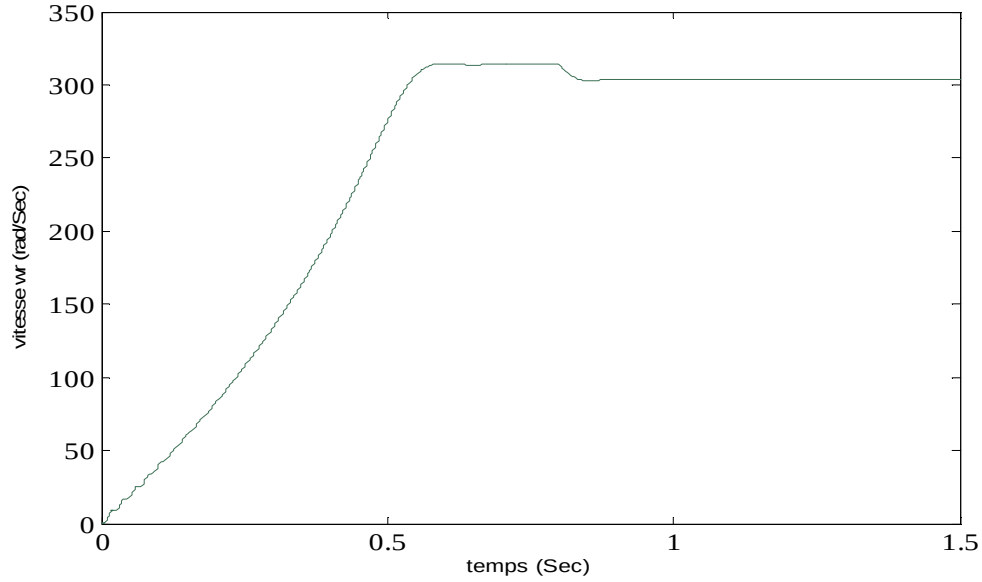
(a)


flux firalpha (Wb)
temps (Sec)

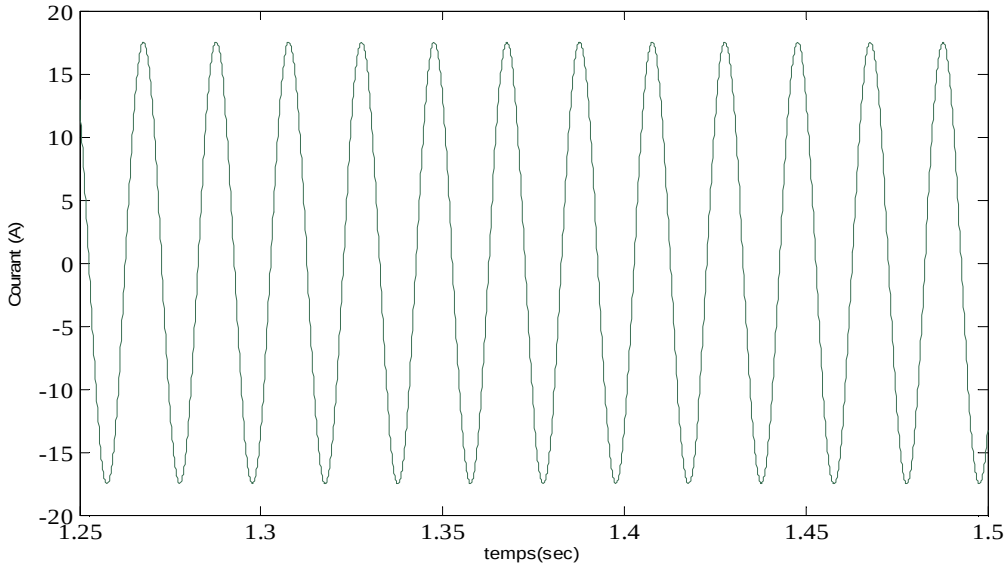
(b)


Ce
Cr
couples (Nm)
temps (Sec)

(c)



(d)



(e)

Figure (II.5) : *Résultats de simulation d'un démarrage directe de la MAS à vide alimentée en tension, suivie d'une application d'une perturbation de* (*Cr=1.5 N.m*) *à* (*t =0.8sec*).

(a) Réponse du courant statorique $i_{s\alpha}$ (A),

(b) Réponse du flux rotorique $\phi_{r\alpha}$ (Wb),

(c) Réponse du couple électromagnétique C_{em} (N.m),

(d) Réponse de la vitesse de rotation w (rad/sec),

(e) Réponse du courant statorique en régime permanent $i_{s\alpha}$ (A).

II.5.3 Interprétation des courbes

Lors de démarrage à vide, un fort appel au courant qui est égal à environ 4 fois le courant nominal présenté sous forme d'oscillations successives, qui se stabilisent en régime permanent.

Le couple est fortement pulsatoire à cause des bruits générés par la partie mécanique. Il atteint une valeur maximale de l'ordre de (0.98) fois le couple nominale, et il tend vers zéro (charge nulle) au régime permanent.

La vitesse fait un accroissement presque linéaire, jusqu'à ce qu'il stabilise à une valeur proche de la vitesse de synchronisme (la machine tourne à vide).

Le flux, fait des oscillations de faible amplitude, jusqu'à ce qu'il stabilise a des oscillations sinusoïdales.

Lors de l'application d'une perturbation, la vitesse et le flux sont diminués mais ils sont restent stable, le couple augmente afin de compenser le couple résistant et le courant aussi est augmenté.

II.6 Modélisation de l'alimentation du MAS

Les onduleurs de tension, associés aux machines à courant alternatif, sont de nos jours très largement utilisés dans les systèmes d'entraînement industriels. En premier lieu, les progrès en matière de semi-conducteur ont permis la réalisation de convertisseurs statiques de plus en plus performants. En second lieu, l'évolution des techniques numériques, notamment l'utilisation sans cesse grandissante des processeurs de signaux (DSP, "*Digital Signal Processing*") et des systèmes à base d'architecture reconfigurable (FPGA,"*Field Programmable Gate Array*"), permet désormais d'exécuter en temps réel des algorithmes complexes de contrôle des convertisseurs **[25]**.

II.6.1 Modélisation de l'onduleur de tension triphasé

Un onduleur de tension triphasé est constitué de trois cellules (bras) de commutation de deux interrupteurs. Pour chaque interrupteur est formé par un transistor (ou un thyristor) monté en tête-bêche avec une diode de récupération. Pour assurer la continuité des courants alternatifs et éviter le court-circuitage de la source, les interrupteurs k_{11} et k_{12}, k_{21} et k_{22}, k_{31} et k_{32} doivent être contrôlé de manière complémentaire **[15, 17 ,19]**.

Le schéma structurel de l'onduleur de tension alimentant le stator du moteur asynchrone est donné par la figure (II.6) :

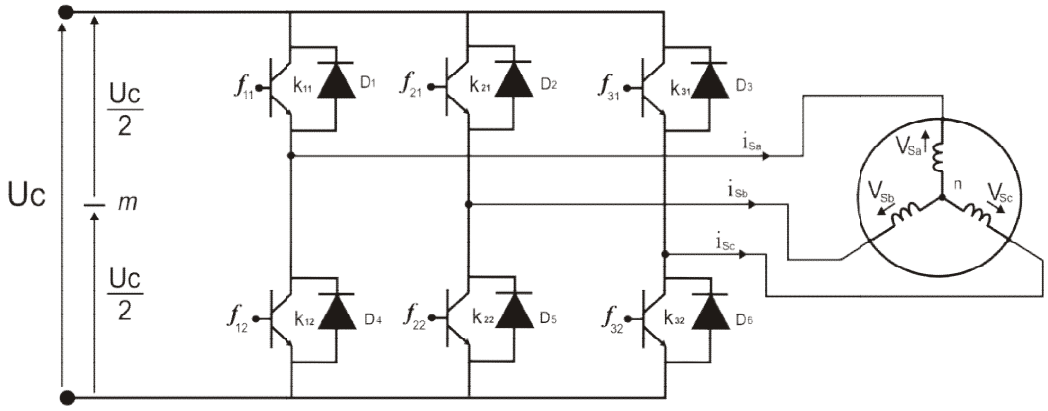


Figure (II.6) : *Schéma d'un onduleur de tension triphasé alimentant le stator du MAS*

Pour simplifier l'étude, on supposera que **[17]** :

- la commutation des interrupteurs est instantanée,
- la chute de tension aux bornes des interrupteurs est négligeable,
- la charge triphasé est équilibrée, couplé en étoile avec un neutre isolé.

D'où on présente chaque paire transistor-diode par un seule interrupteur bidirectionnel.

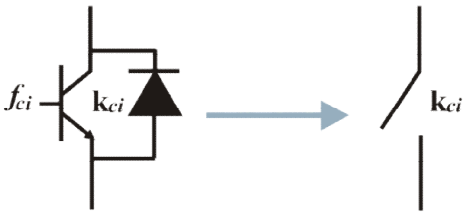


Figure (II.7) : *Interrupteur bidirectionnel de paire transistor-diode*

On définit la fonction de connexion f_{ci} ($c \in \{1, 2, 3\}$, $i \in \{1, 2\}$) comme l'état de l'interrupteur k_{ci}, on a :

$$\begin{cases} f_{11} = 1 - f_{10} \\ f_{21} = 1 - f_{20} \\ f_{31} = 1 - f_{30} \end{cases} \qquad (II.38)$$

avec :

- f_{ci} =1 si l'interrupteur est fermé,
- f_{ci} =0 si l'interrupteur est ouvert.

L'onduleur est alimenté par une source de tension continue constante, d'amplitude «U_c». Les potentiels des nœuds « *a* », « *b* » et « *c* » de l'onduleur triphasé par rapport au point milieu fictif « *n* » sont donnés par les tensions suivantes :

$$\begin{cases} v_{an} = f_{11} U_c \\ v_{bn} = f_{21} U_c \\ v_{cn} = f_{31} U_c \end{cases} \quad (\text{II}.39)$$

Les tensions composés délivrées par cet onduleur sont donnés par :

$$\begin{cases} u_{sab} = U_c (f_{11} - f_{21}) \\ u_{sbc} = U_c (f_{21} - f_{31}) \\ u_{sca} = U_c (f_{31} - f_{11}) \end{cases} \quad (\text{II}.40)$$

Pour une charge triphasé équilibrée, couplé en étoile avec un neutre isolé, les tensions statoriques simples sont reliés par :

$$v_{sa} + v_{sb} + v_{sc} = 0 \quad (\text{II}.41)$$

Les tensions simples sont liées aux tensions composées par :

$$\begin{cases} v_{sa} = \frac{1}{3}(u_{sab} - u_{sca}) \\ v_{sb} = \frac{1}{3}(u_{sbc} - u_{sab}) \\ v_{sc} = \frac{1}{3}(u_{sca} - u_{sbc}) \end{cases} \quad (\text{II}.42)$$

Après arrangement les équations des deux systèmes (II.40) et (II.42), on obtient le système matricielle suivant :

$$\begin{bmatrix} v_{sa} \\ v_{sb} \\ v_{sc} \end{bmatrix} = \frac{U_c}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{21} \\ f_{31} \end{bmatrix} \quad (\text{II}.43)$$

Pour déterminer les fonctions « f_{ci} », on présentera dans ce qui suit la stratégie de commande de l'onduleur.

II.6.2 Commande de l'onduleur par la stratégie triangulo-sinusoïdale

La MLI Sinus-Triangle utilise le principe d'intersection entre une référence sinusoïdale de fréquence f, appelée modulante, et un signal triangulaire de haute fréquence f_p, appelée la porteuse P, pour déterminer les instants de commutation. Le schéma de principe est donné par la figure (II.8) **[25]**.

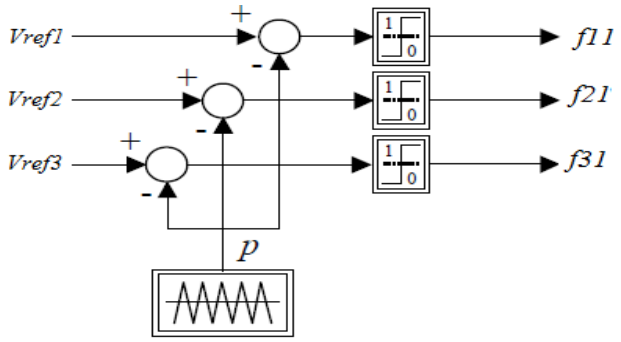


Figure (II.8) : *Principe de la MLI Sinus-Triangle*

Les signaux de référence sont définis par **[13]**:

$$V_{refj}(t) = V_m \sin\left(2\pi ft - 2(j-1)\frac{\pi}{3}\right), \quad j = 1,2,3 \qquad (II.44)$$

La porteuse est donnée par **[13]**:

$$V_p(t) = \begin{cases} V_{pm}\left(\dfrac{4t}{T_p} - (4n+1)\right) & si\ t \in [nT_p, \dfrac{n+1}{2}T_p] \\ V_{pm}\left(-\dfrac{4t}{T_p} + (4n+3)\right) & si\ t \in [\dfrac{n+1}{2}T_p, (n+1)T_p] \end{cases} \qquad (II.45)$$

avec : T_p période de V_p

La commande *MLI* est caractérisée par les deux paramètres **[23] :**

- l'indice de modulation m égal au rapport de la fréquence de modulation sur la fréquence de référence,

– Le coefficient de réglage en tension r égal au rapport de l'amplitude de la tension de référence à la valeur crête (U_c/2) de l'onde de modulation.

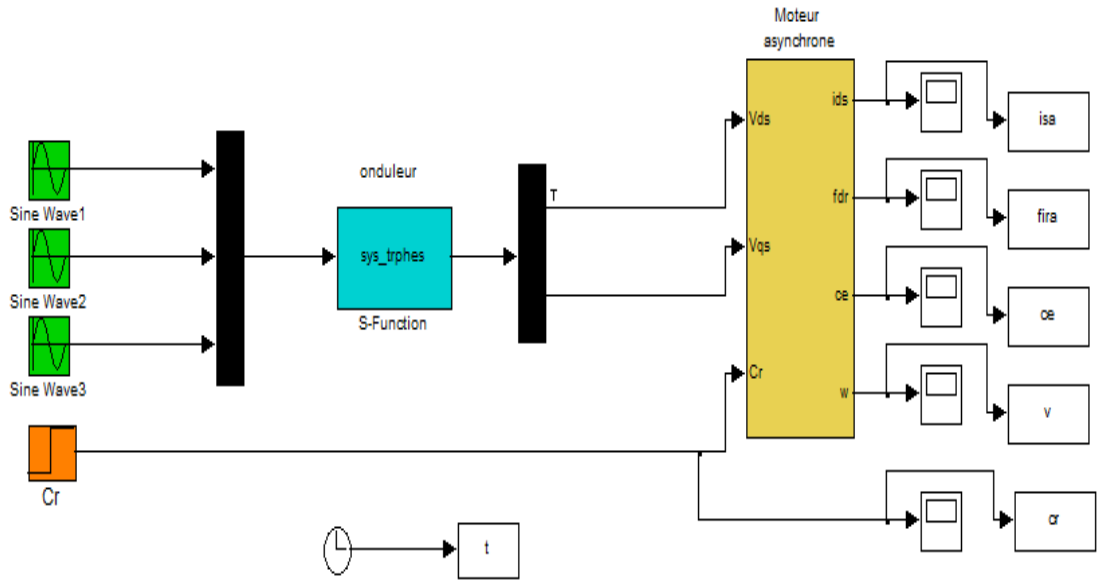
Notez que l'augmentation de m rejette les premiers harmoniques non nuls vers les fréquences élevées et donc facilite le filtrage. Mais m est limité par le temps de commutation des interrupteurs du l'onduleur et donc par la largeur minimale des impulsions [**23**].

Alors, le choix de m procède d'un compromis entre la neutralisation des harmoniques et le rendement de l'onduleur, dans notre travail nous avons prend m=100. Tandis que, par action sur r on peut faire varier la valeur efficace du fondamental de la tension de sortie.

II.6.3 Simulation et interprétation

La figure (II.9) représente une simulation de l'association Onduleur-MAS.

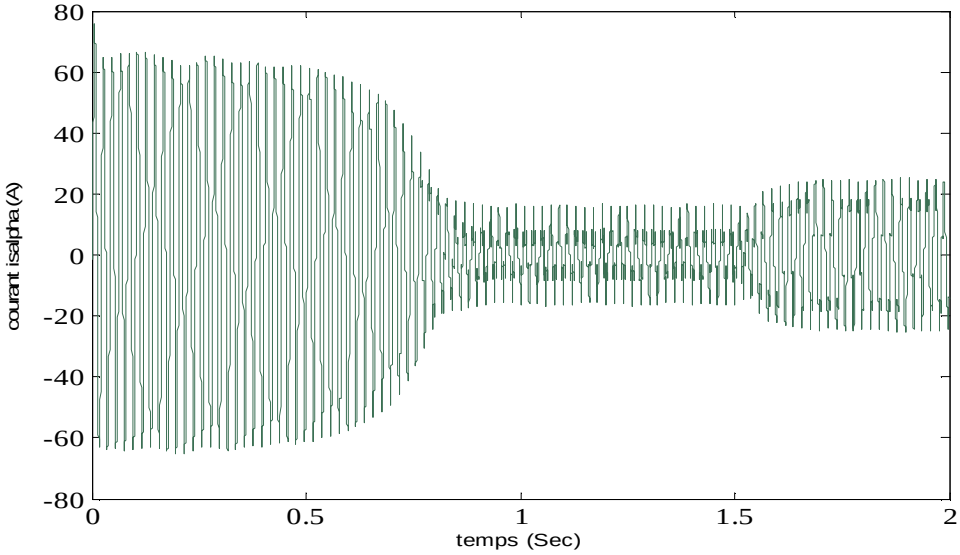
L'onduleur de tension est représenté par un bloc de « *S-Functio*n ».



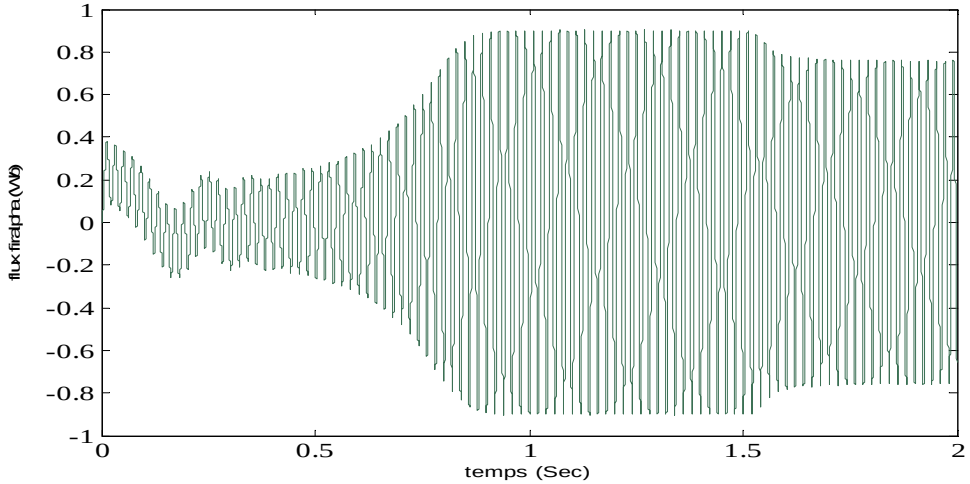
Figure(II.9) : *Le schéma bloc de simulation de l'association Onduleur-MAS*

L'architecture interne du bloc de l'onduleur sera donnée en Annexe (B).

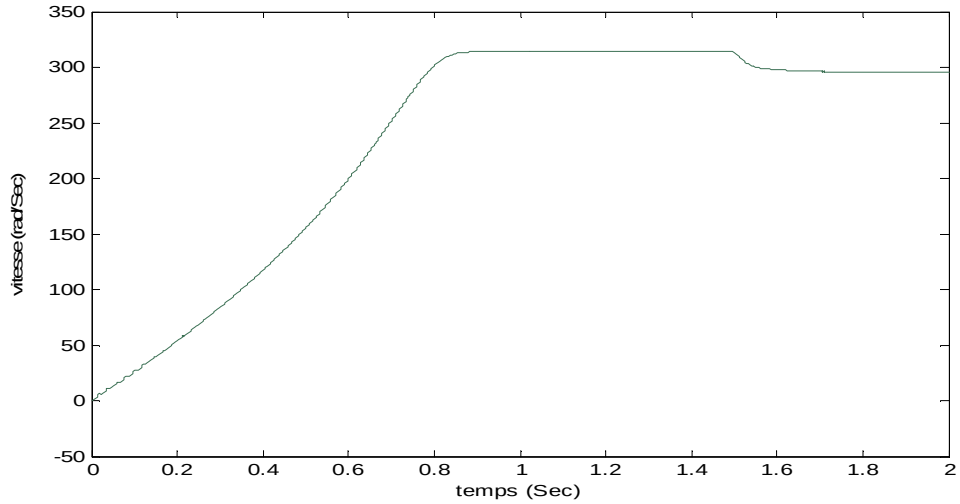
Les résultats de simulation sont présentés par la figure (II.10).



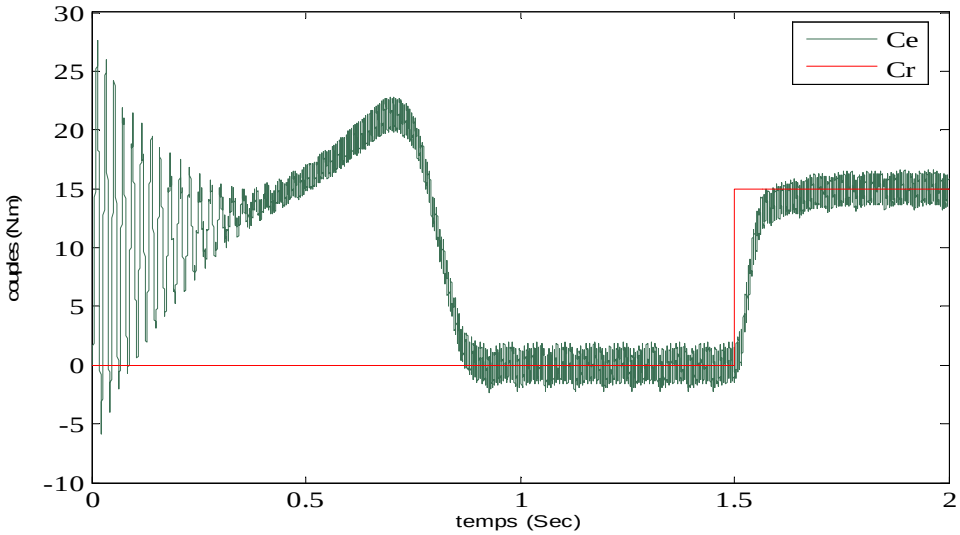
(a) *Réponse du courant statorique $i_{s\alpha}$(A).*



(b) *Réponse du flux rotorique $\phi_{r\alpha}$(Wb).*



(c) *Réponse de la vitesse de rotation w(rad/sec),*



(d) *Réponse du couple électromagnétique C_{em}(N.m),*

Figure (II.10) : *Résultats de simulation de l'association MAS –Onduleur*

On a simulé le modèle de la MAS associé à un onduleur de tension commandé par la technique MLI comme précédemment à vide puis on a appliqué un couple résistant de (15 N.m) à l'instant (t=1.5s). Si on compare ces résultats avec ceux obtenus avec le modèle seul, on constate qu'ils sont similaires mais ils présentent des oscillations autour d'une valeur moyenne, ces oscillations sont dues principalement à la présence des harmoniques dans les tensions délivrées par l'onduleur.

