

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



DÉPARTEMENT D'ÉLECTRONIQUE

Mémoire de fin d'études

En vue de l'obtention du diplôme d'Ingénieur d'Etat en Electronique

Thème :

**Implémentation sur FPGA d'une commande  
MLI à élimination des harmoniques pour un  
onduleur sept-niveaux en pont H cascadé**

Encadré par :

Pr. C. LARBES

Mr. A.GUELLAL

Réalisé par :

Mr. ZOGHBI Abderrezzaq

Mr. YAHIA-AMMAR Mohamed

**Promotion: Juin 2013**

## ملخص:

الهدف من هذا العمل هو التحكم في محول مستمر-متناوب متعدد المستويات المستخدم في السيارة الكهربائية، طريقة التحكم المستعملة هي تقنية تعديل عرض النبضة (MLI) مع حذف توافقيات الإشارة والتحكم في المركبة الأساسية، لهذا الغرض قمنا بمحاكاة التقنية المستعملة باستعمال البرنامجين PSIM و MATLAB من أجل إثبات نجاعتها في حساب زوايا التبديل بطريقة تضمن التحكم في سرعة المحرك اللاتزامني، بعد ذلك قمنا ببرمجة هذه التقنية على دائرة مبرمجة من نوع FPGA باستعمال لغة VHDL.

كلمات مفتاحية: تغيير السرعة، محرك لاتزامني، تعديل عرض النبضة، دراة مبرمجة، محول مستمر-متناوب متعدد المستويات.

## Résumé

L'objectif de ce mémoire est la commande d'un onduleur multi-niveaux utilisé dans la voiture électrique. La commande utilisée est une MLI calculée off-line à élimination d'harmoniques et asservissement du fondamental. C'est une commande scalaire avec V/f constant qui permet d'obtenir un couple maximal et constant, et de donner un bon spectre à la sortie de l'onduleur en éliminant les harmoniques indésirables. Dans ce mémoire nous avons commencé par simuler le système à réaliser dans l'environnement MATLAB et PSIM, puis nous avons implémenté l'algorithme MLI off-line sur un circuit FPGA SPARTAN 3E de XILINX, en utilisant le langage VHDL.

Mots clés: Variateur de vitesse, moteur asynchrone, MLI Elimination harmoniques, FPGA, VHDL, onduleur multiniveaux.

## Abstract

The aim of this work is the control of a multilevel inverter used in electrical vehicle. The technique used is an Off-Line selective harmonic elimination PWM, with fundamental control. It is a scalar technique which allows reducing torque pulsations and generating high-quality output spectra.

We have used a co-simulation based on MATLAB and PSIM to improve the performances of the HEPWM technique used. Then the Spartan-3E development card was used to implement the Off-Line algorithm in a XILINX FPGA, using the VHDL language.

Key words: variable speed drive, induction motor, PWM, harmonics elimination, FPGA,

VHDL, Voltage source multilevel inverter.

---

# REMERCIEMENTS

Nous remercions le bon Dieu de nous avoir accordé toute la patience, le courage, la volonté et la motivation qui nous ont permis d'achever ce travail.

Nous exprimons notre profonde gratitude, notre grand respect et notre sincère reconnaissance à notre promoteur le Pr. LARBES de l'Ecole Nationale Polytechnique pour avoir assumé la lourde responsabilité de nous encadrer, de nous avoir orienté et conseiller tout au long de ce travail ainsi pour la confiance qu'il nous a accordée.

Nous remercions également notre co-promoteur : Mr.GUELLAL du centre de développements des énergies renouvelables (CDER) pour ses précieux conseils, son suivi, sa disponibilité et son aide.

Nous remercions chaleureusement les membres du jury pour l'honneur qu'ils nous ont fait en acceptent d'évaluer notre projet.

Nous remercions également les gens du CDER pour leur aide.

Nous souhaitons aussi remercier tous les enseignants de l'Ecole Nationale Polytechnique d'Alger, et en particulier, Nos professeurs d'Électronique qui nous ont encadrés auparavant et tous nos enseignants pour les connaissances qu'ils nous ont transmis, leur disponibilité et leurs efforts.

Que tous ceux qui ont contribué de près ou de loin à la réalisation de ce modeste travail trouvent ici l'expression de notre sincère gratitude.

# Table des matières

<b>Introduction générale</b>	<b>11</b>
<b>I Généralités</b>	<b>13</b>
I.1 Composition et principe de fonctionnement du véhicule électrique . . . . .	13
I.2 Principales Configurations des Véhicules . . . . .	14
I.2.1 Le véhicule tout électrique . . . . .	14
I.2.2 Les véhicules hybrides . . . . .	14
I.3 Choix du moteur électrique . . . . .	16
I.4 Moteur asynchrone . . . . .	17
I.4.1 Définition et constitution . . . . .	17
I.4.2 Principe de fonctionnement . . . . .	17
I.4.3 Inversion du sens de rotation . . . . .	18
I.4.4 Commande de la machine asynchrone . . . . .	18
I.5 Commande V/F constant . . . . .	18
I.6 Les onduleurs de tension . . . . .	21
I.6.1 Définition et principe . . . . .	21
I.6.2 Défferents types d'onduleurs . . . . .	21
I.7 Les onduleurs multiniveaux . . . . .	22
I.7.1 Introduction . . . . .	22
I.7.2 Intérêt des onduleur multi-niveaux . . . . .	22
I.8 Les topologies des convertisseurs multiniveaux . . . . .	23
I.8.1 Introduction . . . . .	23
I.8.2 Onduleur à diode de bouclage (structure NPC) . . . . .	23
I.8.3 Onduleur à condensateur flottant . . . . .	24
I.8.4 Convertisseur en pont H cascadi . . . . .	25
I.9 Les méthodes de commande des onduleurs . . . . .	26
I.9.1 Introduction . . . . .	26
I.9.2 Classification des méthodes de commandes des onduleurs multi-niveaux . . . . .	27
I.9.3 La méthode MLI engendrée classique . . . . .	27
I.9.4 La méthode MLI véctorielle SVPWM . . . . .	28
I.9.5 Contrôle véctoriel de l'espace (SCV) . . . . .	28
I.9.6 La méthode d'élimination sélective des harmoniques . . . . .	29
<b>II La méthode d'élimination des harmoniques pour onduleurs multi-niveaux en pont H cascadiés</b>	<b>31</b>
II.1 L'onduleur en pont H cascadi . . . . .	31
II.1.1 Onduleur cascadié monophasé . . . . .	31
II.1.2 Structure triphasée . . . . .	33

II.2	Principe de la technique de PATEL et HOFT . . . . .	35
II.3	Série de fourier du signal de sortie . . . . .	35
II.4	Résolution du système . . . . .	37
II.4.1	Principe de la théorie Résultante . . . . .	37
II.4.2	Application de la méthode résultante pour l'élimination de deux harmoniques . . . . .	39
II.4.3	Les résultats des calculs . . . . .	43
II.4.4	Elimination des autres harmoniques . . . . .	45
II.5	Vérification des résultats par la simulation . . . . .	45
II.5.1	Schéma de simulation sous PSIM /SIMULINK . . . . .	45
II.5.2	Résultats de la simulation . . . . .	48
II.5.3	Conclusion . . . . .	54
 <b>III Implémentation de la commande sur FPGA</b>		<b>55</b>
III.1	Introduction . . . . .	55
III.2	Les circuits FPGA . . . . .	56
III.2.1	Définition . . . . .	56
III.2.2	Nomenclature des circuits FPGA . . . . .	57
III.2.3	Application des FPGA . . . . .	57
III.2.4	Architecture des FPGA . . . . .	57
III.2.5	Les blocs du circuits configurables . . . . .	59
III.2.6	La configuration en entrée et en sortie . . . . .	62
III.2.7	Les différents types d'interconnexions . . . . .	62
III.3	Configuration des FPGA . . . . .	65
III.4	Outils de développement des FPGA . . . . .	65
III.5	Langage VHDL . . . . .	66
III.5.1	Introduction . . . . .	66
III.5.2	Bref historique . . . . .	66
III.5.3	Utilité du VHDL . . . . .	66
III.5.4	Spécification . . . . .	67
III.5.5	Simulation . . . . .	67
III.5.6	Conception . . . . .	67
III.5.7	Unité de conception (module VHDL) . . . . .	68
III.5.8	Les deux modes de travail en VHDL . . . . .	70
III.6	Etapes de réalisation d'un projet autour d'un circuit FPGA . . . . .	71
III.6.1	Description . . . . .	71
III.6.2	Vérification des erreurs . . . . .	72
III.6.3	Synthèse . . . . .	72
III.6.4	Implémentation . . . . .	73
III.6.5	Configuration . . . . .	74
III.7	Implémentation de la commande sur un circuit FPGA . . . . .	74
III.7.1	présentation de la carte utilisé . . . . .	74
III.7.2	Description de l'algorithme implémenté . . . . .	76
III.7.3	L'impémentation de la commande sur la carte Spartan-3E . . . . .	80
III.7.4	Test des signaux de sorties . . . . .	83
III.8	Conclusion . . . . .	84
 <b>Conclusion</b>		<b>85</b>

Bibliographie

87

# Table des figures

I.1	Chaine de traction du véhicule électrique . . . . .	13
I.2	Voiture tout électrique . . . . .	14
I.3	Architecture du véhicule hybride série . . . . .	15
I.4	Architecture du véhicule hybride parallèle . . . . .	15
I.5	Architecture du véhicule hybride parallèle-série . . . . .	16
I.6	Rotor à cage d'écuruil . . . . .	17
I.7	shéma équivalent de la MAS[10] . . . . .	19
I.8	Variation du couple en fonction du glissement. . . . .	20
I.9	Caractéristique V/F[11]. . . . .	20
I.10	Schéma de l'onduleur demi-pont . . . . .	21
I.11	Schéma de principe d'un onduleur triphasé de tension . . . . .	22
I.12	Onduleur NPC multi niveaux[16] . . . . .	23
I.13	Convertisseur multi-niveaux avec condensateur flottant . . . . .	24
I.14	Simple pont H . . . . .	25
I.15	structure en pont H cascadié a M niveaux . . . . .	26
I.16	MLI triangolo-sinosoidale avec multi porteuses :(a)porteuses et signal de référence (b)signal de sortie[13] . . . . .	27
I.17	Diagramme vectoriel de l'espace :(a)3 niveaux (b)5 niveaux (c) 7 niveaux[13] . . . . .	28
I.18	Méthode SVC à 21 niveaux [13] . . . . .	29
II.1	Structure d'un onduleur monophasé à M niveaux . . . . .	32
II.2	La forme de la tension d'un onduleur à 7 niveaux . . . . .	33
II.3	structure de l'onduleur triphasé à 7 niveaux . . . . .	33
II.4	Les formes des tensions de phase et de ligne . . . . .	34
II.5	La tension de sortie désirée de l'onduleur . . . . .	35
II.6	Les angles de commutation (deg) en fonction de l'indice de modulation m pour l'élimination des harmonique 5 et 7 . . . . .	43
II.7	Les angles de commutation (deg) en fonction du taux de modulation r pour l'élimination des harmonique 5 et 7 . . . . .	44
II.8	Les angles de commutation calculés par la méthode de Newton-Raphson [31] . . . . .	44
II.9	Shéma bloc de la commande en SIMULINK . . . . .	46
II.10	Schéma de l'onduleur et de la charge en PSIM . . . . .	47
II.11	Les signaux de commandes des six switchs . . . . .	48
II.12	Les formes des tensions de $v_{an}$ et $v_{ab}$ à vide . . . . .	49
II.13	Les spectres fréquentielles des signaux à vide . . . . .	49
II.14	Les formes des tension de $v_{an}$ , $v_{ab}$ et du courant $i_1$ . . . . .	50
II.15	Les spectres fréquentiel de $v_{an}$ , $v_{ab}$ et $i_1$ . . . . .	50
II.16	Les formes des tension de $v_{an}$ , $v_{ab}$ , $i_1$ pour r=0.35 . . . . .	51

II.17 Les spectres fréquentiel des signaux pour $r=0.35$ . . . . .	51
II.18 Les formes des tensions de $v_{an}, v_{ab}, i_1$ pour $r=0.61$ . . . . .	52
II.19 Les spectres fréquentiel des signaux pour $r=0.61$ . . . . .	52
II.20 les formes des tension de $v_{an}, v_{ab}, i_1$ pour $r=0.8$ . . . . .	53
II.21 les spectres fréquentiel des signaux pour $r=0.8$ . . . . .	53
III.1 Domaines d'utilisation des DSPs et des FPGAs[12] . . . . .	56
III.2 Architecture interne d'un FPGA (circuit configurable)[32] . . . . .	58
III.3 Réseaux d'interconnexions configurables . . . . .	58
III.4 Structure d'un cellule SRAM . . . . .	59
III.5 Les entrés/sorties du CLB . . . . .	59
III.6 Cellules logiques(CLB)[27] . . . . .	60
III.7 Input Output Block (IOB)[10] . . . . .	61
III.8 Connexions à usage général et matrice de commutation . . . . .	63
III.9 Les interconnexions directes . . . . .	64
III.10 Les lignes longues . . . . .	64
III.11 Exemple d'un graphe des états . . . . .	65
III.12 Etapes de réalisation d'un projet sur circuit FPGA (design flow) . .	72
III.13 carte de développement utilisée . . . . .	74
III.14 Organigramme du diviseur de fréquence . . . . .	77
III.15 Le code VHDL du programme du diviseur de fréquence . . . . .	78
III.16 Le résultat de la simulation du diviseur . . . . .	78
III.17 Organigramme du programme de génération des signaux . . . . .	79
III.18 Résultat de la simulation pour $r= 0.35$ ( $f= 17.5$ Hz) . . . . .	79
III.19 résultat de la simulation pour $r=0.61$ ( $f= 30.5$ Hz) . . . . .	80
III.20 Résultat de la simulation pour $r=0.8$ ( $f= 40$ Hz) . . . . .	80
III.21 Résultat de la simulation pour $r=1$ ( $f= 50$ Hz) . . . . .	80
III.22 La structure du programme qui donne les signaux de commande . .	81
III.23 Plan ahead des pins . . . . .	81
III.24 Le tableau des contraintes . . . . .	82
III.25 Présentation du circuit . . . . .	82
III.26 Le fichier .ucf . . . . .	82
III.27 Oscilloscope utilisé pour la visualisation . . . . .	83
III.28 Les signaux de commande pour MLI $r=0.35$ . . . . .	83
III.29 Les signaux de commande MLI pour $r=0.61$ . . . . .	84
III.30 Les signaux de commande MLI pour $r=0.8$ . . . . .	84
III.31 Les signaux de commande MLI pour $r=1$ . . . . .	84



# Introduction générale

Le moteur à courant continu est généralement utilisé pour la variation de vitesse mais il présente beaucoup d'inconvénients : coût élevé, présence d'un collecteur, entretien fréquent.

Afin d'éviter ces inconvénients on utilise de plus en plus, le moteur asynchrone à vitesse variable. Il est robuste, de construction facile, simple, de coût réduit et faible encombrement. Cependant la variation de vitesse du moteur asynchrone nécessite dans le cas de la commande statorique, une source d'alimentation alternative variable en amplitude et en fréquence.

Pour cela on utilise un onduleur de tension variable en tension et en fréquence avec le rapport  $V/F$  constant et à Modulation de Largeur d'Impulsion (MLI).

Les onduleurs les plus connus jusqu'ici sont les onduleurs à deux niveaux. Toutefois, certaines applications exigent des variateurs asynchrones triphasés fonctionnant à des puissances et/ou vitesses très élevées. Pour monter en puissance et en tension, on associe généralement plusieurs onduleurs en séries ou en parallèles, d'où une complication dans la commande et une augmentation du coût du système.

Pour remédier à ces inconvénients, la solution naturelle consiste à réaliser une association particulière des éléments conducteurs de manière à réduire la tension à commuter en des valeurs plus petites et directement commutables par les semi-conducteurs actuels. Les onduleurs multiniveaux permettent d'augmenter la tension de sortie des convertisseurs statiques au-delà des limites des semi-conducteurs[2].

Les onduleurs multiniveaux, malgré leurs qualités, qu'ont pu atteindre grâce au développement de l'électronique de puissance, leur fonctionnement est accompagné de pulsations de couple et d'échauffement du moteur dus à la présence d'harmoniques dans le signal de sortie de l'onduleur. Ces pulsations de couple deviennent très gênantes aux faibles vitesses.

Pour résoudre ces problèmes on doit utiliser la commande MLI à élimination sélective des harmoniques. Autrement dit, on doit calculer les angles de commutation avec une plus grande précision. Cette commande nécessite la résolution d'un système d'équations non linéaires, Ce système peut être résolu par des méthodes itératives, Cependant ces méthodes nécessitent une bonne estimation des valeurs initiales et ne donnent pas tout les résultats possibles.

---

Dans ce mémoire une méthode appelé "Résultante" a été proposée pour résoudre le système d'équations non linéaires avec une très grande précision, ainsi cette méthode permet d'avoir tous les résultats possibles du système.

Un autre objectif de notre projet est d'implémenter une commande MLI pour un onduleur multiniveaux avec asservissement du fondamental et élimination harmonique sélective, telle que définie par la technique de Patel et Hoft. Ainsi notre mémoire est organisé en quatre chapitres :

Le premier chapitre introduit des généralités, il présente une vue d'ensemble sur les véhicules électriques, machines asynchrones et les onduleurs multiniveaux.

Le deuxième chapitre, est consacré aux techniques de modulations et on s'intéresse surtout à la technique MLI 'programmée', avec élimination d'harmonique sélective et asservissement du fondamental, de Patel et Hoft. On calcule les angles exacts de commutation par une méthode analytique la Résultante. Pour vérifier l'algorithme une simulation a été faite dans l'environnement MATLAB, SIMULINK et en utilisant le logiciel PSIM.

Dans le troisième chapitre, on aborde les circuits FPGA et on s'intéresse à leurs architectures et leurs caractéristiques. Ainsi On fait une description de la carte de développement SPARTAN E3 qu'on utilise pour l'implémentation, et on termine le chapitre par une explication du langage de description VHDL. A la fin du chapitre on aborde l'implémentation de la commande MLI pour l'onduleur multiniveaux sur la carte SPARTAN E3. On présente les programmes VHDL, on les simule à l'aide du ModelSim(ou Isim) puis on les visualise sur un oscilloscope. Après la réalisation nous avons consacré une partie aux relevés des résultats et à leurs interprétations.

Une conclusion générale termine ce mémoire.

# Chapitre I

## Généralités

---

Le véhicule électrique représente actuellement un regain d'intérêt car il présente l'avantage d'être non polluant. Sa vocation urbaine est d'autre part renforcée par le fait qu'il présente un agrément de conduite évident : silence et absence de passage de vitesse[1].

### I.1 Composition et principe de fonctionnement du véhicule électrique

le véhicule électrique contient deux éléments majeurs : une batterie et un moteur électrique. La batterie fournit l'électricité nécessaire au fonctionnement du moteur. Elle est reliée au moteur électrique avec un convertisseur commandé de façon à régler l'intensité du courant qui actionne le moteur. Le convertisseur utilisé dépend du type du moteur utilisé, qui peut être à courant continu ou alternatif.

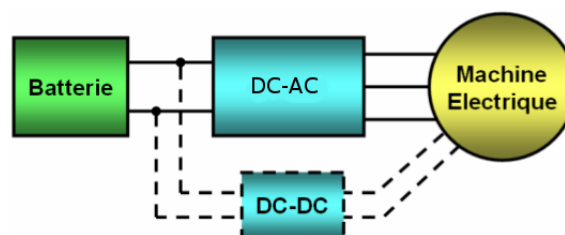


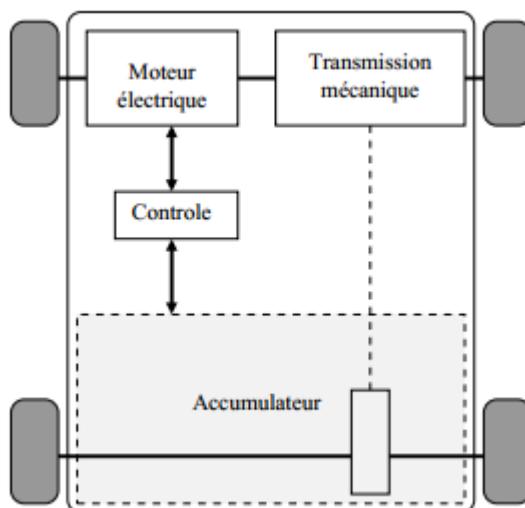
Figure I.1 – Chaîne de traction du véhicule électrique

### I.2 Principales Configurations des Véhicules

Selon le type d'énergie embarqué on distingue deux grandes familles des véhicules électriques : le véhicule tout électrique et le véhicule hybride [3].

#### I.2.1 Le véhicule tout électrique

Il s'agit d'un véhicule qui possède uniquement un accumulateur comme source d'énergie [4]. La structure est donnée par le schéma de la figure I.2.



**Figure I.2** – Voiture tout électrique

Les véhicules tout électriques (EV) proposés actuellement sont exclusivement urbains, leur autonomie est comprise entre 70 et 120 km avec des technologies de batteries relativement conventionnelles (plomb-acide et cadmium-nikel) et 150 à 200 km avec des technologies plus avancées (nikel-métal-hydrure et lithium). Le freinage récupératif permet d'accroître sensiblement l'autonomie, surtout en cycle urbain (d'environ 20%). Il permet en outre d'obtenir un frein moteur. Pour ces raisons, il est nécessaire d'avoir un système de conversion réversible [5].

#### I.2.2 Les véhicules hybrides

Les véhicules hybrides (HEV) utilisent des sources d'énergie différentes pour fournir la force motrice. Généralement, ils combinent un moteur à combustion interne et une ou plusieurs machines électriques (moteur/générateur) associées à un système de stockage. En comparaison avec un véhicule électrique de même dimension et de même puissance, un véhicule hybride est plus spacieux, plus léger et offre plus d'autonomie.

Par rapport aux véhicules conventionnels, les véhicules hybrides consomment moins de carburant et émettent donc moins de polluants [6]. On distingue alors trois concepts de véhicule hybride :

## I.2. PRINCIPALES CONFIGURATIONS DES VÉHICULES

### a. Hybride série

C'est configuration la plus simple des structures des HEV. La sortie mécanique du moteur à combustion est convertie en électricité par un alternateur. Cette énergie produite peut alimenter le moteur électrique ou recharger la batterie.

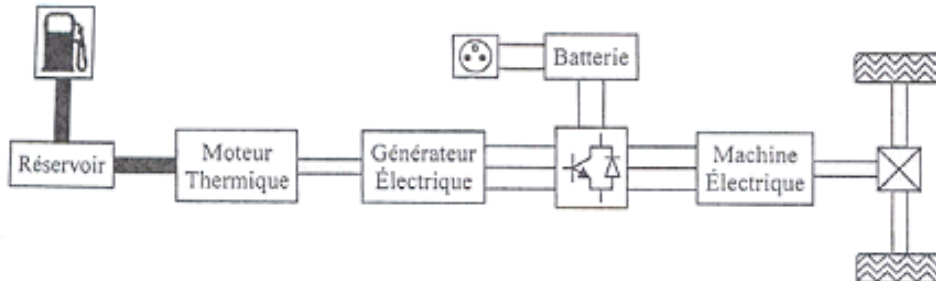


Figure I.3 – Architecture du véhicule hybride série

### b. Hybride parallèle

Dans cette configuration, le moteur à combustion et le moteur électrique sont couplés au système de transmission, de sorte qu'ils peuvent travailler séparément.

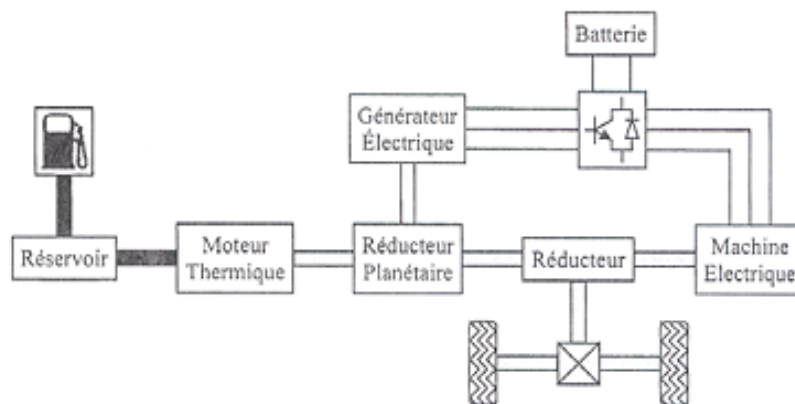


Figure I.4 – Architecture du véhicule hybride parallèle

### c. Le bimode (série-parallèle)

Dans cette configuration, un générateur est ajouté entre le moteur et le convertisseur de puissance. La méthode de contrôle dans cette configuration est plus compliquée que dans les configurations précédentes. Les modes de fonctionnement dans cette configuration est divisée en deux groupes, électrique lourd où le moteur électrique est plus actif et moteur lourd lorsque le interne Moteur à combustion est plus actif dans l'opération.