II.7 Commande vectorielle de la MAS

L'objectif principal de la commande vectorielle de la machine asynchrone est d'améliorer leur comportement statique et dynamique, grâce à une structure de contrôle similaire à celle d'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 **[29]**.

II.7.1 Principe de la commande par orientation de flux

La commande par orientation du flux consiste à régler le flux par une des deux composantes du courant et le couple par l'autre composante. Pour cela, il faut choisir un système d'axes (*d*, *q*) et une loi de commande assurant le découplage du couple et du flux.

Pour simplifier la commande, il est nécessaire de faire un choix judicieux de référentiel. On se place donc dans un référentiel (*d, q*) lié au champ tournant tel que l'axe *d* coïncide avec la direction désiré du flux (figure II.11) **[18]**.

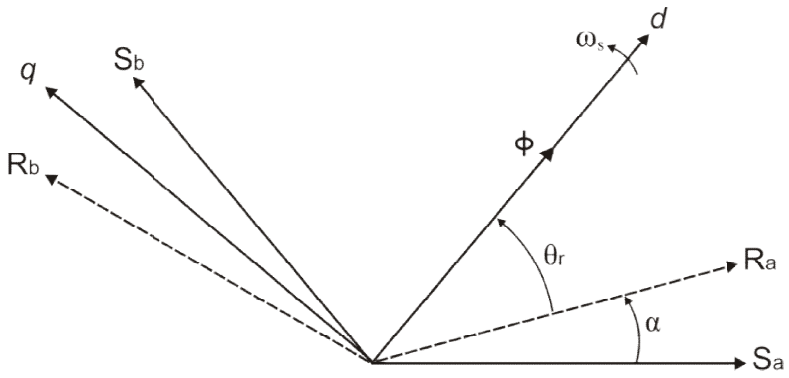


Figure (II.11) : *Principe de commande par orientation du flux*

Trois choix sont possibles pour fixer l'orientation du flux représenté dans la figure (II.11), soit **[23, 18]**:

- orienter le flux rotorique avec la condition : $\phi_{rd} = \phi_r \;\; et \;\; \phi_{rq} = 0,$ $(II.46)$
- orienter le flux statorique avec la condition : $\phi_{sd} = \phi_s \;\; et \;\; \phi_{sq} = 0,$ $(II.47)$
- orienter le flux d'entrefer avec la condition : $\phi_{ed} = \phi_e \;\; et \;\; \phi_{eq} = 0,$ $(II.48)$

La commande vectorielle à orientation du flux rotorique est la plus utilisée car elle permet d'obtenir un couple de démarrage important **[23]**, ainsi elle élimine l'influence des réactances de fuite rotorique et statorique et donnent de meilleurs résultats que les méthodes basées sur l'orientation du flux statorique ou d'entrefer **[29]**.

Pour cette raison, dans notre étude nous avons choisi l'orientation du flux rotorique. D'après la condition (II.46) l'expression du couple électromagnétique devient :

$$C_{em} = p.\frac{M}{L_r}.\phi_r.i_{sq} \qquad (II.49)$$

La troisième équation du système (I.36) devient :

$$T_r \frac{d\phi_r}{dt} + \phi_r = .M.i_{sd} \qquad (II.50)$$

Nous pouvons remarquer d'après les relations (II.49) et (II.50) que seule la composante directe i_{sd} détermine l'amplitude du flux rotorique, alors que le couple ne dépend que de la composante en quadrature i_{sq} si le flux rotorique est maintenu constant.

Ainsi, nous avons réalisé la décomposition du courant statorique en deux termes correspondants respectivement au flux et au couple, et par conséquence, nous avons obtenu une structure semblable à celle d'une machine à courant continu.

La régulation de flux peut être soit direct ou indirect **[23]** :

- contrôle direct : le flux est régulé par une contre-réaction. Il doit donc être mesuré (rarement) ou estimé. La pulsation statorique ω_s est directement évaluée à partir de la position du flux dans le repère lié au stator.
- contrôle indirect : le flux n'est ni mesuré ni reconstruit. Il est fixé en boucle ouverte. Les tensions ou les courants assurant l'orientation du flux et le découplage sont évalués à partir d'un modèle de la machine en régime transitoire.

II.7.2 Commande vectorielle indirecte par orientation du flux rotorique

Le système d'équations d'état (II.36) de la machine dans un référentiel lié au champ tournant nous permet d'exprimer (v_{sd} , v_{sq} , ϕ_r , ω_r *et* C_e) avec ($\phi_{rd} = \phi_r$) *et* ($\phi_{rq} = 0$) :

$$\begin{cases} v_{sd} = R_s\, i_{sd} + \sigma L_s \dfrac{di_{sd}}{dt} + \dfrac{M}{L_r}\dfrac{d\phi_r}{dt} - \omega_s \sigma L_s i_{sq} \\ v_{sq} = R_s\, i_{sq} + \sigma L_s \dfrac{di_{sq}}{dt} + \omega_s \dfrac{M}{L_r}\phi_r + \omega_s \sigma L_s i_{sd} \\ T_r \dfrac{d\phi_r}{dt} + \phi_r = M i_{sd} \\ \dfrac{M}{L_r}\, i_{sq} = (\omega_s - \omega)\phi_r \end{cases} \qquad (\text{II}.51)$$

Après passage par une transformation de Laplace nous obtenons le système d'équations (II.52) suivant :

$$\begin{cases} v_{sd} = (R_s + s\sigma L_s) i_{sd} + s\dfrac{M}{L_r}\phi_r - \omega_s \sigma L_s i_{sq} \\ v_{sq} = (R_s + s\sigma L_s) i_{sq} + \omega_s \dfrac{M}{L_r}\phi_r + \omega_s \sigma L_s i_{sd} \\ \phi_r = \dfrac{M}{1 + sT_r} i_{sd} \\ \omega_r = \dfrac{M}{L_r \phi_r} i_{sq} \end{cases} \qquad (II.52)$$

avec : *s* est l'opérateur de Laplace.

L'angle de *Park* θ_s est calculé à partir de la pulsation statorique, elle-même reconstituée à l'aide de la vitesse de la machine et de la pulsation rotorique ω_r, est donné par la formule suivante **[29]** :

$$\theta_s = \int (p\Omega + \frac{i_{sq}^*}{T_r\, i_{sd}^*} i_{sd})dt \qquad où : \; i_{sd}^* = \frac{\phi_r^*}{M} \qquad (II.53)$$

où (*) : indice indique la consigne de la grandeur.

Le schéma de principe de la commande vectorielle indirecte par orientation du flux rotorique du moteur asynchrone est représenté par la figure (II.12).

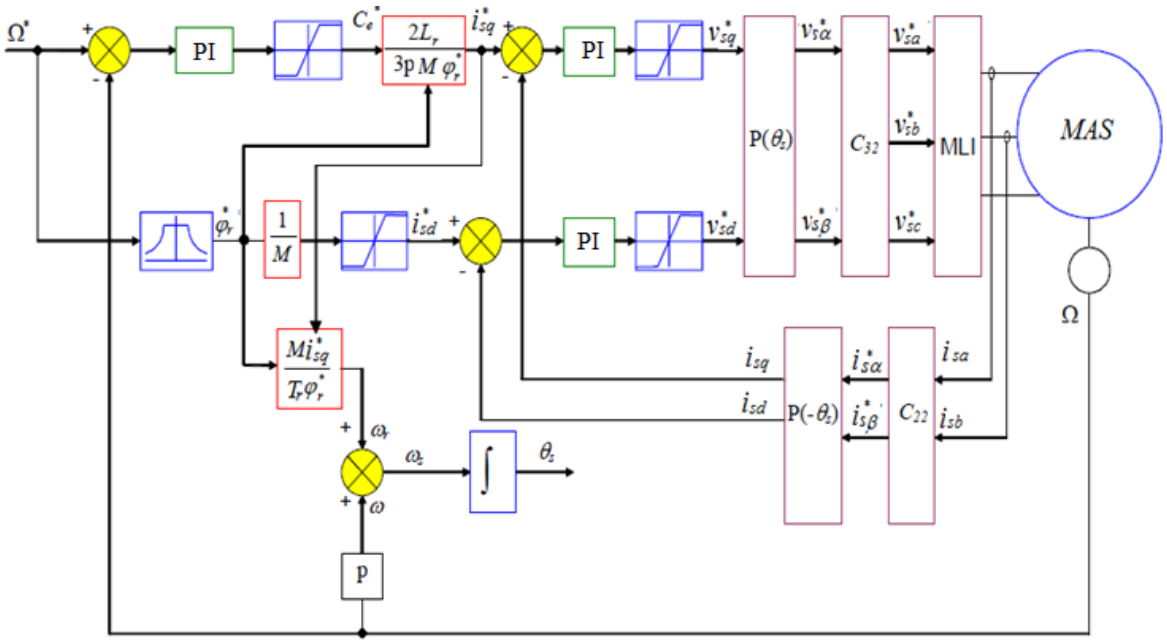


Figure (II.12) : *Régulation de vitesse par la commande vectorielle indirecte.*

II.7.3 Simulation du comportement du système en état sain

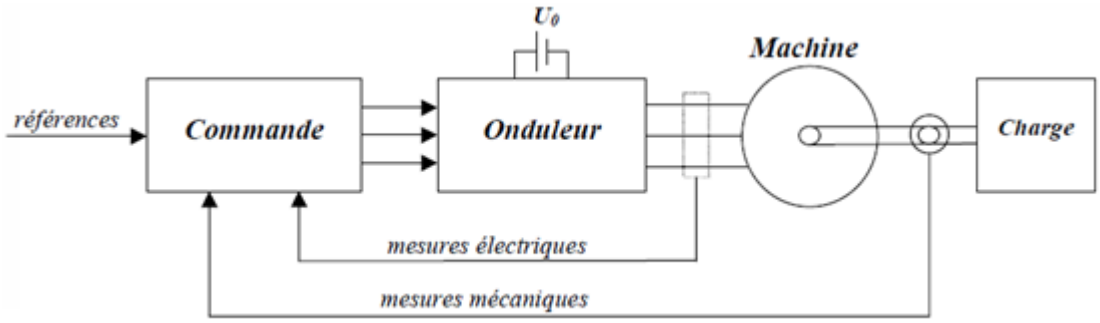


Figure (II.13) : *Ensemble Commande-Onduleur-Machine*

La figure (II.14) illustre le schéma bloc de la commande du *MAS* fait par logiciel *Simulink* sous *Matlab*.

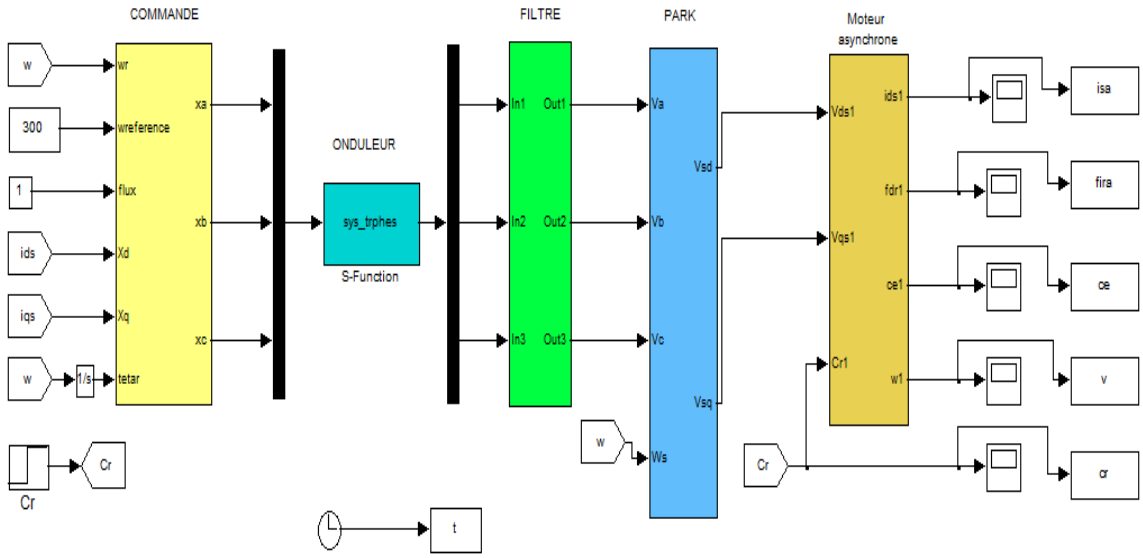
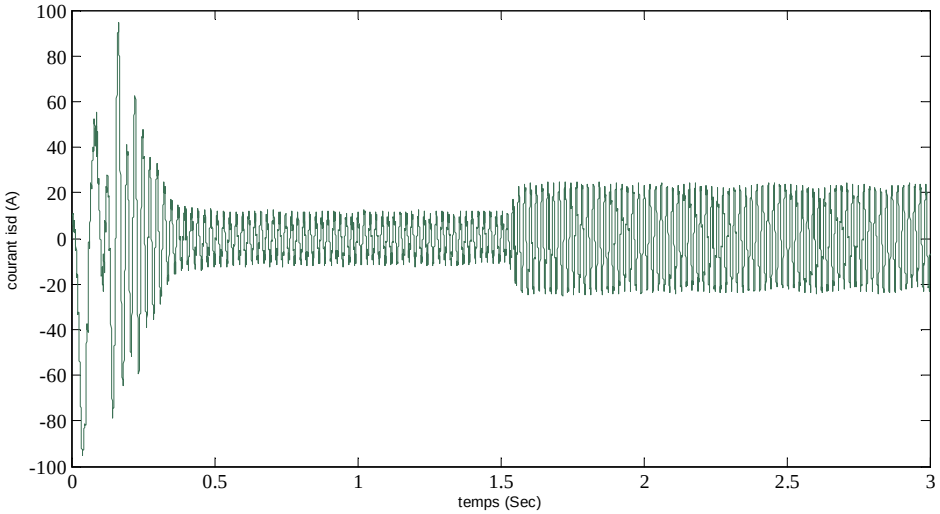
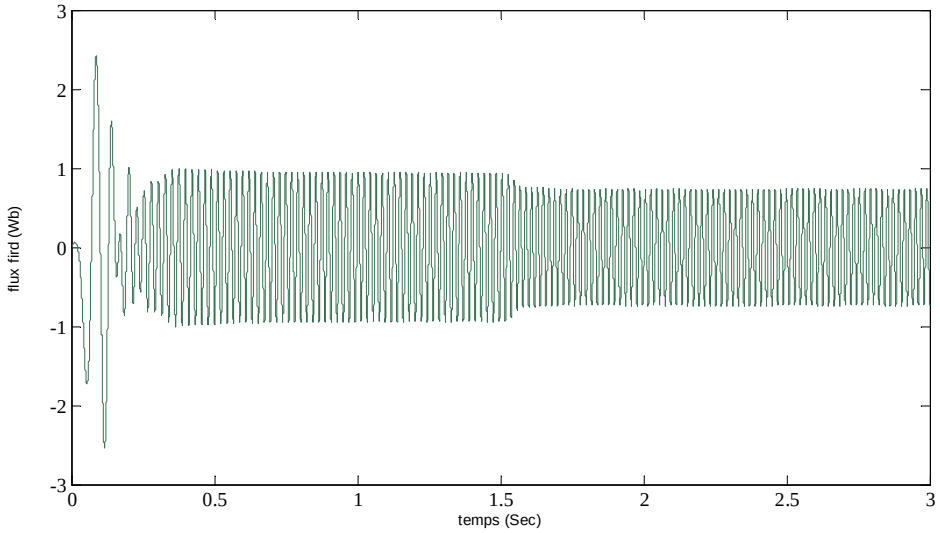


Figure (II.14) : *Bloc de simulation de la commande du MAS*

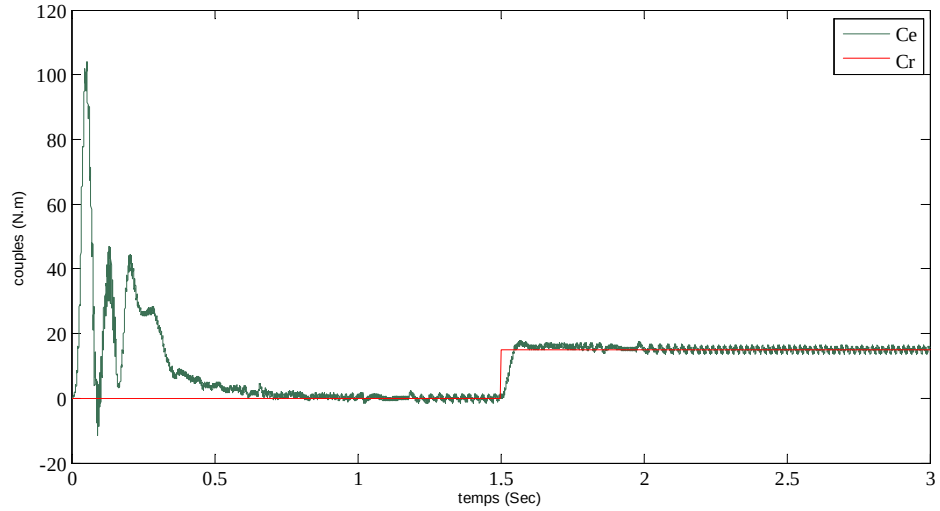
La figure (II.15) donne les résultats de simulation du système (Commande-Onduleur-Mas) en état sein (régime de fonctionnement normal), tel que on a illustré : le courant statorique, le flux rotorique, le couple électromagnétique et de la vitesse de rotation du MAS en présence de la charge.



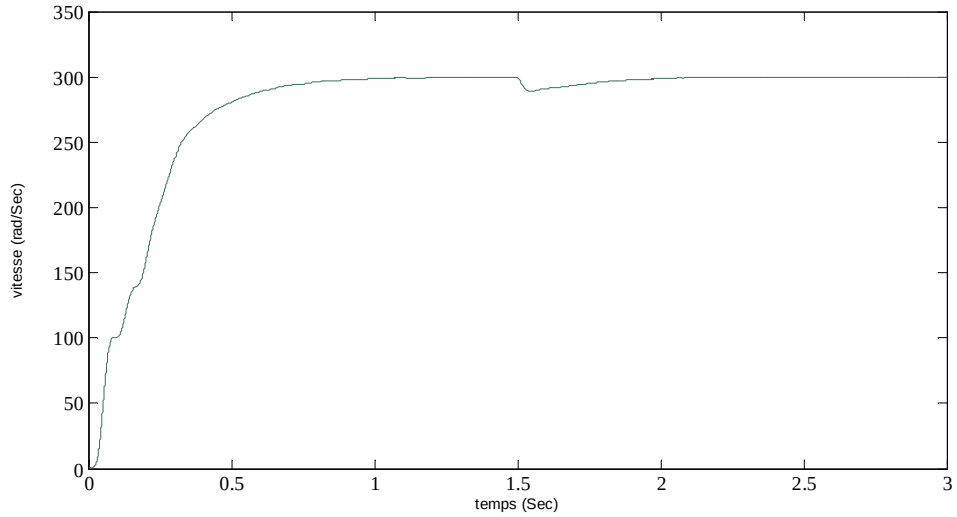
(a) *Réponse du courant statorique $i_{sd}(A)$*



(b) *Réponse du flux rotorique $\phi_{rd}(Wb)$*



(c) *Réponse du couple électromagnétique $C_{em}(N.m)$*



(d) *Réponse de la vitesse de rotation w(rad/sec)*

Figure (II.15) : *Résultat de simulation de la MAS en état sein lors d'une application d'un couple résistant (Cr =15N.m) à (t=1.5 s).*

II.7.4 Interprétation des courbes

Si en comparent ces résultats a ceux qui sont obtenus par la simulation du MAS alimenté par un onduleur triphasé sans commande vectorielle, on note la diminution du courant absorbé par le moteur lors de démarrage.

On note que la vitesse de rotation est atteint la consigne (300 rad/s) dans un taux plus rapide, l'application d'une perturbation à l'instant (t=1.5s) provoque une chute de la vitesse afin qu'il est restitué à sa valeur désiré. Ainsi que les oscillations du couple ont disparu lors du démarrage.

II.8 Simulation du comportement du système en cas de défaut

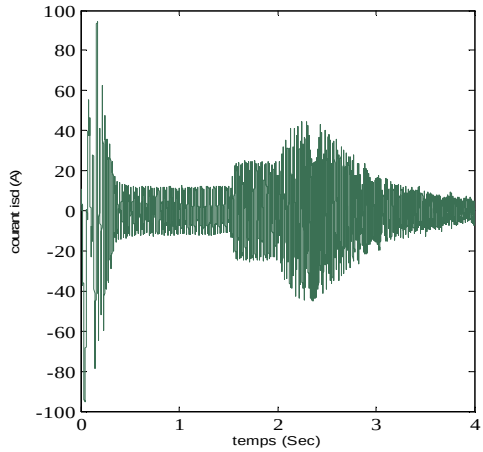
On va étudie les défauts de la tension d'alimentation, tels que la coupure des phases à savoir :

- coupure monophasé de la tension d'alimentation,
- coupure biphasé de la tension d'alimentation,

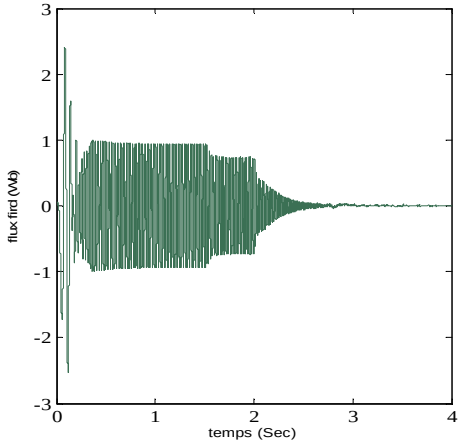
II.8.1 Simulation du comportement en cas de coupure monophasé

On va crée une coupure sur la phase une à l'instant (t=2s).

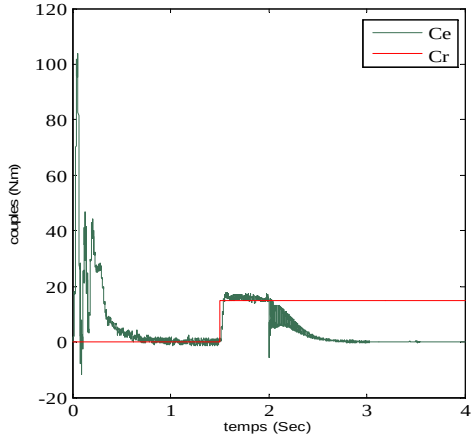
L'évolution du courant statorique, du flux rotorique, du couple électromagnétique et de la vitesse de rotation du MAS alimenté par un onduleur triphasé commandé par une vectorielle indirecte à flux rotorique orienté en présence de la charge (Cr= 15N.m) à (t=1.5s), sont illustrés à la figure (II.16).



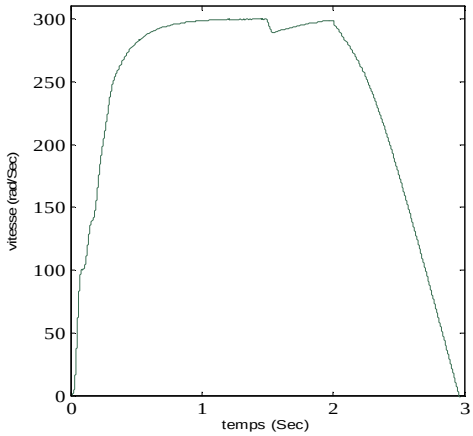
(a) *Réponse du courant statorique*



(b) *Réponse du flux rotorique*



(c) *Réponse du couple électromagnétique*



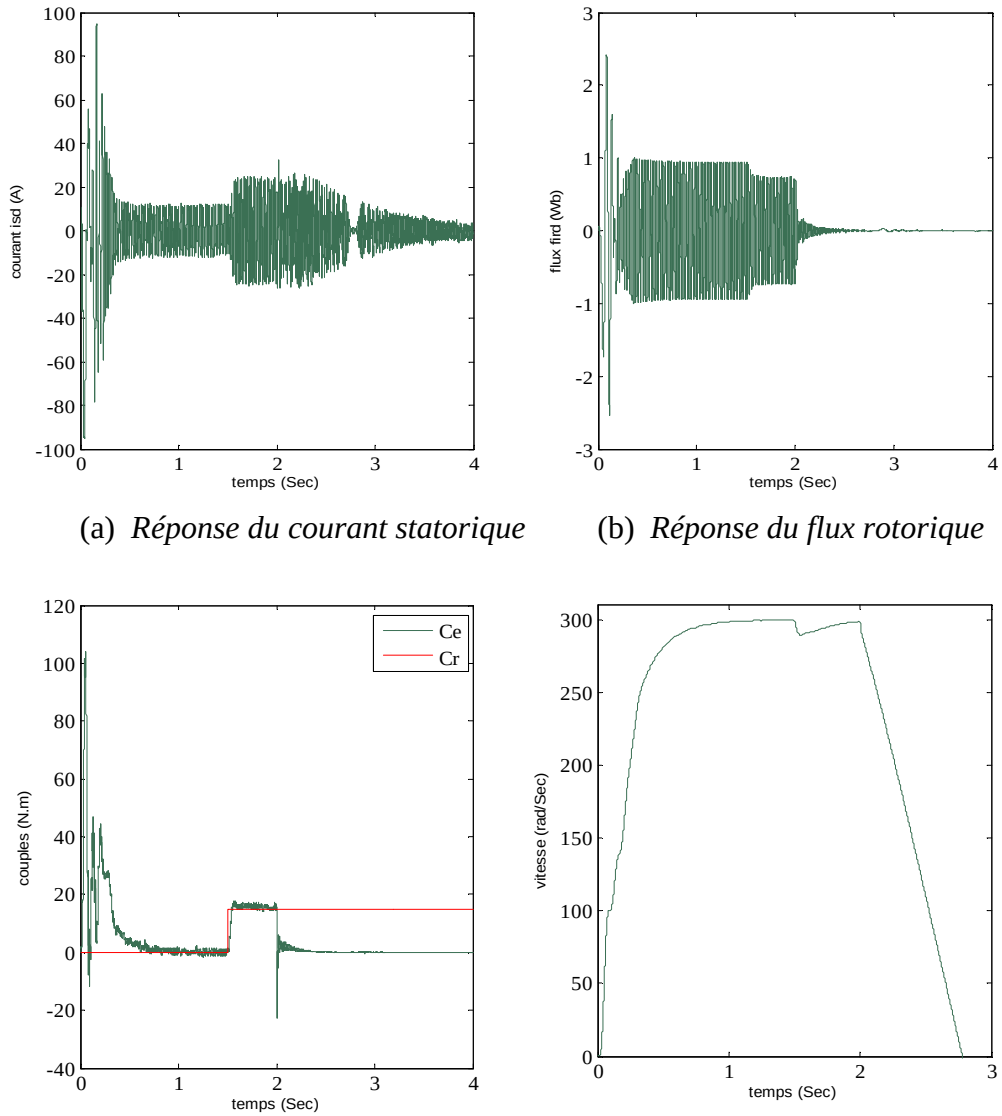
(d) *Réponse de la vitesse de rotation w(rad/sec)*

Figure (II.16) : *Résultat de simulation de la MAS en cas de coupure monophasé de la tension d'alimentation à (t=2 s), avec une charge de (Cr =15N.m) à (t=1.5 s).*

II.8.2 Simulation du comportement en cas de coupure biphasé

Dans ce cas, on va crée une coupure sur les deux phases (1) et (2) instantanément à l'instant (t=2s).

Les résultats de simulation sont représentés à la figure (II.17).



(a) *Réponse du courant statorique* (b) *Réponse du flux rotorique*

(c) *Réponse du couple électromagnétique* (d) *Réponse de la vitesse de rotation w(rad/sec)*

Figure (II.17) : *Résultat de simulation de la MAS en cas de coupure biphasé de la tension d'alimentation à (t=2 s), avec une charge de (Cr =15N.m) à (t=1.5 s).*

II.8.3 Interprétation des résultats

Les résultats de simulation sont présentés dans les Figures [II.16, II.17]. Ces dernières représentent les courbes d'évolution des différents paramètres : courant, flux, couple et vitesse.

Le caractère des graphes montre qui en phase de démarrage le moteur subit un allongement du régime transitoire. Dans les régimes anormaux,

Les grandeurs électriques sont caractérises par une variation brusque au moment d'application du défaut (2 Sec et 3 Sec). Il est a signale que les défauts de coupure de tension influent sur les comportements mécaniques du moteur, cette influence est caractérisé par une chute de la valeur de la vitesse selon le type du défaut, ainsi que la variation (oscillation) du couple électromagnétique.

II.9 Conclusion

On a présenté dans ce chapitre la modélisation de la machine asynchrone triphasée, en mettant en exigence la complexité et le non linéarité du modèle. Par la suite, et en se basant sur un ensemble d'hypothèses simplificatrices, le modèle de la MAS dans le repère de Park a été établi dans le but de linéariser le système et faciliter l'étude.

Puis, on a abordé la modélisation de la partie d'alimentation, le principe de fonctionnement et de commande de l'onduleur de tension triphasé a été présenté en donnant le principe de MLI triangulo-sinusoïdale, ainsi que l'application de la commande vectorielle.

Dans la dernière partie, on a présenté la simulation de la machine asynchrone dans les deux cas : état sein et d'autre en présence des défauts.

Après l'étude du comportement de la machine asynchrone commandée par la technique du flux orienté. En état sein et en état du défaut, nous consacrons le troisième chapitre a l'étude des circuits reconfigurable notamment les FPGA en vue de préparer la plate forme de l'implémentation des réseaux de neurones artificiels pour la détection des défauts de la machine asynchrone commandée la technique du flux oriente.

Chapitre III

Supports physiques des circuits reconfigurables

III .1 Introduction

L'évolution technologique de la microélectronique a permis de voir naître un nouveau type de composants électroniques, dits circuits programmables. 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 différents algorithmes de commande caractérisés par leurs grande complexité.

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 algorithmes de commande numérique, 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

Les supports 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 microprocesseur 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) **[8]**

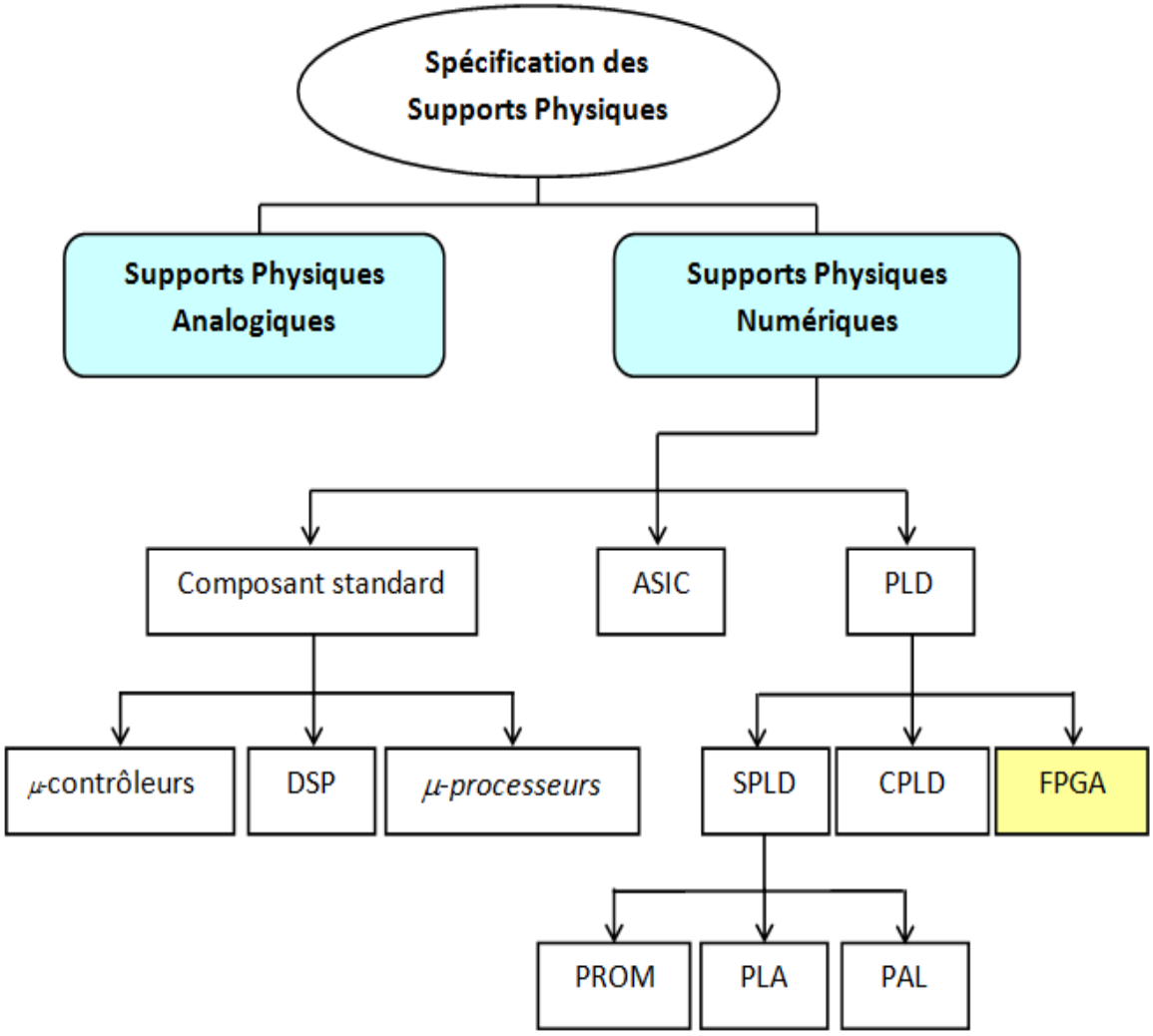


Figure (III.1): *Différentes solutions technologiques*

III .2.1 Support physique analogique [8]

Avant que le microprocesseur ne soit utilisé largement dans l'implémentation des commandes électriques, les supports analogiques ont dominés pendant longtemps cette implémentation. L'ensemble de 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 effet, le vieillissement, les bruits et les contraintes thermiques inhérentes à cette solution d'implantation sont à l'origine de variations comportementales assez néfastes des composants analogiques qui la définissent. Cette contrainte physique implique un réajustement régulier des paramètres du système électronique. 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.

Finalement, si la conception théorique d'une commande analogique est simple, sa réalisation peut s'avérer délicate compte tenu des contraintes liées à la tolérance des composants. Enfin, les problèmes, décrits ci-dessus, et le coût de la maintenance pour un système totalement câblé, deviennent vite un handicap. De plus, la complexité croissante des algorithmes de commande de systèmes électriques impose des calculs de plus en plus difficiles à gérer par une technologie purement analogique. La mise en œuvre de ces algorithmes entraîne progressivement l'abandon de cette voie d'implantation.

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 [8]

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). Nous distinguons :

- *Les composants standards* : cette famille englobe les toutes anciennes technologie 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 logicielles spécifique à la technologie du circuit à programmer. Nous distinguons trois types principaux de circuits programmables, à savoir : les SPLD (Simple Programmable Logic Device), les CPLD et les FPGA.

III .3 Les composants Standards (circuits à fonctionnement programmable)

III .3.1 Microprocesseur [8]

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 exécuté de manière séquentielle.

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 se qui rend le temps d'exécution d'un calcul très considérable, surtout lors du traitement des algorithmes complexes tels que ceux de la commande électrique.

Une architecture de microprocesseur peut alors être plus adaptée pour une tache 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 [8]

Les DSP (Digital Signal Processing) 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 **[8]**.

La représentation de ces microprocesseurs est liée au fait qu'ils sont actuellement les architectures les plus prisées pour l'implantation de commandes de systèmes électriques. 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, 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 .3.3 Les ASICs (Application Specific Integrated Circuits)

Ces circuits présentent d'énormes atouts, à savoir :

- *une consommation faible :* dû à une conception ciblée de circuit.
- *une perspective de mixte d'intégration* : l'association des technologies numériques et analogiques (où l'intégration de la commande ainsi que les fonctionnalités de l'interface et des convertisseurs peut être envisagée sur une même puce de silicium),
- *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 **[8]**.

III .3.4 PLD (Les Circuits à structure Programmable)

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 ; les SPLDs, les CPLDs et les FPGA (voir figure III.1) **[8]**.

III .3.4.1 SPLD (Simple Programmable Logic Device)

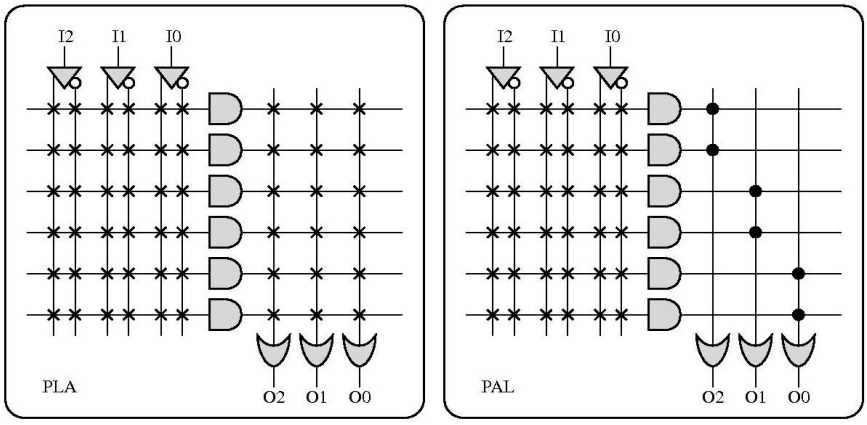
Les SPLDs, sont composés d'une grille de portes ET et d'une grille de portes OU, les deux étant reliées. Les entrées du système peuvent être connectées aux portes ET, et le résultat des portes OU correspond à la sortie du système (figure III.3). Dans ces circuits, les connexions sont préexistantes, les différentes lignes étant reliées par des fusibles, des transistors EPROM, ou des transistors EEPROM. En brûlant certains de ces fusibles, ou en programmant les transistors, il est alors possible de réaliser différentes fonctions logiques **[8]**.

III .3.4.1.1 PLA

Les PLAs (Programmable Logic Array) apparurent aux environs de l'année 1975 dans le but de pallier les limitations des PROMs. En effet, dans un PLA (figure III.2.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 même 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 **[8]**.

III .3.4.1.2 PAL

Vers la fin des années 1970, les PALs (Programmable Array Logic) furent introduites afin de contrer le problème de vitesse de propagation des PLAs. Dans un PAL (figure III.2.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 **[8]**.



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

Figure (III.2) : *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.

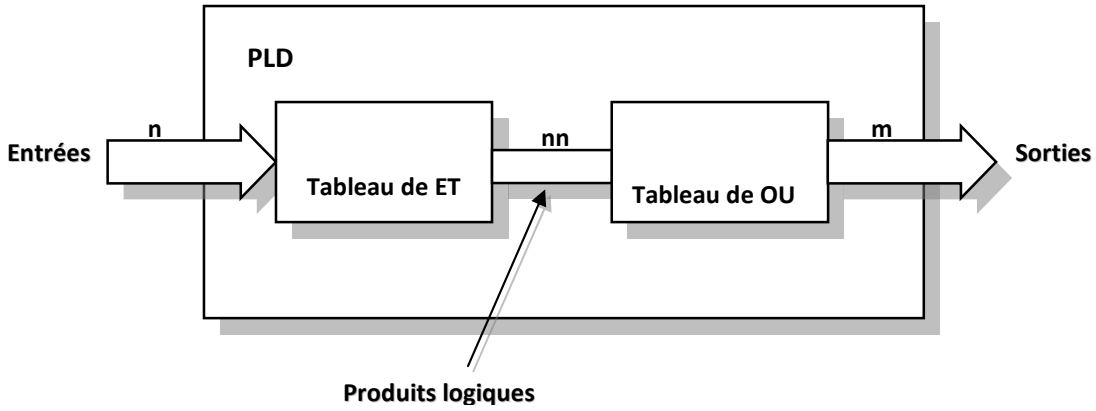


Figure (III.3): *Architecture d'un SPLD*

III .3.4.2 CPLD [8]

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.3.4.3 Les FPGA

III.3.4.3.1 Introduction

Nous en arrivons au point central de ce chapitre, à savoir les circuits de type FPGA (*Field Programmable Gate Array*).

Au début des années 80, les développeurs disposaient des circuits de type PLD, facilement configurables, mais ne pouvant contenir des designs trop complexes. Les ASICs, quant à eux, supportaient des systèmes de grande complexité, mais n'avaient pas les propriétés de configuration des PLDs. Il manquait donc un type de circuits permettant la réalisation de systèmes complexes, tout en offrant une reconfiguration rapide et peu onéreuse. C'est pourquoi en 1984, Ross Freeman, Bernie Vonderschmitt, et Jim Barnett fondent la compagnie Xilinx. En 1985, ils introduisent sur le marché le premier FPGA, le XC2064, et offrent ainsi une alternative aux précédentes approches **[42]**.

III.3.4.3.2 Qu'est ce que un FPGA ?

Le circuit FPGA est un dispositif logique programmable, qu'il comporte de millier de portes de logique, et qui peut implémenter des circuits logiques en programmant la fonction requit **[32]**. La combinaison entre ces portes forme un bloc logique configurable CLB (*Configurable Logic Block*). Des interconnexions de portes utilisant *software* sont définies par SRAM ou ROM. Ceci fournit la flexibilité de modifier le circuit conçu sans changer la pièce (de matériel), facile et la modification rapide de circuit **[40]**.

III.3.4.3.3 Architectures reconfigurables [27]

POUR QUOI LES ARCHETECTURES RECONFIGURABLES?

Depuis quelques années, les progrès technologiques réalisés ont permis l'émergence d'un nouveau type d'architectures : les architectures reconfigurables. L'idée de base de ces architectures est d'offrir aux concepteurs la flexibilité d'une architecture programmable et les performances temporelles d'un circuit dédié. Ces architectures ont donc été très largement utilisées dans des domaines très volatiles, où les normes font défauts et où les performances d'un microprocesseur seul ne sont pas suffisantes. Elles offraient cependant de très faibles capacités d'intégration et n'offraient pas des performances de calcul très élevées.

L'étude sémantique du mot reconfigurable en donne une définition plus précise. En effet, **configuration**, signifie à l'origine *façonner à la ressemblance de* et a pris le sens de *disposition relative d'éléments*. Une *architecture* est constituée d'une *disposition relative d'éléments* organisés selon un certain schéma. La reconfiguration, en permettant un choix des éléments d'une part, et de leur disposition relative d'autre part, autorise une variabilité des schémas et donc des architectures. Cette définition recouvre alors un large ensemble d'architectures dont les FPGAs font partis.

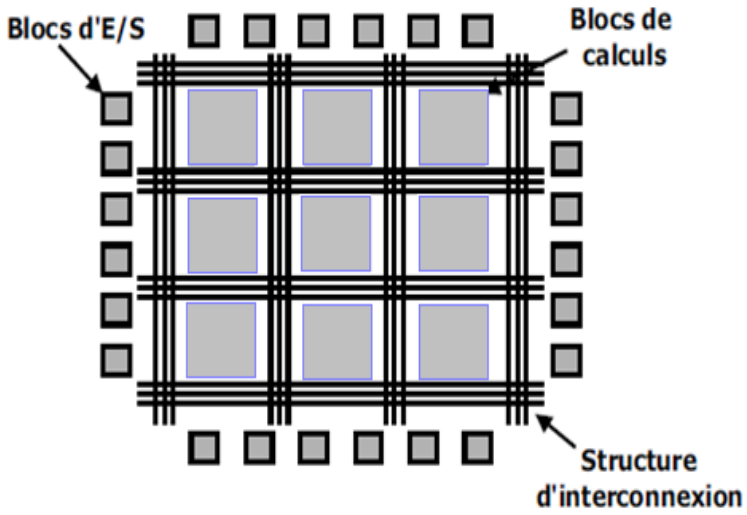


Figure (III.4) : *Architecture reconfigurable générique.*

Les architectures sont en général organisées selon un tableau régulier d'éléments de calcul (figure III.6). Ces ressources de calcul sont interconnectées par un réseau plus ou moins flexible et plus ou moins performant. Les connexions sont réalisées par des matrices

d'interconnexions construites autour de portes de transmission permettant de créer des connexions entre les segments arrivant sur la matrice.