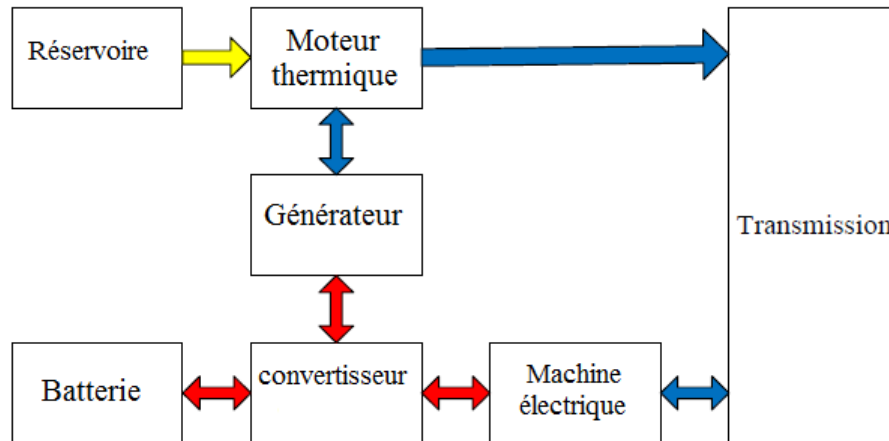


Figure I.5 – Architecture du véhicule hybride parallèle-série

### I.3 Choix du moteur électrique

Le choix du moteur influe directement sur les performances du véhicule électrique. A ce stade trois grandes catégories de moteurs sont disponibles : le moteur à courant continu, le moteur asynchrone (MAS) et le moteur synchrone.

La solution de référence a été pendant longtemps la motorisation à courant continu [7], mais l'utilisation de ce type de moteur électrique pose plusieurs problèmes :

- rendement limité (90%).
- vitesse de rotation élevée.
- pertes thermiques situées au rotor, donc difficiles à évacuer (l'échauffement fait encore plus diminuer le rendement).
- usure des balais.

Les problèmes cités précédemment et l'évolution de l'électronique de puissance et des matériaux tels que les aimants permanents ont conduit aujourd'hui à s'orienter vers des solutions plus performantes telles que les motorisations synchrones ou asynchrone.

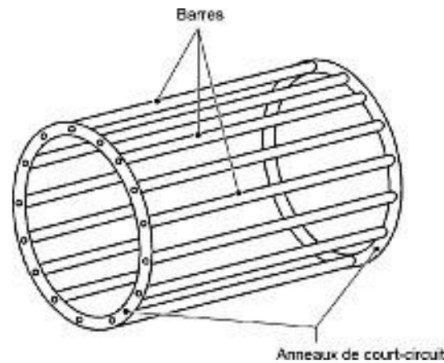
Le moteur synchrone a l'avantage d'avoir un très bon rendement (pas de glissement) mais il est plus délicat à piloter et potentiellement moins robuste.

Le MAS est le candidat le plus adapté pour propulser les véhicules hybrides électriques (HEV) [8] [3], car il est très fiable (aucun entretien), facile à produire très robuste en particulier le moteur asynchrone à cage d'écurie qui est facilement industrialisable et il a une puissance massique assez élevée, ce qui conduit à un bon rendement pour la chaîne de traction.

## I.4 Moteur asynchrone

### I.4.1 Définition et constitution

Un moteur asynchrone est une machine à  $2p$  poles, alimentée à partir du réseau alternatif de fréquence  $f$ , et qui ne tourne pas exactement à la vitesse synchrone  $N_s$  du champ tournant. Il est constitué d'une partie fixe, le stator qui comporte le bobinage et d'une partie rotative, le rotor qui est soit bobiné, soit en cage d'écuréuil selon le type du moteur asynchrone (figure I.6) [9].



**Figure I.6** – Rotor à cage d'écuréuil

### I.4.2 Principe de fonctionnement

Le moteur asynchrone à cage est le moteur le plus répandu dans l'industrie. Il est robuste, fiable, normalisé, économique, disponible et peu encombrant.

Son principe est le suivant :

Lorsque les 3 bobines du stator sont parcourues par des courants alternatifs de fréquence  $f$  (50 Hz) décalés électriquement, le stator produit un champ magnétique tournant à la fréquence de synchronisme  $N_s$ .

$$N_s = \frac{60f}{p} [tr/min] \quad (I.1)$$

et la vitesse angulaire synchrone :

$$\Omega = \frac{\omega}{p} \quad (I.2)$$

Ce flux balayant les enroulements rotoriques y induit des f.e.m., donc des courants puisque ces bobinages sont en court-circuit. Si le rotor tournait à la vitesse de synchronisme  $N_s$ , le flux à travers ses enroulements ne varierait plus, d'où absence de courant rotorique et de couple. Le moteur tourne à une vitesse  $N$  d'autant plus inférieure à  $N_s$  que le couple demandé sur l'arbre est important.

On appelle glissement l'écart des vitesses angulaires synchrone  $\Omega$  et réelle  $\Omega'$  rapporté à la vitesse synchrone  $\Omega$  :

$$g = \frac{\Omega - \Omega'}{\Omega} = \frac{\omega - \omega'}{\omega} = \frac{N_s - N}{N_s} \quad (\text{I.3})$$

Avec :

$$N_s = \frac{\Omega}{2\pi} \quad (\text{I.4})$$

et

$$N = \frac{\Omega'}{2\pi} (t/s) \quad (\text{I.5})$$

### I.4.3 Inversion du sens de rotation

Le sens de rotation du moteur asynchrone triphasé s'inverse en inversant le sens de rotation du flux tournant statorique. Il suffit pour cela de permuter les entrées de deux phases. L'inversion du sens du flux tournant pendant la rotation du moteur entraîne d'abord le freinage puis l'arrêt et enfin la rotation en sens inverse du moteur.

### I.4.4 Commande de la machine asynchrone

**Commande scalaire :**

Il y a deux types de commande scalaire, en agissant sur le courant ou sur la tension. La commande scalaire la plus utilisée est la commande en V/F constant qu'on va détailler dans une section plus tard.

**Commande vectoriel**

La Commande vectorielle est un terme générique désignant l'ensemble des commandes tenant compte en temps réel des équations du système qu'elle commande. Le nom de ces commandes vient du fait que les relations finales sont vectorielles à la différence des commandes scalaires. Les relations ainsi obtenues sont bien plus complexes que celles des commandes scalaires, mais en contrepartie elles permettent d'obtenir de meilleures performances lors des régimes transitoires. Il existe des commandes vectorielles pour tous les moteurs à courant alternatif.

À chaque période de fonctionnement de l'onduleur, la commande doit ouvrir ou fermer les interrupteurs de puissance (IGBT ou autre) de manière à créer dans la machine électrique un champ magnétique résultant dont le module et la direction sont optimaux pour répondre aux consignes de vitesse et de couple.

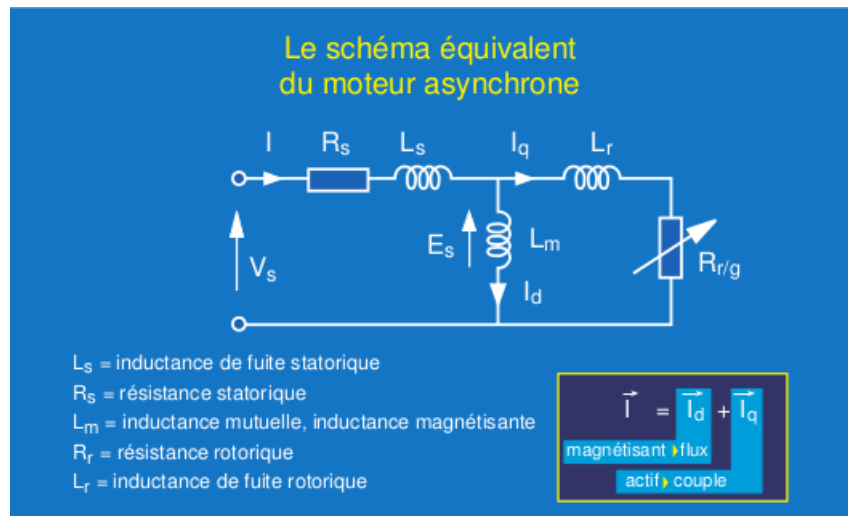
## I.5 Commande V/F constant

C'est la commande scalaire la plus utilisée, son principe est de maintenir  $V/f = \text{Constant}$  ce qui signifie de garder le flux constant.

La figure II.14 représente le schéma équivalent par phase d'une machine asynchrone. Le flux est créé par le courant circulant dans l'inductance magnétisante  $L_m$ .



Les performances optimales du moteur sont obtenues si le flux, et donc le courant magnétisant, est maintenu sensiblement constant sur toute la gamme de vitesse.



**Figure I.7** – schéma équivalent de la MAS[10]

Le courant magnétisant peut être calculé par l'expression :

$$I_d = \frac{E_s}{L_m \omega} \quad (\text{I.6})$$

avec

$$E_s = V_s - (R_s + L_s \omega) I \quad (\text{I.7})$$

En négligeant

$$(R_s + L_s \omega) I \quad (\text{I.8})$$

on obtient :

$$I_d = \frac{V_s}{\omega} \frac{I}{L_m} = \frac{V_s}{f} \frac{I}{2\pi L_m} \quad (\text{I.9})$$

Le courant magnétisant peut donc être maintenu constant en maintenant ce rapport sensiblement constant.

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

$$C_{max} = \frac{3p}{2N_r'} \left( \frac{V_s}{\omega_s} \right)^2 \quad (\text{I.10})$$

On voit bien que le couple est directement proportionnel au carré du rapport de la tension sur la fréquence statorique.

En maintenant ce rapport constant et en jouant sur la fréquence statorique, on déplace la courbe du couple électromagnétique (en régime quasi-statique) de la machine asynchrone. (figure I.8)

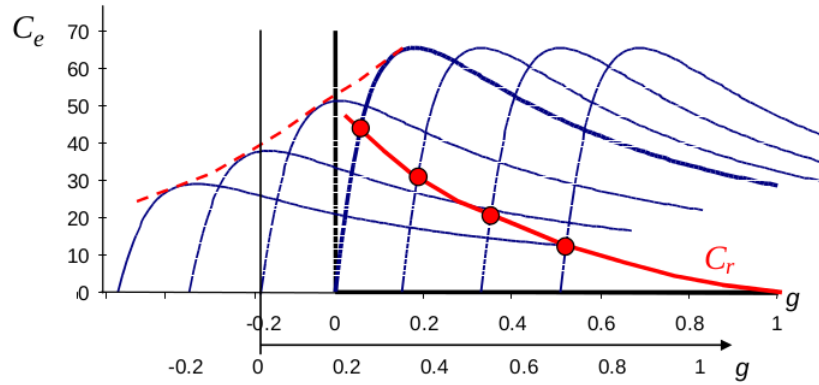


Figure I.8 – Variation du couple en fonction du glissement.

Pour la régulation de vitesse la figure suivante représente la caractéristique  $V/f$  du moteur, qui est composé de trois intervalles : comme le présente la figure I.9

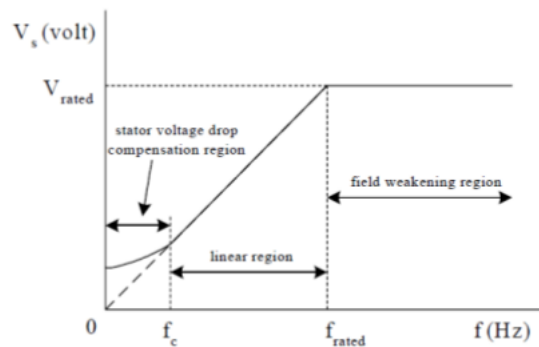


Figure I.9 – Caractéristique  $V/F$ [11].

- **Entre  $0 - f_c$  Hz :** une tension est nécessaire, de sorte que la chute de tension aux bornes de la résistance statorique peut pas être négligée et doit être compensée par une augmentation de la  $V_s$ . Ainsi, le profil en  $V/F$  n'est pas linéaire. La fréquence de coupure ( $f_c$ ) et les tensions de stator appropriés peuvent être calculées analytiquement du circuit équivalent en régime permanent avec  $R_s \neq 0$ .
- **Entre Hz  $f_c - f_{rated}$  :** il suit la relation  $V/F$  constant. La pente représente réellement l'air la quantité de flux d'entrefer..
- **Aux fréquences supérieures à  $f_{rated}$  :** le rapport  $V_s/F$  constant ne peut être satisfaite car les tensions statoriques serait limitées à la valeur nominale, afin d'éviter une rupture d'isolation à des enroulements de stator. Par conséquent, le flux d'entrefer résultant serait réduit, et cela va inévitablement causer la décroissance du couple développé en conséquence. Cette région est généralement dite "Région fieldweakening". Pour cela, le principe  $V/F$  constant est également violé à ces fréquences.[11]

## I.6 Les onduleurs de tension

### I.6.1 Définition et principe

Un onduleur est un dispositif permettant de transformer en alternatif une énergie électrique de type continue. Il est utilisé pour [12] :

- Fournir des tensions ou courants alternatifs de fréquence et amplitudes variables.

**Exemple :** C'est le cas des onduleurs servant à alimenter des moteurs à courant alternatif devant tourner à vitesse variable par exemple (la vitesse est liée à la fréquence des courants qui traversent la machine).

- Fournir une ou des tensions alternatives de fréquence et d'amplitude fixes.

**Exemple :** C'est le cas en particulier des alimentations de sécurité destinées à se substituer au réseau en cas de défaillance de celui-ci par exemple. L'énergie stockée dans les batteries de secours est restituée sous forme continue, l'onduleur est alors nécessaire pour recréer la forme de tension et fréquence du réseau. On distingue les onduleurs de tension et les onduleurs de courant, en fonction de la source d'entrée continue : source de tension ou source de courant. La technologie des onduleurs de tension est la plus maîtrisée et est présente dans la plupart des systèmes industriels, dans toutes les gammes de puissance (de quelques Watts à plusieurs MW).

### I.6.2 Différents types d'onduleurs

#### Les Onduleurs monophasés de tension

Ce type d'onduleurs est destiné à alimenter des charges alternatives monophasées.

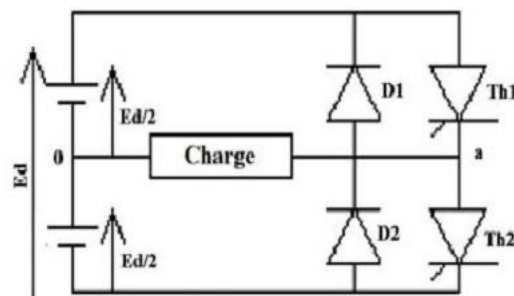


Figure I.10 – Schéma de l'onduleur demi-pont

#### Les onduleurs triphasés

Les onduleurs monophasés sont utilisés pour des applications de faible puissance, alors que les onduleurs triphasés couvrent la gamme des moyennes et des fortes puissances. L'objectif de cette topologie est de fournir une source de tension triphasée, dont l'amplitude, la phase et la fréquence sont contrôlables [12].

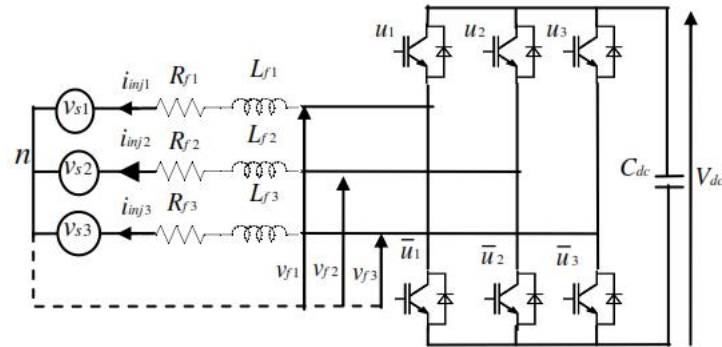


Figure I.11 – Schéma de principe d'un onduleur triphasé de tension

## I.7 Les onduleurs multiniveaux

### I.7.1 Introduction

Au début des années 90, l'évolution des propriétés des semi-conducteurs de puissance a stimulé la croissance du marché des convertisseurs dédiés aux applications moyenne tension/moyenne et forte puissance allant de quelques kilowatts à plusieurs mégawatts dans la gamme du kilovolt.

Cette augmentation incessante des niveaux de puissance mis en jeu s'est traduit par une augmentation des niveaux de tension et de courant des convertisseurs, et a conduit à l'apparition et le perfectionnement de nouveaux composants de puissance tels que les MOSFET, GTO, IGBT et IGCT et le développement des structures de conversion multiniveaux dont le principe fondamental est essentiellement basé sur une combinaison parallèle/série des composants de puissance.

### I.7.2 Intérêt des onduleur multi-niveaux

Deux motivations principales sont à l'origine des onduleurs multiniveaux, à savoir d'une part l'augmentation de puissance par le biais de la génération de tensions plus élevées, au-delà de celles compatibles avec les tensions de blocage des dispositifs à semi-conducteurs de puissance.

D'autre part, on cherche à obtenir des grandeurs de sortie ayant une meilleure définition, c'est-à-dire qui présentent un contenu harmonique réduit.

D'autres avantages peuvent être cités[13] :

- La tension commutée est réduite à la valeur du pas du convertisseur, c'est-à-dire à la valeur de tension bloquée par ses interrupteurs. Cela réduit d'autant les pertes par commutation. D'autre part, la fréquence de pulsation de chacun de ses éléments est plus basse que la fréquence de pulsation apparente de la tension appliquée à la charge. Cela autorise une augmentation de cette fréquence de pulsation qui permet une réduction de la dimension des filtres ou une amélioration de la qualité des signaux filtrés.
- En terme de production, selon la topologie choisie, il est possible de réaliser un convertisseur modulaire composé de modules identiques. Le nombre de modules peut éventuellement être adapté à la tension de service du convertisseur.

## I.8 Les topologies des convertisseurs multiniveaux

### I.8.1 Introduction

Il existe trois principaux types de convertisseurs multiniveaux : diode de bouclage, condensateur flottant, et en ponts en H cascadié [14]. Les avantages et les inconvénients détaillés des trois convertisseurs multiniveaux seront abordés dans ce qui suit.

### I.8.2 Onduleur à diode de bouclage (structure NPC)

La première topologie la plus pratique d'onduleur de tension multi-niveaux est le NPC (Neutral-Point-Clamped). Elle a été proposée, la première fois en 1981, par Nabae et Al[15].

L'Onduleur NPC est constitué de deux paires de commutateurs séries (supérieure et inférieure) en parallèle avec deux condensateurs en série où l'anode de la diode supérieure est reliée au point milieu (neutre) des condensateurs et sa cathode au point milieu de la paire supérieure de commutateurs, la cathode de la diode inférieure est connectée au point milieu des condensateurs et divise la tension continue en de plus petites tensions principal, comme illustré par la figure I.12 .

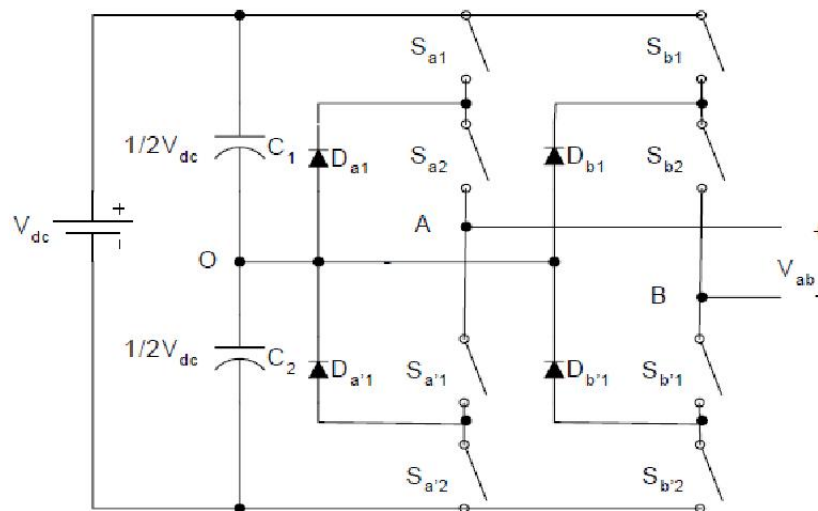


Figure I.12 – Onduleur NPC multi niveaux[16]

Dans cet exemple, la tension continue principale est divisée en deux. Si le point O est pris comme référence à la masse, les trois sorties de tension de phase possibles sont  $V_{dc}$ , 0, ou  $-V_{dc}$ .

Pour générer une tension triphasée, trois phases sont nécessaires.

Certains inconvénients des structures NPC peuvent être observés.

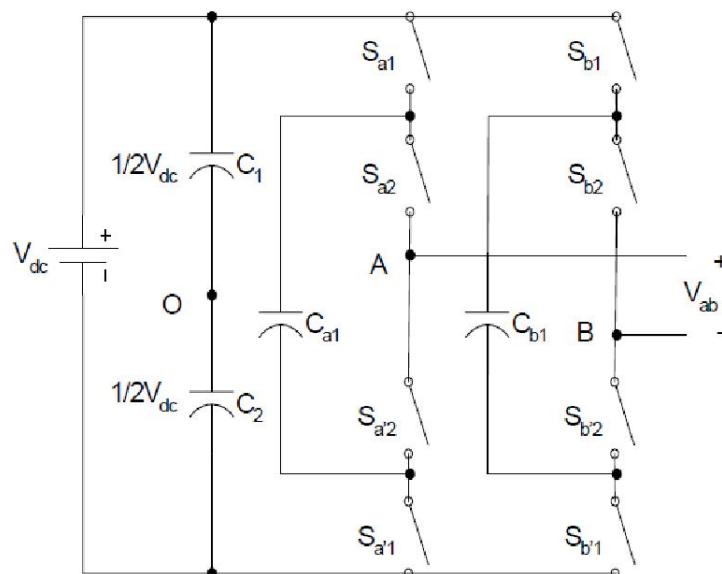
- L'utilisation des diodes supplémentaires en série devient impraticable lorsque le nombre de niveaux  $M$  augmente, exigeant  $(m-1) \times (m-2)$  diodes par phase si toutes les diodes ont des tensions de blocage égales. Notez que les tensions des diodes dans différentes positions ne sont pas équilibrées. Par exemple, la diode  $D_{a2}$  doit bloquer deux tensions de condensateur.  $D_{a(m-2)}$  doit bloquer  $(m-2)$  tensions de condensateur.
- En outre, le cycle de commutation est différent pour certains des interrupteurs nécessitant différents calibres. En outre, les condensateurs ne partagent pas la même décharge ou courant de charge entraînant un déséquilibre de tension des condensateurs en série. Le déséquilibre de tension de condensateur peut être contrôlée en utilisant une topologie dos-à-dos, La connexion des résistances en parallèle avec les condensateurs, ou à l'aide des états redondants de tension [15].

Néanmoins cette topologie possède les avantages suivantes :

- Un grand nombre de niveaux donne une petite distorsion harmonique.
- Toutes les phases partagent le même bus DC.
- Flux de puissance réactive peut être contrôlée.
- Le contrôle est simple.

### I.8.3 Onduleur à condensateur flottant

La topologie de l'onduleur multi-niveaux à condensateur flottant (flying capacitor multilevel inverter), donnée par la Fig.I.13, a été proposée en 1992. Elle est considérée comme l'alternative la plus sérieuse à la topologie de l'onduleur NPC. L'avantage de cette topologie est qu'elle élimine le problème des diodes de bouclage présent dans les topologies des onduleurs NPC multi-niveaux[17].



**Figure I.13** – Convertisseur multi-niveaux avec condensateur flottant

En plus, cette topologie limite naturellement les contraintes en tension imposées aux composants de puissance (faible valeur de  $\frac{dv}{dt}$  aux bornes des composants) et introduit des états de commutation additionnels qui peuvent être utilisés pour aider à maintenir l'équilibre des charges dans les condensateurs.

La topologie de l'onduleur à condensateur flottant a assez d'états de commutation pour contrôler l'équilibre des charges dans chaque bras d'onduleur ayant n'importe quel nombre de niveaux, ce qui n'est pas le cas dans l'onduleur NPC.

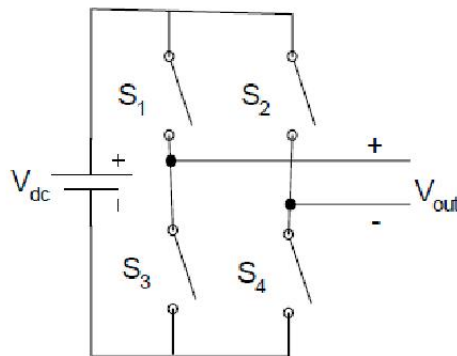
Actuellement il semble que cette topologie a quelques inconvénients. Néanmoins, quelques points faibles doivent toujours être explorés :

- le contrôle de la charge du condensateur ajoute la complexité au contrôle du circuit entier.
- la topologie de l'onduleur à condensateur flottant à multi-niveaux peut exiger plus de condensateurs que la topologie de l'onduleur NPC. De plus, il est évident que des courants de grandes valeurs efficaces circuleront à travers ces condensateurs.
- il y a un potentiel de résonance parasite entre les condensateurs découplés[13].

### I.8.4 Convertisseur en pont H cascadié

Un convertisseur en pont H cascadié ( Cascaded H-bridge Converter) est constitué de plusieurs ponts H en configuration série [18], [19], [20].

Un seul pont en H est montré à la figure I.14.

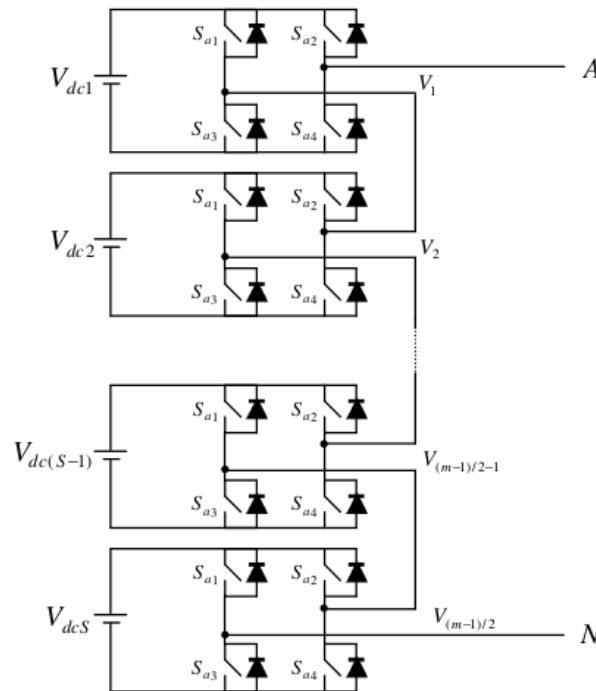


**Figure I.14** – Simple pont H

Un seul pont en H est un convertisseur à trois niveaux. Les quatre interrupteurs  $S_1$ ,  $S_2$ ,  $S_3$  et  $S_4$  sont contrôlés afin de générer trois sorties discrètes  $V_{out}$  ayant des niveaux  $V_{dc}$ , 0 et  $-V_{dc}$ .