III.3.4.3.4 Architecture des FPGAs

Il y a une large variété d'architectures pour FPGAs de différents fournisseurs comprenant Xilinx, Altera, Lattice,...etc., bien que la structure exacte de ces FPGAs varie, tout les FPGAs consistent de trois composants fondamentaux, comme vu en figure (III.5) [41] :

1. blocs logiques qui sont capables d'implémenter des fonctions logiques multiples,
2. blocs I/O ou E/S pour la communication avec le monde extérieur,
3. fixe, aussi bien que programmable, ressources de routinage employées pour réaliser toutes les interconnexions requis (exigé) entre les blocs.

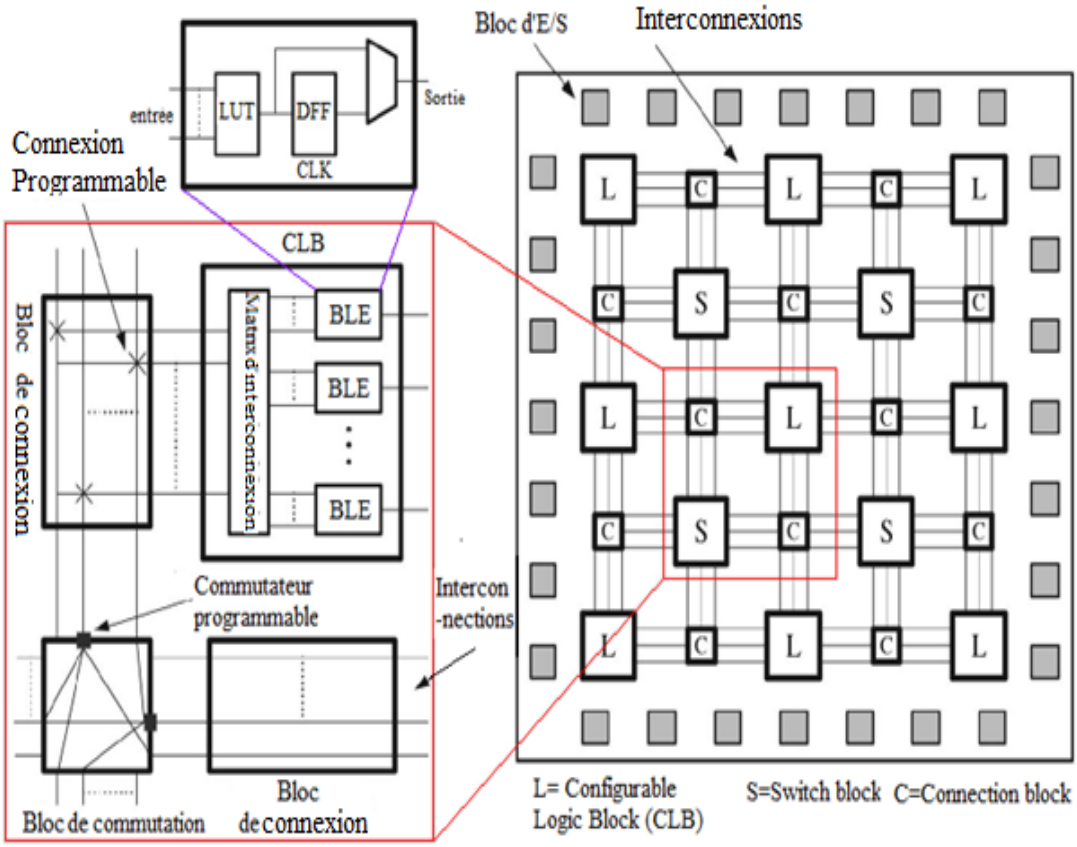


Figure (III.5) : *Architecture de FPGA (Island-Style)*

III.3.4.3.4.1 Blocs logiques

Beaucoup de fournisseurs emploient une architecture de FPGA dite *Island-Style*. Les blocs de logique dans cette architecture sont mentionnés comme des CLBs et sont arrangés comme rangée symétrique. Les voies de routinage ont la géométrie de *Manhattan* ; c'est-à-dire, elles sont horizontales ou verticales **[40]**.

Un CLB est un bloc de logique combinatoire **[38],** dans la plupart de FPGAs commercial se compose d'un ou plusieurs BLE *(Basic Logic Element)* **[40]**, chaque BLE se compose habituellement (suivant les indications de la figure III.5) de **[32]** :

- une LUT (*Look Up Table*) pour implémenter la fonctionnalité logique,
- une Bascule pour implémenter le classement logique,
- un multiplexeur pour créer la connectivité logique.
- La LUT est un élément qui dispose de K-entrées, il existe donc 2^K combinaisons différentes de ces entrées. L'idée consiste à mémoriser la sortie correspondant à chaque combinaison d'entrée dans une petite table de 2^K bits, la LUT devient ainsi un petit bloc générateur de fonctions **[38]**.

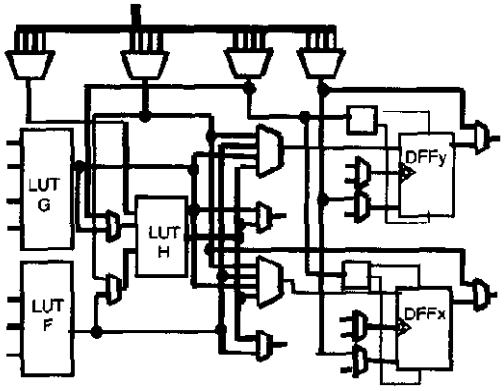
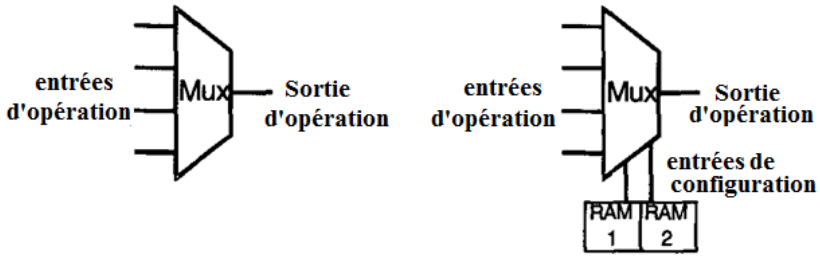


Figure (III.6) : *Cellule Logic de la famille XILINX 4000*

Comme exemple, la figure(III.6) tirer de **[32]** donne une cellule logique de la famille de XILINX 4000 avec 3 LUTs, 2 bascules et environ 12 multiplexeurs pour le raccordement entre les entrées de cellules, les bascules et les sorties de cellules.

Les cellules de RAM ne sont pas habituellement représentées dans la description de FPGA en tant que celle donnée sur la figure (III.6), mais elles sont évidemment présentes et relié aux « entrés configurables » du multiplexeur configurable et du LUT **[32]**.



a) Représentation de Mux du FPGA *b) Représentation complète*

Figure (III.7) : *Représentation d'un multiplexeur configurable*

Comme exemple la figure (III.7.a) donne la représentation habituelle d'un multiplexeur configurable où seulement les entrées d'opération apparaissent ; tandis que la figure (III.7.b) donne la représentation complète avec les entrées d'opération et modules d'entrées de configuration peut être reliée à n'importe quel signal interne actuel sur le routinage de FPGA tandis que les entrées de configuration sont seulement reliées aux cellules de RAM de configuration **[32]**. .

III.3.4.3.4.2 Blocs I/O

Ces blocs d'entrée/sortie permettent l'interface entre les broches du composant FPGA et la logique interne développée à l'intérieur du composant. Ils sont présents sur toute la périphérie du circuit FPGA. Chaque bloc d'E/S contrôle une broche du composant et il peut être défini en entrée, en sortie, en signal bidirectionnel **[38]**.

III.3.4.3.4.3 Réseaux d'interconnexion

Au sein d'un composant programmable, le réseau d'interconnexion assure la distribution des données entre les différentes unités fonctionnelles. Sa topologie a un impact de tout premier ordre sur les performances, la consommation et la flexibilité du système **[41]**.

Les FPGAs utilisent généralement un réseau d'interconnexion de type *mesh* (maillage) [28] très complexes, où chaque bit doit pouvoir être orienté individuellement. Pour les composants Xilinx les interconnexions transitent à travers différents types de ligne suivant la distance, le temps de propagation et le nombre de portes connectées. Des lignes prédéfinies, dites directes, connectent efficacement les proches voisins. Des lignes, dites

générales, parcourent l'ensemble du circuit et sont organisées suivant un motif de type *mesh* généralisé **[41]**.

La structure détaillée des voies de routinage se compose de trois composants **[40]** :

- blocs de connexion,
- switchs bloc,
- canaux de routinage.

Un bloc de connexion est employé pour relier un CLB aux canaux de routinage par l'intermédiaire des interconnections programmables. Les pins de chaque passage de CLB ininterrompu par le bloc d'interconnexion et ont l'option de la « fusion » à quelques segments de canal.

Le switch bloc est une matrice de commutation qui est employée pour relier les fils dans un segment de canal à d'autres fils.

Cette structure flexible de routinage permet à chaque CLB d'avoir des raccordements avec toute autre pin de CLB ou I/O, selon le nombre de voies dans les canaux de routinage.

III.3.4.3.5 Technologies de programmation [42]

Nous avons déjà présenté les différentes technologies de programmation. Les FPGAs actuels ne tirent partie que des technologies, EEPROM/FLASH et SRAM, le tableau (III.1) résumant les caractéristiques de ces approches.

Une nouvelle technologie est peut-être en train de voir le jour. Le 12 juin 2002, « Fujitsu » annonçait la réalisation d'un nouveau FPGA utilisant une technologie RAM ferroélectrique (FRAM). L'avantage de cette nouvelle approche réside dans le fait que la mémoire ferroélectrique est non-volatile, et que donc le circuit est directement opérationnel au démarrage. De plus le temps de configuration du FPGA est vingt fois plus rapide qu'une technologie EEPROM ou FLASH, et ne nécessite qu'un courant de 3,3V. L'avenir nous dira si cette technologie saura s'imposer face à celles existantes.

Tableau (III.1) : *Comparaison des caractéristiques des différentes technologies de programmation appliquées aux FPGAs.*

Caractéristique	SRAM	EEPROM/FLASH
Technologie	état de l'art	une ou deux générations de retard
Reprogrammable	Oui	Oui
Vitesse de Programmation	Rapide	3× plus lent que SRAM
Volatile	Oui	Non
Prototypage	Excellent	Bon
Sécurité IP	Non	Oui
Taille des cellules	Grande (6 transistors)	Moyen (2transistors)
Consommation	Moyenne	Moyenne
Sensible aux radiations	Oui	Plutôt oui

III.3.4.3.6 Niveaux de reconfiguration [41].

Trois niveaux de reconfiguration sont-ils proposés : système, fonctionnel et logique. Ils reflètent directement les circuits FPGA.

a) Niveau système

Typiquement, ce niveau de reconfiguration est celui des processeurs programmables. Il concerne essentiellement la fonctionnalité des unités de calcul et l'orientation du chemin de données (de/vers la mémoire et/ou les registres).

b) Niveau fonctionnel

Les architectures de ce niveau peuvent être illustrées par la figure (III.8). L'architecture générique consiste en un ensemble d'opérateurs arithmétiques dédiés ou programmables, reliés par un réseau d'interconnexions, qui communiquent vers l'extérieur au moyen de blocs d'entrées/sorties configurables. En fonction de l'aspect programmable

ou dédié des ressources de calcul, les modèles de programmation peuvent être très diversifiés.

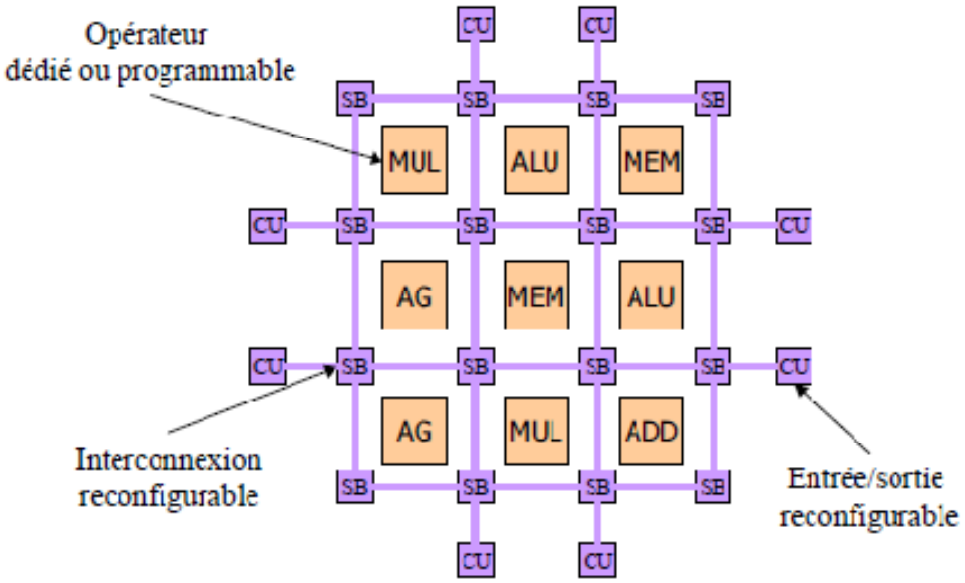


Figure (III.8) : *Architecture générique des composants reconfigurables au niveau fonctionnel.*

c) Niveau logique

Les architectures qui se situent à ce niveau sont qualifiées de grain fin en raison de la faible largeur de leurs chemins de données. La programmation opère au niveau logique sur des primitives de calcul manipulant des données binaires et sur leurs interconnexions. Étant donnée la quantité de ressources nécessaires à la définition de fonctions évoluées, ces réseaux sont très complexes et nécessitent un très grand nombre de données pour spécifier à la fois le traitement réalisé sur chacune des primitives de calcul et l'ensemble de leurs interconnexions.

Les circuits FPGA au détriment des circuits de type PAL ou CPLD qui ne disposent pas d'une complexité suffisante pour implémenter des systèmes complets.

L'architecture générique de ces composants est représentée sur un exemple de la figure (III.9). Elle consiste en une matrice d'éléments configurables, reliés par un réseau d'interconnexions, qui communiquent vers l'extérieur au moyen de blocs d'entrées/sorties configurables. La taille, le type et le nombre de cellules, de même que leurs interconnexions varient suivant les familles et les constructeurs.

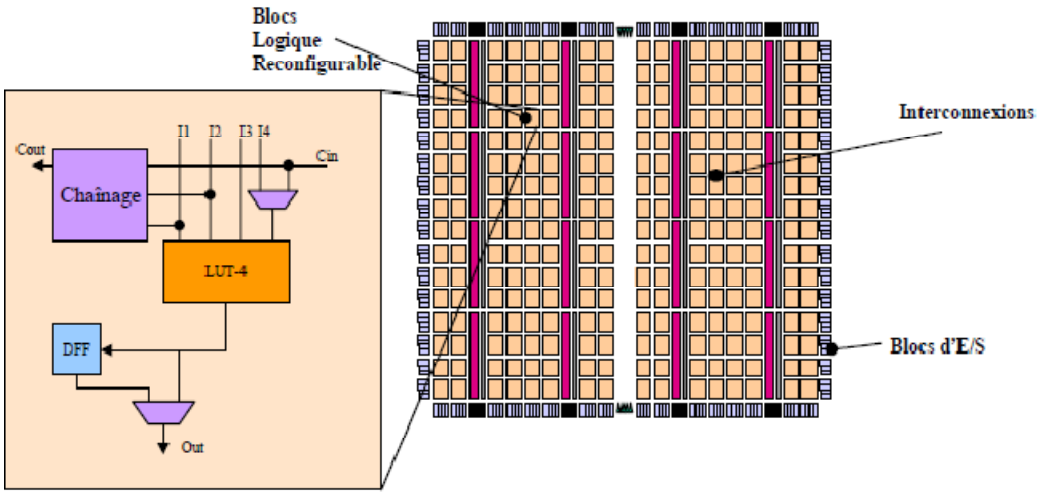


Figure (III.9) : *Architecture générique des composants reconfigurables au niveau logique.*

III.3.4.3.7 Les ressources de calculs

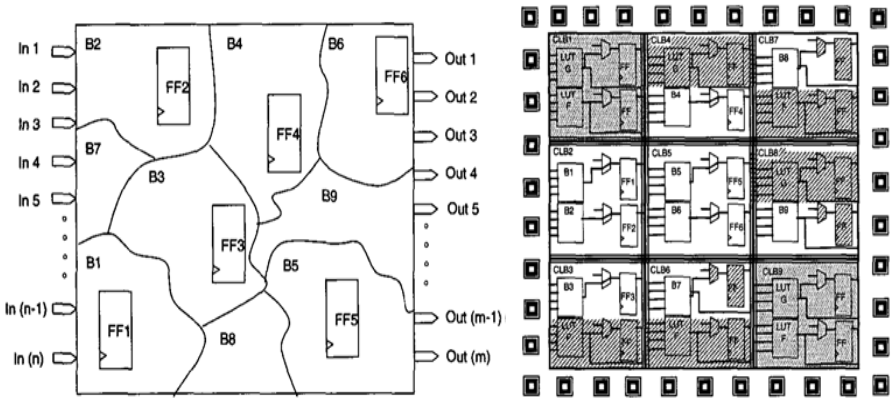
La cellule de base est un élément mémoire associé à un module combinatoire. Construire un système consiste à associer ces cellules par le biais d'un réseau d'interconnexions programmable.

La logique combinatoire est implantée sous forme de LUTs, ce module combinatoire est associé à un élément de mémorisation au sein de la primitive de calcul : le bloc CLB pour la famille Xilinx. Ces briques de base intègrent également des fonctionnalités intéressantes telles que le chaînage de retenue pour accélérer l'opération d'addition ou de petits modules mémoire (figure III.9). Les dernières générations de FPGA, le VIRTEX pour Xilinx, offrent de très fortes capacités d'intégration (4,000,000 portes équivalentes) et une fréquence de fonctionnement élevée.

III.3.4.3.8 Implémentation sur FPGA [32]

Pour employer un FPGA il faut le configurer et puis l'actionner avec sa fonctionnalité d'implémentation. N'importe quel circuit peut être tracé dans un FPGA dans la mesure où il contient assez de ressources dans la limite des portes logiques équivalentes et de routinage. Il est évident que la conception de grands circuits exige une grande matrice de FPGA qui est plus chère qu'une petite matrice.

Évidemment, la taille de la matrice, le FPGA visé, est habituellement définie en fonction de la taille du circuit à tracer. Et ainsi, quand le circuit est tracé dans la matrice visée, seulement quelque CLBs restent non utilisé par l'implémentation.



a) le circuit *b) l'implémentation de la FPGA*

Figure (III.10) : *Le circuit et leur implémentation de la FPGA*

La figure (III.10.a) donne une représentation conceptuelle du circuit défini pour l'utilisateur avec des blocs (Bi) et des bascules (FFi). Noter que le circuit est séquentiel et quelques blocs incluent des bascules. Dans la figure (III.10.b) le circuit est implémenté dans une FPGA représenté par une matrice didactique (3×3).

Nous observons que :

- 2 CLBs restent non utilisés par l'implémentation (CLB1 et CLB9),
- 3 CLBs sont implémenté des blocs combinatoires et n'emploient pas des bascules (CLB6, CLB7, CLB8),
- 2 CLBs sont implémenté des blocs séquentiels utilisant seulement 1 bascule (CLB3, CLB4).

III.4 Xilinx [42]

A l'heure actuelle, Xilinx, le premier fabricant de FPGAs, propose principalement deux familles, Virtex et Spartan, toutes deux de type SRAM, basées sur une architecture LUT. La différence entre les deux familles est minime, et tient principalement du nombre d'éléments proposés ainsi que du type de process utilisé, les Spartan étant positionnées bas coût en comparaison des Virtex. L'élément de base, le CLB , est composé de deux Slices, eux-mêmes comprenant deux 4-LUT et deux bascules. Les versions Virtex-II, Virtex-4 et

Spartan-3 diffèrent toutefois, le CLB y contenant quatre Slices. Les deux familles contiennent des blocs de RAM, pouvant être utilisés en single ou dual port.

Alors que les séries Spartan-3, Virtex-II et Virtex-4 sont presque identiques en terme d'architecture, la série Virtex-II Pro introduit un ou deux processeurs de type PowerPC. Le plus imposant de la série Spartan-3 propose 5M de portes équivalentes, dont 104 multiplicateur de 18_18 bits et 1'872K bits de RAM. En comparaison, les plus gros Virtex sont le XC4VLX200, de la famille Virtex-4LX et le XC4VFX140, un Virtex-FX. Le premier contient 178'176 éléments logiques, un élément logique étant une LUT à 4 entrées et une bascule. A ceci s'ajoutent 6'048 Kbits de RAM configurables, ainsi que 96 multiplicateurs 18x18 bits, pour un total de 15M de portes équivalentes. Le deuxième, le XC4VFX140, est composé de 126'336 éléments logiques, de 9'936 Kbits de RAM, 192 multiplicateurs, et 2 processeurs de type PowerPC.0

Nous pouvons finalement noter que Xilinx, à l'instar d'Altera, propose une solution appelée EasyPath, permettant la réalisation en ASIC d'un design implémenté sur un Virtex-II, Virtex-II Pro ou un Spartan-3.

III.5 Conclusion

Nous avons présenté des circuits reprogrammables, prêtant une attention particulière aux FPGAs, qui sont les circuits reconfigurables les plus flexibles et offrant le plus de fonctionnalités. Ils sont les meilleurs candidats pour la réalisation rapide de prototype, de par leur capacité à être reconfigurés un très grand nombre de fois.

Dans le chapitre suivant nous verrons qu'ils ont été utilisés pour l'implémentation des réseaux de neurones artificiels en utilisant la Co-simulation puis la détection des défauts de la machine asynchrone.

.

Chapitre IV

Co-simulation et RNA sur FPGA

IV.1 Introduction

L'objectif de ce chapitre est l'implantation de réseau de neurone artificiel sur un 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.

En effet, nous programmons ce RNA sur le System Generator. Ce dernier permet de générer le code VHDL. Ce code est vérifié et implémenté sur un circuit FPGA de type Virtex4 par le logiciel ISE fundation de Xilinx (voir Figure (IV.1)).

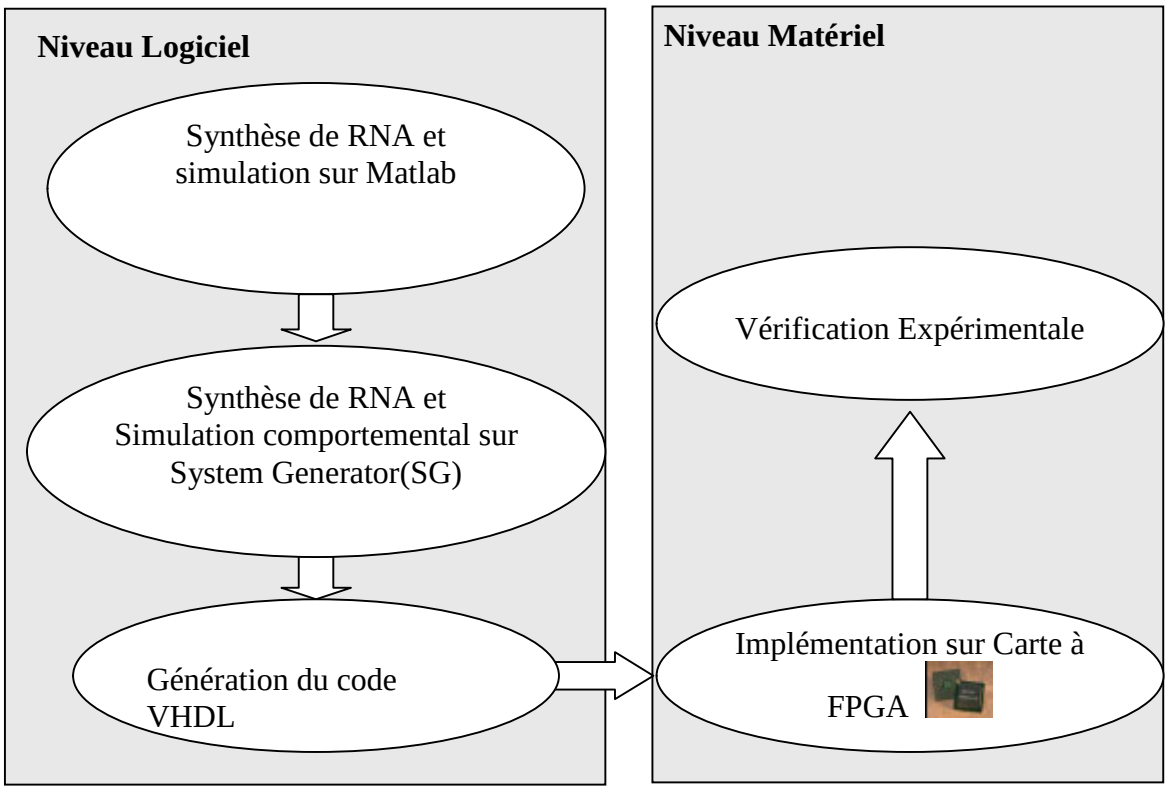


Figure (IV.1): *Phases d'implémentation sur FPGA*

En suite, pour s'approcher au cas réel nous implémentons ce réseau de neurone sur le même circuit en utilisant la Co-simulation afin de valider l'efficacité des circuits FPGA en terme d'implémentation et de simulation (voire figure IV.2).

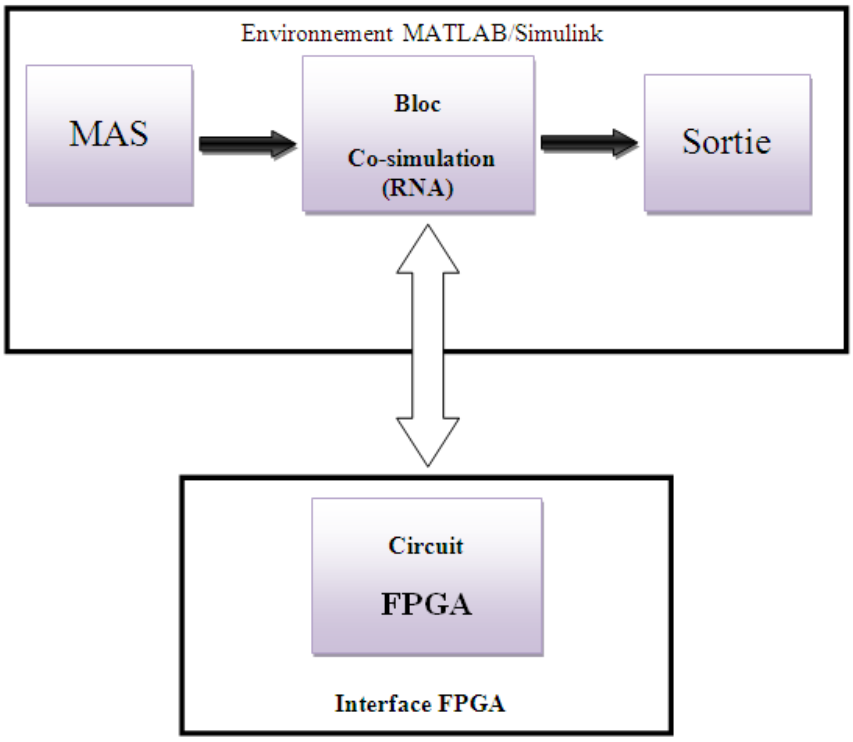


Figure (IV.2): *Implémentation et Co-simulation sur FPGA*

IV.2 Synthèse de RNA et simulation sur MATLAB

IV.2.1 Description du réseau de neurones utilisé

Le réseau de neurones proposé est un réseau multicouche de (3-5-3) dont l'architecture adoptée est illustrée sur la figure (IV.3).

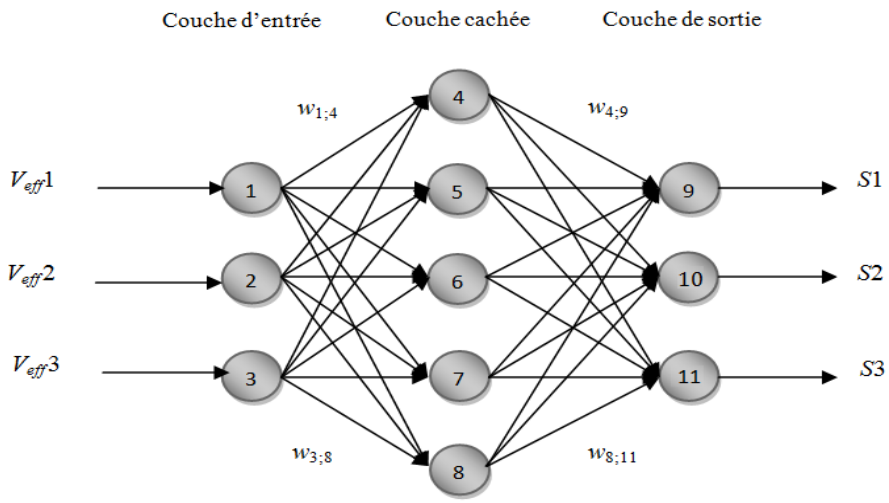


Figure (IV.3) : *L'architecture du réseau neuronal proposé.*

Chaque neurone est connecté à l'ensemble des neurones de la couche suivante par des connexions dont les poids sont des nombres réels quelconques. On note $w_{x,y}$ le poids de la connexion entre les neurones *x* et *y*.

IV.2.2 Étude de réseau de neurone utilisé

Il y avait quatre études principales :

- construction du bloc RNA,
- acquisition des données (base d'apprentissage),
- test de réseau.

IV.2.2.1 Construction du bloc de réseau de neurones

La figure (IV.3) montre bien que notre réseau est constitué de trois couches à savoir :

- Une couche d'entrée composée de trois neurones, dont le rôle est de transmettre les valeurs des entrées qui correspondent aux variables ($V_{eff\ 1}$, $V_{eff\ 2}$ et $V_{eff\ 3}$) vers la couche suivante appelée *couche cachée*.
- Une couche cachée dotée de cinq neurones avec des fonctions d'activations choisies de type sigmoïde.
- une couche de sortie, cette couche est composée de trois neurones, la sortie de chaque neurone soit 0 ou 1.

Les valeurs efficaces des tensions V_1, V_2 et V_3 sont calculés par des schémas blocs sous Matlab/Simulink (voir Annexe C).

IV.2.2.2 Acquisition des données (base d'apprentissage)

Avant la construction du système du bloc RNA On doit tout d'abord accéder à la d'apprentissage. Celle-ci peut se mettre sous forme de tableau (matrice). Ce dernier est constitué de vecteurs (qui représentent la couche d'entrée du RNA), où chaque vecteur est constitué de 3 paramètres.

On construire (fonctionnements normaux et anormaux) une base de données très riches, qui possèdent beaucoup d'informations sur les différents types de défauts. Pour cette phase on a réalisé les taches suivantes:

- la machine a été simulée en régime normal (état sein),
- la machine a été simulée en régime anormale (en présence de défauts : coupure monophasé et coupure biphasé),
- puis nous a vous prenant les valeurs efficaces dans chaque cas y compris l'état sein.

En fait, pour passer à l'étape de classification nous disposons pour chacun des paramètres, 3 types de défauts y compris l'état sein (le tableau IV.1).

Tableau IV.1. *Classification des différents défauts*

Catégorie	**Type de défaut**	**Symbole**	**Code**		
1	État sain	ES	0	0	0
2	Coupure monophasé 1	CM1	1	0	0
3	Coupure monophasé 2	CM2	0	1	0
4	Coupure monophasé 3	CM3	0	0	1
5	Coupure biphasé 1	CB1	1	1	0
7	Coupure biphasé 2	CB2	0	1	1
6	Coupure biphasé 3	CB3	1	0	1

IV.2.2.2.1 Résultats d'essais de réseau

On a effectué un apprentissage automatique à l'aide du logiciel Matlab jusqu'à où on obtenu une erreur quadratique plus petite de « 7.69704e-014 » (voir figures IV.4) on a obtenu la plus petite erreur après 56 itérations.

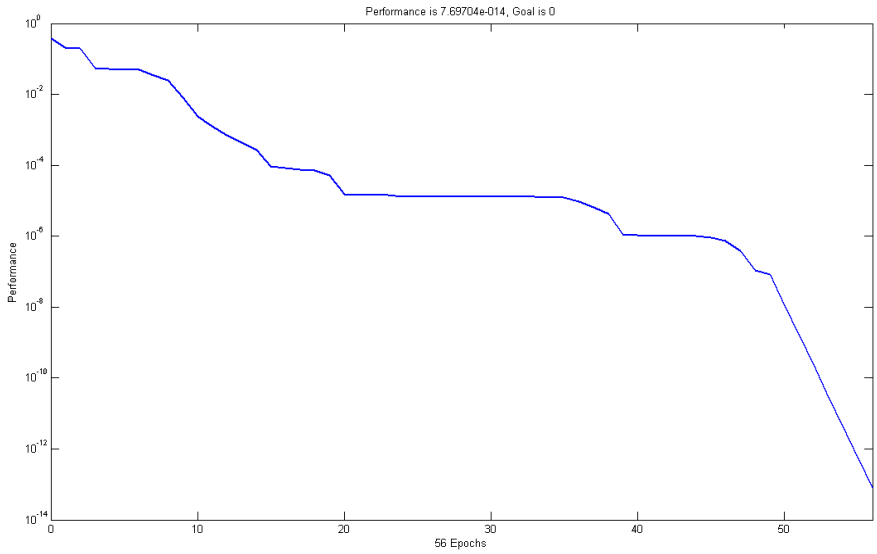
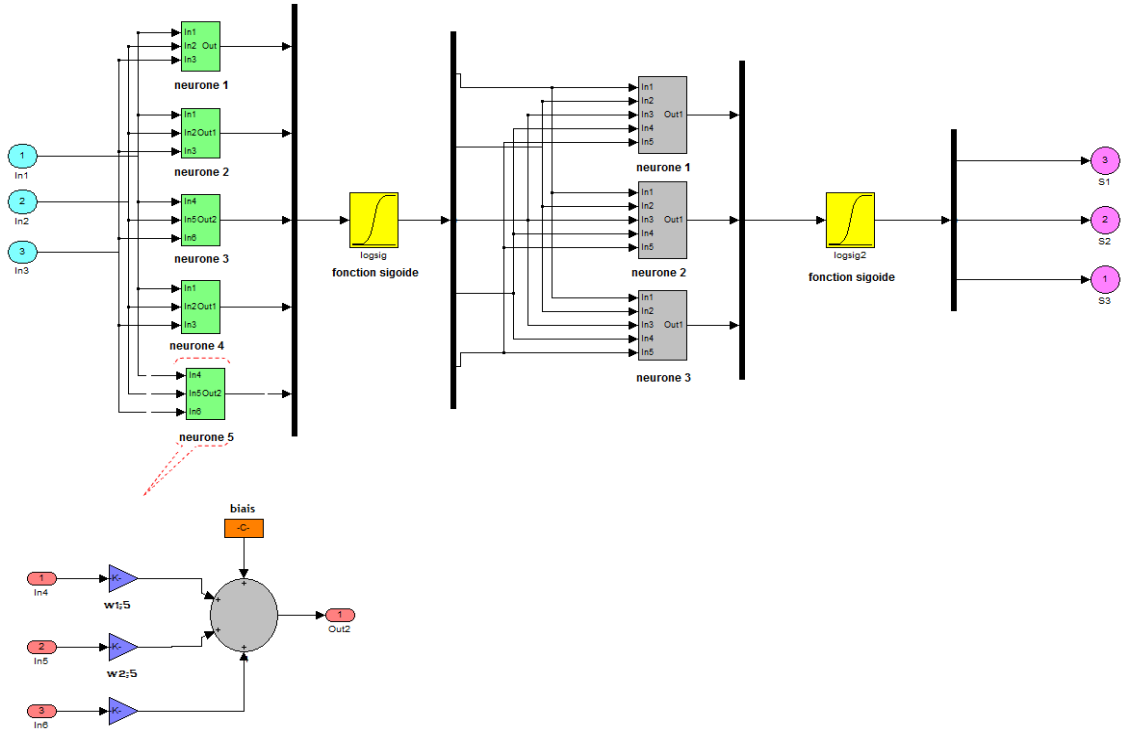


Figure (IV.4): *Evaluation de l'erreur quadratique en fonction du nombre d'itérations d'apprentissage (en utilisant la méthode de rétro propagation de gradient).*

IV.2.2.2.2 Conception de RNA sous Simulink

La conception de RNA sous Simulink est représentée sur la figure (Figure IV.5)



Figure(IV.5) : *Réseau de neurones artificiel fait par Simulink*

IV.2.2.3 Test de réseau

Alors que les tests de généralisations concernent la vérification des performances d'un réseau de neurones et sa capacité de généralisation. Une fois le réseau calculé, il faut toujours procéder à des tests afin de vérifier que notre réseau réagit correctement.

Tableau IV.2 : *Classification des défauts*

Instants d'application	**Type de défaut**
à t =2s	Coupure monophasé 1
à t = 4s	régime normale
à t = 5s	Coupure monophasé 2
à t = 7s	régime normale
à t = 8s	Coupure monophasé 3
à t = 10s	régime normale
à t =11s	Coupure biphasé 2

Cette fois nous avons appliqué des défauts à plusieurs instants sur le RNA (tableau IV.2)

La figure (IV.6) représente le résultat du test, donné par la réponse du courant statorique sur différents états de fonctionnement**.**

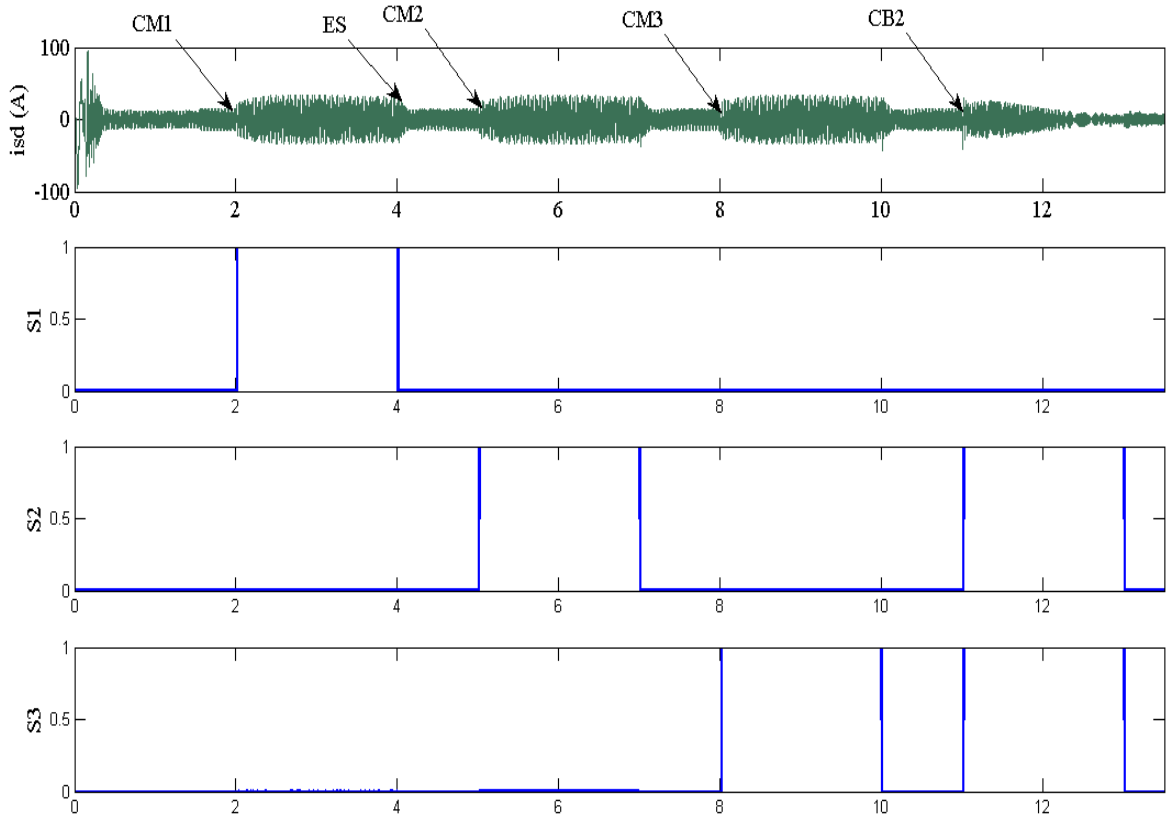


Figure (IV.6) : *évolution du courant statorique sur différents états de fonctionnement (état en appliquant des défauts biphasés)***.**

IV.2.2.3.1 Interprétation des résultats

D'après les résultats obtenus dans la phase du test, on constate que les sorties de réseau suive conformément les sorties désirées préétablies auparavant.

En effet, il est évident que les tests de réseau de neurone sur les exemples ont donné de meilleurs résultats, car tous les types de fonctionnement (les défauts et le fonctionnement normale) ont été identifiés exactement par le réseau; cela peut être expliqué par les résultats obtenus dans la phase d'apprentissage (dont les valeurs des erreurs quadratiques moyennes sont proches de zéro).

IV.3 Synthèse de RNA et Simulation sur Xilinx

IV.3.1 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.

1. On place tout le contenu de chaque couche du bloc de RNA dans le même subsystem avec un seul system *Generator* (voir Annexe E), ensuite on génère le code VHDL avec les caractéristiques suivantes :

   - *Compilation: HDL Netlist*
   - *Part: Virtex4 xc4vsx35-1011668*
   - *Synthesis Tool: XST*
   - *Target Directory: Votre choix de répertoire*
   - *Create Testbench: Checked*
   - *FPGA System Clock Period (ns): 10*

2. On ouvre le bloc *System Generator* :

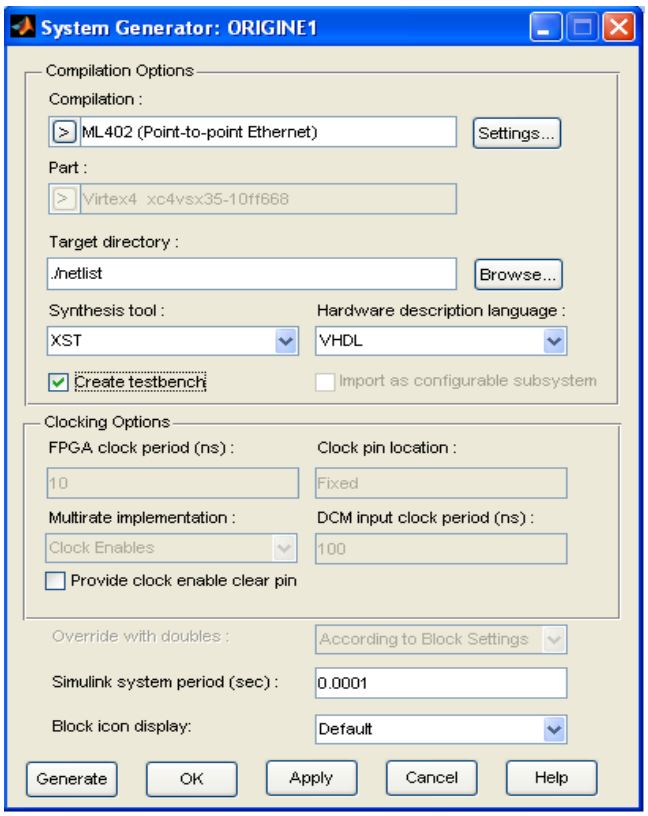


Figure (IV.7): *La fenêtre de system Generator*

3. On choisi un répertoire de travail désire dans *Target Directory***.** On met «. /netlist » qui va créer un sous répertoire dans le répertoire courant.