Lorsque  $S_1$  et  $S_4$  sont allumés, la sortie est  $V_{dc}$  lorsque  $S_2$  et  $S_3$  sont allumés, la sortie est  $-V_{dc}$ . Lorsque la paire  $S_1$  et  $S_2$  paire ou  $S_3$  et  $S_4$  sont allumés, la sortie est 0.

Afin de générer une tension de sortie à M niveaux il suffit de connecter en cascade  $(M - 1)/2$  ponts monophasés complets . Chaque pont a sa propre alimentation à courant continu, comme le montre la figure I.15[2]



**Figure I.15** – structure en pont H cascadié a M niveaux

Les avantages des onduleurs multi-niveaux en cascade sont les suivants[13]

- La structure de la série permet, une configuration de circuit modulaire évolutive et l’emballage en raison de la structure identique de chaque pont en H.
- Pas de diodes de blocage supplémentaires ou de condensateurs d’équilibrage de tensions
- Redondance pour des niveaux de tension interne est possible parce que la tension de phase est la somme de la production de chaque pont.
- La commande de l’onduleur est simple.

L’inconvénient des convertisseur multi-niveaux en pont H cascadiés est qu’il a besoin de sources de courant continu distincts.

## I.9 Les méthodes de commande des onduleurs

### I.9.1 Introduction

L’objectif principal de techniques de commande, appliquées aux onduleurs, est de permettre l’obtention d’ondes de tension alternatives, d’amplitude et de fréquence fondamentale réglables, en éliminant ou en repoussant le plus loin possible les composantes harmoniques parasites résultant du découpage. Quelle que soit la forme de l’onde alternative recherchée (le plus souvent sinusoïdale), l’établissement de la stratégie de commande devra, prendre en compte la façon dont ces techniques vont pouvoir s’insérer dans les boucles de contrôle et de régulation qui sont obligatoirement présentes dans toutes les applications des onduleurs.[21]



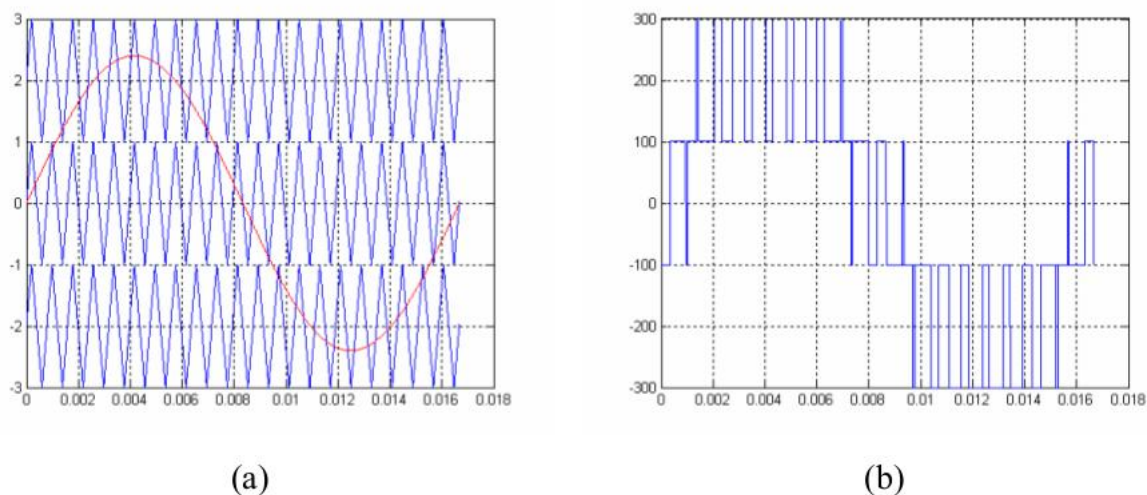
### I.9.2 Classification des méthodes de commandes des onduleurs multi-niveaux

Les méthodes de modulation employées dans les convertisseurs multiniveaux peuvent être classées selon la fréquence de commutation [14], [22], [23] :

- Les méthodes qui fonctionnent avec des hautes fréquences de commutation, représentées par la MLI engendrée classique (sinusoïdale avec porteuse) et la MLI vectoriel.
- Les méthodes qui fonctionnent avec des basses fréquences de commutation, représenté par la méthode d'élimination des harmoniques (selective harmonic elimination) et la méthode de modulation par vecteur d'espace (space vector modulation).

### I.9.3 La méthode MLI engendrée classique

La commande MLI Triangulo-sinusoidale consiste à comparer une valeur de tension de référence de fréquence  $F_r$ , image du signal souhaité à la sortie appelée modulante, à une porteuse triangulaire ou en dent de scie de fréquence  $F_p$ . Pour les onduleur multi-niveaux des porteuses multiples doivent être utilisées. Chaque source de DC a besoin de sa propre porteuse. Les points d'intersection entre la modulante et les porteuses engendrent l'enclenchement/déclenchement constituant ainsi une impulsion de durée variable et l'ensemble de ces impulsions reconstitue, de ce fait, le fondamental de la sinusoïde de référence[20][13]. Ceci peut être montré sur la figure I.16 (a). La tension de sortie de l'onduleur est montrée sur la figure I.16 (b).

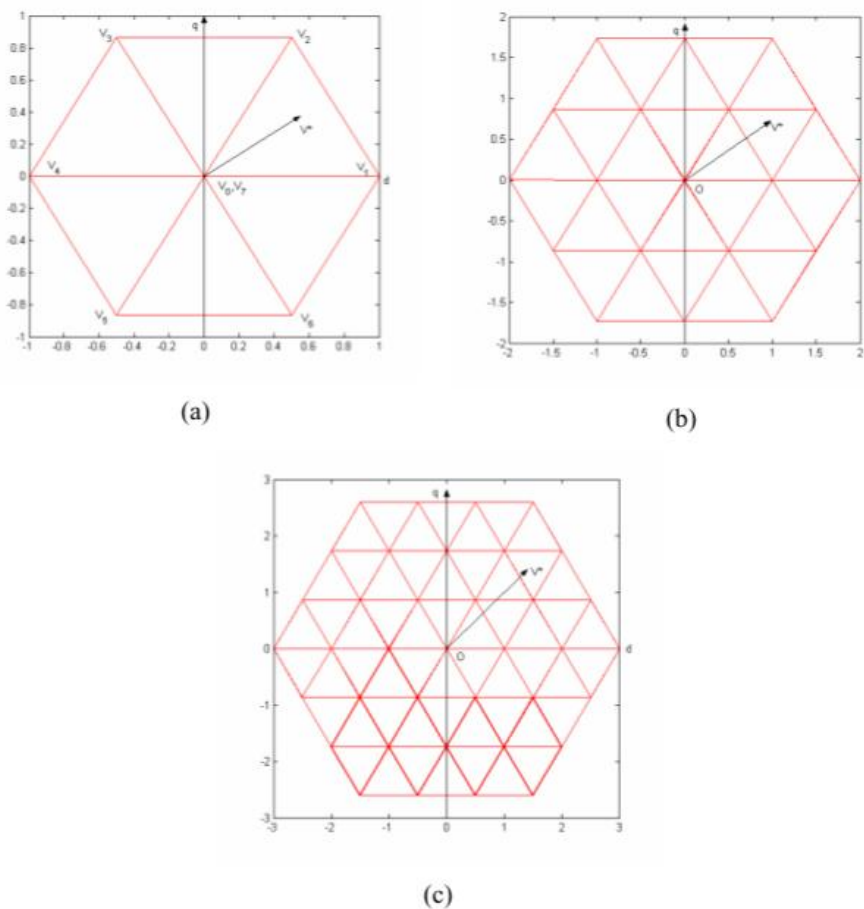


**Figure I.16** – MLI triangulo-sinusoidale avec multi porteuses : (a) porteuses et signal de référence (b) signal de sortie[13]

### I.9.4 La méthode MLI vectorielle SVPWM

La technique de modulation du vecteur d'espace PWM est populaire pour la commande de convertisseur à deux niveaux. Elle peut être prolongée aux convertisseurs multiniveaux. [24] – [25] La technique de modulation vectorielle est une technique numérique pour laquelle l'objectif est de générer une onde MLI de tension  $v(t)$  à la sortie de l'onduleur dont la valeur moyenne sur chaque période de découpage  $T$  est égale à celle de tension sinusoïdale de référence  $V_r(t)$  sur cette période. Ceci est effectué à chaque période d'échantillonnage en sélectionnant les états appropriés des interrupteurs parmi la table d'excitation de l'onduleur à deux niveaux et en déterminant la durée d'application de chacun des états.

la figure I.17. montre les vecteurs de l'espace pour les onduleurs à trois, cinq, sept niveaux de tension.



**Figure I.17** – Diagramme vectoriel de l'espace :(a)3 niveaux (b)5 niveaux (c) 7 niveaux[13]

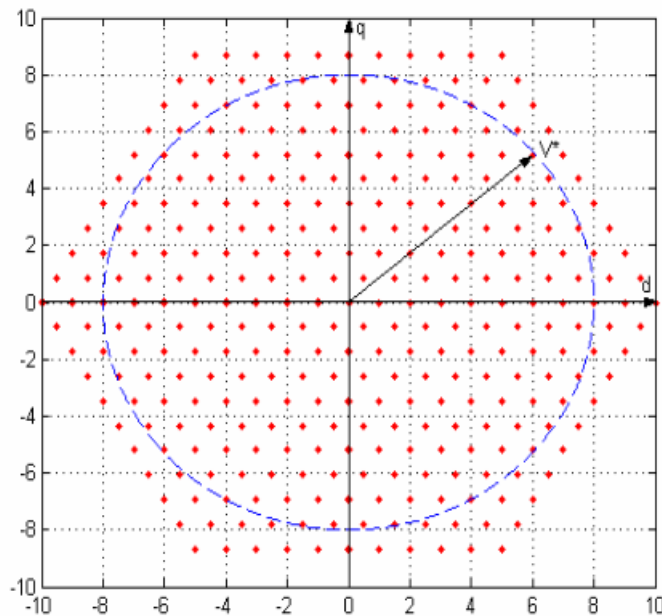
### I.9.5 Contrôle vectoriel de l'espace (SCV)

C'est un procédé de commande conceptuellement différent pour les convertisseurs multi-niveaux basé sur la théorie du vecteur spatial, (SVC)[23]. cette méthode travaille avec des basses fréquences de commutation et ne génère pas la valeur moyenne de la tension désirée à chaque point de commutation, comme le

fait la SVPWM.

la Figure I.18. montre les 311 différents vecteurs de l'espace générés par un convertisseur 21-niveaux. Le vecteur de tension de référence est également inclus dans ce chiffre.

L'idée principale de la méthode SVC est d'offrir à la charge un vecteur de tension qui minimise l'erreur ou la distance d'espace entre ce vecteur et le vecteur de référence. Donc en augmentant la densité des vecteurs on minimise (en augmentant le nombre de niveaux) on va bien minimiser les erreurs. Cette méthode est simple et attrayante pour un nombre élevé de niveaux. Les avantages de SVC sont que le calcul est très simple et la fréquence de commutation est très faible, proche de celle de la méthode d'élimination des harmoniques. Mais l'inconvénient de cette méthode est que plus le nombre de niveaux diminue, l'erreur en termes de vecteurs générés par rapport à la référence sera plus grande [14].



**Figure I.18** – Méthode SVC à 21 niveaux [13]

### I.9.6 La méthode d'élimination sélective des harmoniques

Cette technique est une modulation MLI calculée ou programmée, Elle consiste à calculer les instants de commutation (séquences de fonctionnement) de manière à répondre à certains critères portant sur le spectre fréquentiel de l'onde délivrée par l'onduleur. Ces séquences de fonctionnement sont alors mémorisées et restituées cycliquement pour assurer la commande des interrupteurs. Les critères usuellement retenus sont : l'élimination d'harmoniques de rangs spécifiés ou l'élimination d'harmoniques dans une bande de fréquences spécifiée [27].

Le principe de cette méthode est basée sur l'algorithme de Patel et Hoft. Dans cette technique, il est possible d'asservir le fondamental de la tension MLI et d'anuler les amplitudes des  $(m-1)$  premiers harmoniques.[26]

Pour calculer les angles exacts de commutation  $\alpha_1, \dots, \alpha_m$ , on doit résoudre un système de  $m$  équations non linéaires à  $m$  inconnues  $\alpha_1, \dots, \alpha_m$ .

En conclusion, on peut dire que la technique MLI "programmée" présente de nombreux avantages :

- Asservissement de la tension  $V$  du fondamental
- Variation de la fréquence  $f$  du fondamental en utilisant la relation de conversion d'une valeur angulaire en valeur temporelle :  $\alpha = 2\pi ft$ .
- Élimination des  $(m-1)$  premiers harmoniques.

Ces avantages permettent de remplacer l'alimentation sinusoïdale idéale avec une alimentation pratique ayant un taux d'harmoniques que l'on peut réduire à volonté. La technique MLI programmée avec asservissement du fondamental et élimination harmonique sera décrite en détail et appliquée sur l'onduleur multiniveaux cascade dans le chapitre suivant.

# Chapitre

# II

## La méthode d'élimination des harmoniques pour onduleurs multi-niveaux en pont H cascades

---

Dans le chapitre précédent, on a présenté les différentes topologies et les méthodes de commande des onduleurs multi-niveaux, Dans ce qui suivent, la théorie d'élimination des harmoniques sera discutée et appliquée à la commande d'un onduleur multi-niveau cascade.

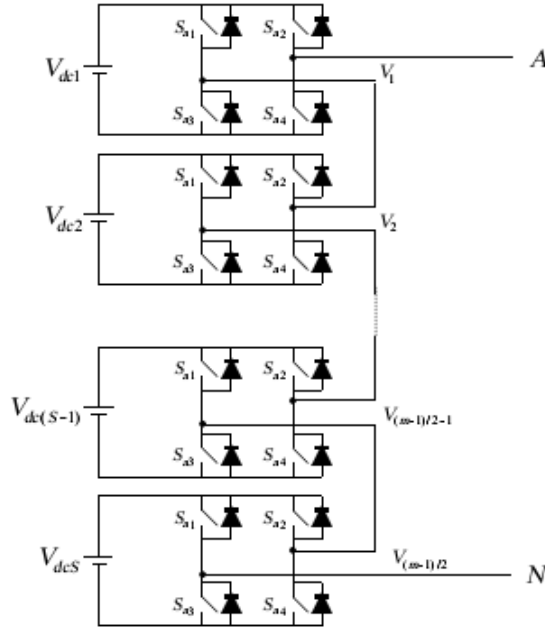
### II.1 L'onduleur en pont H cascade

#### II.1.1 Onduleur cascade monophasé

Pour synthétiser la forme d'onde d'un onduleur multi-niveaux cascade, les sorties des cellules en pont H sont reliées en série. La forme d'onde synthétisée de tension est, donc la somme des sorties de chaque cellule.

## II.1. L'ONDULEUR EN PONT H CASCADÉ

Le nombre de niveaux de tension de phase d'un onduleur multi-niveaux cascadi est défini par :  $M = 2s + 1$ . Ou  $s$  est le nombre de sources de tension DC (c'est le nombre de cellules). Par exemple, une forme d'onde de tension de phase d'un onduleur à neuf-niveaux peut être obtenue avec quatre sources DC séparées et quatre cellules en pont H. La figure II.1 montre le schéma d'une phase de l'onduleur à  $M$  niveaux.



**Figure II.1** – Structure d'un onduleur monophasé à  $M$  niveaux

De la figure II.1, la tension de phase est la somme des sorties de chaque pont H et est donnée par :

$$V_{an} = V_{dc1} + V_{dc2} + \dots + V_{dc(s-1)} + V_{dc(s)} \quad (\text{II.1})$$

Puisque la tension nulle est commune pour toutes les sorties de l'onduleur, le nombre total de niveaux de l'onduleur devient  $2s + 1$ .

Une forme d'onde de tension de phase d'un onduleur à sept-niveaux cascadi, et les formes de tension de chaque cellule en pont H sont montrées dans la figure II.2. Dans ce mémoire, on assume que toutes les tensions DC sont égales.

## II.1. L'ONDULEUR EN PONT H CASCADÉ

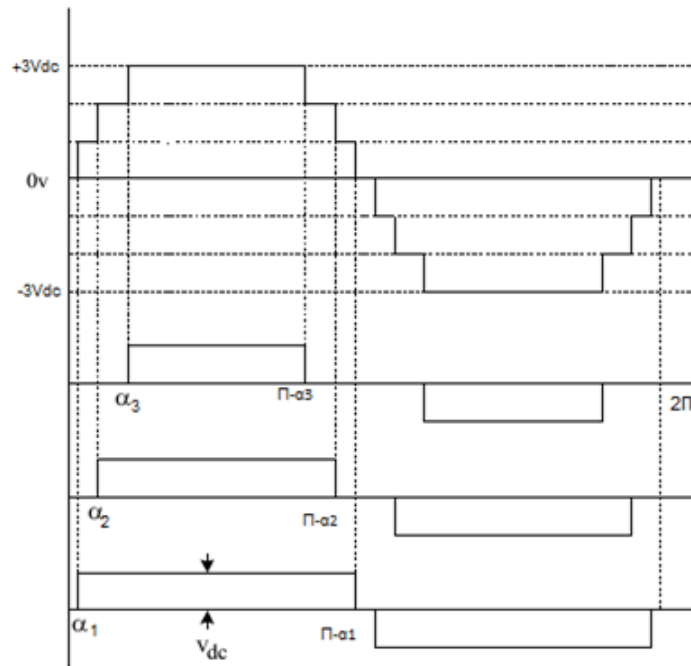


Figure II.2 – La forme de la tension d'un onduleur à 7 niveaux

### II.1.2 Structure triphasée

Pour un système triphasé, trois structures monophasés sont combinées, Soit en étoile ou en triangle. La figue II.3 illustre le schéma de principe de l'onduleur à sept niveaux en utilisant trois cellules en pont H et trois sources de tension séparées par phase.

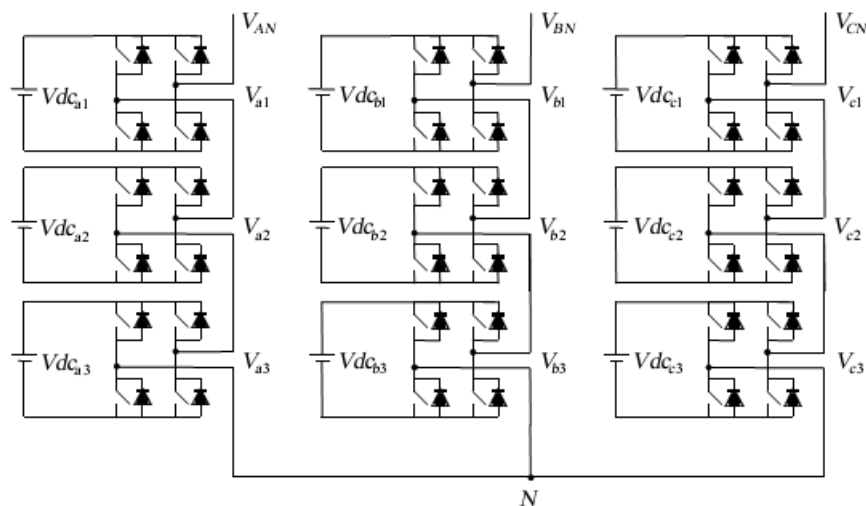


Figure II.3 – structure de l'onduleur triphasé à 7 niveaux

## II.1. L'ONDULEUR EN PONT H CASCADÉ

De la figure II.3, la tension  $V_{an}$  de la phase A est la somme de  $V_{a1}$ ,  $V_{a2}$ , et  $V_{a3}$ . Elle a la forme d'une tension simple (de phase). Les tensions des autres phase B et C sont similaires à celle de la phase A sauf qu'elles sont décalées de 120 degrés pour chaque phase.

La tension entre phases ( de ligne) peut être obtenue par la relation :

$$V_{ab} = V_{an} - V_{bn}$$

ou  $V_{an}$  avec la tension de la phase a et  $V_{bn}$  et la tension de la phase b.

Théoriquement, le nombre maximum de niveaux dans la tension de ligne est  $2M-1$ , où  $M$  est le nombre de niveaux de tension de phase. Le nombre de niveaux de la tension de la tension entre phase dépend de l'indice de modulation et des harmoniques à éliminer. Dans notre exemple, l'onduleur cascadié à sept niveaux peut avoir une tension de ligne jusqu'à 13 niveaux.

L'avantage du système triphasé est que toutes les harmoniques multiple de trois dans la tension de ligne sont éliminés, donc on doit éliminés seulement les autres harmoniques impaire : 5,7 . La figure II.4 montre la tension de sortie de la phase A, la tension de la phase B  $V_{bn}$  et la tension de ligne (entre phases)  $V_{ab}$ , de l'onduleur cascadié à 7 niveaux présentés dans la figure II.3.

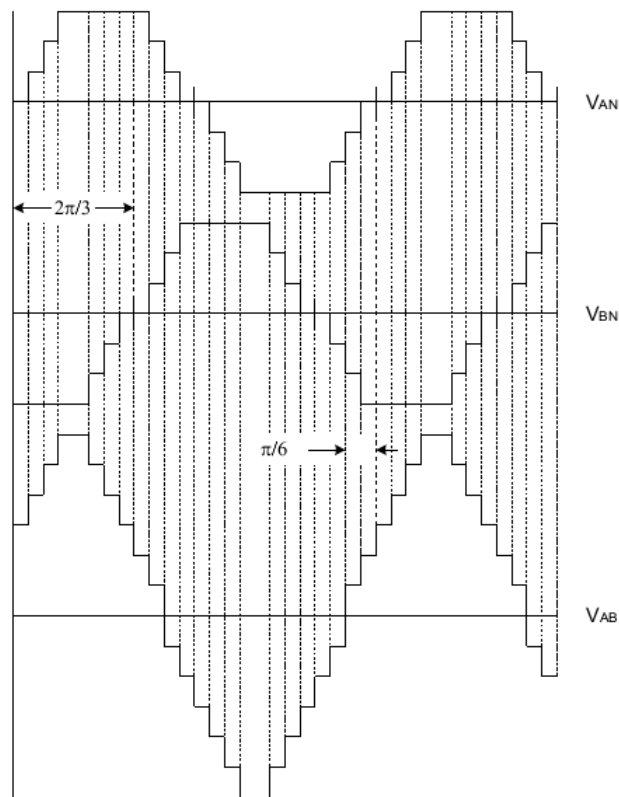


Figure II.4 – Les formes des tensions de phase et de ligne



## II.2 Principe de la technique de PATEL et HOFT

La tension à la sortie des onduleurs multi-niveaux est formée de plusieurs créneaux en tension continue, dont les instants de commutation sont calculés directement par la MLI utilisée. Le principe de la stratégie d'élimination d'harmoniques consiste à imposer ces instants tel que le spectre d'harmoniques de la tension simple de l'onduleur ne contient pas (C-1) harmoniques indésirables. Pour cela, il faut d'abord imposer un motif de tension de bras contenant C angles de commutation par le quart de la période afin d'éliminer (C-1) harmoniques. Ensuite, extraire l'expression générale de l'amplitude des harmoniques en fonction de C angles de commutation. Ceci est obtenu par la décomposition en série de Fourier de la tension du bras [27].

Le motif adéquat de la tension du bras de l'onduleur à sept niveaux dans le cas où trois harmoniques sont éliminés est donné par la figure II.5

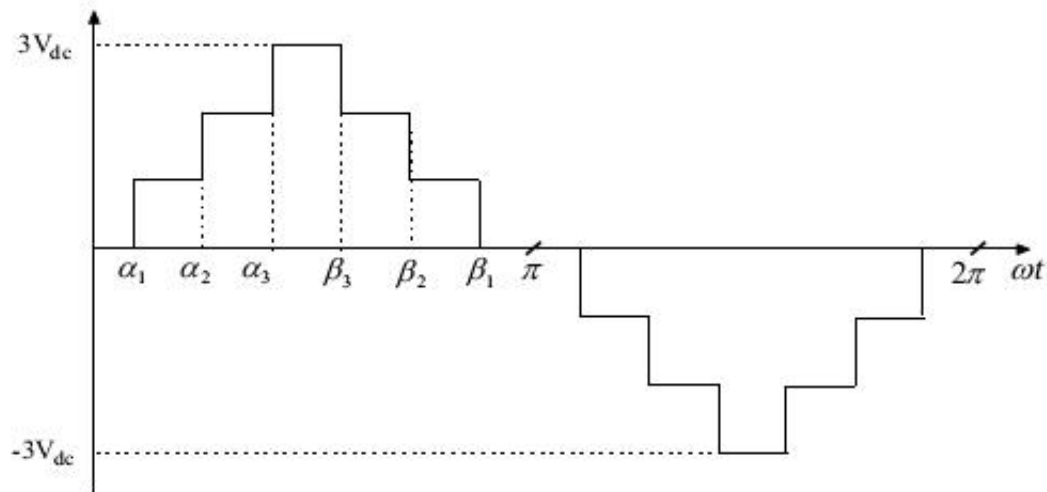


Figure II.5 – La tension de sortie désirée de l'onduleur

## II.3 Série de fourier du signal de sortie

La tension de sortie de l'onduleur triphasé à sept niveaux est représentée dans la Figure II.5. Les angles de commutation  $\alpha_1, \alpha_2, \alpha_3$  définissent les transitions entre les différents niveaux de tension.