4. . On Choisi les paramètres suivant dans le reste de la fenêtre :

   - *Compilation: HDL Netlist*
   - *Part: Virtex4 xc4vsx35-1011668*
   - *Synthesis Tool: XST*

5. On met 10 dans *FPGA System Clock Period (ns).* Ceci sera la contrainte d'horloge pour le logiciel ISE.

6. A la fin on Cliqué sur *Generate***,** qui donne le programme en *VHDL*.

La figure (IV.8) présente le résultat obtenu :

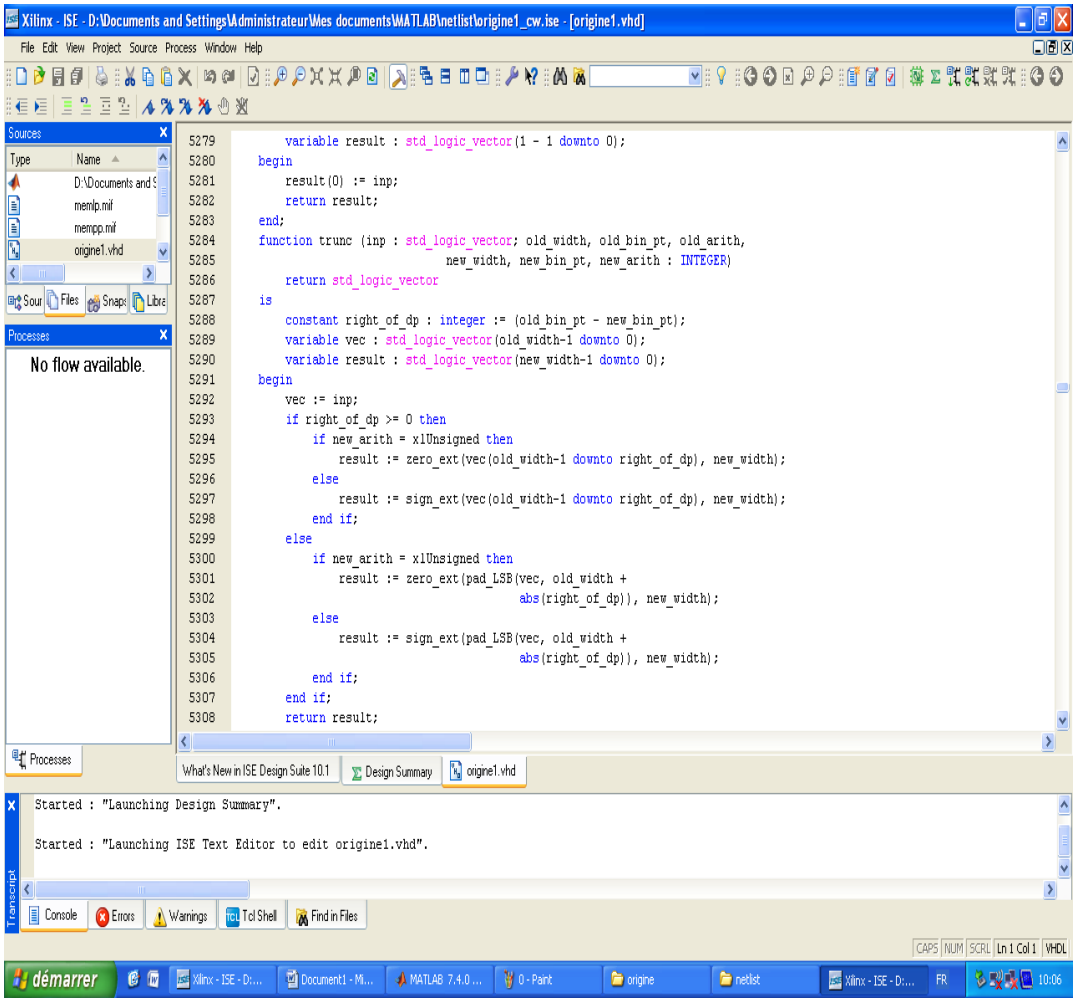


Figure (IV.8) : *Le résultat de code VHDL obtenu*

IV.3.2 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 System Generator de Xilinx.

- **Synthésize :** on Cliqué sur *Synthésize* pour connaître l'espace des ressources occupées par le code VHDL voir la figure (IV.9) :

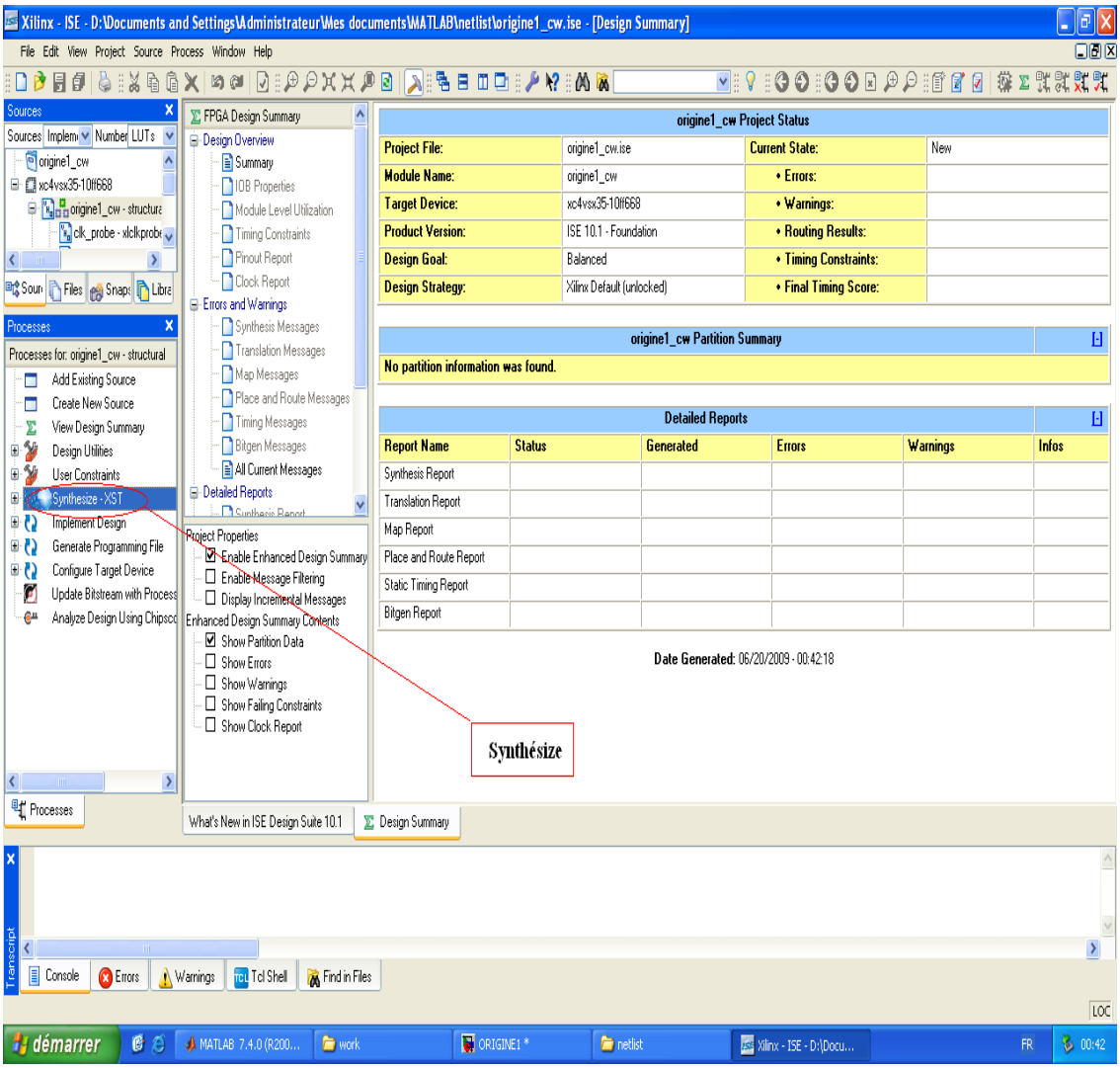


Figure (IV.9) : *La réalisation de la tâche Synthésize*

Le résultat de Synthésize est affiché dans le tableau (IV.3).

Tableau (IV.3) : *Résumé de l'espace utilisé sur le FPGA*

origine1_cw Project Status (06/20/2009 - 01:33:05)			
Project File:	origine1_cw.ise	**Current State:**	Programming File Generated
Module Name:	origine1_cw	**• Errors:**	No Errors
Target Device:	xc4vsx35-10ff668	**• Warnings:**	2718 Warnings
Product Version:	ISE 10.1 - Foundation	**• Routing Results:**	All Signals Completely Routed
Design Goal:	Balanced	**• Timing Constraints:**	X 1 Failing Constraint
Design Strategy:	Xilinx Default (unlocked)	**• Final Timing Score:**	1176420 (Timing Report)

origine1_cw Partition Summary	[-]
No partition information was found.	

Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	6,633	30,720	21%	
Number of 4 input LUTs	11,264	30,720	36%	
Logic Distribution				
Number of occupied Slices	6,613	15,360	43%	
Number of Slices containing only related logic	6,613	6,613	100%	
Number of Slices containing unrelated logic	0	6,613	0%	
Total Number of 4 input LUTs	11,726	30,720	38%	
Number used as logic	11,264			
Number used as a route-thru	462			
Number of bonded IOBs	115	448	25%	
Number of BUFG/BUFGCTRLs	1	32	3%	
Number used as BUFGs	1			
Number of RPM macros	62			

Detailed Reports					[-]
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	sam. 20. juin 00:53:16 2009	0	2704 Warnings	1 Info
Translation Report	Current	sam. 20. juin 01:10:06 2009	0	11 Warnings	0
Map Report	Current	sam. 20. juin 01:17:11 2009	0	2 Warnings	121 Infos
Place and Route Report	Current	sam. 20. juin 01:27:08 2009	0	1 Warning	0
Static Timing Report	Current	sam. 20. juin 01:27:52 2009	0	0	2 Infos
Bitgen Report	Current	sam. 20. juin 01:33:03 2009	0	0	1 Info

IV.3.3 Discutions des résultats

D'après les résultats obtenus sur le tableau on remarque que le code VHDL occupe un espace de 21% sur FPGA, signifie que le circuit FPGA de type **Virtex4 xc4vsx35-1011668** supporté ce code VHDL.

IV.4 La Co-simulation [21]

Dans le cadre de la co-simulation, on doit pouvoir connecter cette entité avec d'autres simulateurs permettant de reproduire l'environnement externe du composant et ce, quelque soit le niveau d'abstraction fait sur les entrées et sorties.

Intégrer le FPGA dans un environnement de co-simulation consiste alors à mettre en place ses interfaces.

En effet, la co-simulation est basée sur l'exécution conjointe de simulateurs, chaque simulateur représentant une partie spécifique du circuit à concevoir ou de son environnement.

Une interface de co-simulation doit permettre l'échange de données entre les simulateurs. Le concepteur doit pouvoir tester son circuit, en reliant celui-ci avec d'autres simulateurs sans que cette interface n'intervienne dans les spécifications de son système.

Initialement, la co-simulation qui correspond à la simulation conjointe du logiciel et du matériel ne fait intervenir que des composants numériques. La connexion entre les simulateurs se fait de façon particulière en introduisant la notion de " bus de co-simulation".

Dans le but d'utiliser la co-simulation, il est nécessaire de construire une interface spécifique entre un simulateur FPGA et un logiciel permettant de simuler le procédé à commander qui, dans notre cas, correspond au logiciel Matlab.

IV.4.1 Définition d'un bus de co-simulation [21]

La communication entre deux simulateurs peut se faire selon deux modes différents

- Le premier consiste à utiliser une relation de type maître-esclave entre les deux simulateurs (ils s'exécutent alors au sein d'un même processus au sens informatique du terme). Dans ce cas, l'un des simulateurs (par convention définit comme maître) appelle l'autre (alors défini comme esclave) par le biais d'une procédure (figure IV.10.a). Ce type de communication peut s'avérer complexe car il est nécessaire que le simulateur esclave puisse sauver l'ensemble du contexte à la fin de son appel pour pouvoir repartir du même état (en rechargeant le contexte) au prochain appel (figure IV.10.b). Le

lancement du simulateur esclave, la sauvegarde et la restitution peuvent s'avérer très coûteux en terme de ressources et temps de simulation.

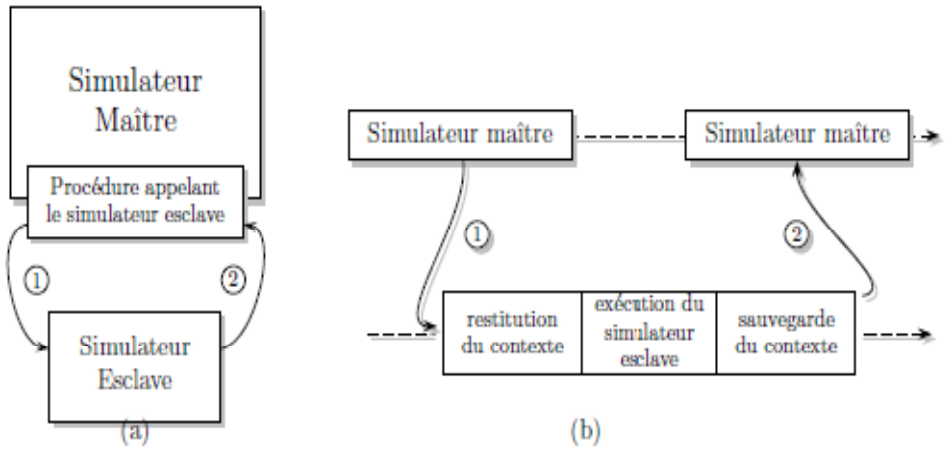


Figure (IV.10) : *Représentation du mode de fonctionnement maître/esclave pour l'échange de données entre deux simulateurs.*

Le second mode consiste à utiliser un système de simulation distribuée. Les deux simulateurs fonctionnent en parallèle et s'échangent des informations via un canal de communication (figure IV.11).

C'est ce deuxième mode de communication qui est généralement utilise en co-simulation.

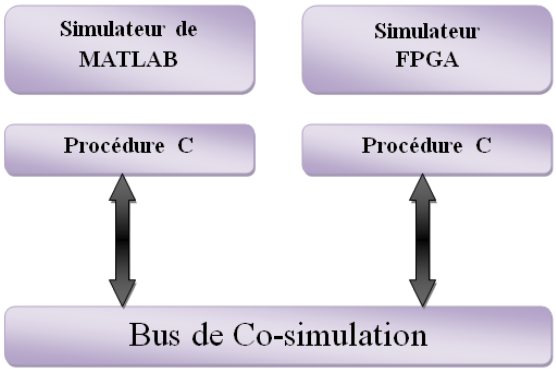


Figure (IV.11) : *Utilisation d'un bus de co-simulation*

IV.4.2 Les différents types de communication inter-processus [21]

La mise en place d'un canal de communication peut être faite en utilisant différentes technologies informatiques : on peut répertorier les mécanismes de communication interprocessus du système Unix, ainsi que le mécanisme des sockets et des **RPC**.

Les sockets peuvent être utilisés sur une station de travail unique mais il est aussi possible de mettre en place des communications réseaux (**ethernet**) utilisant des protocoles tels que **UDP** ou **TCP**.

IV.4.3 Application de la Co-simulation au diagnostic

IV.4.3.1 Etape de la co-simulation

On a travaillé avec les quatre étapes suivantes :

1. construire le bloc en *Xilinx Simulink*
2. tester le bloc en *Xilinx* Simulink
3. Générer le bloc de la co-simulation
4. Implémenter le bloc de la co-simulation

IV.4.3.1.1 Construire le bloc en Xilinx Simulink

La figure suivante le schéma bloc de réseau de neurones sous *Xilinx Simulink*, dans cette figure on illustré le comportement du deuxième neurone de la couche cachée, ainsi la fonction sigmoïde**.**

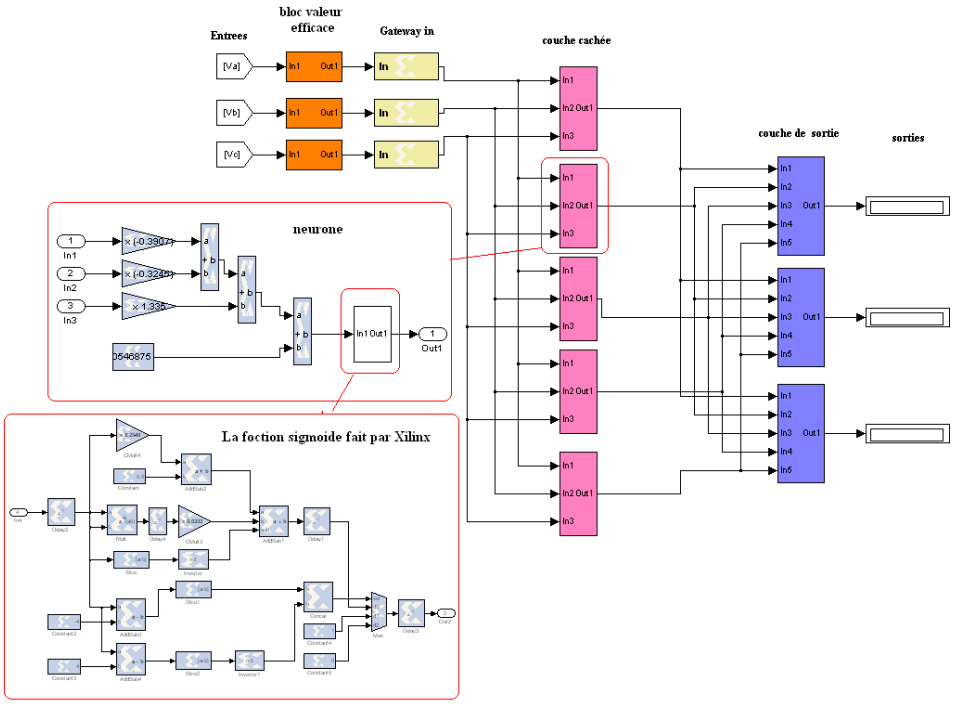


Figure (IV.12) : *Representation deRNA fait par Xilinx Simulink*

IV.4.3.1.2 Test de réseau fait par Xilinx Simulink

La teste se faite de la meme manière que precedament (test de RNA fait par simulink), on a suivit les conditions du tableau IV.2.

Le resultat du test est donnée par la figure (IV.13)

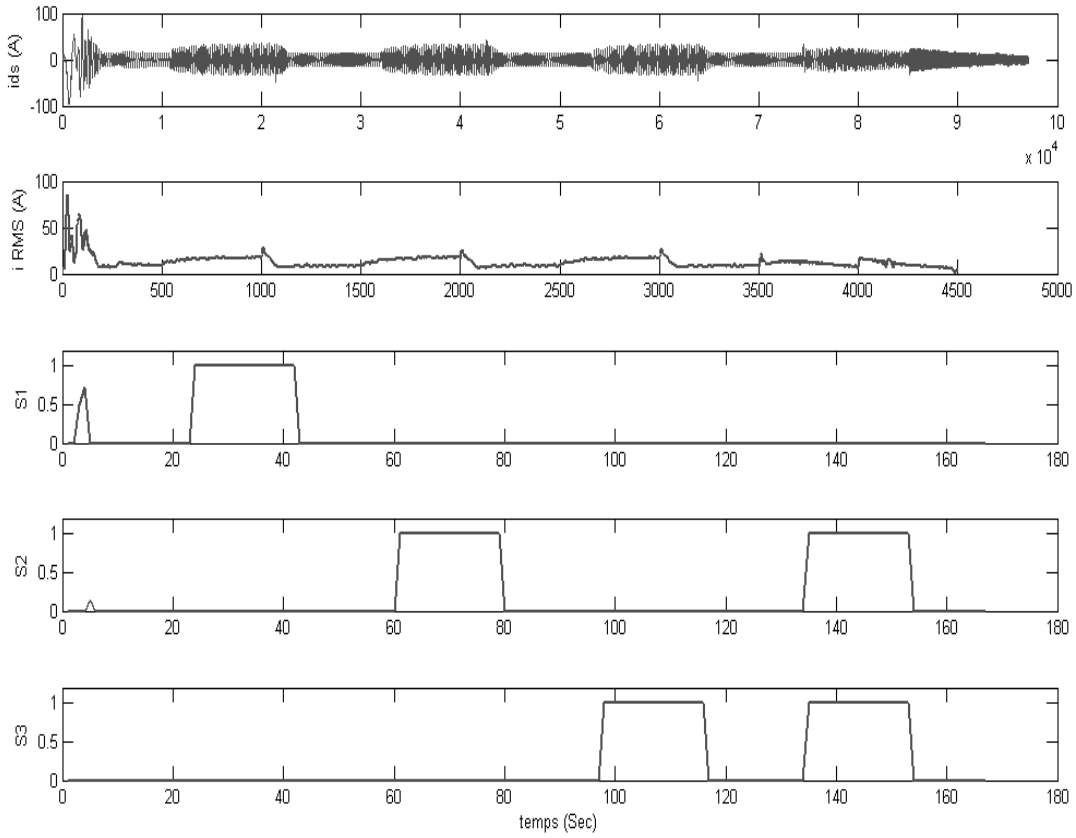


Figure (IV.13) : *Test du RNA fait par Xilinx Simulink*

Il est apparait que le test a été donné des bons résultats.

IV.4.3.1.3 Génération du bloc de la co-simulation

On a appliqué la co-simulation sur la fonction d'activation sigmoïde tel qu'on a suivit les étapes suivantes :

1. Premièrement, on ouvre le bloc *System Generator* sous Simulink suivant la figure(IV.14), (voir l'approximation de la fonction sigmoïde dans l'annexe H)

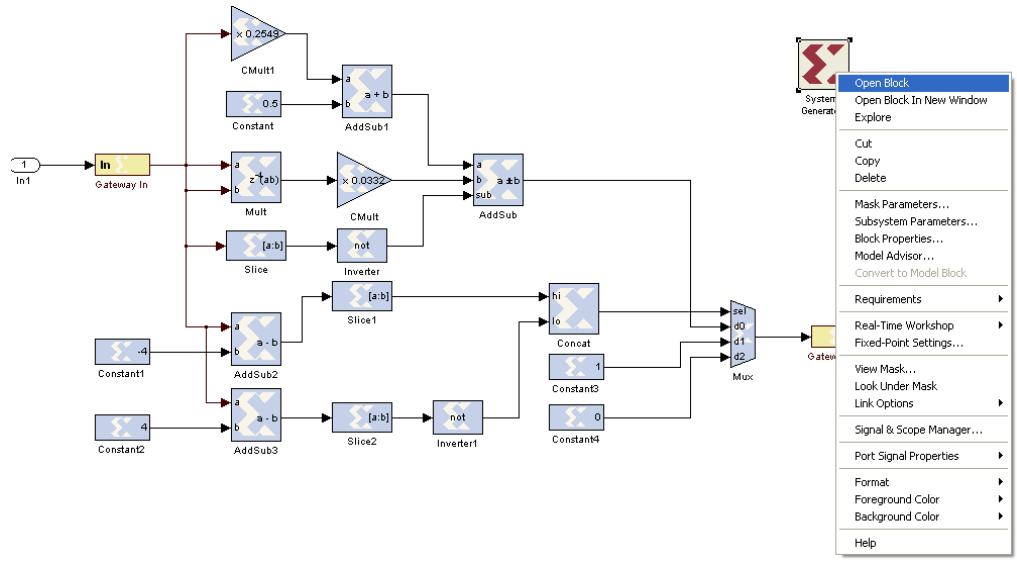


Figure (IV.14): *La fenêtre de system Generator*

2. On choisi un répertoire de travail désire dans *Target Directory*. On met. /netlist qui va créer un sous répertoire dans le répertoire courant.

3. On Choisi les paramètres suivant dans le reste de la fenêtre de la figure (IV.15):

   - **Compilation: ML 402 (Point-to-point Ethernet)**
   - **Synthesis Tool: XST**

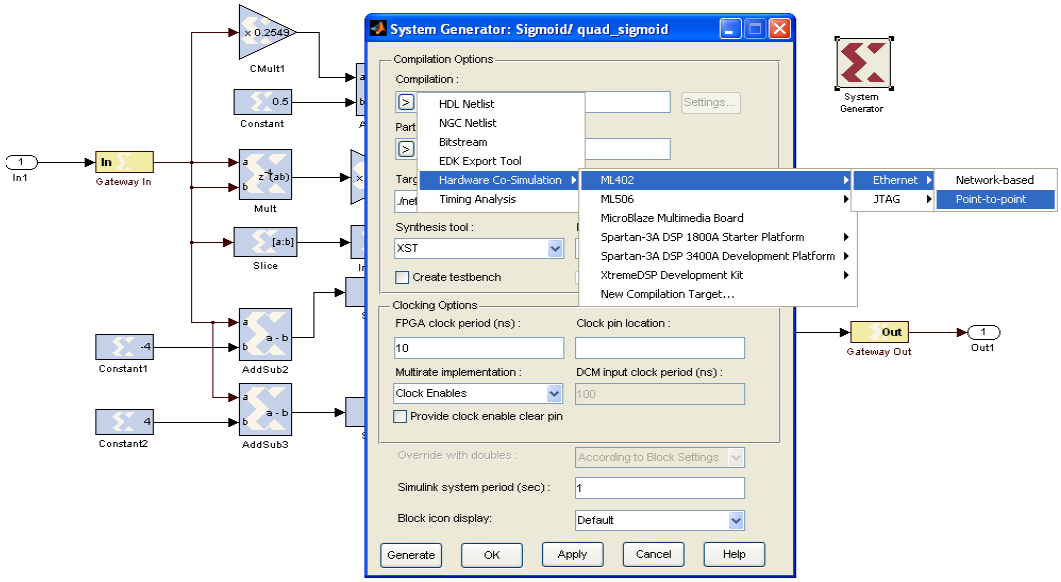


Figure (IV.15): *Sous fenêtre de system Generator*

4. A la fin on Cliqué sur *Generate***,** qui donne le bloc de Co-simulation**.**

La figure (IV.16) présente la résultat obtenu :

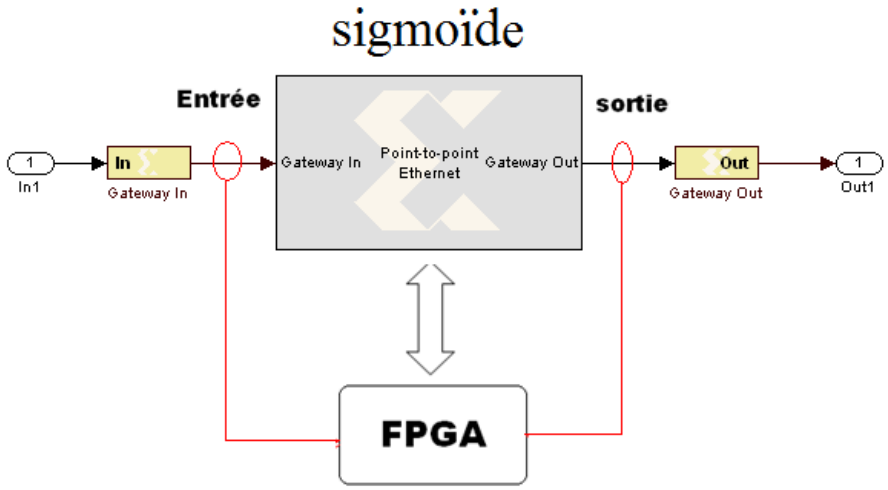


Figure (IV.16): *Principe de l'implémentation du bloc sigmoïde*

IV.4.3.1.4 Implémentation du bloc de la co-simulation

Pour cette dernière phase, on a implémenté le bloc sur un circuit FPGA **Virtex4 xc4vsx35-1011668** (voir Annexe G). A travers d'un câble de connexion (ou ethernet), voir la figure (IV.17).

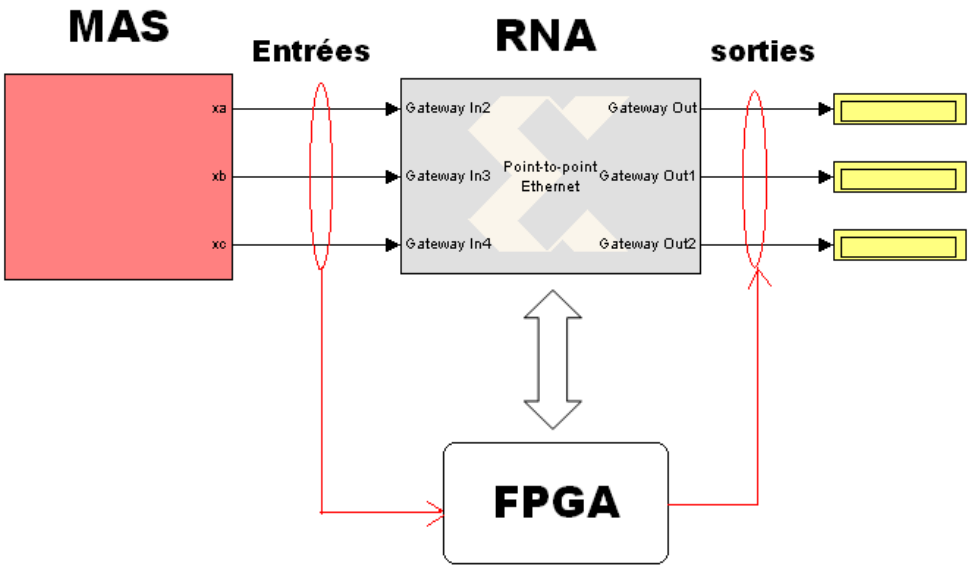


Figure (IV.17): *Principe de l'implémentation du bloc RNA sur FPGA*

IV.5 Conclusion

Dans ce chapitre, nous avons étudié le réseau de neurones qui utilise des entrées simples telles que les valeurs efficaces des trois tensions d'alimentation.

Par ailleurs, pour la mise en œuvre du réseau de neurones on a tout d'abord passé par plusieurs études paramétriques (choix du type de réseau, choix des entrées, choix des sorties...). Ces études ont été précédées par l'opération d'acquisition des données, qui a pour but d'établir la base d'apprentissage de réseau.

Nous venons de voir dans ce chapitre que les réseaux de neurones peuvent être utilisés pour le diagnostic des défaillances dans les systèmes électromécaniques. L'efficacité de ces architectures a été démontrée par des exemples de simulation et des résultats satisfaisants ont été obtenus.

Nous avons aussi 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 **Virtex4** 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 sigmoïde ainsi que pour le réseau de neurones montrent que les résultats obtenus sous Xilinx ont donné les mêmes performances que la fonction sigmoïde obtenus sous Simulink ainsi que pour le réseau de neurones étudié.

Conclusion générale

Conclusion générale

Dans ce travail on a étudié une tache principale de la surveillance, qui est le diagnostic des défauts dans une MAS commandée par la technique de l'orientation du flux rotorique.

Nous avons commencé 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 mieux que l'automatisme ordinaire telles que les méthodes internes.

Ensuite, on a effectué la modélisation de l'ensemble moteur-convertisseur statistique et leur commande afin de révéler le comportement de la machine vis-à-vis des différentes défaillances qui peuvent surgir sur cet ensemble.

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 été 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 ou sur Xilinx sous Simulink.

Enfin, 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 librairie de Xilinx (system Generator) pour que cette dernière soit directement transformée en langage VHDL ou donnée sous forme de bloc permettant la Co-simulation. Cette dernière nous a permis de s'approcher au cas réel pour le diagnostic des défaillances de la machine asynchrone par réseau de neurones en temps réel.

Annexe

Annexe A

Paramètres de la machine asynchrone

Pour toute simulation dans ce mémoire, nous avons adopté une machine asynchrone de puissance 5.5Kw dont les caractéristiques suivantes :

- La tension nominale : U=220/380V.
- Le courant nominal : 21/12V.
- Fréquence du réseau : 50 Hz.
- Vitesse nominale : Nn= 1420 tr/mn.
- Nombre de paires de pôles : p = 1.
- Résistance statorique : Rs = 2.25 Ohm.
- Inductance cyclique statorique : Ls = 0.1232 H.
- Résistance rotorique : Rr = 0.70 Ohm.
- Inductance cyclique rotorique : Lr = 0.1122 H
- Inductance mutuelle cyclique : M = 0.1118 H.
- Constante de temps rotorique : T_r = 0.1600 s.
- Constante de temps statorique : T_s = 0.0546 s.
- Coefficient de dispersion : σ=0.1133.
- Moment d'inertie : J =0.038 kg.m2.

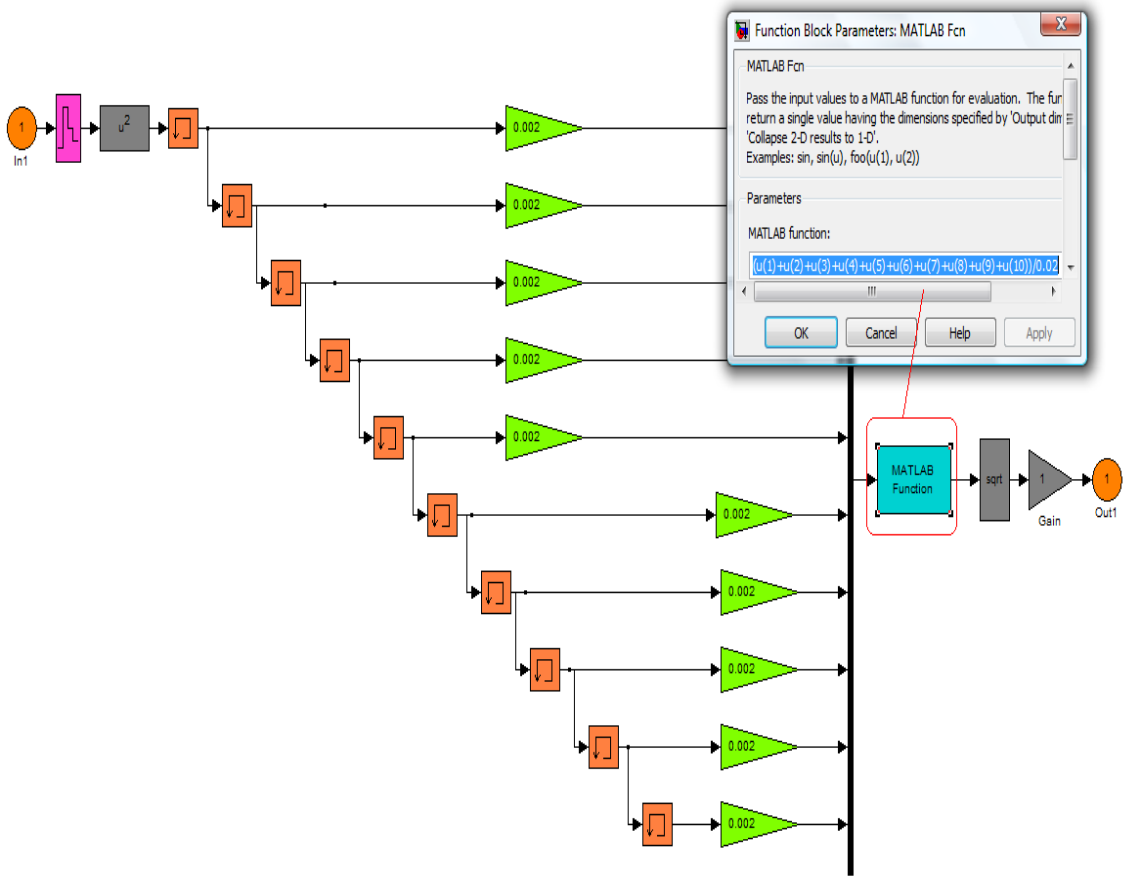
Annexe B

S-Function de l'onduleur de tension

```
function [sys,x0,str,ts]=sys_trphes(t,x,u,flag)
Ts=0.0001;E=400;vp=390;f=50;thettas=0;m=100;fp=f*m;
if flag == 0
sys=[0;0;3;3;0;1;1];
str = [];
x0  = [];
ts  = [Ts 0];
elseif flag == 3
v1=u(1);
v2=u(2);
v3=u(3);
v=vp*asin(sin(2*pi*fp*t))*2/pi;
if v1>v
    sa=1;
else
    sa=0;
end
if v2>v
    sb=1;
else
    sb=0;
end
if v3>v
    sc=1;
else
    sc=0;
end
u1=(2*sa-sb-sc)*E/3;
u2=(2*sb-sa-sc)*E/3;
u3=(2*sc-sb-sa)*E/3;
vd=sqrt(2/3)*(cos(thettas)*u1+cos(thettas-(2*pi/3))*u2+cos(thettas+(2*pi/3))*u3);
vq=sqrt(2/3)*(-sin(thettas)*u1-(sin(thettas-(2*pi/3)))*u2-(sin(thettas+(2*pi/3))*u3));
sys=[u1;u2;u3];
else
sys =[];
end
```

Annexe C

Calcul de la valeur efficace (RMS)



Cette figure représente un schéma bloc fait par Simulink, pour calculer la valeur efficace des tensions statoriques.

Annexe D

Algorithme de rétropropagation du gradient

Dans cette annexe on va détailler l'algorithme de rétropropagation du gradient **[30]** :
La fonction d'activation choisie pour le réseau étudié est la fonction sigmoïde qui est donnée par :

$$S(x) = \frac{1}{1+e^{-x}} \qquad (A.1)$$

Sa dérivée est :

$$S'(x) = S(x).(1-S(x)) \qquad (A.2)$$

Cet algorithme est basé sur une extension de la règle delta. On va donc utiliser un gradient stochastique pour tenter de minimiser l'erreur quadratique sur la sortie.

$$p_i^k = \sum w_{ij} \cdot s_j^k + \theta_i \qquad (A.3)$$

et :

$$s_i^k = \varphi(p_i^k) \qquad (A.4)$$

avec :

- p_i^k : le potentiel du neurone *i* pour l'exemple *k*,
- w_{ij} : le poids de la connexion du neurone *j* vers le neurone *i*,
- θ_i : le seuil de neurone *i*,
- s_i^k : le signal de sortie du neurone *i* pour l'exemple *k*,
- φ : la fonction de transfert.

L'erreur sur l'exemple *k* est définie comme :

$$E^k = \frac{1}{2}\sum_{o=1}^{O}(d_o^k - s_o^k)^2 \qquad (A.5)$$

d_o^k le signal désiré pour l'exemple *k* sur le neurone de sortie *o*, et *O* le nombre de neurones sur la couche de sortie. L'erreur totale est :

$$E = \sum E^k \qquad (A.6)$$

L'application de la règle delta donne :

$$\Delta^k w_{ji} = -\eta \cdot \frac{\partial E^k}{\partial w_{ji}} \qquad (A.7)$$

On peut poser :

$$\frac{\partial E^k}{\partial w_{ji}} = \frac{\partial E^k}{\partial p_i^k} \cdot \frac{\partial p_i^k}{\partial w_{ji}} \qquad (A.8)$$

De (A.3) on déduit :

$$\frac{\partial p_i^k}{\partial w_{ji}} = s_j^k \qquad (A.9)$$

On définit le *signal d'erreur* :

$$\delta_i^k = -\frac{\partial E^k}{\partial p_i^k} \qquad (A.10)$$

De (A.7), (A.8), (A.9), (A.10) on déduit la fonction d'apprentissage :

$$\Delta^k w_{ji} = \eta \cdot \delta_i^k \cdot s_j^k \qquad (A.11)$$

On corrige donc les coefficients synaptiques en fonction opposée au signal d'erreur. La correction apportée au coefficient synaptique est également fonction du signal reçu.

Le problème se pose maintenant pour la détermination du signal d'erreur. On peut écrire :

$$\delta_i^k = -\frac{\partial E^k}{\partial p_i^k} = -\frac{\partial E^k}{\partial s_i^k} \cdot \frac{\partial s_i^k}{\partial p_i^k} \qquad (A.12)$$

De (A.4) on déduit :

$$\frac{\partial s_i^k}{\partial p_i^k} = \varphi'(p_i^k) \qquad (A.13)$$

À partir de là, on peut traiter la couche de sortie. En effet, de (A.5) on déduit :

$$\frac{\partial E^k}{\partial s_o^k} = -(d_o^k - s_o^k) \qquad (A.14)$$

Soit après (A.12), (A.13) et (A.14) :

$$\delta_o^k = (d_o^k - s_o^k) \cdot \varphi'(p_o^k) \qquad (A.15)$$

On a donc d'après (A.8), (A.9), (A.10) et (A.15) :

$$\frac{\partial E^k}{\partial w_{jo}} = -\delta_o^k \cdot s_j^k = -(d_o^k - s_o^k) \cdot \varphi'(p_o^k) \cdot s_j^k \qquad (A.16)$$

Par ceci, l'équation (A.7) donne la règle d'apprentissage suivante pour les neurones de sortie :

$$\Delta^k w_{jo} = \eta \cdot (d_o^k - s_o^k) \cdot \varphi'(p_o^k) \cdot s_j^k \qquad (A.17)$$

Pour chaque neurone de sortie, on corrige l'erreur induite par les neurones de la couche cachée qui lui sont liés. Cette correction est proportionnelle à l'ampleur de l'erreur et à la fonction du signal reçu.

Cas de fonction de transfert sigmoïde

Prenons le cas où la fonction de transfert est la fonction logistique. On a d'après (A.1) :

$$\varphi(x) = s(x) = \frac{1}{1+e^{-x}} \qquad (A.18)$$

De (A.2) on déduit :

$$\varphi'(p_o^k) = \varphi(p_o^k) \cdot (1 - \varphi(p_o^k)) \qquad (A.19)$$

$$\varphi'(p_o^k) = \frac{1}{1+e^{-p_o^k}} \cdot (1 - \frac{1}{1+e^{-p_o^k}}) = s_o^k \cdot (1 - s_o^k) \qquad (A.20)$$

En substituant dans (A.17) :

$$\Delta^k w_{jo} = \eta \cdot (d_o^k - s_o^k) \cdot s_o^k \cdot (1 - s_o^k) \cdot s_j^k \qquad (A.21)$$

Pour les neurones cachés, on ne peut pas calculer directement les corrections puisque l'on ne connait pas leurs contributions respectives directes à l'erreur sur la sortie liée. Toutefois, il est possible d'exprimer l'erreur sur la sortie comme fonction des différents potentiels des neurones cachés. Soit :

$$E^k = E^k(p_0^k, p_1^k, ..., p_H^k) \qquad (A.22)$$

Pout *H* neurones de la couche cachée.

On a :

$$\frac{\partial E^k}{\partial s_h^k} = \sum_{o=1}^{O} \frac{\partial E^k}{\partial p_o^k} \cdot \frac{\partial p_o^k}{\partial s_h^k} = \sum_{o=1}^{O} \frac{\partial E^k}{\partial p_o^k} \cdot \frac{\partial}{\partial s_h^k} \sum_{j=1}^{H} w_{jo}\, s_j^k \qquad (A.23)$$

$$\frac{\partial E^k}{\partial s_h^k} = \sum_{o=1}^{O} \frac{\partial E^k}{\partial p_o^k} w_{ho} = -\sum_{o=1}^{O} \delta_o^k\, w_{ho} \qquad (A.24)$$

Soit d'après (A.12) et (A.13) :

$$\delta_h^k = \varphi'(p_h^k) \cdot \sum_{o=1}^{O} \delta_o^k \cdot w_{ho} \qquad (A.25)$$

Le signal d'erreur sur le neurone caché est fonction de la moyenne des erreurs sur les neurones de sortie qui lui sont liés, pondérée par les poids de connexion.

Pour les réseaux comportant plus d'une couche cachée, on applique la même règle de manière récursive, la couche cachée suivante joue alors le même rôle que la couche de sortie. Soit pour une couche cachée ***h*** connectée à la couche suivante ***h'*** :

$$\delta_h^k = \varphi'(p_h^k) \cdot \sum_{o=1}^{H'} \delta_{h'}^k \cdot w_{hh'} \qquad (A.26)$$

De (A.11) on déduit :

$$\Delta^k w_{ih} = \eta \cdot \varphi'(p_h^k) \cdot \sum_{o=1}^{O} \delta_o^k \cdot w_{ho} \cdot s_i^k \qquad (A.27)$$

Avec la fonction logistique, la règle d'apprentissage devient d'après (A.20) et (A.27) :

$$\Delta^k w_{ih} = \eta \cdot (s_h^k \cdot (1 - s_h^k)) \cdot \sum_{o=1}^{O} \delta_o^k \cdot w_{ho} \cdot s_i^k \qquad (A.28)$$

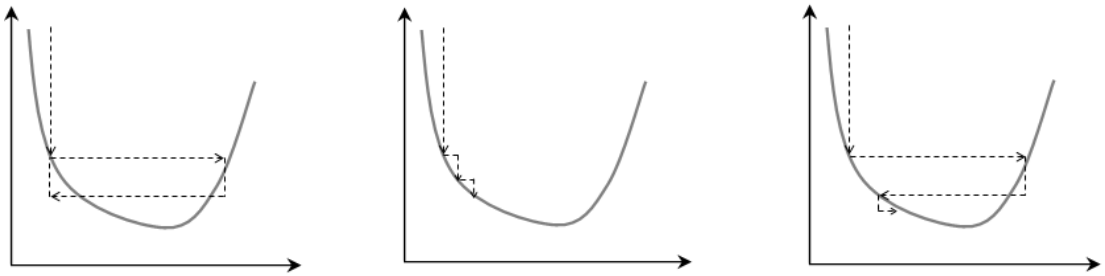
$$\Delta^k w_{ih} = \eta \cdot (s_h^k \cdot (1 - s_h^k)) \cdot \sum_{o=1}^{O} (d_o^k - s_o^k) \cdot s_o^k \cdot (1 - s_o^k) \cdot w_{ho} \cdot s_i^k \qquad (A.29)$$

avec :

- η : Valeur constante qui sert à contrôler la vitesse de convergence de l'apprentissage (pas d'apprentissage - *learning rate*). Cette valeur est comprise dans l'intervalle [0..1] .

Pas d’apprentissage

Le pas d’apprentissage (momontum) η est essentiel au bon fonctionnement de la rétropropagation. Trop élevé, il conduit à des oscillations, trop faible il ralentit la convergence. Dans l’idéal, il doit être élevé dans les descentes est se réduire quand se produit une oscillation, c’est-à-dire un changement de signe de la dérivée de la fonction d’erreur (figure A.1).