Puisque la tension de sortie  $f(\omega t)$  est périodique, donc on peut la développer en série de Fourier tel que  $f(\alpha) = f(\omega t)$  :

$$f(\alpha) = a_0 + \sum_{n=1}^{\infty} (a_n \cos(n\alpha) + b_n \sin(n\beta)) \quad (\text{II.2})$$

Les coefficients  $a_n$  et  $b_n$  sont donnés par :

$$a_0 = \frac{1}{2\pi} \int_0^{2\pi} f(\alpha) d\alpha \quad (\text{II.3})$$

$$a_n = \frac{1}{\pi} \int_0^{2\pi} f(\alpha) \sin(n\alpha) d\alpha \quad (\text{II.4})$$

### II.3. SÉRIE DE FOURIER DU SIGNAL DE SORTIE

$$b_n = \frac{1}{\pi} \int_0^{2\pi} f(\alpha) \cos(n\alpha) d\alpha \quad (\text{II.5})$$

tq  $n = 1, 2, 3, \dots$

D'autre part comme  $f(\alpha)$  présente une symétrie demi-onde :  $f(\alpha + \pi) = -f(\alpha)$  La valeur moyenne  $a_0$  est nulle et seulement les harmoniques impairs existent. Par conséquent, l'indice  $n$  prend les valeurs impaires 1, 3, 5, 7, 9, ...

La forme de la tension de sortie de la figure II.5 présente une symétrie quart d'onde et demi-onde. D'où les harmoniques pairs sont nuls. De plus la fonction  $f(\omega t)$  est impaire. Les coefficients  $a_n$  de la série de Fourier sont nuls. Il en résulte, que le développement en série de Fourier peut se faire sur le quart de la période.

En effet le coefficient  $b_n$  s'écrit comme suit :

$$\begin{cases} b_n = 0 & \text{si } n \text{ pair} \\ b_n = \frac{4}{\pi} \int_0^{\frac{\pi}{2}} f(\alpha) \cos(n\alpha) d\alpha & \text{si } n \text{ impair} \end{cases} \quad (\text{II.6})$$

Où  $n$  représente le rang de l'harmonique  $n = 1, 3, 5, 7, \dots$

Donc l'expression II.6 peut s'écrire de la manière suivante :

$$b_n = \frac{4}{\pi} \int_{\alpha_1}^{\alpha_2} (U_c) \cos(n\alpha) d\alpha + \frac{4}{\pi} \int_{\alpha_2}^{\alpha_3} (2U_c) \cos(n\alpha) d\alpha + \frac{4}{\pi} \int_{\alpha_3}^{\frac{\pi}{2}} (3U_c) \cos(n\alpha) d\alpha \quad (\text{II.7})$$

Les paramètres  $\alpha_1 \alpha_2 \alpha_3$  représentent les angles de commutation à déterminer.

Sachant que  $\cos(n\frac{\pi}{2}) = 0$  pour  $n$  impair, l'équation II.7 se réduit donc à :

$$b_n = \frac{4U_c}{n\pi} [\cos(n\alpha_1) + \cos(n\alpha_2) + \cos(n\alpha_3)] \quad (\text{II.8})$$

En remplaçant  $n$  dans l'expression II.8, on abouti à un système d'équations non linéaires suivant :

$$\begin{cases} \cos(\alpha_1) + \cos(\alpha_2) + \cos(\alpha_3) = \frac{3\pi r}{4} \\ \cos(5\alpha_1) + \cos(5\alpha_2) + \cos(5\alpha_3) = 0 \\ \cos(7\alpha_1) + \cos(7\alpha_2) + \cos(7\alpha_3) = 0 \end{cases} \quad (\text{II.9})$$

Où  $r$  représente le taux de modulation.

Pour simplifier l'équation on prend  $m = \frac{3\pi r}{4}$  = l'indice de modulation, et donc le system II.9 devient :

$$\begin{cases} \cos(\alpha_1) + \cos(\alpha_2) + \cos(\alpha_3) = m \\ \cos(5\alpha_1) + \cos(5\alpha_2) + \cos(5\alpha_3) = 0 \\ \cos(7\alpha_1) + \cos(7\alpha_2) + \cos(7\alpha_3) = 0 \end{cases} \quad (\text{II.10})$$

On a abouti à un système de 3 équations non-linéaires à 3 inconnues  $\alpha_1, \alpha_2, \alpha_3$  en fonction de l'indice de modulation m.

## II.4 Résolution du système

Le système II.10 contient C équations non linéaires à C inconnues en fonction de l'indice de modulation m. La détermination des  $\alpha_i$  est effectuée par le principe de la théorie résultante et les polynômes symétriques.

### II.4.1 Principe de la théorie Résultante

On suppose qu'on a les deux polynômes suivants en fonction de  $x_1$  et  $x_2$  :

$$a(x_1, x_2) = a_3(x_1)x_2^3 + a_2(x_1)x_2^2 + a_1(x_1)x_2 + a_0(x_1) = 0 \quad (\text{II.11})$$

et

$$b(x_1, x_2) = b_3(x_1)x_2^3 + b_2(x_1)x_2^2 + b_1(x_1)x_2 + b_0(x_1) = 0 \quad (\text{II.12})$$

Où  $a_3(x_1), a_2(x_1), a_1(x_1), a_0(x_1), b_3(x_1), b_2(x_1), b_1(x_1), et b_0(x_1)$  : sont des polynômes en  $x_1$

ainsi, on définit les deux polynômes en  $x_1$  et  $x_2$  :

$$\alpha(x_1, x_2) = \alpha_2(x_1)x_2^2 + \alpha_1(x_1)x_2 + \alpha_0(x_1) \quad (\text{II.13})$$

et

$$\beta(x_1, x_2) = \beta_2(x_1)x_2^2 + \beta_1(x_1)x_2 + \beta_0(x_1) \quad (\text{II.14})$$

Où  $\alpha_2(x_1), \alpha_1(x_1), \alpha_0(x_1), \beta_2(x_1), \beta_1(x_1)$  ,et  $\beta_0(x_1)$  sont des polynomes en  $x_1$  .

Donnant une valeur à  $x_1$  on peut voir des équations de II.11 au II.14 :

$a(x_1, x_2)$  et  $b(x_1, x_2)$  ne sont pas premier entre eux si et seulement si il existe un polynôme  $\alpha(x_1, x_2)$  et  $\beta(x_1, x_2)$  tel que :

$$\frac{a(x_1, x_2)}{b(x_1, x_2)} = \frac{\alpha(x_1, x_2)}{\beta(x_1, x_2)} \quad (\text{II.15})$$

Or :

$$b(x_1, x_2)\alpha(x_1, x_2) - a(x_1, x_2)\beta(x_1, x_2) = 0 \quad (\text{II.16})$$

si cette équation est écrite sous forme matricielle :

$$S(x_1)V(x_1) = \vec{0} \quad (\text{II.17})$$

Où les coefficients de  $x_2, k = 0, 1, \dots, 5$ , sont égale à zero car cette équation est vérifié pour chaque  $x_2$ . L'équation matricielle résultante est la suivante[13]

$$\begin{bmatrix} b_0(x_1) & a_0(x_1) & 0 & 0 & 0 & 0 \\ b_1(x_1) & a_1(x_1) & b_0(x_1) & a_0(x_1) & 0 & 0 \\ b_2(x_1) & a_2(x_1) & b_1(x_1) & a_1(x_1) & b_0(x_1) & a_0(x_1) \\ b_3(x_1) & a_3(x_1) & b_2(x_1) & a_2(x_1) & b_1(x_1) & a_1(x_1) \\ 0 & 0 & b_3(x_1) & a_3(x_1) & b_2(x_1) & a_2(x_1) \\ 0 & 0 & 0 & 0 & b_3(x_1) & a_3(x_1) \end{bmatrix} \times \begin{bmatrix} -\alpha_0(x_1) \\ \beta_0(x_1) \\ -\alpha_1(x_1) \\ \beta_1(x_1) \\ -\alpha_2(x_1) \\ \beta_2(x_1) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (\text{II.18})$$

la matrice carré  $6 \times 6$   $S(x_1)$  dans II.18 est connue par la matrice Résultante de SYLVESTER.

En général, la matrice Résultante de SYLVESTER va être de dimensions  $(d_1 + d_2) \times (d_1 + d_2)$ , où  $d_1$  et  $d_2$  sont les degrés des deux polynômes  $a(x_1, x_2)$  et  $b(x_1, x_2)$  en  $x_2$ , respectivement.

D'après l'équation II.14, en donnant une valeur particulière à  $x_1$ , la seule solution du system II.18 est la solution de l'équation  $V(x_1) = \vec{0}$  si  $a(x_1, x_2)$  et  $b(x_1, x_2)$  sont premier entre eux, sinon le determinant de  $S(x_1)$  est zéro :

$$R(x_1) = \det \left( \begin{bmatrix} b_0(x_1) & a_0(x_1) & 0 & 0 & 0 & 0 \\ b_1(x_1) & a_1(x_1) & b_0(x_1) & a_0(x_1) & 0 & 0 \\ b_2(x_1) & a_2(x_1) & b_1(x_1) & a_1(x_1) & b_0(x_1) & a_0(x_1) \\ b_3(x_1) & a_3(x_1) & b_2(x_1) & a_2(x_1) & b_1(x_1) & a_1(x_1) \\ 0 & 0 & b_3(x_1) & a_3(x_1) & b_2(x_1) & a_2(x_1) \\ 0 & 0 & 0 & 0 & b_3(x_1) & a_3(x_1) \end{bmatrix} \right) = 0 \quad (\text{II.19})$$

$R(x_1)$ , est fonction seulement de  $x_1$ , est le polynôme résultant.

Les solutions de II.19 peuvent être trouvées puis remplacées dans l'équation :

$$a(x_1, x_2) = 0 \quad (\text{II.20})$$

pour avoir les solutions de II.20.

Les solutions communes sont les solutions des deux équations II.11 et II.12.[13]  
[28]

## II.4.2 Application de la méthode résultante pour l'élimination de deux harmoniques

Ceci revient à résoudre le système II.10 de trois équations. En premier lieu, on pose le changement de variable suivant :

$$x_1 = \cos(\alpha_1), x_2 = \cos(\alpha_2), x_3 = \cos(\alpha_3) \quad (\text{II.21})$$

puis on exprime les termes  $\cos(5\alpha_i), \cos(7\alpha_i)$  en fonction des  $\cos(\alpha_i)$  seulement.

$$\cos(5\alpha_i) = 5\cos(\alpha_i) - 20\cos^3(\alpha_i) + 16\cos^5(\alpha_i) \quad (\text{II.22})$$

$$\cos(7\alpha_i) = -7\cos(\alpha_i) + 56\cos^3(\alpha_i) - 112\cos^5(\alpha_i) + 64\cos^7(\alpha_i) \quad (\text{II.23})$$

En remplaçant les équations de II.21 au II.23 dans le système II.10 on aura :

$$\begin{cases} p_1(x_1, x_2, x_3) = 0 = x_1 + x_2 + x_3 - m \\ p_5(x_1, x_2, x_3) = 0 = \sum_{i=1}^3 (5x_i - 20x_i^3 + 16x_i^5) \\ p_7(x_1, x_2, x_3) = 0 = \sum_{i=1}^3 (-7x_i + 56x_i^3 - 112x_i^5 + 64x_i^7) \end{cases} \quad (\text{II.24})$$

avec :

$$x_i = (x_1, x_2, x_3)$$

Les solutions doivent vérifier la condition suivante :

$$0 \leq \alpha_1 \leq \alpha_2 \leq \alpha_3 \leq \frac{\pi}{2} \quad (\text{II.25})$$

c.a.d

$$0 \leq x_3 \leq x_2 \leq x_1 \leq 1 \quad (\text{II.26})$$

Utilisant le polynôme  $p_1$  pour exprimer  $x_1$  en fonction des deux autres variables :

$$x_1 = m - x_2 - x_3 \quad (\text{II.27})$$

En remplaçant l'équation II.27 dans  $p_5$  et  $p_7$  :

$$\begin{cases} p_5(x_2, x_3) = 5(m - x_2 - x_3) - 20(m - x_2 - x_3)^3 + 16(m - x_2 - x_3)^5 + 5x_2 - 20x_2^3 + 16x_2^5 \\ \quad + 5x_3 - 20x_3^3 + 16x_3^5 \\ p_7(x_2, x_3) = -7(m - x_2 - x_3) + 56(m - x_2 - x_3)^3 - 112(m - x_2 - x_3)^5 + 64(m - x_2 - x_3)^7 \\ \quad + -7x_2 + 56x_2^3 - 112x_2^5 + 64x_2^7 + -7x_3 + 56x_3^3 - 112x_3^5 + 64x_3^7 \end{cases} \quad (\text{II.28})$$

Après l'élimination de  $x_1$ , on peut maintenant appliquer la théorie résultante pour éliminer  $x_2$ .

Pour faire le calcul de la résultante, on utilise l'un des logiciels de calculs mathématiques.

La fonction résultante est intégrée dans plusieurs logiciels, dont les plus connus sont les suivants : MATHEMATICA, MAPLE, et enfin le MATLAB 2013.

les deux premiers logiciels (Maple, Mathématique) sont spécialisés dans la résolution des équations, Ils se caractérisent par leur vitesse de calcul par rapport au MATLAB.

Dans notre travail on utilise le MATLAB 2013 qui nous permet de programmer tout l'algorithme de calcul dans un script.

La fonction qu'on utilise pour calculer la résultante est la suivante :

```

1 R1=evalin(symengine, "polylib::resultant(5 *(m-x2-x3) - 20 ...
    *(m-x2-x3)^3 + 16 *(m-x2-x3)^5 + 5*x2 - 20*x2^3 + 16*x2^5 + ...
    (5*x3 - 20*x3^3 + 16*x3^5), (-7 *(m-x2-x3) + 56 *(m-x2-x3)^3 ...
    - 112 *(m-x2-x3)^5 + 64 *(m-x2-x3)^7) + (-7*x2 + 56*x2^3 - ...
    112*x2^5 + 64*x2^7) + (-7*x3 + 56*x3^3 - 112*x3^5 + ...
    64*x3^7), x2)");
```

Cette fonction donne un polynôme de  $x_3$  de degré 22 dont la résolution est très difficile.

Une nouvelle méthode est utilisée pour réduire le degré des équations basée sur la théorie des polynômes symétriques.

Les trois équations du système II.24 peuvent se réduire en s'écrivant en fonction des polynômes symétriques de degré inférieur  $s_1, s_2, s_3$  [29] [30].

Pour exprimer les polynômes  $p_1, p_5$  et  $p_7$  en fonction de  $s_1, s_2, s_3$  on utilise la fonction (symmetricReduction) in Mathématique ou on utilise Matlab 2013 en utilisant la fonction :

```

1 p1=evalin(symengine,"polylib::representByElemSym(poly(5 *x1 - ...
    20 *x1^3 + 16 *x1^5 + 5*x2 - 20*x2^3 + 16*x2^5 + (5*x3 - ...
    20*x3^3 + 16*x3^5)), [m, s2, s3]);");

```

Alors on obtient un système de trois équations en fonction de  $s_1$ ,  $s_2$  et  $s_3$  définies par :

$$\begin{cases} s_1 = x_1 + x_2 + x_3 \\ s_2 = x_1x_2 + x_1x_3 + x_2x_3 \\ s_3 = x_1x_2x_3 \end{cases} \quad (\text{II.29})$$

et notre système est

$$\begin{cases} q_1(s_1) = 0 = s_1 - m \\ q_5(s_1, s_2, s_3) = 0 = 16m^5 - 80 * m^3s_2 - 20m^3 + 80m^2 * s_3 + 80ms_2^2 + 60ms_2 + 5m \\ \quad - 80s_2s_3 - 60s_3 \\ q_7(s_1, s_2, s_3) = 0 = 64m^7 - 448m^5s_2 - 112m^5 + 448m^4s_3 + 896m^3s_2^2 + 560m^3s_2 + 56m^3 \\ \quad - 1344m^2s_2s_3 - 560m^2s_3 - 448ms_2^2 - 560ms_2^2 - 168ms_2 + 448ms_3^2 - 7m \\ \quad + 448s_2^2s_3 + 560s_2s_3 + 168s_3 \end{cases} \quad (\text{II.30})$$

De  $q_1$  en II.30 on a  $s_1 = m$ . On remplace  $s_1$  par  $m$  dans  $q_5$  et  $q_7$ , et on les réécrit de la façon suivante :

$$\begin{cases} q_5(s_2, s_3) = 5m - 20m^3 + 16m^5 + 60ms_2 - 80m^3s_2 + 80ms_2^2 - 60s_3 + 80m^2s_3 - 80s_2s_3 \\ q_7(s_2, s_3) = 0 = -7m + 56m^3 - 112m^5 + 64m^7 - 168ms_2 + 560m^3s_2 - 448m^5s_2 - 560ms_2^2 \\ \quad + 896m^3s_2^2 - 448ms_2^3 + 168s_3 - 560m^2s_3 + 448m^4s_3 + 560s_2s_3 - 1344m^2s_2s_3 \\ \quad + 448s_2^2s_3 + 448ms_3^2 \end{cases} \quad (\text{II.31})$$

Remarquons que :

$$\begin{aligned} \deg_{s_2}[q_5(s_2, s_3)] &= 2, \deg_{s_3}[q_5(s_2, s_3)] = 1 \\ \deg_{s_2}[q_7(s_2, s_3)] &= 3, \deg_{s_3}[q_7(s_2, s_3)] = 2 \end{aligned}$$

## II.4. RÉSOLUTION DU SYSTÈME

On note que le degré des polynomes en  $s_2$  et  $s_3$  sont inférieurs à ceux de  $p_5(x_2, x_3)$  et  $p_7(x_2, x_3)$ .

Par conséquent la matrice de sylvester de la paire  $[q_5(s_2, s_3), q_7(s_2, s_3)]$  est  $3 \times 3$  (si  $s_3$  est éliminé) à la place de  $10 \times 10$  dans le cas de  $[p_5(x_2, x_3), p_7(x_2, x_3)]$ .

L'élimination de  $s_3$  peut se faire par la fonction :

```

1 R1=evalin(symengine, "polylib::resultant(16*m^5 - 80*m^3*s2 - ...
    20*m^3 + 80*m^2*s3 + 80*m*s2^2 + 60*m*s2 + 5*m - 80*s2*s3 - ...
    60*s3, 64*m^7 - 448*m^5*s2 - 112*m^5 + 448*m^4*s3 + ...
    896*m^3*s2^2 + 560*m^3*s2 + 56*m^3 - 1344*m^2*s2*s3 - ...
    560*m^2*s3 - 448*m*s2^3 - 560*m*s2^2 - 168*m*s2 + 448*m*s3^2 ...
    - 7*m + 448*s2^2*s3 + 560*s2*s3 + 168*s3, s3)");

```

Elle donne comme résultat :

$$\begin{aligned}
 & -49152m^{11} + 327680m^9s_2 + 245760m^9 - 737280m^7s_2^2 - 1249280m^7s_2 - 450560m^7 \\
 & + 573440m^5s_2^3 + 1863680m^5s_2^2 + 1648640m^5s_2 + 385280m^5 - 716800m^3s_2^3 - 1433600m^3s_2^2 \\
 & - 896000m^3s_2 - 156800m^3 + 179200ms_2^3 + 313600ms_2^2 + 168000ms_2 + 25200m
 \end{aligned} \tag{II.32}$$

A la fin de l'opération on aboutit à une seule équation non linéaire, de degré 3, en fonction d'une seule variable  $S_2$ . Cette équation est résolue pour chaque valeur de  $m$ , et  $S_1$  et  $S_3$  peuvent être déduites des relations II.30 et II.31. et on aboutit enfin à des triples  $(s_1, s_2, s_3)$ , Il nous reste maintenant de trouver  $(x_1, x_2, x_3)$ .

La déduction des triples  $(x_1, x_2, x_3)$  se fait par la résolution du system non-linéaire suivant en utilisant la méthode de la Résultante :

$$\begin{cases} f_1(x_1, x_2, x_3) = s_1 - (x_1 + x_2 + x_3) = 0 \\ f_2(x_1, x_2, x_3) = s_2 - (x_1x_2 + x_1x_3 + x_2x_3) = 0 \\ f_3(x_1, x_2, x_3) = s_3 - x_1x_2x_3 = 0 \end{cases} \tag{II.33}$$

On calcul donc les résultants suivants :

$$R_{12} = res(f_1(x_1, x_2, x_3), f_2(x_1, x_2, x_3), x_1) \tag{II.34}$$

$$R_{13} = res(f_1(x_1, x_2, x_3), f_3(x_1, x_2, x_3), x_1) \tag{II.35}$$

et enfin :

$$R(x_3) = res(R_{12}(x_2, x_3), R_{13}(x_2, x_3)) \tag{II.36}$$

La procédure maintenant est de remplacer les triples  $(s_1, s_2, s_3)$  trouvés précédement dans l'équation II.36 pour trouver les solutions  $x_{3i}$ .



Pour chaque valeur  $x_{3i}$  on résout l'équation II.34 pour trouver les valeurs  $x_{2j}$  qui annule  $R_{12}(x_2, x_3)$ , et finalement on remplace chaque  $(x_{3i}, x_{2j})$  dans  $f_1(x_1, x_2, x_3)$  pour trouver  $x_{1j}$ .  
 enfin on aura les triples :

$$(x_1, x_2, x_3) = (x_{1j}, x_{2j}, x_{3i})$$

On prend celle qui satisfait la condition :

$$0 \leq x_1, x_2, x_3 \leq 1$$

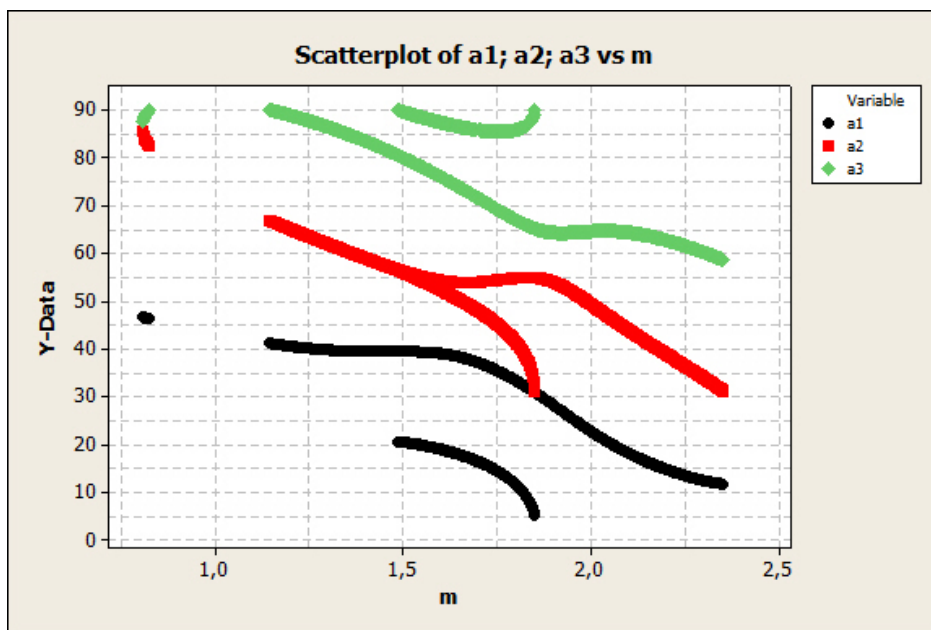
A partir de ces résultats on calcule les angles  $(\alpha_1, \alpha_2, \alpha_3)$  telle que :

$$\begin{cases} \alpha_1 = \arccos(x_1) \\ \alpha_2 = \arccos(x_2) \\ \alpha_3 = \arccos(x_3) \end{cases} \quad (\text{II.37})$$

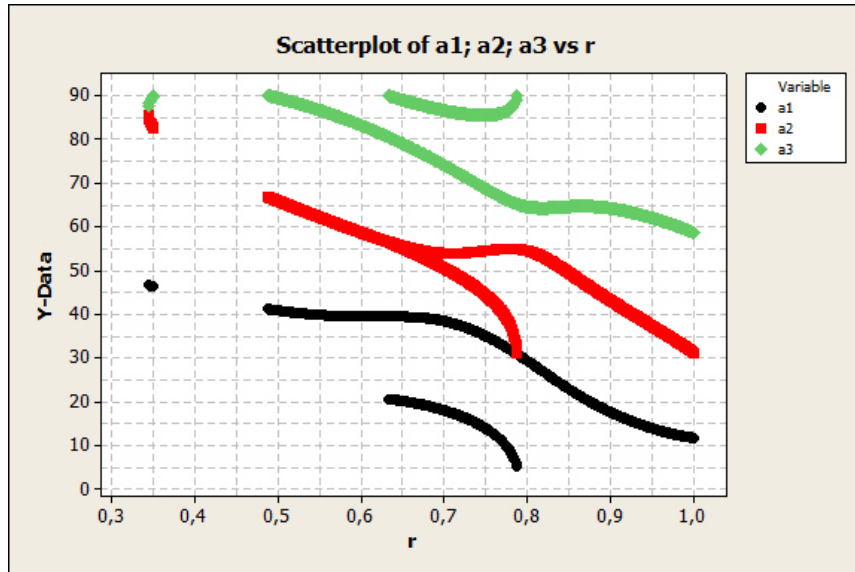
### II.4.3 Les résultats des calculs

A l'aide du logiciel Matlab, on fait un script qui calcule les angles de commutation  $(\alpha_1, \alpha_2, \alpha_3)$  pour plusieurs valeurs (on prend 1000 valeurs) de l'indice de modulation  $m$  (ou le taux de modulation  $r$ ). On obtient alors un tableau des angles pour chaque valeur de  $m$ .

A l'aide du logiciel Minitab on trace le graph des angles en fonction de l'indice de modulation  $m$  (figure II.6) et en fonction du taux de modulation  $r$  (figure II.7).



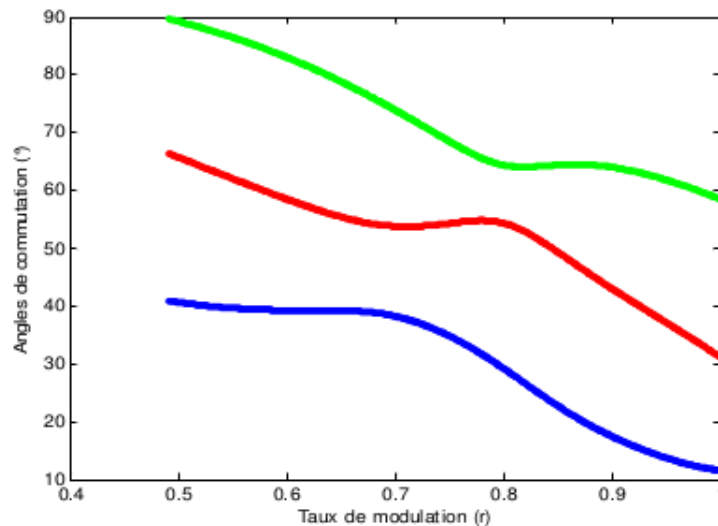
**Figure II.6** – Les angles de commutation (deg) en fonction de l'indice de modulation  $m$  pour l'élimination des harmonique 5 et 7



**Figure II.7** – Les angles de commutation (deg) en fonction du taux de modulation  $r$  pour l'élimination des harmonique 5 et 7

### Remarques et constatations

Des anciennes thèses et publications avait fait ces calculs par la méthode itérative de Newton-Raphson, La figure I.9 qui suivent représente les angles de commutation calculés par la méthode de Newton-Raphson [13][31] .



**Figure II.8** – Les angles de commutation calculés par la méthode de Newton-Raphson [31]

Des résultats obtenus, nous avons constaté que la méthode basée sur la théorie résultante est plus rigoureuse que la méthode itérative. Elle permet de déterminer toutes les solutions possibles des systèmes d'équations non-linéaires. En effet, elle élargit l'intervalle du taux de modulation  $r$ .

### II.4.4 Elimination des autres harmoniques

Avec le même principe et la théorie de l'élimination par résultante on peut déterminer les angles de commutation pour éliminer les harmoniques spécifiques, tels que le 11 et le 13 jusqu'à la 19<sup>ème</sup> harmonique .

Cependant, comme le nombre de niveaux augmente, les degrés des polynômes dans ces équations sont grands et on atteint les limites de la capacité d'un logiciel de calcul (par exemple, mathematica et maple) pour résoudre le système d'équations polynômes à l'aide la théorie de la résultante.

Une solution est proposée pour résoudre ce problème des harmoniques d'ordre supérieur à 19, est d'utiliser la méthode de Newton-Raphson en utilisant comme valeur initiale la valeur obtenu par la méthode de la résultante du niveau juste inférieur.[13]

## II.5 Vérification des résultats par la simulation

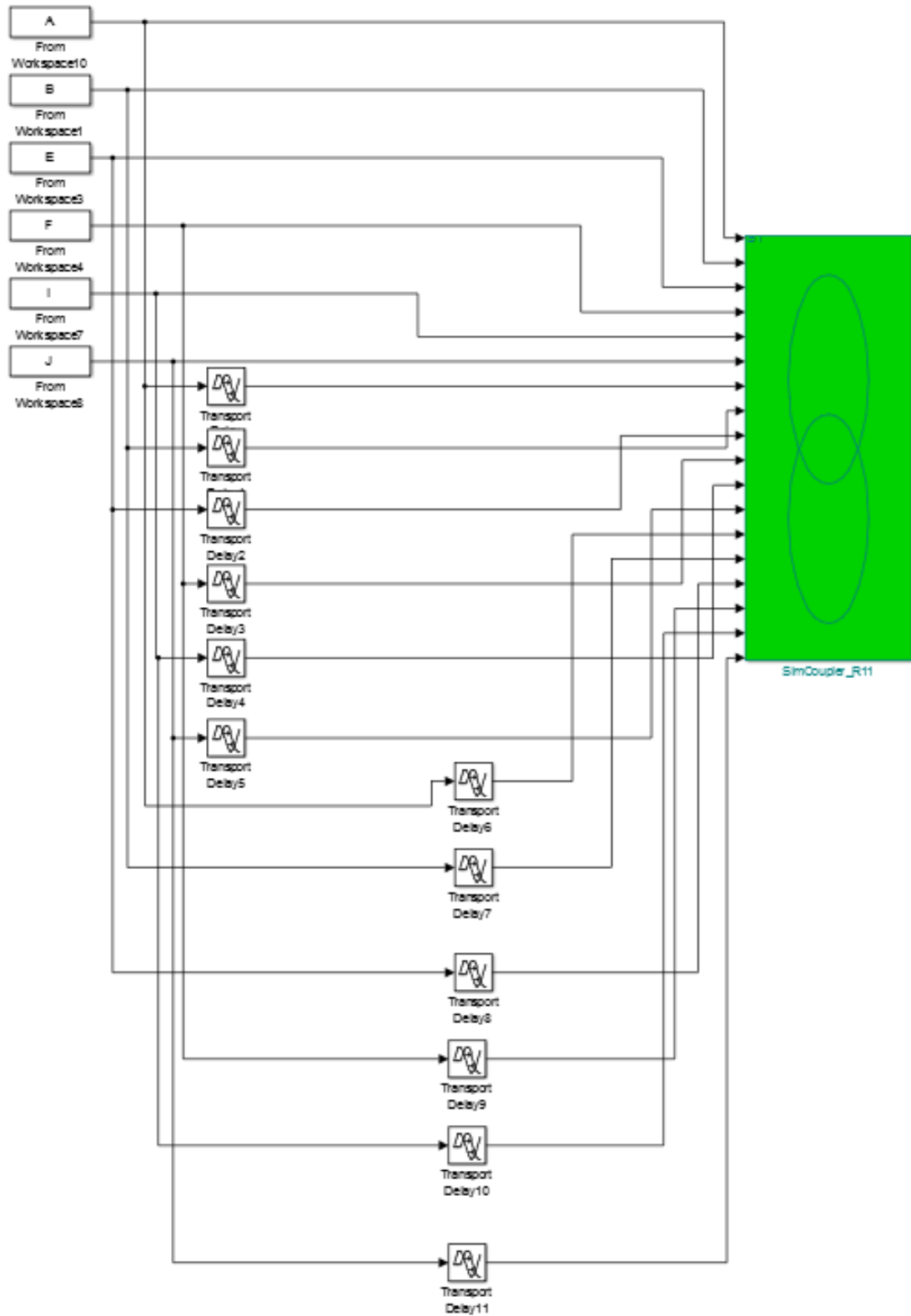
Afin de confirmer la validité des solutions obtenues, on a reconstitué la tension du bras (figureII.5) par les angles de commutation trouvés, et ceci en faisant la simulation de l'onduleur et ses commande par une cosimulation PSIM/SIMULINK.

La partie puissance du système, constitué de l'onduleur multi-niveaux cascadié triphasé et la charge (machine asynchrone), est simulée avec le logiciel PSIM de Power sim spécialisé dans l'électronique de puissance, alors que les signaux de commande sont générés par un scripte Matlab (command.m) qui transmet les signaux à un fichier SIMULINK contenant un bloc, appelé `simcoupler` qui sert à communiquer avec le PSIM .

### II.5.1 Schéma de simulation sous PSIM /SIMULINK

La figure II.10 donne le schéma de l'étage de puissance sous PSIM, il est constitué d'un onduleur multi-niveaux cascade triphasée alimentant un moteur asynchrone. Alors que la figure II.9 donne le shéma de simulink contenant le bloc `simcoupler`.

## II.5. VÉRIFICATION DES RÉSULTATS PAR LA SIMULATION



**Figure II.9** – Schéma bloc de la commande en SIMULINK

## II.5. VÉRIFICATION DES RÉSULTATS PAR LA SIMULATION

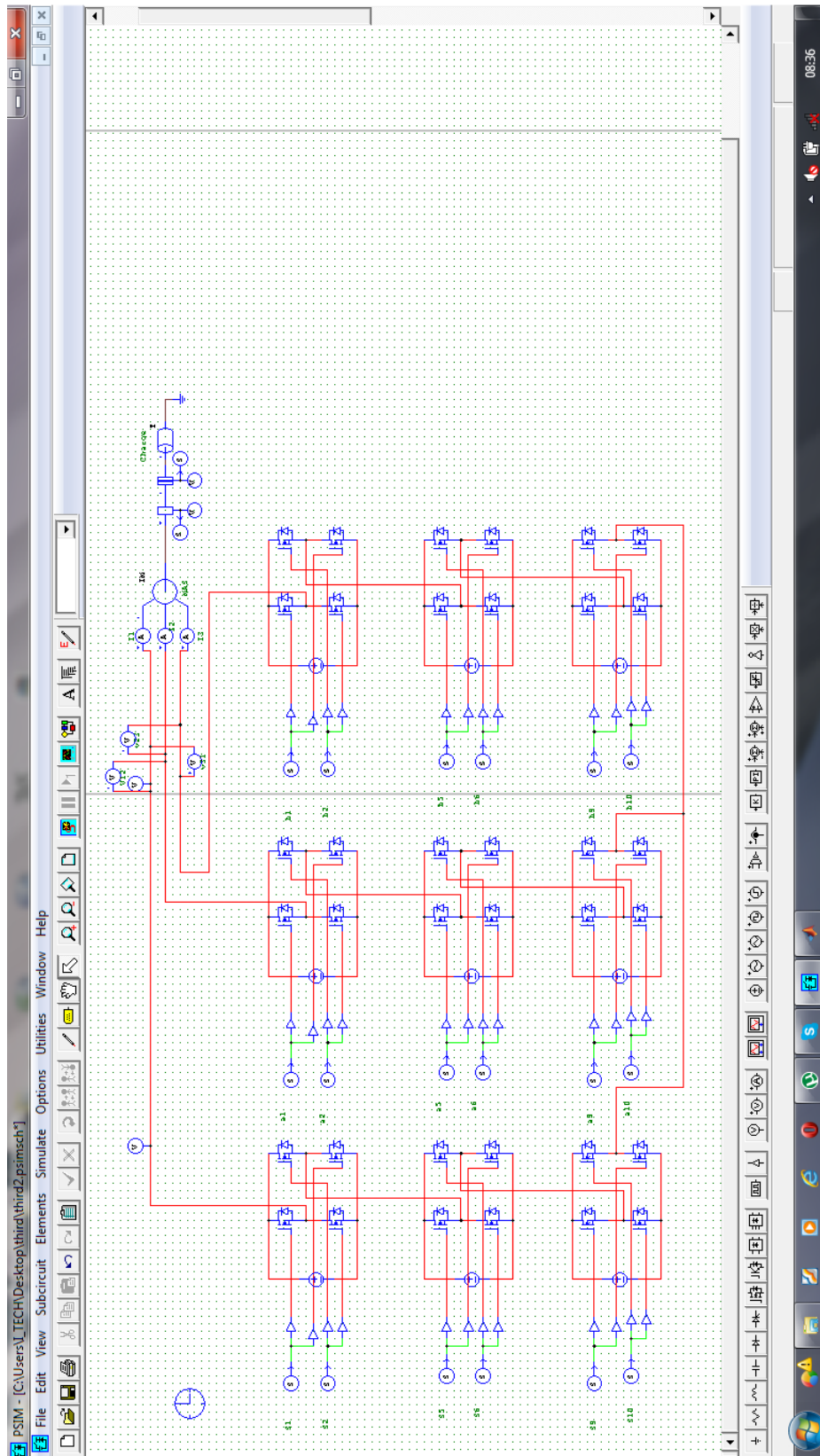


Figure II.10 – Schéma de l'onduleur et de la charge en PSIM

## II.5. VÉRIFICATION DES RÉSULTATS PAR LA SIMULATION

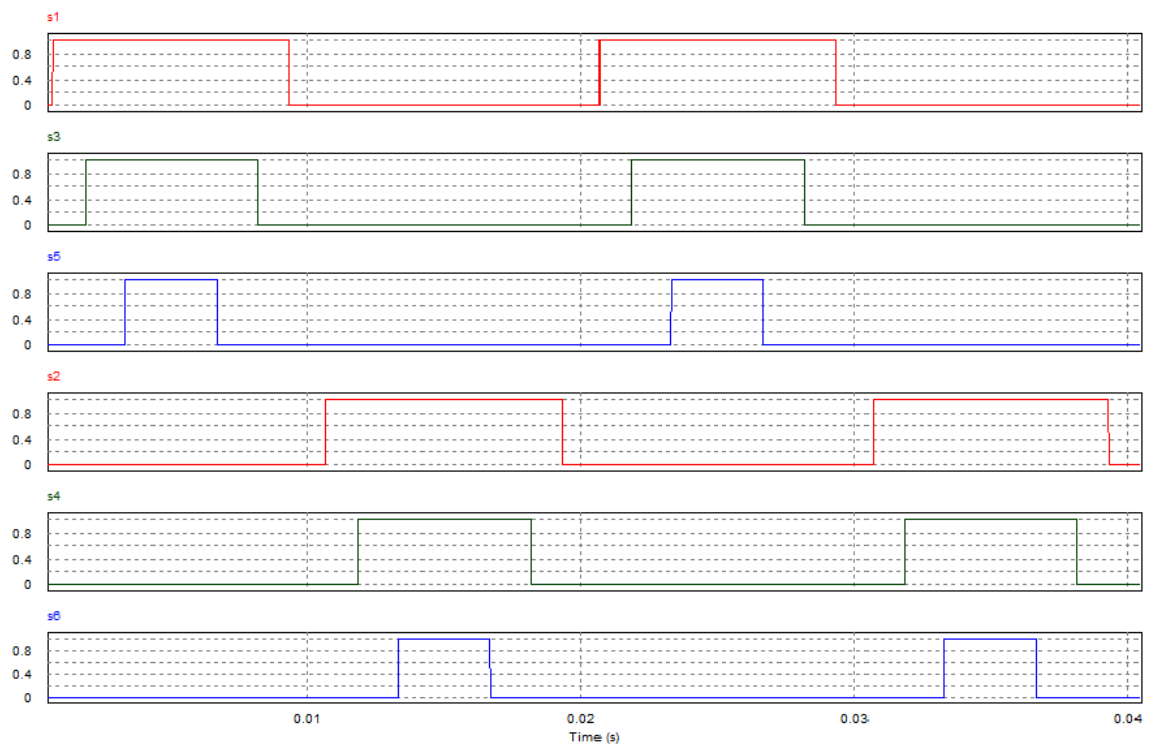
### II.5.2 Résultats de la simulation

Pour la simulation on choisit des valeurs de l'indice de modulation  $m$  qui sont présentées dans la tableau suivant :

r	im	$\alpha_1$	$\alpha_2$	$\alpha_3$
0.35	0,82425	0.808048834	1.437658461	1.569783467
0.61	1.43655	0.688073615	1.010751094	1.436812496
0.8	1.884	0.510255698	0.950128345	1.125464627
1	2.355	0.204911721450922	0.549986370837494	1.025829874198489

**Table II.1** – des valeurs choisit de  $m$

Pour l'indice de modulation  $m=2.35$  on est à la fréquence 50 HZ, on a tracé les signaux de commande des six switches d'une phase figure II.11. Après on a visualisé en premier temps les formes des signaux de sortie de l'onduleur à vide. les signaux de sortie sont présentés sur les figures II.12 et leurs spectres sur la figure II.13 :



**Figure II.11** – Les signaux de commandes des six switches

## II.5. VÉRIFICATION DES RÉSULTATS PAR LA SIMULATION

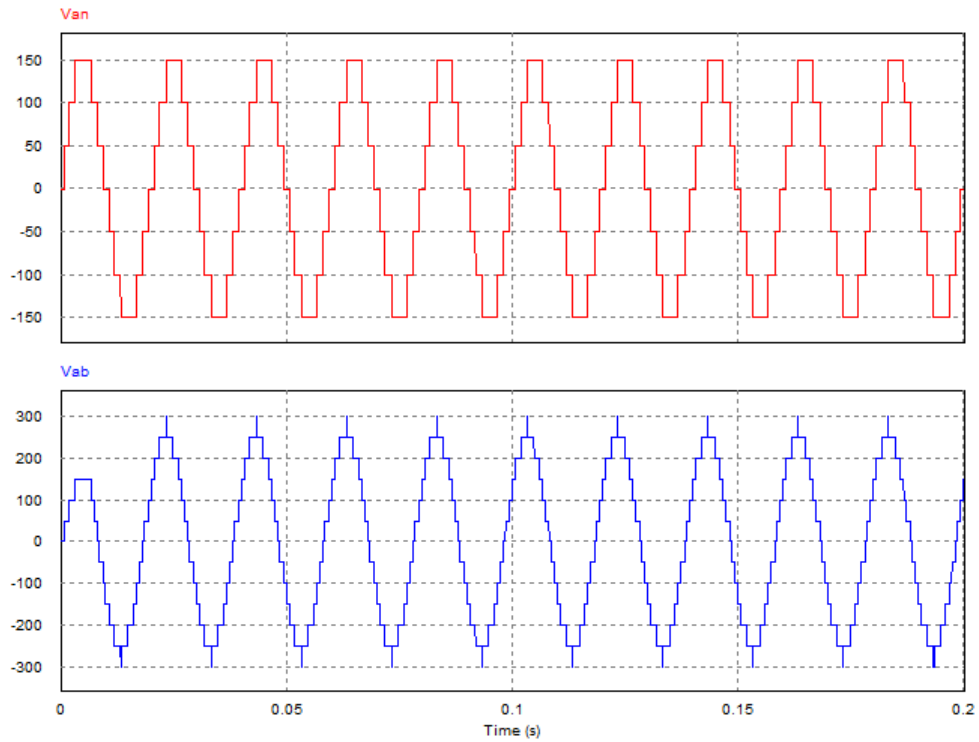


Figure II.12 – Les formes des tensions de  $v_{an}$  et  $v_{ab}$  à vide

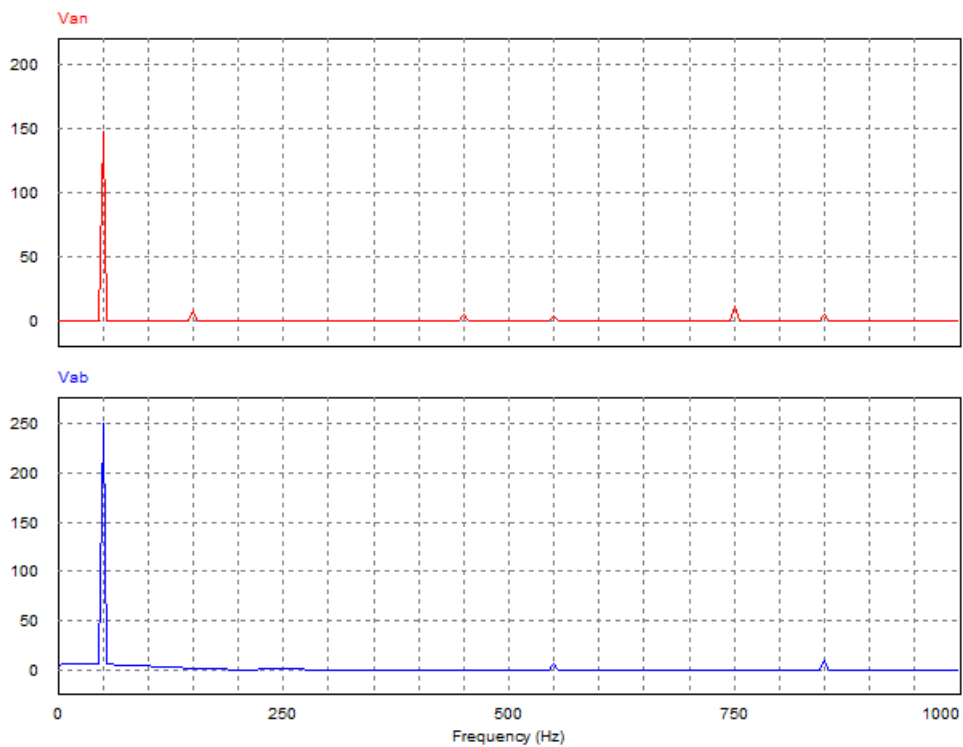


Figure II.13 – Les spectres fréquentielles des signaux à vide

On remarque bien que les harmonique 5 et 7 sont éliminés dans le spectres de la tension de phase. Pour la tension entre phase les harmoniques multiples de 3 sont ainsi éliminés.

## II.5. VÉRIFICATION DES RÉSULTATS PAR LA SIMULATION

Le premier harmonique non éliminé est celui du rang 11 (550 Hz).

Pour le fonctionnement en charge (machine asynchrone) on a tracé les formes des tensions  $V_{an}$ ,  $V_{ab}$  et du courant  $i_1$  ainsi que leurs spectres fréquentiels.

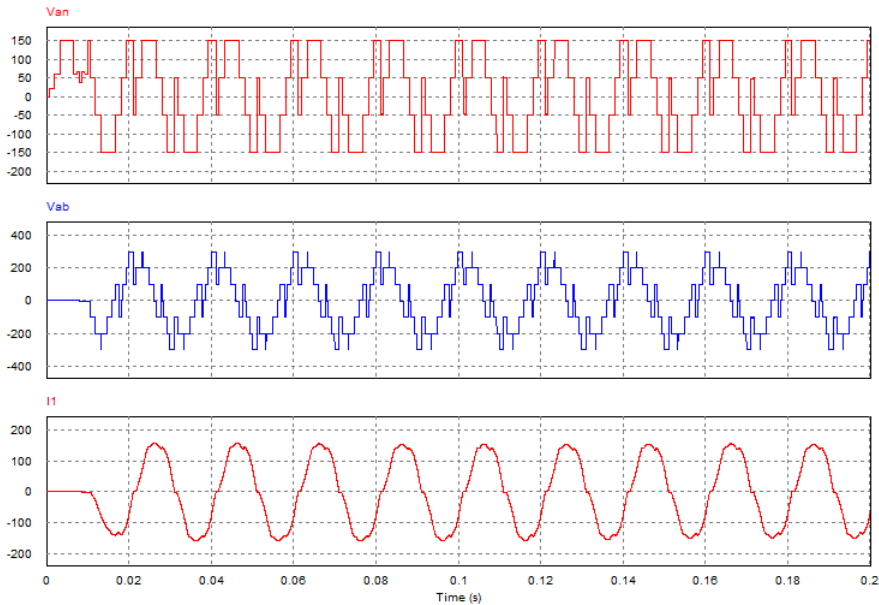


Figure II.14 – Les formes des tension de  $v_{an}$ ,  $v_{ab}$  et du courant  $i_1$

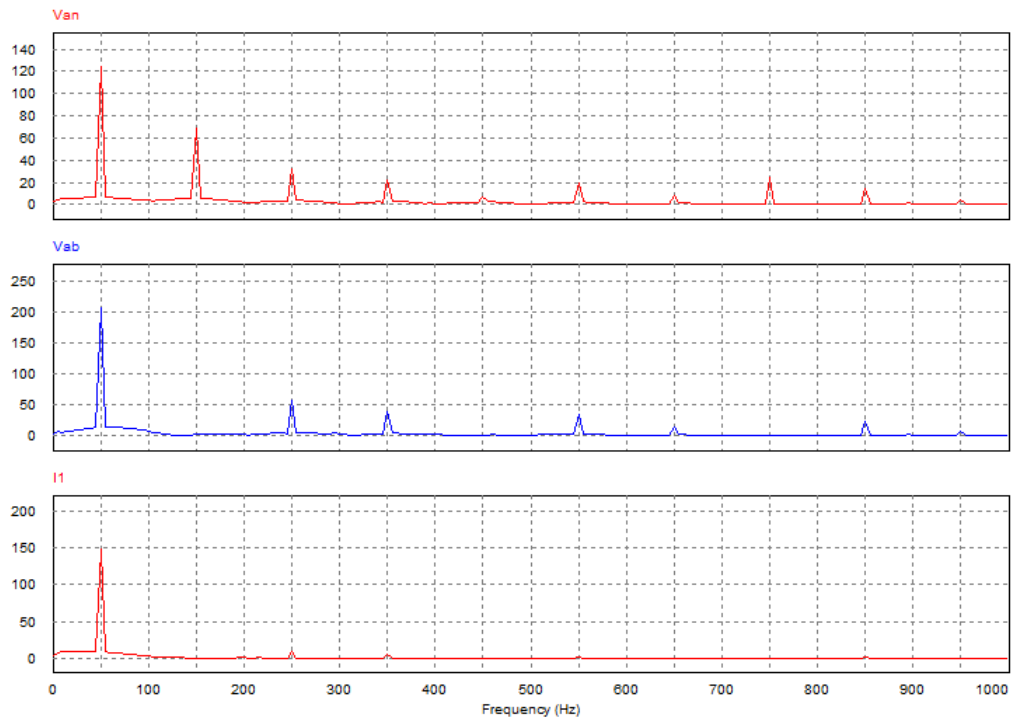


Figure II.15 – Les spectres fréquentiel de  $v_{an}$ ,  $v_{ab}$  et  $i_1$



## II.5. VÉRIFICATION DES RÉSULTATS PAR LA SIMULATION

On remarque l'absence des harmoniques 3,5,7,et 9 dans le spectre du courant ce qui rend la forme du courant similaire à un sinusoïde de fréquence 50 Hz.