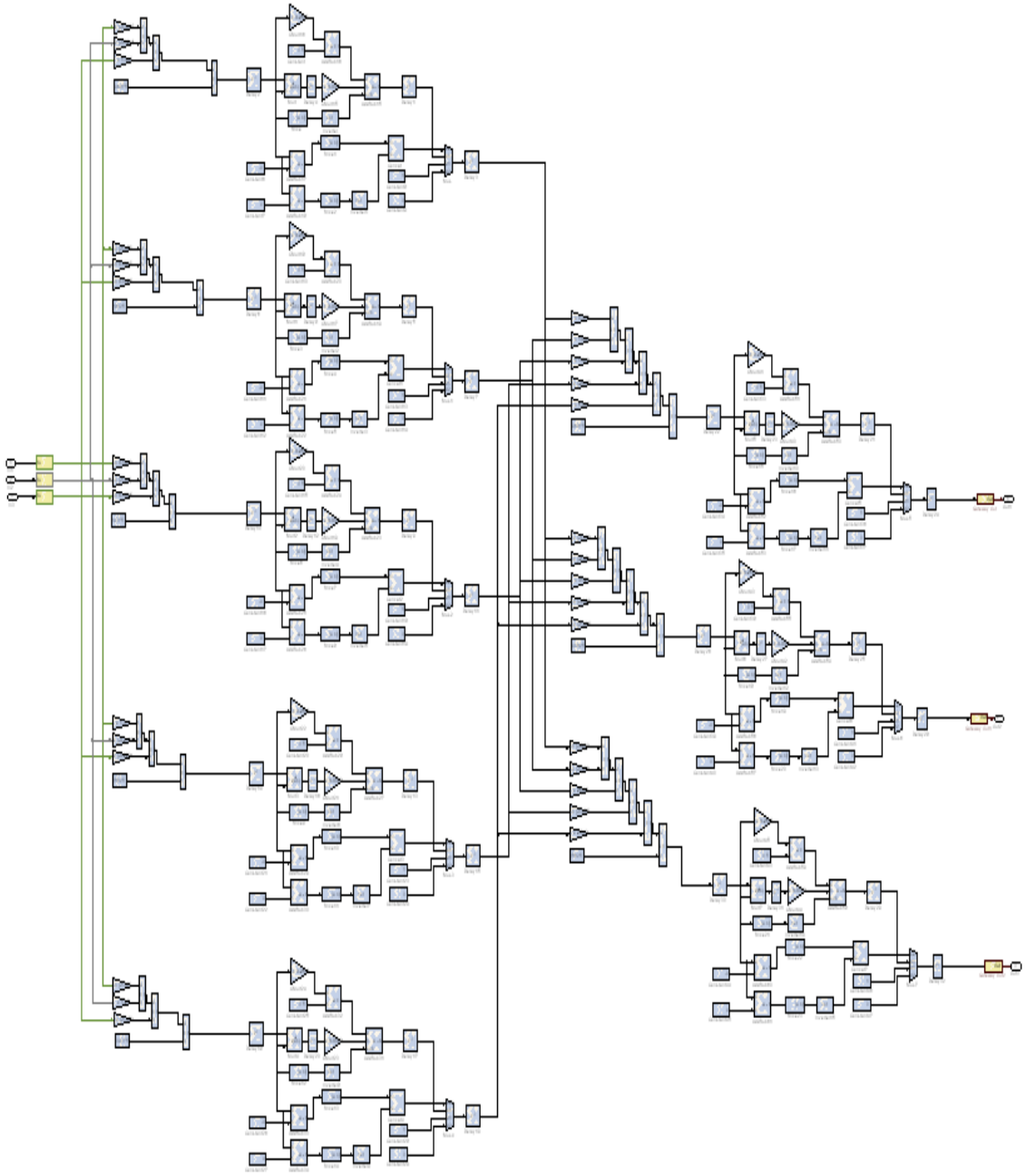


a) η fort *b) η faible* *c) η correct*

Figure (A.1) : *Différents pas d’apprentissage*

Annexe E

Réseau de neurones fait par Xilinx



Annexe F

F1. Approximation de la fonction sigmoïde

Nous avons l'intention d'approximer la fonction sigmoïde sous forme polynomiale selon la matrice de *Vangove*, où la fonction (F.1) devient (F.2):

$$f\ (x) = \frac{1}{1+e^{-cx}} \qquad (F.1)$$

$$f\ (x) = c + bx + ax^2 \qquad (F.2)$$

pour la détermination des coefficients **c, b, a** on applique tout d'abord :

$$\begin{cases} c = \dfrac{1}{1+e^{-a1}} \\ b = \dfrac{1}{1+e^{-a2}} \\ a = \dfrac{1}{1+e^{-a3}} \end{cases} \qquad (F.3)$$

avec :

$\vec{a} = \begin{bmatrix} a1 \\ a2 \\ a3 \end{bmatrix}$ est un vecteur qui obtenue a partir de $\vec{a} = \vec{V} / \vec{F}$

$$où : \quad \vec{a} = \begin{bmatrix} 1 & 0^1 & 0^2 \\ 1 & 2^1 & 2^2 \\ 1 & 4^1 & 4^2 \end{bmatrix}, \quad \vec{F} = \begin{bmatrix} 0 \\ 2 \\ 4 \end{bmatrix} \qquad (F.6)$$

F.2 Génération du code VHDL de la sigmoïde

Tableau. F.1 : *Génération du code VHDL de la sigmoïde*

quad_sigmoid_cw Project Status			
Project File:	quad_sigmoid_cw.ise	**Current State:**	Programming File Generated
Module Name:	quad_sigmoid_cw	**• Errors:**	
Target Device:	xc4vsx35-10ff668	**• Warnings:**	
Product Version:	ISE 10.1 - Foundation	**• Routing Results:**	All Signals Completely Routed
Design Goal:	Balanced	**• Timing Constraints:**	All Constraints Met
Design Strategy:	Xilinx Default (unlocked)	**• Final Timing Score:**	0 (Timing Report)

quad_sigmoid_cw Partition Summary	[-]
No partition information was found.	

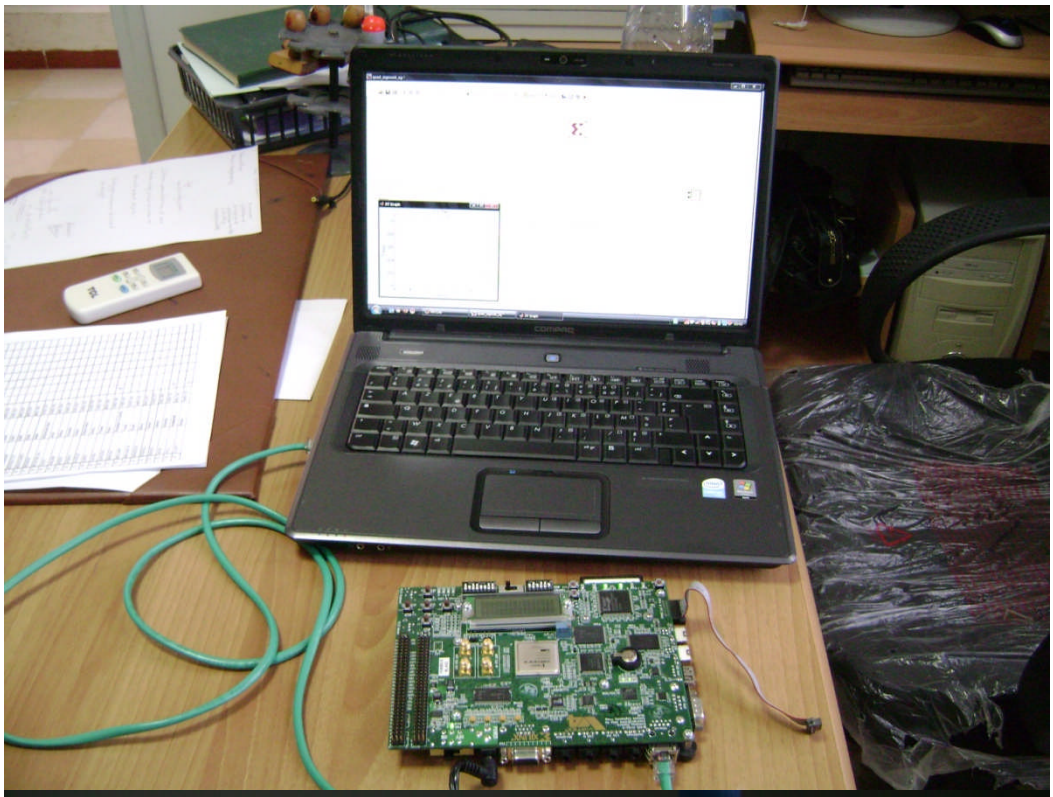
Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	364	30,720	1%	
Number of 4 input LUTs	488	30,720	1%	
Logic Distribution				
Number of occupied Slices	300	15,360	1%	
Number of Slices containing only related logic	300	300	100%	
Number of Slices containing unrelated logic	0	300	0%	
Total Number of 4 input LUTs	511	30,720	1%	
Number used as logic	488			
Number used as a route-thru	23			
Number of bonded IOBs	39	448	8%	
Number of BUFG/BUFGCTRLs	1	32	3%	
Number used as BUFGs	1			
Number of RPM macros	4			

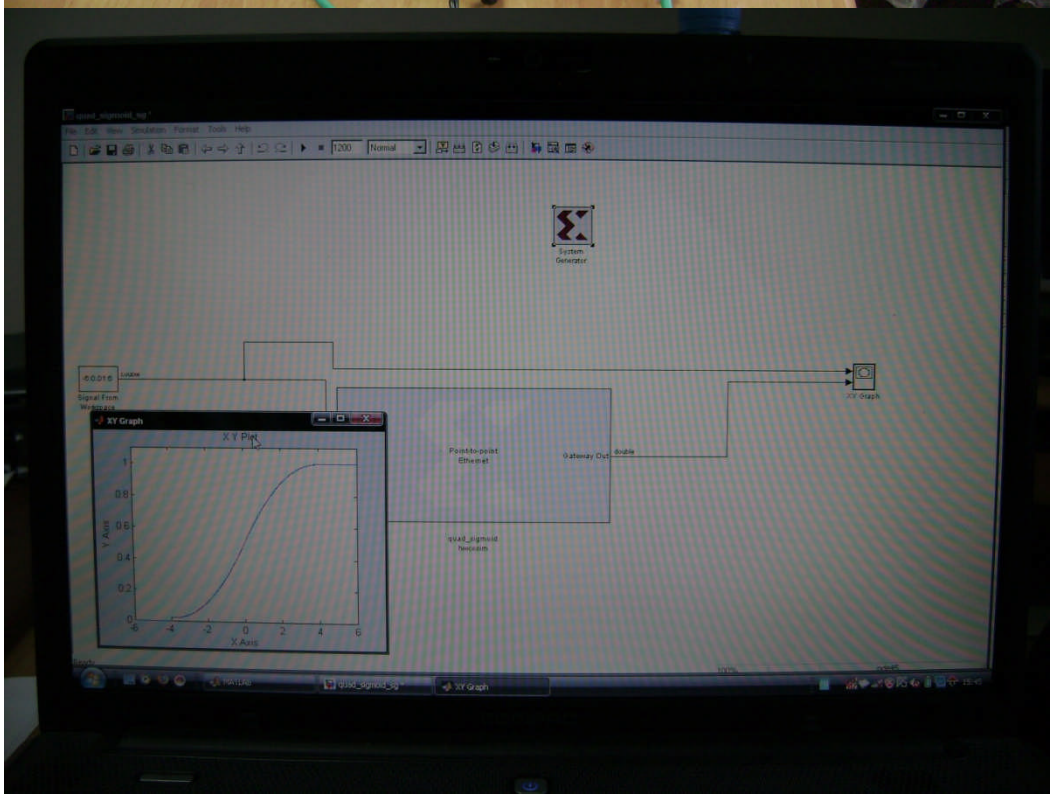
Performance Summary			[-]
Final Timing Score:	0	**Pinout Data:**	Pinout Report
Routing Results:	All Signals Completely Routed	**Clock Data:**	Clock Report
Timing Constraints:	All Constraints Met		

Detailed Reports					[-]
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	sam. 20. juin 01:52:58 2009			
Translation Report	Current	sam. 20. juin 01:53:30 2009			
Map Report	Current	sam. 20. juin 01:54:21 2009			
Place and Route Report	Current	sam. 20. juin 01:54:44 2009			
Static Timing Report	Current	sam. 20. juin 01:54:56 2009			
Bitgen Report	Current	sam. 20. juin 01:56:51 2009			

Annexe G

La carte VIRTEX-4




System Generator
XY Graph
XY Plot
X Axis
Y Axis
Point-to-point Ethernet
Gateway Out
double

Bibliographie

Bibliographie

[1] S.Touaf, *" Diagnostic Logique des Systèmes Complexes Dynamiques dans un Contexte Multi-Agent"*, Thèse de Doctorat de L'université Joseph Fourier – Grenoble 1, Grenoble, France, Mars 2005.

[2] O. Ondel, E. Boutleux and G. Clerc, *"A Method to Detect Broken Bars in Induction Machine Using Pattern Recognition Techniques"*, IEEE Transactions On Industry Applications, Vol. 42, No. 4, July/August 2006.

[3] R. Isermann, *"Model-Based Fault Detection And Diagnosis -Status And Applications-"*, *IFAC* Institute of Automatic Control, Darmstadt University of Technology, Darmstadt, Germany 2004.

[4] M.L. Sin, W.L. Soong and N. Ertugrul, *"Induction Machine On-Line Condition Monitoring and Fault Diagnosis-A Survey"*, University of Adelaide, 2003.

[5] N. Moubayed, *"Detection et Localisation Des Défauts Dans Les Convertisseurs Statiques"*, 6TH International Conference on Electromechanical and Power Systems, Chişinău, Rep. Moldova, October, 2007.

[6] H. Razik, *"Le Contenu Spectral Du Courant Absorbé Par La Machine Asynchrone En Cas De Défaillance, Un Etat De L'Art"*, La revue IEEE n°29, Page 48-52, Juin, 2002.

[7] G. Didier et H. Razik,*"Sur La Détection D'un Défaut Au Rotor Des Moteurs Asynchrones"*, La revue IEEE n°27, Page 48-52, Décembre, 2001.

[8] R. Benguettaf et R. Zaoui, *" Implémentation des réseaux de neurones artificiels sur FPGA pour le diagnostic des défaillances de la machine asynchrone"*, Mémoire de fin d'étude de L'Université de M'sila, France, Juin 2008.

[9] O. Ondel,*"Diagnostic Par Reconnaissance Des Formes : Application à un Ensemble Convertisseur-Machine Asynchrone"*, Thèse de Doctorat de L'école Centrale De Lyon, France, Octobre 2006.

[10] G. Zwingelstein, *"Diagnostic des Défaillance, théorie et pratique pour les systèmes industriels"*, Edition Hermes, Paris 1995.

[11] R. Casimir, *"Diagnostic Des Défauts Des Machines Asynchrones Par Reconnaissance des Formes : Application A Un Ensemble Convertisseur-Machine Asynchrone"*, Thèse de Doctorat de L'école Centrale De Lyon, France, Décembre 2003.

[12] G. Didier, *"Modélisation et diagnostic de la machine asynchrone en présence de défaillances"*, Thèse de Doctorat de l'Université Henri Poincaré, Nancy-I, France, Octobre 2004.

[13] T. Toudja, *"Commande non linéaire robuste d'un moteur synchrone à aimant permanent"*, Thèse de Magister de l'Ecole Nationale Polytechnique, Alger, Février 2008.

[14] R. Champagne, *"Simulation en temps réel à l'aide de la représentation d'état : Application à un entraînement électrique basé sur une machine asynchrone"*, Thèse de Doctorat de l'École de Technologie Supérieure Université de Québec, Montréal, Canada, Juillet 2001.

[15] A. Aïssa, *"Commande sans capteur de vitesse par DTC d'une machine synchrone à aimants permanents dotée d'un observateur d'ordre complet à mode glissants"*, Thèse de Magister de Université de Batna, Octobre 2005.

[16] G. Buche, *"Commande vectorielle de la machine asynchrone en environnement temps réel Matlab/Simulink"*, Mémoire d'ingénieur de C.N.A.M de Grenoble, France, Mars 2001.

[17] M. Yousfi, et Z. Bouhenaf, *"Commande directe du couple de la machine synchrone double étoile alimentée par deux types d'onduleurs"*, P.F.E de l'École Nationale Polytechnique, Alger, Juin 2007.

[18] R. Achouri, et M. Hidouche, *"Asynchrone, Simulation sur logiciel SIMPLORER, Validation expérimentale sur banc d'essai "*, P.F.E de l'École Nationale Polytechnique, Alger, Juin 2007.

[19] J.P. Caron, et J.P. Hautier, *"Modélisation et Commande de la Machine Asynchrone"*, Edition Technip, Paris 1995.

[20] G. Sturtzer, et E. Smigiel, *"Modélisation et Commande des Moteurs Triphasés, commande vectorielle des moteurs synchrones, commande numérique par contrôleurs DSP"*, Edition Ellipses, Paris 2000.

[21] R. Ruelland, *" Apport de la co-simulation dans la conception de l'architecture des dispositifs de commande numérique pour les systèmes électriques "*, Thèse de doctorat de L'Institut National Polytechnique de Toulouse, France, Septembre 2002.

[22] G-O. Cimuca, *"Système Inertiel de Stockage d'Energie Associe à des Générateurs Eoliens"*, Thèse de Doctorat de l'Ecole Nationale Supérieure d'Arts et Métiers Centre de Lille, France, 2005.

[23] G. Grellet et G. Clerc *"Actionneurs Électriques, principes modèles commande"*, Edition Eyrolles, Paris 2000.

[24] S. Bachir, *"Contribution Au Diagnostic de la Machine Asynchrone Par Estimation Paramétrique"*, Thèse de Doctorat de L'université de Poitiers, France, Décembre 2002.

[25] D. Hadiouche, *"Contribution à l'étude de la machine asynchrone double étoile : modélisation, alimentation et structure"*, Thèse de Doctorat de l'Université Henri Poincaré, Nancy-I, France, Décembre 2001.

[26] T. M. Layadi, L. Abed et N. Khenfer, *" Structure Neuronale Hybride pour le Diagnostic des défauts statoriques de la Machine Asynchrone"*, 4th International Conference on Computer Integrated Manufacturing CIP'2007, November 2007.

[27] S. Pillement, R. David et O. Sentieys, *" Architectures reconfigurables : opportunités pour la faible consommation"*, article, Université de Rennes, France.

[28] A. Akhenak, *" Conception d'observateurs non linéaires par approche multimodèle : application au diagnostic "*, Thèse de Doctorat de l'Institut National Polytechnique de Lorraine, France, Décembre 2004.

[29] L.Baghli, *" Contribution à la commande de la machine asynchrone, utilisation de la logique floue, des réseaux de neurones et des algorithmes génétiques"*, Thèse de Doctorat de l'Université Henri Poincaré Nancy-I, France, Janvier 1999.

[30] J. Ph. Rennard, *"Réseaux Neuronaux, une introduction accompagnée d'un modèle Java"*, Edition Vuibert, Paris, 2006.

[31] K. Roy-Neogi and C. Sechen, *"Multiple FPGA Partitioning with Performance Optimization"*, IEEE Computer Society, Proceedings of the Third International ACM Symposium on Field-Programmable Gate Arrays (FPGA'95), 1995.

[32] K. Renovell, P. Faure, J.M. Portal, J. Figueras and Y. Zorian, *"IS-FPGA: A New Symmetric FPGA Architecture With Implicit SCAN"*, IEEE ITC International Test Conference, Paper 33.1, 0-7803-7169-0/2001.

[33] J. Ph. Mague, *"Explicitation des Connaissances d'un Réseau de Neurones Artificiels au Moyen d'un Moniteur Génétique"*, Rapport de stage Magistère d'informatique 1ère année, Université Joseph Fourier, Septembre 1999.

[34] V. Lemaire, *"Une nouvelle fonction de coût régularisante dans les réseaux de neurones artificiels : Application à l'estimation des temps de blocage dans un nœud ATM"*, Thèse de Doctorat de l'Université de PARIS VI, Paris, France, Septembre1999.

[35] H. El Ayech et A. Trabelsi, *"Les réseaux de neurones artificiels pour la prévision du trafic aérien de passagers. Construction et comparaison avec l'analyse Box-Jenkins"*, Institut Supérieur de Gestion de Tunis, Tunisie, mars 2003.

[36] M. L. Doumbia et A. Traoré, *" Modélisation et Simulation d'une Machine Asynchrone à Cage à l'Aide du Logiciel Matlab/Simulink"*, article, Montréal, Canada.

[37] A. Aïb, *" Conception et implémentation d'un algorithme de vision artificielle sur FPGA "*, P.F.E de l'École Nationale Polytechnique, Alger, Juin 2008.

[38] W. Merrouche et M. Mossi Idrissa, *" Implémentation d'un Modulateur OFDM sur un circuit FPGA "*, P.F.E de l'École Nationale Polytechnique, Alger, Juin 2007.

[39] A. Benmosbah et C. A. Mecheraoui, *" Implémentation sur FPGA des méthodes MPPT : « P&O » et « floue optimisée par les Algorithmes Génétiques » "*, P.F.E de l'École Nationale Polytechnique, Alger, Juin 2006.

[40] S. Areibi, G. Grewal, D. Banerji and P. Du *"Hierarchical FPGA Placement"*, CAN. J. ELECT. COMPUT. ENG., VOL. 32, NO. 1, Winter 2007.

[41] R. David, D. Lavenier et S. Pillement, *"Du micro-processeur au circuit FPGA une analyse sous l'angle de la reconfiguration"*, LASTI - ENSSAT / INRIA - R2D2, IRISA – CNRS, 2e soumission à Technique et science informatiques, Juillet 2003.

[42] Y. Thoma, *"Tissu Numérique Cellulaire À Routage et Configuration Dynamiques"*, Thèse de Doctorat de l'École Polytechnique Fédérale De Lausanne, Lausanne, France, 2005.

[43] J. Li et C-K. Cheng, *"Routability Improvement Using Dynamic Interconnect Architecture "*, IEEE, 0-8186-7086X/95, Dept. of Computer Sci. & Engr. University of California, San Diego, 1995.

[44] F. Li, Y. Lin and L. He, *"Power Modeling And Characteristics Of Field Programmable Gate Arrays"*, IEEE Transactions On Computer-Aided Design Of Integrated Circuits And Systems, Vol. 24, No. 11, November 2005.

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

OPTION: ELCTROMECANIQUE

Proposé et dirigé par : Dr. KHODJA Djalal Eddine

Présenté par: MEZAACHE Farouq
MENASRI Abderrachid

Thème :

IMPLEMENTATION DES "RNA" SUR "FPGA" POUR LE DIAGNOSTIC DES DEFAILLANCES DE LA MACHINE ASYNCHRONE EN UTILISANT LA CO-SIMULATION

Résumé :

Dans ce travail nous proposons l'implémentation des réseaux de neurones artificiels sur FPGA pour le diagnostic des défaillances de l'association moteur convertisseur et leur commande en utilisant la Co-simulation. Cette dernière qui fait partie des applications de « Hardware in the loop » permet d'approcher la simulation au cas réel. Les circuits FPGA 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 'Virtex4'. 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, Circuit reconfigurable, FPGA, Xilinx, Co-simulation.