Les tensions de phase et entre phases sont déformées à cause des effets de la machine asynchrone.

### Résultats pour différentes valeurs de $r$

- Pour la valeur  $r=0.35$  la fréquence  $f$  est égale à 17.5HZ, on trace les tension de ligne et de phase à vide et ces spectres, les résultats sont présenté sur les figures II.16 et II.17.

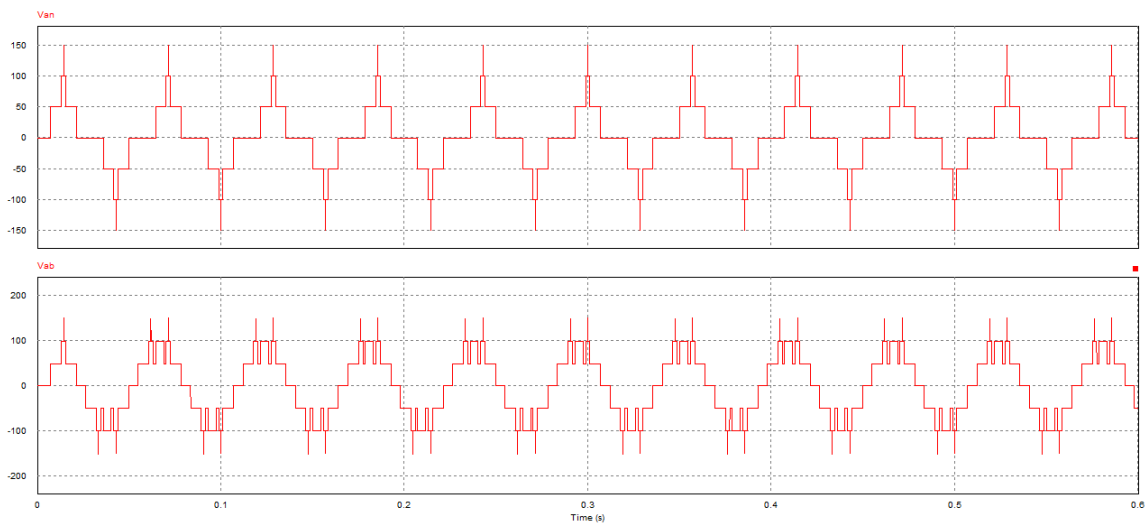


Figure II.16 – Les formes des tension de  $v_{an}$ ,  $v_{ab}$ ,  $i_1$  pour  $r=0.35$

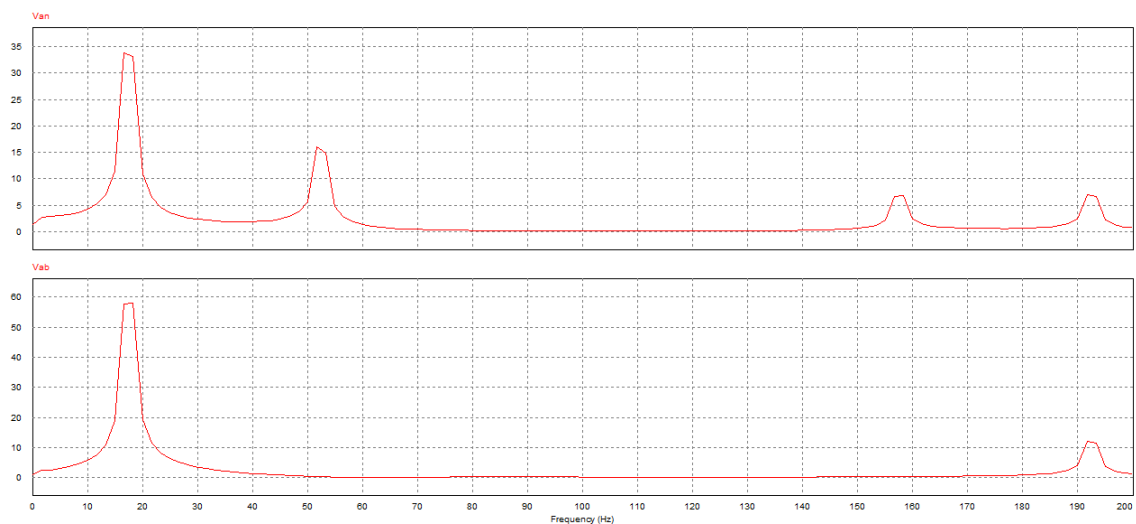
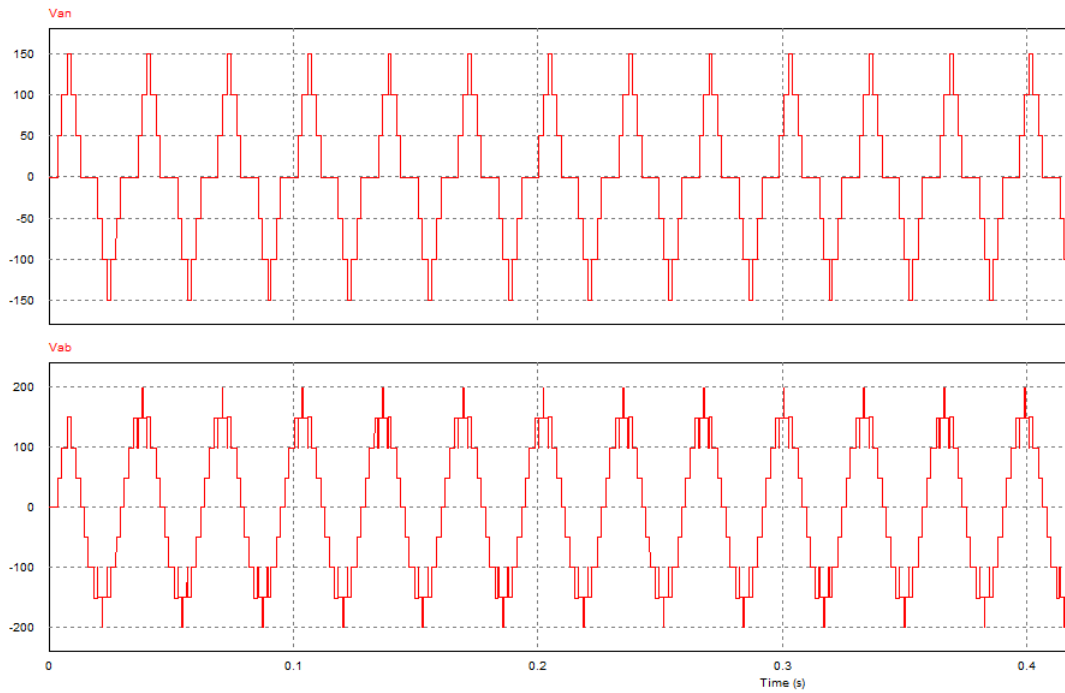


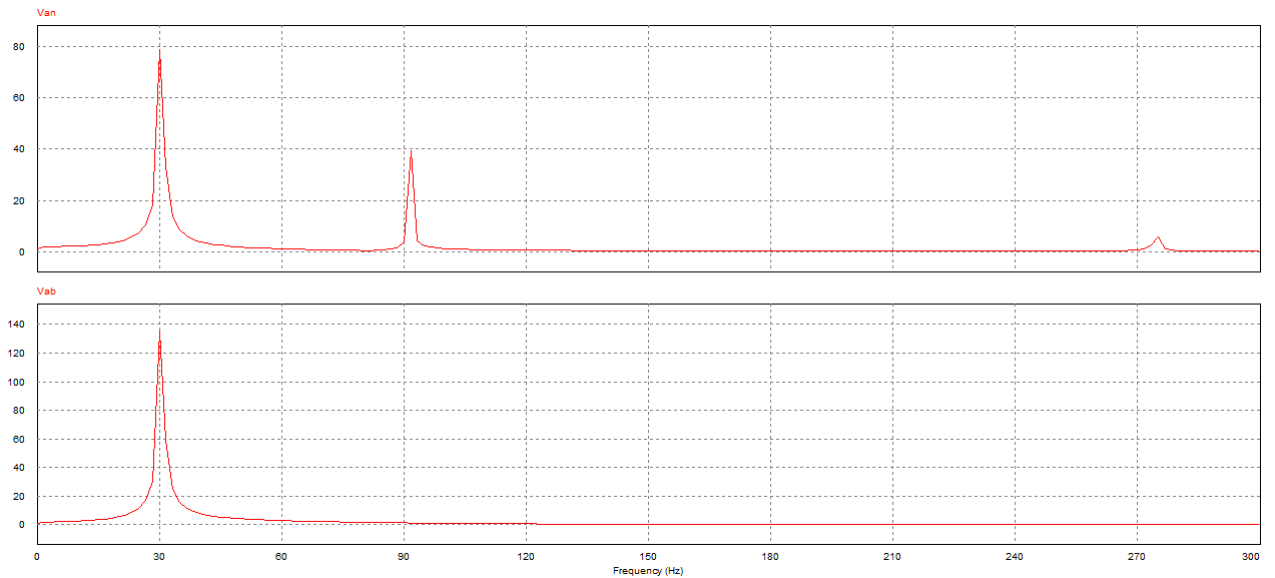
Figure II.17 – Les spectres fréquentiel des signaux pour  $r=0.35$

## II.5. VÉRIFICATION DES RÉSULTATS PAR LA SIMULATION

- Pour la valeur  $r=0.61$  la fréquence est égale à 30.5Hz on trace les tensions de ligne et de phase à vide et ces spectres, les résultats sont présentés sur les figures II.18 et II.19



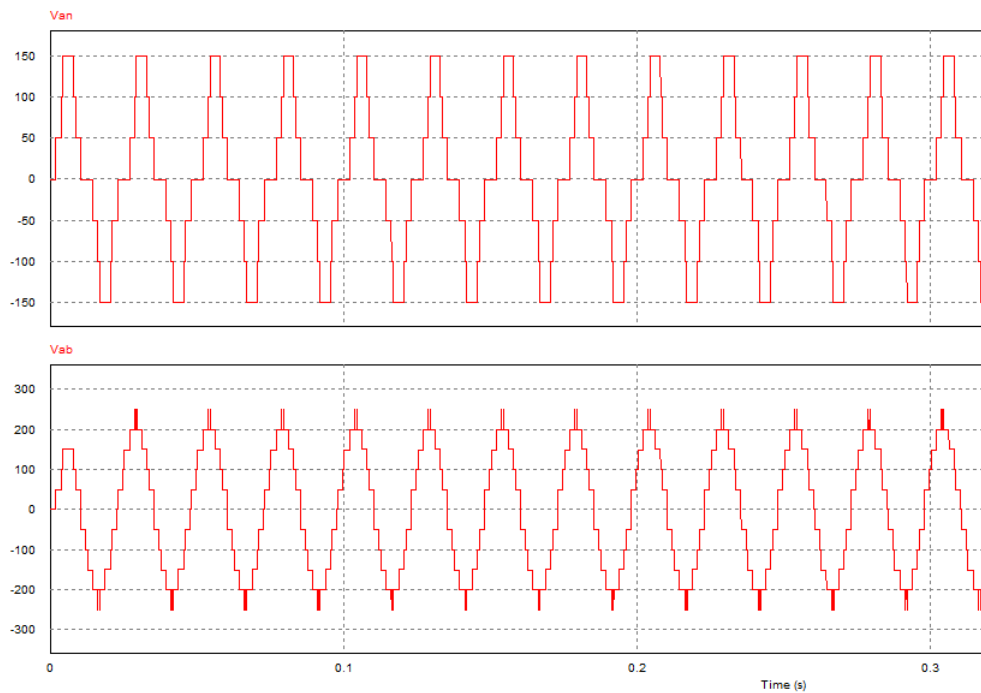
**Figure II.18** – Les formes des tensions de  $v_{an}$ ,  $v_{ab}$ ,  $i_1$  pour  $r=0.61$



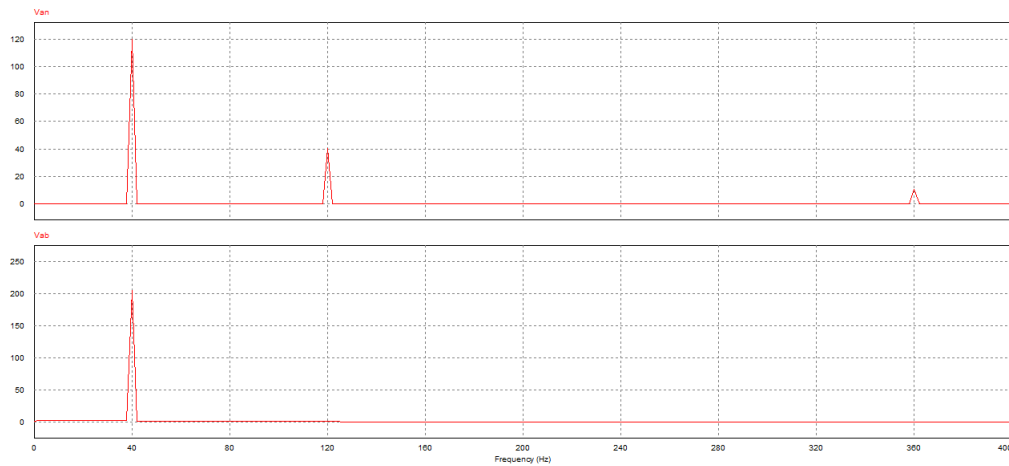
**Figure II.19** – Les spectres fréquentiel des signaux pour  $r=0.61$

## II.5. VÉRIFICATION DES RÉSULTATS PAR LA SIMULATION

- Pour  $r=0.8$  la fréquence est égale à 40Hz on trace les tensions de ligne et de phase à vide et ses spectres, les résultats sont présentés sur les figures II.20 et II.21



**Figure II.20** – les formes des tension de  $v_{an}$ ,  $v_{ab}$ ,  $i_1$  pour  $r=0.8$



**Figure II.21** – les spectres fréquentiel des signaux pour  $r=0.8$

### Interprétation des résultats

D'après les résultats présentés précédemment pour différentes valeurs de  $m$  on remarque bien l'élimination des harmoniques désirés.

On remarque que le nombre de niveaux de la tension entre phase varie avec l'indice de modulation, et à un maximum de 13 niveaux dans notre cas.

### II.5.3 Conclusion

L'algorithme de la méthode d'élimination des harmoniques élaboré précédemment présente une grande précision dans le calcul des angles de commutation. Il donne une efficacité dans l'élimination des harmoniques voulus. La cosimulation de cet algorithme dans PSIM et SIMULINK a prouvé son efficacité dans la commande de la vitesse des moteurs à induction.

Pour valider pratiquement l'efficacité de cet algorithme nous allons l'implémenter sur une architecture hardware de type FPGA.

# Chapitre

# III

## Implémentation de la commande sur FPGA

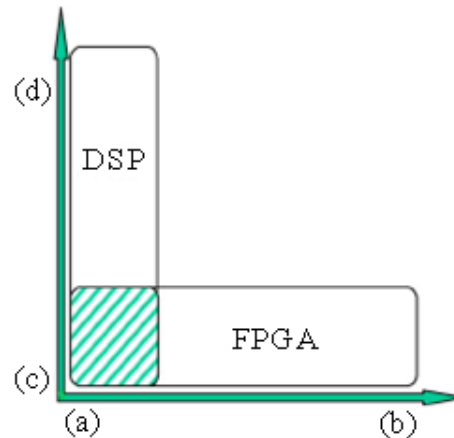
---

### III.1 Introduction

Les composants programmables sont disponibles depuis les années 1970, mais leur utilisation est restée limitée essentiellement pour des raisons technologiques. Le premier composant XILINX a été fabriqué en 1984, constitué de 85000 transistors, l'équivalent de 1000 portes logiques, fabriqué en  $2\mu m$ .

Actuellement, les FPGAs récents sont fabriqués avec une technologie de 65 nm, leur densité est d'environ 10 millions portes logiques [32]. Lors de la conception des systèmes électroniques industriels (circuits) plusieurs critères doivent être pris en considération : le prix, la consommation de l'énergie (surtout dans le cas des systèmes embarqués), les performances demandées par l'application et avant tous est ce que le matériel est bien choisi et correspond à l'algorithme à implémenter.

Actuellement les deux moyens pour implémenter un contrôleur sont les DSPs et les FPGAs. Selon la nature de l'algorithme, le concepteur peut faire un choix entre ces deux possibilités.[12] La figure III.1 illustre ce principe.



**Figure III.1** – Domaines d'utilisation des DSPs et des FPGAs[12]

L'axe X de ce graphe représente les contraintes temporelles de l'algorithme, alors que l'axe Y représente la complexité de l'algorithme.

Dans le chapitre précédent nous avons vu la méthode de commande à élimination des harmoniques pour les onduleurs multi-niveaux, et nous avons élaboré un l'algorithme permettant la commande de la vitesse d'un moteur asynchrone triphasé. Pour vérifier l'algorithme et valider expérimentalement les résultats de simulation nous avons choisi d'implémenter cet algorithme sur un circuit reconfigurable de type FPGA.

## III.2 Les circuits FPGA

Depuis la commercialisation du premier prédéfini programmable (FPGA) en 1985, l'utilisation de ces circuits ne cesse de s'étendre à des domaines et applications variés, parmi lesquels nous pouvons citer le traitement d'image [34] [35], les réseaux de neurones [36], la comparaison de séquences génétiques [37], l'architecture des ordinateurs [38] etc...

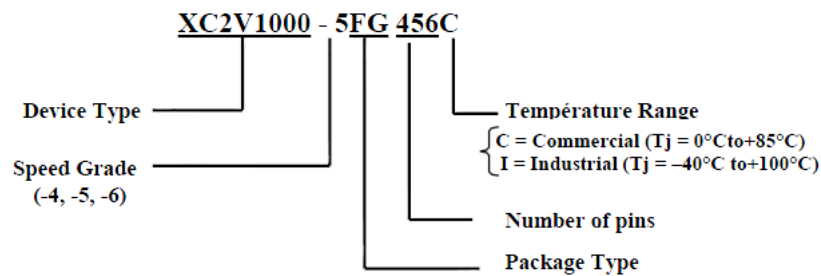
L'intérêt suscité par les FPGA est dû essentiellement à leurs prix abordables, facilité de mise en oeuvre et flexibilité. En outre, les coûts fixes et délais de fabrications (NRE), en comparaison avec les circuits spécifiques (ASIC), sont totalement éliminés. Cependant, ils présentent une faible densité d'intégration de portes logiques et atteignent des fréquences de travail relativement faibles devant les ASIC.

### III.2.1 Définition

Les FPGA (Field Programmable Gate Arrays ou réseaux logiques programmables) sont des composants entièrement reconfigurables ce qui permet de les reprogrammer à volonté afin d'accélérer notablement certaines phases de calculs. L'avantage de ce genre de circuit est sa grande souplesse qui permet de les réutiliser à volonté dans des algorithmes différents en un temps très court.[39]

### III.2.2 Nomenclature des circuits FPGA

Les circuits FPGA suivent la nomenclature suivante, selon un exemple donné[21] :



### III.2.3 Application des FPGA

Les FPGA sont utilisés dans de nombreuses applications, on en cite dans ce qui suit quelques unes :

- Prototypage de nouveaux circuits.
- Fabrication de composants spéciaux en petite série.
- Adaptation aux besoins rencontrés lors de l'utilisation.
- Systèmes de commande en temps réel.
- DSP (Digital Signal Processor).
- Imagerie médicale.

### III.2.4 Architecture des FPGA

Les circuits FPGA sont constitués d'une matrice de blocs logiques programmables entourés de blocs d'entrées sorties programmables. L'ensemble est relié par un réseau d'interconnexions programmable.

Les FPGA sont bien distincts des autres familles de circuits programmables tout en offrant le plus haut niveau d'intégration logique. Il existe actuellement plusieurs fabricants de circuits FPGA dont Xilinx et Altera qui sont les plus connus, et plusieurs technologies et principes organisationnels.

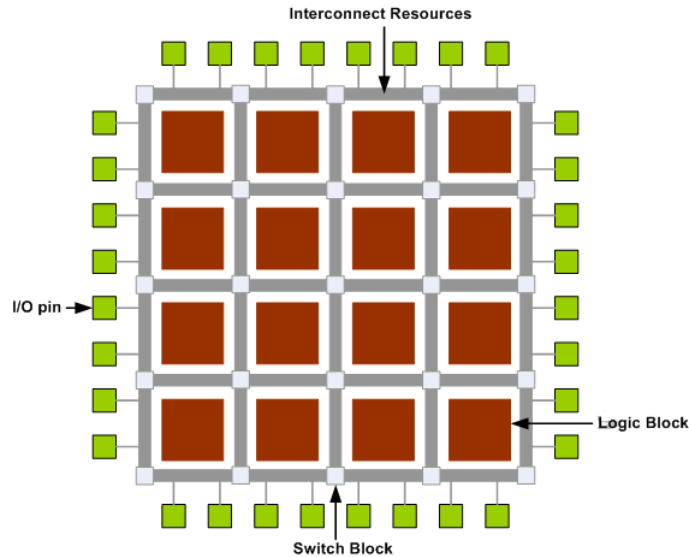
L'architecture, retenue par Xilinx, se présente sous forme de deux couches :

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

#### Circuit configurable

La couche dite "circuit configurable" est constituée d'une matrice de blocs logiques configurables **CLB** permettant de réaliser des fonctions combinatoires et des fonctions séquentielles. Tout autour de ces blocs logiques configurables, nous trouvons des blocs entrées/sorties **IOB** dont le rôle est de gérer les entrées-sorties réalisant l'interface avec les modules extérieurs.

La programmation du circuit FPGA appelée aussi LCA (Logic Cells Arrays) consistera par le biais de l'application d'un potentiel adéquat sur la grille de certains transistors à effet de champ à interconnecter les éléments des CLB et des IOB afin de réaliser les fonctions souhaitées et d'assurer la propagation des signaux. Ces potentiels sont tout simplement mémorisés dans le réseau mémoire SRAM[32].

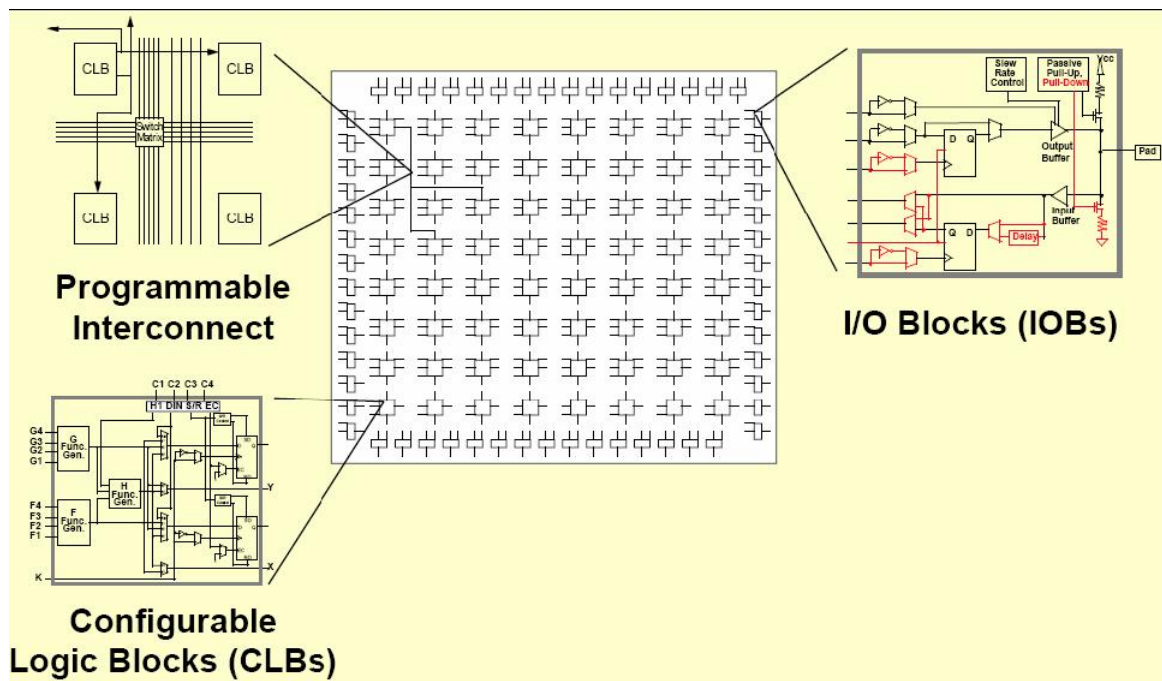


**Figure III.2** – Architecture interne d'un FPGA (circuit configurable)[32]

Les circuits FPGA du fabricant Xilinx utilisent deux types de cellules de base :

- les cellules d'entrées/sorties appelés IOB (input output bloc).
- les cellules logiques appelées CLB (configurable logic bloc).

Ces différentes cellules sont reliées entre elles par un réseau d'interconnexions configurable.



**Figure III.3** – Réseaux d'interconnexions configurables



### Réseau mémoire SRAM

La programmation d'un circuit FPGA est volatile, la configuration du circuit est donc mémorisée sur la couche réseau SRAM et stockée dans une ROM externe. Un dispositif interne Permet à chaque mise sous tension de charger la SRAM interne à partir de la ROM. Ainsi on conçoit aisément qu'un même circuit puisse être exploité successivement avec des ROM différentes puisque sa programmation interne n'est jamais définitive. On voit tout le parti que l'on peut tirer de cette souplesse en particulier lors d'une phase de mise au point. Une erreur n'est pas rédhibitoire, mais peut aisément être réparée.

La mise au point d'une configuration s'effectue en deux temps. Une première étape purement logicielle va consister à dessiner puis simuler logiquement le circuit fini. Dans la seconde étape, on effectuera une simulation matérielle en configurant un circuit réel. On pourra alors vérifier si le fonctionnement réel correspond bien à l'attente du concepteur, et ses besoin, et identifier les anomalies liées généralement à des temps de transit réels légèrement différents de ceux supposés lors de la simulation logicielle, ce qui peut conduire à des états instables voire même erronés.

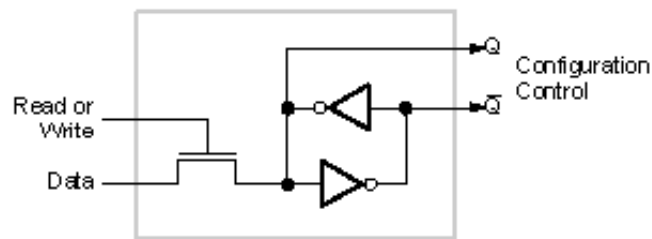


Figure III.4 – Structure d'une cellule SRAM

### III.2.5 Les blocs du circuits configurables

#### Les CLB (configurable logic bloc)

Les blocs logiques configurables sont les éléments déterminants des performances du FPGA. Chaque bloc est composé d'un bloc de logique combinatoire composé de deux générateurs de fonctions à quatre entrées et d'un bloc de mémorisation synchronisation composé de deux bascules D. Quatre autres entrées permettent d'effectuer les connexions internes entre les différents éléments du CLB. La figure ci-dessous, nous montre le schéma d'un CLB.

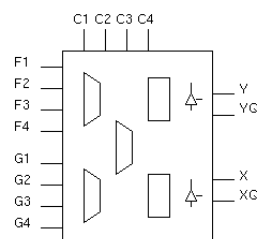
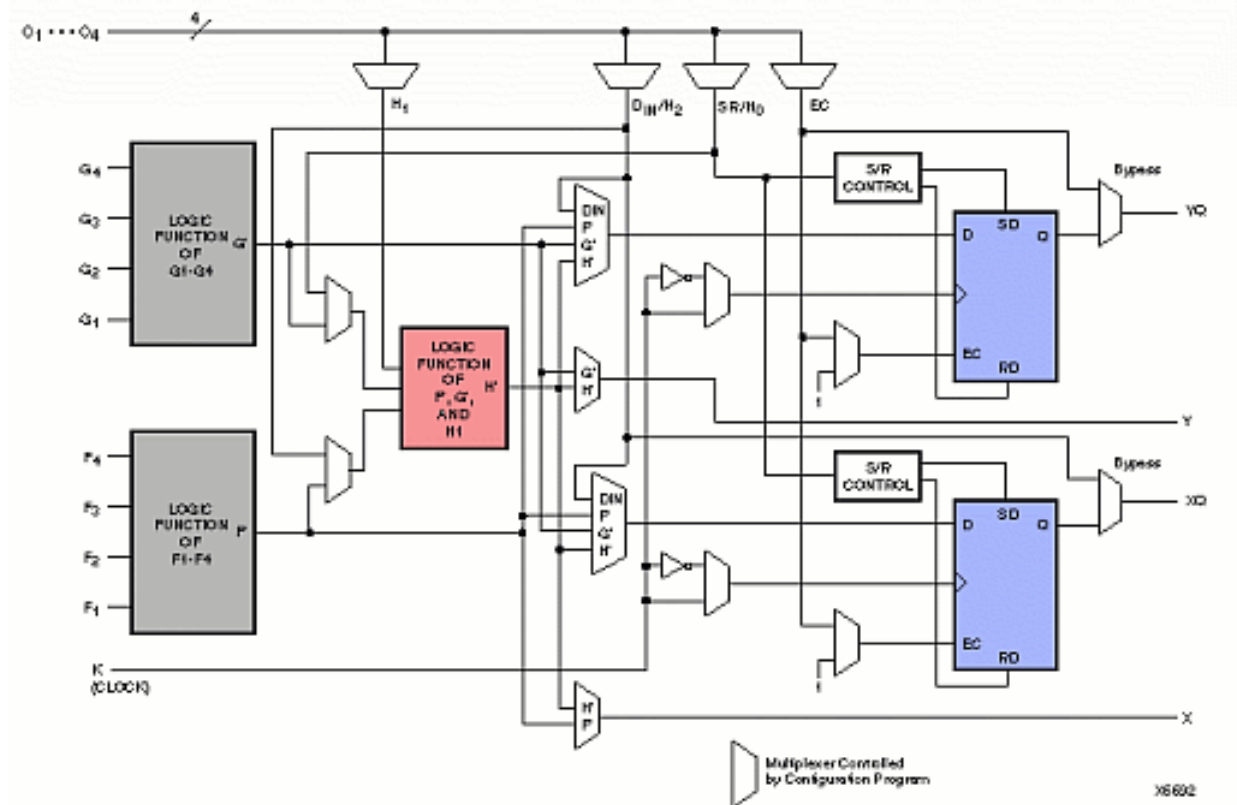


Figure III.5 – Les entrées/sorties du CLB



**Figure III.6** – Cellules logiques(CLB)[27]

Voyons d’abord le bloc logique combinatoire qui possède deux générateurs de fonctions  $F'$  et  $G'$  à quatre entrées indépendantes ( $F_1...F_4$ ,  $G_1...G_4$ ), lesquelles offrent aux concepteurs une flexibilité de développement importante car la majorité des fonctions aléatoires à concevoir n’excèdent pas quatre variables. Les deux fonctions sont générées à partir d’une table de vérité câblée inscrite dans une zone mémoire, rendant ainsi les délais de propagation pour chaque générateur de fonction indépendants de celle à réaliser. Une troisième fonction  $H'$  est réalisée à partir des sorties  $F'$  et  $G'$  et d’une troisième variable d’entrée  $H_1$  sortant d’un bloc composé de quatre signaux de contrôle  $H_1$ ,  $D_{in}$ ,  $S/R$ ,  $Ec$ . Les signaux des générateurs de fonction peuvent sortir du CLB, soit par la sortie  $X$ , pour les fonctions  $F'$  et  $G'$ , soit  $Y$  pour les fonctions  $G'$  et  $H'$ . Ainsi un CLB peut être utilisé pour réaliser :

- deux fonctions indépendantes à quatre entrées indépendantes
- ou une seule fonction à cinq variables
- ou deux fonctions, une à quatre variables et une autre à cinq variables.

L’intégration de fonctions à nombreuses variables diminue le nombre de CLB nécessaires, les délais de propagation des signaux et par conséquent augmente la densité et la vitesse du circuit. Les sorties de ces blocs logiques peuvent être appliquées à des bascules au nombre de deux ou directement à la sortie du CLB (sorties  $X$  et  $Y$ ). Chaque bascule présente deux modes de fonctionnement : un mode ‘flip-flop’ avec comme donnée à mémoriser, soit l’une des fonctions  $F'$ ,  $G'$ ,  $H'$  soit l’entrée directe  $D_{in}$ . La donnée peut être mémorisée sur un front montant ou descendant de l’horloge ( $CLK$ ).

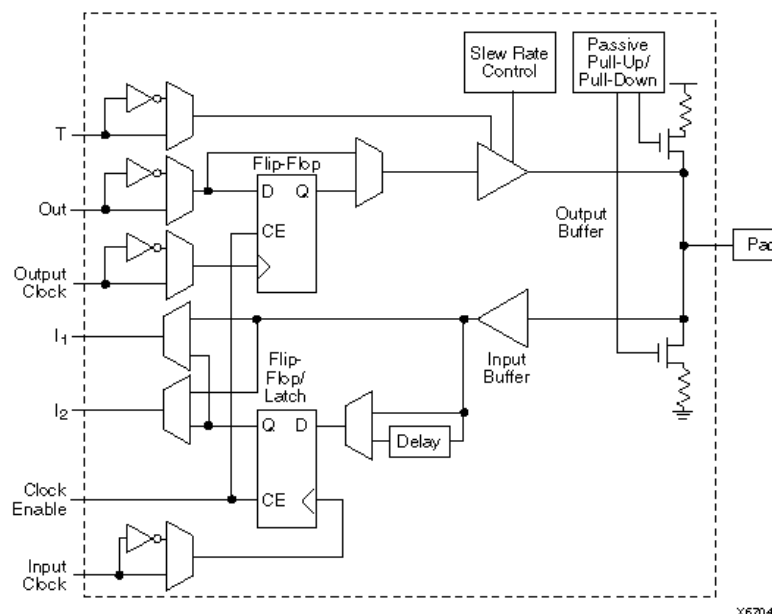
Les sorties de ces deux bascules correspondent aux sorties du CLB XQ et YQ. Un mode dit de "verrouillage" exploite une entrée S/R qui peut être programmée soit en mode SET, mise à 1 de la bascule, soit en Reset, mise à zéro de la bascule. Ces deux entrées coexistent avec une autre entrée laquelle n'est pas représentée sur la figure III.6 appelée le global Set/Reset. Cette entrée initialise le circuit FPGA à chaque mise sous tension, à chaque configuration, en commandant toutes les bascules au même instant soit à '1', soit à '0'. Elle agit également lors d'un niveau actif sur le fil RESET lequel peut être connecté à n'importe quelle entrée du circuit FPGA.

Un mode optionnel des CLB est la configuration en mémoire RAM de  $16 \times 2$ bits ou  $32 \times 1$ bit. Les entrées F1 à F4 et G1 à G4 deviennent des lignes d'adresses sélectionnant une cellule mémoire particulière .

La fonctionnalité des signaux de contrôle est modifiée dans cette configuration, les lignes H1, DIN et S/R deviennent respectivement les deux données D0, D1 (RAM  $16 \times 2$ bits) d'entrée et le signal de validation d'écriture WE. Le contenu de la cellule mémoire (D0 et D1) est accessible aux sorties des générateurs de fonctions F' et G'. Ces données peuvent sortir du CLB à travers ses sorties X et Y ou alors en passant par les deux bascules.

### Les IOB (Input Output Block)

La figure présente la structure de ce bloc. Ces blocs 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 IOB contrôle une broche du composant et il peut être défini en entrée, en sortie, en signaux bidirectionnels ou être inutilisé (haute impédance).



**Figure III.7 – Input Output Block (IOB)[10]**

### III.2.6 La configuration en entrée et en sortie

#### Configuration en entrée

Premièrement, le signal d'entrée traverse un buffer qui selon sa programmation peut détecter soit des seuils TTL ou soit des seuils CMOS. Il peut être routé directement sur une entrée directe de la logique du circuit FPGA ou sur une entrée synchronisée. Cette synchronisation est réalisée à l'aide d'une bascule de type D, Le changement d'état peut se faire sur un front montant ou descendant. De plus, cette entrée peut être retardée de quelques nanosecondes pour compenser le retard pris par le signal d'horloge lors de son passage par l'amplificateur. Le choix de la configuration de l'entrée s'effectue grâce à un multiplexeur (program controlled multiplexer). Un bit positionné dans une case mémoire commande ce dernier.

#### Configuration en sortie

Nous distinguons les possibilités suivantes :

- Inversion ou non du signal avant son application à l'IOB,
- Synchronisation du signal sur des fronts montants ou descendants d'horloge,
- Mise en place d'un " pull-up " ou " pull-down " dans le but de limiter la consommation des entrées sorties inutilisées,
- Signaux en logique trois états ou deux états. Le contrôle de mise en haute impédance et la réalisation des lignes bidirectionnelles sont commandés par le signal de commande Out Enable lequel peut être inversé ou non. Chaque sortie peut délivrer un courant de 12mA. Ainsi toutes ces possibilités permettent au concepteur de connecter au mieux une architecture avec les périphériques extérieurs.

### III.2.7 Les différents types d'interconnexions

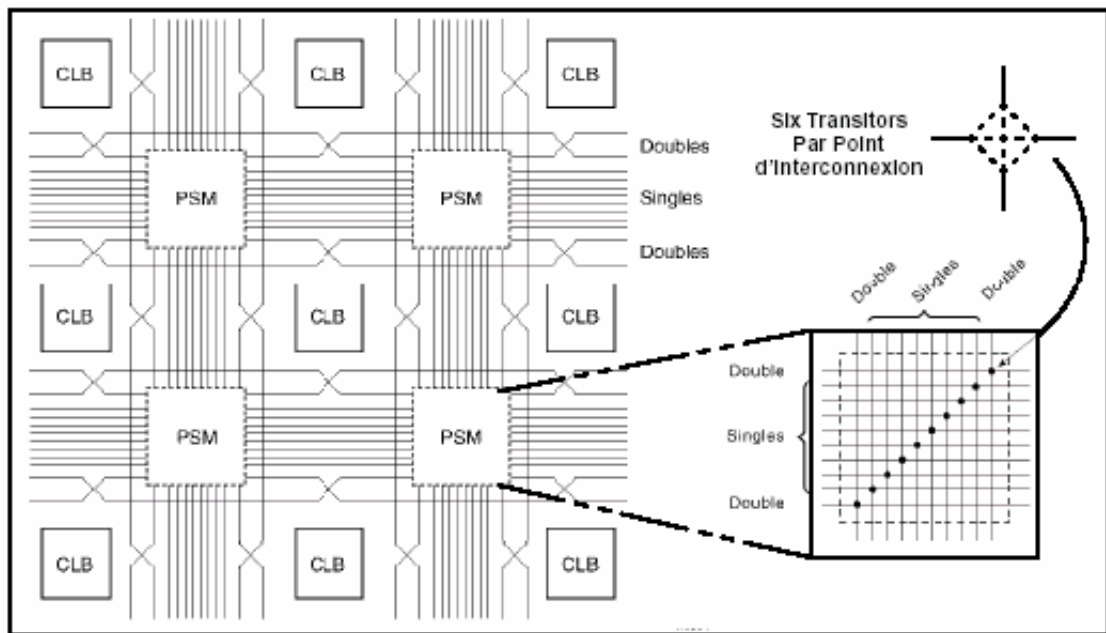
Les connexions internes dans les circuits FPGA sont composées de segments métallisés. Parallèlement à ces lignes, nous trouvons des matrices programmables réparties sur la totalité du circuit, horizontalement et verticalement entre les divers CLB. Elles permettent les connexions entre les diverses lignes. Celles-ci sont assurées par des transistors MOS dont l'état est contrôlé par des cellules de mémoire vive ou RAM. Le rôle de ces interconnexions est de relier avec un maximum d'efficacité les blocs logiques et les entrées/sorties afin que le taux d'utilisation dans un circuit donné soit le plus élevé possible. Pour parvenir à cet objectif, Xilinx propose trois sortes d'interconnexions selon la longueur et la destination des liaisons. Nous disposons des interconnexions suivantes :

- Interconnexions à usage général,
- Interconnexions directes,
- Les lignes Longues .

#### Les interconnexions à usage général

Ce système fonctionne en une grille de cinq segments métalliques verticaux et quatre segments horizontaux positionnés entre les rangées et les colonnes de CLB et de l'IOB. Des aiguilleurs appelés aussi matrices de commutation sont situés à

chaque intersection. Leur rôle est de raccorder les segments entre eux selon diverses configurations, ils assurent ainsi la communication des signaux d'une voie sur l'autre. Ces interconnexions sont utilisées pour relier un CLB à n'importe quel autre. Pour éviter que les signaux traversant les grandes lignes ne soient affaiblis, nous trouvons généralement des buffers implantés en haut et à droite de chaque matrice de commutation.



**Figure III.8** – Connexions à usage général et matrice de commutation

### Les interconnexions directes

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

Pour chaque bloc logique configurable, la sortie X peut être connectée directement aux entrées C ou D du CLB situées au-dessus et les entrées A ou B du CLB situées au-dessous. Quant à la sortie Y, elle peut être connectée à l'entrée B du CLB placé immédiatement à sa droite. Pour chaque bloc logique adjacent à un bloc entrée/sortie, les connexions sont possibles avec les entrées I ou les sorties O suivant leur position sur le circuit.

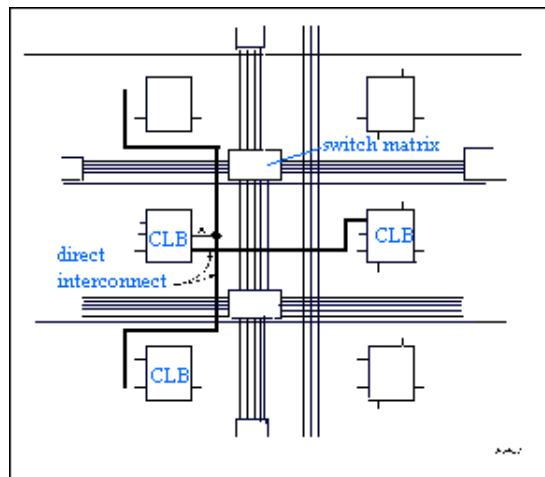


Figure III.9 – Les interconnexions directes

### Les lignes longues

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

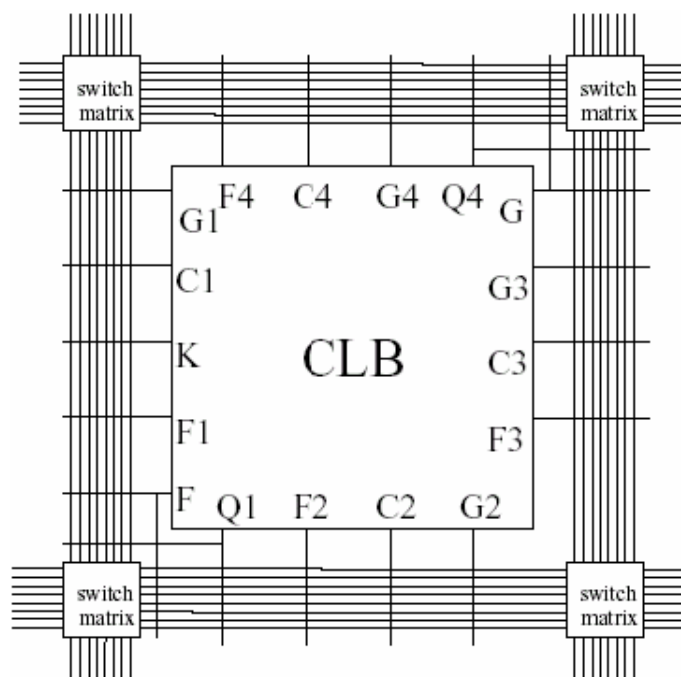


Figure III.10 – Les lignes longues

### III.3 Configuration des FPGA

Les circuits FPGA se configurent selon plusieurs modes à titre d'exemple on cite les plus utilisés par XILINX [32] :

1. Le mode série qui utilise une EEPROM série caractérisée par sa capacité considérable et faible rapidité.
2. Le mode parallèle 8 bits utilisant une EPROM classique qui est plus rapide.
3. Le mode parallèle asynchrone : le circuit FPGA se comporte comme un périphérique de microprocesseur ce qui lui offre une possibilité de reconfiguration partielle intelligente.

Dans chaque mode, plusieurs FPGA peuvent être chaînés en mode maître/esclave.

### III.4 Outils de développement des FPGA

Les outils de développement permettent au concepteur de programmer le circuit à partir de la description de la fonction à réaliser.

La description du fonctionnement des circuits peut se faire de plusieurs façons, soit :

- Par un schéma à base de fonctions logique élémentaires (Portes ET,OU,NON, ... bascules, compteurs, registres à décalages).
- En utilisant un langage de description comportementale H.D.L. (Hardware Description Language). Les plus anciens sont PALASM, ORCAD/PLD, ABEL (utilisé par la plus part des systèmes de développements). Enfin les langages dit de haut niveau, VHDL (Very high speed Hardware Description Language) et VERILOG sont en général utilisés pour des circuits complexes[40].
- En utilisant un schéma et des descriptions en langage de haut niveau de type VHDL.
- Par l'utilisation de graphes d'états. (FSM : Flow States Machines).

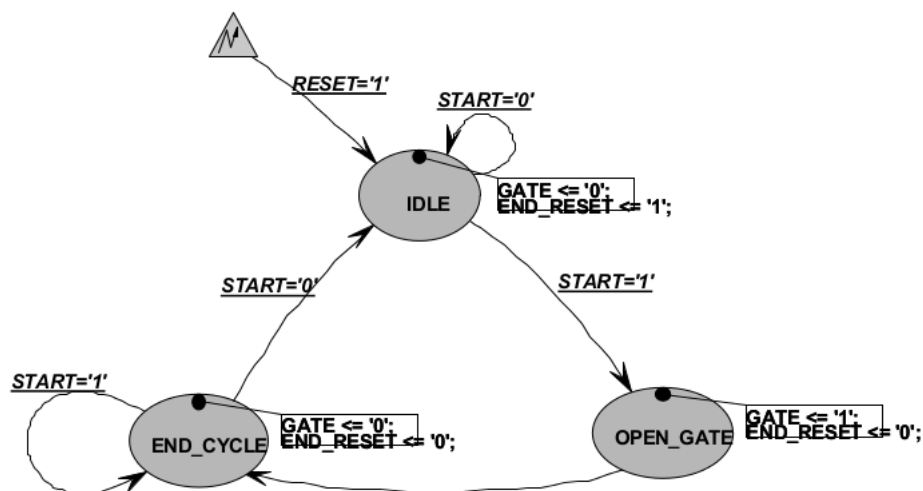


Figure III.11 – Exemple d'un graphe des états

Notre application a été réalisée en utilisant le langage VHDL qui est un langage de description hardware de haut-niveau.

## III.5 Langage VHDL

### III.5.1 Introduction

Cette introduction au VHDL n'a pas la prétention de couvrir tous les aspects de ce langage. En fait, elle suppose une connaissance préalable du langage ADA. Nous nous attarderons d'avantage sur une méthode de travail avec le VHDL que sur la grammaire de ce langage. Il existe pour cette dernière une quantité de livres dont certains sont cités dans la bibliographie. De plus, pour une référence sûre, il est également possible de commander la norme VHDL 1993 (IEEE 1164).

### III.5.2 Bref historique

Au début des années 80, le département de la défense américaine (DOD) désire standardiser un langage de description et de documentation des systèmes matériels ainsi qu'un langage logiciel afin d'avoir une indépendance vis-à-vis de leurs fournisseurs. C'est pourquoi, le DOD a décidé de définir un langage de spécification. Il a ainsi mandaté des sociétés pour établir un langage. Parmi les langages proposés, le DOD a retenu le langage VHDL qui fut ensuite normalisé par IEEE. Le langage ADA est très proche, car celui-ci a servi de base pour l'établissement du langage VHDL. La standardisation du VHDL s'effectuera jusqu'en 1987, époque à laquelle elle sera normalisée par l'IEEE (Institute of Electrical and Electronics Engineers). Cette première normalisation a comme objectif :

- La spécification par la description de circuits et de systèmes.
- La simulation afin de vérifier la fonctionnalité du système.
- La conception afin de tester une fonctionnalité identique mais décrite avec des solutions d'implémentations de différents niveaux d'abstraction.

En 1993, une nouvelle normalisation par l'IEEE du VHDL a permis d'étendre le domaine d'utilisation du VHDL vers :

- La synthèse automatique de circuit à partir des descriptions.
- La vérification des contraintes temporelles.
- La preuve formelle d'équivalence de circuits.

Il existe un autre langage de haut niveau pour la description de matériel. Il s'agit du langage VERILOG. La société Cadence est à l'origine de ce langage. Ce langage est aussi normalisé par IEEE

### III.5.3 Utilité du VHDL

Le VHDL est un langage normalisé, cela lui assure une pérennité. Il est indépendant d'un fournisseur d'outils. Il est devenu un standard reconnu par tous les vendeurs d'outils EDA. Cela permet aux industriels d'investir sur un outil qui n'est pas qu'un mode éphémère, c'est un produit commercialement inévitable. Techniquement, il est incontournable car c'est un langage puissant, moderne et qui permet une excellente lisibilité, une haute modularité et une meilleure productivité des descriptions. Il permet de mettre en oeuvre les nouvelles méthodes de conception.



Il est à noter toutefois un inconvénient qui est la complexité du langage. En effet, ce langage s'adresse à des concepteurs de systèmes électroniques, qui n'ont pas forcément de grandes connaissances en langages de programmation.

D'autres fausses idées circulent sur le langage VHDL. Celui-ci n'assure pas la qualité du résultat, la portabilité et la synthèse des descriptions. Une méthodologie est indispensable pour combler ces lacunes.

### III.5.4 Spécification

Établi en premier lieu pour la spécification, c'est dans ce domaine que la norme est actuellement la mieux établie. Il est tout-à-fait possible de décrire un circuit en un VHDL standard pour qu'il soit lisible de tous. Certains fabricants (de circuits ou de CAO) adaptent ce langage pour donner à l'utilisateur quelques facilités supplémentaires, au détriment de la portabilité du code. Heureusement, il y a une nette tendance de la part des fabricants à revoir leurs positions et à uniformiser le VHDL. Le rapprochement se fait, comme précité, autour du VHDL de Synopsys. Il est donc probable que l'on s'approche d'un vrai standard VHDL et non plus d'un standard théorique. Il y aura toujours des ajouts de la part des fabricants, mais il ne s'agira plus d'une modification du langage (aussi légère soit elle), mais de macros offertes à l'utilisateur pour optimiser le code VHDL en fonction du circuit cible (en vue de la synthèse).

Cette possibilité de décrire des circuits dans un langage universel est aussi très pratique pour éviter les problèmes de langue. De longues explications dans une langue peuvent ainsi être complétées par du code VHDL pour en faciliter la compréhension.

### III.5.5 Simulation

Le VHDL est également un langage de simulation. Pour ce faire, la notion de temps, sous différentes formes, y a été introduite. Des modules, destinés uniquement à la simulation, peuvent ainsi être créés et utilisés pour valider un fonctionnement logique ou temporel du code VHDL.

La possibilité de simuler avec des programmes VHDL devrait considérablement faciliter l'écriture de tests avant la programmation du circuit et éviter ainsi de nombreux essais sur un prototype qui sont beaucoup plus coûteux et dont les erreurs sont plus difficiles à trouver.

Bien que la simulation offre de grandes facilités de test, il est toujours nécessaire de concevoir les circuits en vue des tests de fabrication, c'est-à-dire en permettant l'accès à certains signaux internes.

### III.5.6 Conception

Le VHDL permet la conception de circuits avec une grande quantité de portes. Les circuits actuels comprennent, pour les FPGA par exemple, entre 500 et 100'000

portes et ce nombre augmente très rapidement. L'avantage d'un langage tel que celui-ci par rapport aux langages précédents de conception matérielle est comparable à l'avantage d'un langage informatique de haut niveau (Pascal, ADA, C) vis-à-vis de l'assembleur. Ce qui veut aussi dire que malgré l'évolution fulgurante de la taille des circuits, la longueur du code VHDL n'a pas suivi la même courbe. Cependant, ce langage étant conçu en premier lieu pour de la spécification, certaines variantes du langage ne sont pour l'instant pas utilisables pour la conception.

Il faut noter que le VHDL, bien que facilement accessible dans ses bases, peut devenir extrêmement compliqué s'il s'agit d'optimiser le code pour une architecture de circuit. C'est pour cette raison que de plus en plus de fabricants offrent des macros, gratuites pour les fonctions sans grandes difficultés et payantes pour les autres. Donc avant de concevoir une ALU, un processeur RISC, une interface PCI ou d'autres éléments de cette complexité, il peut être judicieux de choisir un circuit cible en fonction des besoins et d'acheter la macro offerte par le constructeur. Il est bien évident qu'il faudra évaluer les besoins (performance du code nécessaire, quantité de pièces à produire) et le coût d'une telle macro.

### III.5.7 Unité de conception (module VHDL)

L'unité de conception est un ensemble d'éléments VHDL avec lesquels nous allons décrire un système numérique. Celui-ci peut-être constitué d'une simple porte logique jusqu'à un système complexe. Nous parlerons aussi de module VHDL. L'unité de conception est constituée d'une entité (définit l'interface), une ou plusieurs architectures (défini le fonctionnement), des bibliothèques et de la configuration. Les bibliothèques regroupent un ensemble de définitions, déclarations, fonctions, etc.. nécessaire à tous les modules.

#### L'entité

L'entité est une vue externe du module, elle définit toutes les entrées et sorties. Syntaxe générale de l'entité :

```
entity NomDuModule is
port ( NomEntrée 1 :inType Du Signal;
Nom Entrée 2 :inType Du Signal;
Nom Sortie 1 :outType Du Signal;_ Nom E S 1 :inoutType Du Signal );
end Nom Du Module;
```

#### Remarque :

Les entrées et les sorties d'un module sont toujours représentées par un port, il en existe 4 différents types, qui sont :

- in : port d'entrée.
- out : port de sortie.
- inout : port bidirectionnel (utilisation pour des bus).
- buffer : port de sortie qui peut être relu

Un signal de type buffer ne peut pas être connecté directement avec un signal de

type out (le VHDL est typé). Il est nécessaire d'utiliser un cast. Cela n'est pas très pratique. Nous déconseillons l'utilisation de ce type. Nous verrons plus tard comment se passer de ce type en utilisant un signal interne dans l'architecture.

## L'architecture

L'architecture décrit le comportement du module. Le langage VHDL permet de décrire le comportement de différentes façons. Nous allons traiter les descriptions synthétisables.

Nous allons commencer par les descriptions synthétisables. Dans la littérature, ce niveau de description est appelé RTL (Register Transfert Level). Nous distinguerons différentes possibilités de descriptions synthétisables, soit :

- **Logique** : Equations logiques, description du fonctionnement directement avec des équations logiques.
- **TDV** : Table de vérité, description du fonctionnement avec une table de vérité (utilisation de l'instruction `with ... select`).
- **Flot Don** : Flot de données, description du fonctionnement très proche du fonctionnement logique (utilisation des instructions concurrentes "`<<`", "`when ... else` ou `with ... select`)
- **Comport** : Comportementale, description du fonctionnement en décrivant le comportement (exemple :description avec un process et de l'instruction `if... then ... else`)
- **M Etat** : Machine d'état, description du fonctionnement à l'aide d'une machine d'état (diagramme d'états).
- **Struct** : Structurelle, description du module en donnant les interconnexions entre différents sous modules (description hiérarchique avec l'instanciation de composant)

## Syntaxe générale d'une architecture

Voici la syntaxe d'une architecture.

```
architecture Type D Architecture of Nom Du Module is
--Zone de déclaration
begin_ Instructions Concurrentes;
process ( Liste De Sensibilité )
begin
Instructions Séquentielles;
end process;
end Type D Architecture;
```

Dans un circuit logique, toutes les portes fonctionnent simultanément. On dit alors que les portes fonctionnent de manière concurrente, c'est à dire que l'ensemble des opérations se déroulent en parallèle. Il est simple de comprendre que toutes les parties d'un circuit fonctionnent simultanément. Le langage VHDL a été créé dans le but de pouvoir décrire le comportement d'un circuit. Le langage dispose donc d'instructions concurrentes. Cette notion est particulière au langage de description de matériel. Elle n'existe pas dans les langages de programmation conventionnel.

Parfois, il est plus performant de décrire le comportement en utilisant un algorithme en utilisant une instruction tel que le `if .. then .. else`. C'est la raison de l'existence, dans le langage VHDL, d'une instruction concurrente particulière, l'instruction `process`. Celle-ci permet de décrire le comportement du circuit avec des instructions séquentielles. A l'intérieur de cette instruction le déroulement des instructions est séquentiel.

### Les paquetages

Le langage VHDL doit son succès grâce aux notions de bibliothèque et de paquetage. Il est ainsi possible de fournir un ensemble de fonctionnalités rassemblées dans un paquetage.

C'est le cas par exemple des définitions des types standards indispensables en synthèse, soit :

```
Library IEEE;
use IEEE.Std logic 1164.all;
```

Il sera aussi possible de définir son propre paquetage afin de regrouper des définitions, des sous-programmes, .. pour son projet.

### III.5.8 Les deux modes de travail en VHDL

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

#### Le mode combinatoire

En mode combinatoire (ou concurrent), toutes les instructions d'une description VHDL sont évaluées et affectent les signaux de sortie en même temps, l'ordre dans lequel les instructions sont écrites n'a donc aucune importance. En effet la description génère des structures électroniques, c'est la grande différence entre une description VHDL et un langage informatique classique.

Dans un système à microprocesseur, les instructions sont exécutées les unes à la suite des autres. Avec VHDL il faut essayer de penser à la structure qui va être générée par le synthétiseur pour écrire une bonne description, cela n'est pas toujours évident.

#### Le mode séquentiel

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

microprocesseurs.

Dans le mode séquentiel, on distingue :

- Le mode séquentiel asynchrone, dans lequel les changements d'état des sorties peuvent se produire à des instants difficilement prédictibles (retards formés par le basculement d'un nombre souvent inconnu de fonctions),
- Le mode séquentiel synchrone, dans lequel les changements d'état des sorties se produisent tous à des instants quasiment connus (un temps de retard après un front actif de l'horloge).

### III.6 Etapes de réalisation d'un projet autour d'un circuit FPGA

L'expansion du monde des circuits logiques programmables et en particulier des circuits FPGAs est étroitement liée au développement de la technologie de fabrication des circuits intégrés permettant la fabrication d'unités de plus en plus denses. Mais cette expansion n'aurait jamais pu avoir une telle ampleur sans le développement, en parallèle, d'outils performants permettant de faciliter le processus de conception des circuits logiques avec de la logique programmable. Ces outils assurent d'une part la description aisée des circuits logiques complexes et d'autre part la réalisation automatique de circuit à partir de ces descriptions. La réalisation d'un circuit logique, en fait le passage de sa description à l'information nécessaire pour la réalisation d'une implémentation physique du circuit dans un FPGA (i.e la configuration), est dans la littérature anglo-saxonne appelée design flow. Ce processus est habituellement décomposé en plusieurs étapes distinctes. Actuellement, grâce aux outils informatisés, certaines étapes de ce processus peuvent être réalisées de façon entièrement automatique (notamment la synthèse, l'implémentation ainsi que la génération de la configuration pour le FPGA cible) [12] [43].

#### III.6.1 Description

La première étape du design flow comprend la description formelle du circuit à partir des spécifications. Aujourd'hui la description des circuits logiques est faite la plupart de temps à l'aide des langages de description de matériel (Hardware Description Language - HDL) à haut niveau d'abstraction tels que VHDL ou Verilog. Souvent, lorsque l'utilisateur souhaite rester proche du matériel afin d'obtenir la meilleure performance possible de son circuit, la description est faite de façon schématique. Il existe d'autres formalismes permettant de décrire un circuit logique, souvent proposés par un producteur de FPGA particulier, comme par exemple l'éditeur graphique des machines à états finis (FSM) chez Xilinx. La description d'un circuit peut être faite exclusivement à l'aide d'un formalisme bien particulier, ou combiner des formalismes différents. Les différentes parties d'un même circuit logique complexe peuvent donc être décrites de façons différentes avant d'être traduites en une configuration particulière pour un (ou plusieurs) FPGA cible.

## III.6. ETAPES DE RÉALISATION D'UN PROJET AUTOUR D'UN CIRCUIT FPGA

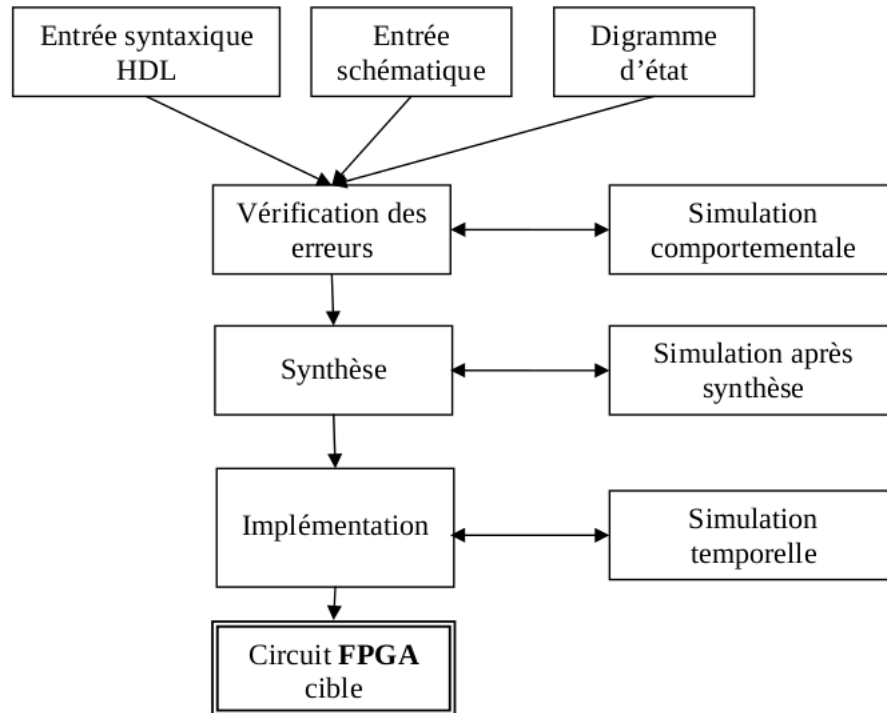


Figure III.12 – Etapes de réalisation d'un projet sur circuit FPGA (design flow)

### III.6.2 Vérification des erreurs

La vérification fonctionnelle de la description se fait à l'aide d'un simulateur, en l'occurrence le logiciel ModelSim. A la description de circuit sont associés les différents stimuli sous forme des bancs d'essai, décrits également en langage VHDL. La description du circuit et le fichier de stimulus sont ensuite passés au logiciel de simulation, permettant d'effectuer une vérification fonctionnelle de la description.

### III.6.3 Synthèse

Lors de l'étape de la synthèse, les différentes descriptions établies lors de la première étape sont ramenées à une description du circuit unique, optimisée pour une performance maximale ou une occupation minimale des ressources. Après l'étape de la synthèse il est possible d'effectuer une analyse fonctionnelle du circuit (vérification de la description à travers la simulation) et de se rendre compte d'éventuelles erreurs, avant de passer à l'étape de l'implémentation. L'analyse fonctionnelle n'est qu'une vérification partielle car elle ne peut tenir compte des délais introduits par le circuit réel, celui-ci n'étant pas encore réalisé. Lors d'une analyse fonctionnelle tous les délais du circuit obtenu après la synthèse sont considérés comme nuls. Après la vérification de la description, la synthèse du circuit peut être effectuée à l'aide du programme XST - Xilinx Synthesis Technology. Outre la possibilité de visualiser le circuit sous forme d'un schéma RTL, nous pouvons disposer d'une première estimation de la performance pour le circuit décrit. Bien entendu la performance donnée ne sera qu'une indication, car la performance réelle ne pourra être obtenue qu'après le placement et le routage définitifs [41] [42].

### III.6.4 Implémentation

L'étape de synthèse fourni au processus d'implémentation un fichier contenant la description physique de circuit, sous forme d'un ensemble de primitives (briques de base) des circuits FPGAs Xilinx (il s'agit d'un fichier portant une extension NGD, Native Generic Database). Mis à part la description du circuit, le processus d'implémentation proprement dit, en fait les trois sous-étapes : translate, map et place and route. Le mapping consiste à traduire le circuit logique obtenu par le processus de synthèse en mémoires des cellules logiques élémentaires d'un circuit FPGA particulier. Lors de la sous-étape de placement, une cellule logique existante, avec sa position unique au sein du FPGA cible, est attribuée à chaque cellule logique nécessaire pour la réalisation du circuit. Ceci est bien entendu fait sous la contrainte des longueurs minimales des connexions entre les différentes cellules. Enfin la dernière sous-étape, le routage, consiste à établir les différentes connexions physiques entre les cellules élémentaires placées. Ces étapes nécessitent la définition d'un ensemble de paramètres permettant de contrôler leur déroulement. Ces paramètres peuvent être classés en deux groupes distincts : les paramètres des programmes d'implémentation et les contraintes de la performance.

#### Paramètres des programmes d'implémentation

Les paramètres liés aux différents programmes d'implémentation sont introduits via l'interface graphique de la plate-forme de développement ISE. Il s'agit principalement : de l'effort que le programme de placement et de routage va fournir afin d'obtenir la performance du circuit souhaité et le choix d'une des 100 différentes tables de coûts utilisée pour le placement. L'influence de choix d'une table de coût sur la performance du circuit définitif peut être significative car chaque table produira un placement particulier. Les expériences ont montré qu'une différence en performance de 30 même description de circuit mais pour les différentes tables de coût. Lorsque l'on souhaite pousser la performance d'un circuit au maximum, alors l'implémentation peut être faite pour un certain nombre de tables de coûts (ou toutes) dans une implémentation dite Multi-Pass Place and Route - MPR. La table de coût offrant la meilleure performance serait alors retenue pour le placement et le routage définitifs.

#### Les contraintes

Les contraintes sur la performance de circuit que l'on souhaite obtenir sont introduites dans le processus d'implémentation à l'aide des applications graphiques Constraint Editor et/ou PACE, ou tout simplement à la main, à l'aide d'un fichier texte contenant les descriptions des contraintes avec une syntaxe particulière. Pour les circuits synchrones, la spécification des contraintes peut être faite de manière globale à l'aide de trois types de contraintes différentes :

- Période minimale (fréquence maximale) : Cette contrainte agit sur le délai maximal qui peut y avoir entre tout élément synchrone source et tout élément synchrone destination. C'est donc la fréquence maximale avec laquelle le circuit synchrone peut fonctionner dans le FPGA.
- Contrainte sur les entrées (Pad to setup time) : Comme le circuit FPGA reçoit les données du monde extérieur, il faut également tenir compte de délais liés au

## III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA

transit de données depuis la frontière physique du FPGA (les pattes d'entrées) jusqu'au premier élément synchrone. Même si le premier élément synchrone se trouve dans un IOB du FPGA, ces délais sont néanmoins à prendre en compte.

- Contrainte sur les sorties (Clock to Pad) : Le même raisonnement est fait pour les sorties du FPGA.

### III.6.5 Configuration

Le processus de design flow se termine par la génération du fichier de configuration correspondant au circuit réalisé. Ce fichier peut être directement transféré dans le circuit FPGA cible depuis la plate-forme de développement (en utilisant l'interface JTAG, ou USB dans notre cas) ou depuis une mémoire programmable (ISP PROM) dans le cas d'une solution embarquée. Le circuit conçu devient alors opérationnel [44].

## III.7 Implémentation de la commande sur un circuit FPGA

### III.7.1 présentation de la carte utilisé

la carte utilisé est une Spartan 3E de Xilinx (figure III.13).

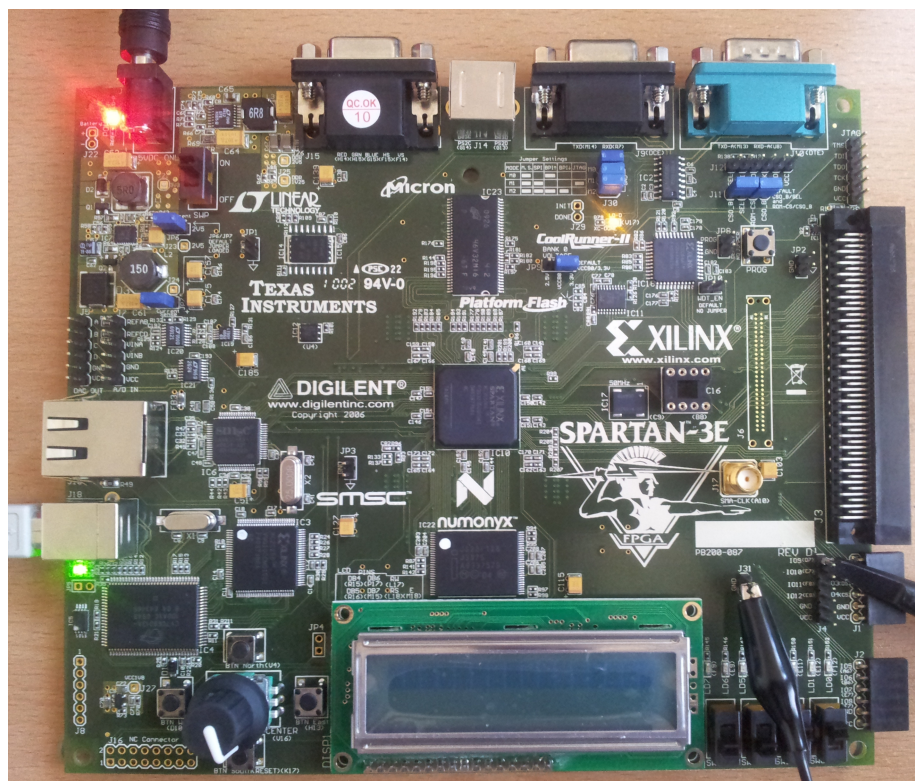


Figure III.13 – carte de développement utilisée



### III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA

---

Les dispositifs principaux de la carte Spartan-3E sont :

- Xilinx XC3S500E Spartan-3E FPGA
  - Jusqu'à 232 bornes d'entrée/sortie utilisateur.
  - 320 pins FBGA package.
  - Plus de 10.000 cellules logiques.
- Mémoire PROM de Xilinx à 4 Mbit de configuration instantané.
- 64 macrocell XC2C64A CoolRunner™ CPLD de Xilinx.
- DDR SDRAM (512 Mbit) , interface des données ×16.
- Flash NOR parallèle 16 Mbit (128 Mbit) (Intel StrataFlash).
  - Stockage de la configuration du circuit FPGA.
  - Stockage du code de MicroBlaze/filature.
- flash série SPI de 16 Mbits (STMicro).
  - Stockage de configuration de FPGA.
  - Filature de code de MicroBlaze.
- 2 lignes, d'écran LCD de 16 caractères.
- Port PS/2 (de la souris ou de clavier).
- Port d'affichage VGA.
- Port Ethernet PHY 10/100 (exige le MAC d'Ethernet dans FPGA)
- Deux 9 ports d RS-232 (DTE et DCI-modèle)
- Port USB.
- Horloge de 50 mégahertz.
- EEPROM périodique pour le système anti-copie de bitstream.
- Connecteur d'expansion de Hirose FX2.
- Quatre convertisseur numérique-analogique SPI (DAC).
- Deux entrée de convertisseur analogique-numérique SPI-basé (CDA).
- Port d'élimination des imperfections de ChipScope™ SoftTouch.
- Rotatoire-encodeur avec l'axe à bouton-poussoir.
- Huit LED discrètes.
- Quatre contacts coulissants.
- Quatre commutateurs à bouton-poussoir.
- Entrée d'horloge de SMA.
- douille d'IMMERSION de 8 bornes pour l'oscillateur auxiliaire d'horloge.

### III.7.2 Description de l'algorithme implémenté

#### Introduction

Le but de notre projet est d'implémenter la commande off-line de l'onduleur multi-niveau sur un circuit FPGA. Pour cela on utilise le langage VHDL. Dans une première étape on stocke les angles de commutation calculées précédemment dans la mémoire de la carte, Dans l'étape suivante on réalise un programme qui à partir des angles de commutation donne les six signaux de commande d'une phase, les signaux des autres phases ont la même forme sauf qu'ils sont décalés de  $120^\circ$  pour chaque phase. Dans la dernière étape on essaye de simuler ces signaux de commande obtenues.

#### Transformation des angles de commutation

Le calcul des angles de commutation exactes a été fait par la résolution du système non linéaire à l'aide de la méthode de la Résultante décrite dans le chapitre 2. On stocke dans l'ordre ces résultats sur la carte pour des valeurs différentes du taux de modulation  $r$ .

Pour programmer les signaux de commande, il faut transformer ces angles  $\alpha(i)$  en degrés à ses instants temporels équivalents  $T(i)$ . On a :

$$\alpha(i) = 2 \times 180 \times f \times T(i) \quad (\text{III.1})$$

$$T(i) = \alpha(i)/(2 \times 180 \times f) \quad (\text{III.2})$$

Pour avoir le rapport  $V/f$  constant, on a :

$$V/V_n = r \quad (\text{III.3})$$

Où  $r$  est le taux de modulation, et  $V_n$  est la tension maximale qui est égale à :

$$\frac{3}{4\pi} \times V_{dc} \quad (\text{III.4})$$

$V_{dc}$  est la tension de la source.

Et on a

$$f/f_n = r \quad (\text{III.5})$$

D'où

$$V/f = (V_n \times r)/(f_n \times r) = V_n/f_n = \text{constante} \quad (\text{III.6})$$

Des équations III.2 à III.6, on déduit :

$$f = V \times (f_n/V_n) = (r \times V_n) \times (f_n/V_n) = r \times f_n \quad (\text{III.7})$$

l'équation III.2 devient

$$T(i) = \alpha(i)/(2 \times 180 \times r \times f_n) \quad (\text{III.8})$$

avec  $T(i)$  en seconde,  $f_n$  est la fréquence maximale qui est égale à 50 Hz.

### III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA

#### Génération des signaux de commande

Pour la génération des signaux de commande, un compteur est utilisé. Chaque fois que les valeurs stockées sont atteintes par le compteur le signal de commande correspondant commute.

Le choix de la fréquence du compteur dépend de la précision voulue, pour notre cas on choisit une fréquence de 1MHZ pour laquelle on aura une précision de  $1\mu s$

On commence par la génération du compteur désiré, et pour cela on utilise un diviseur de fréquence variable dont son principe est présenté par l'organigramme suivant (figureIII.14)

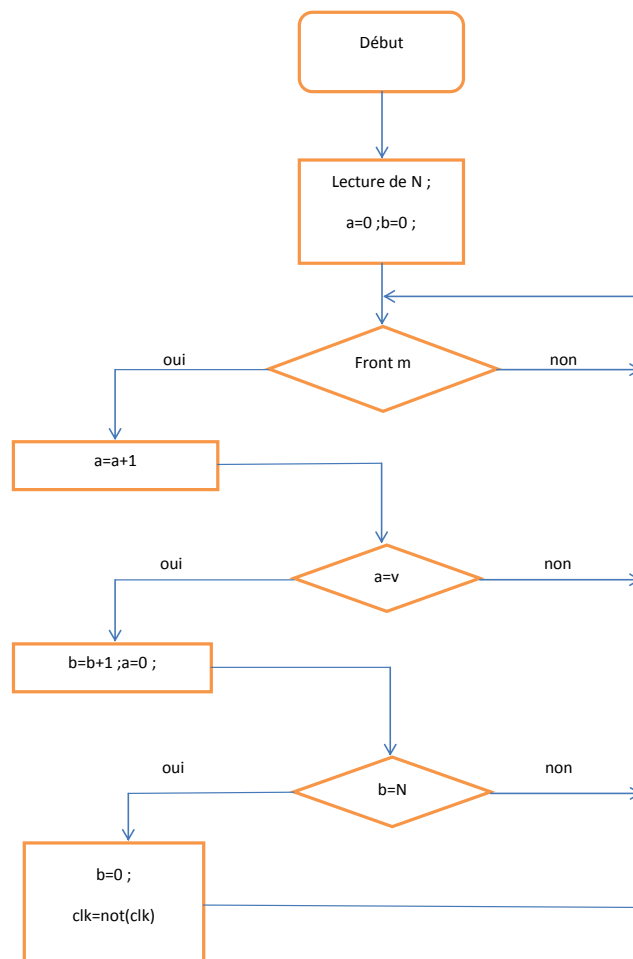


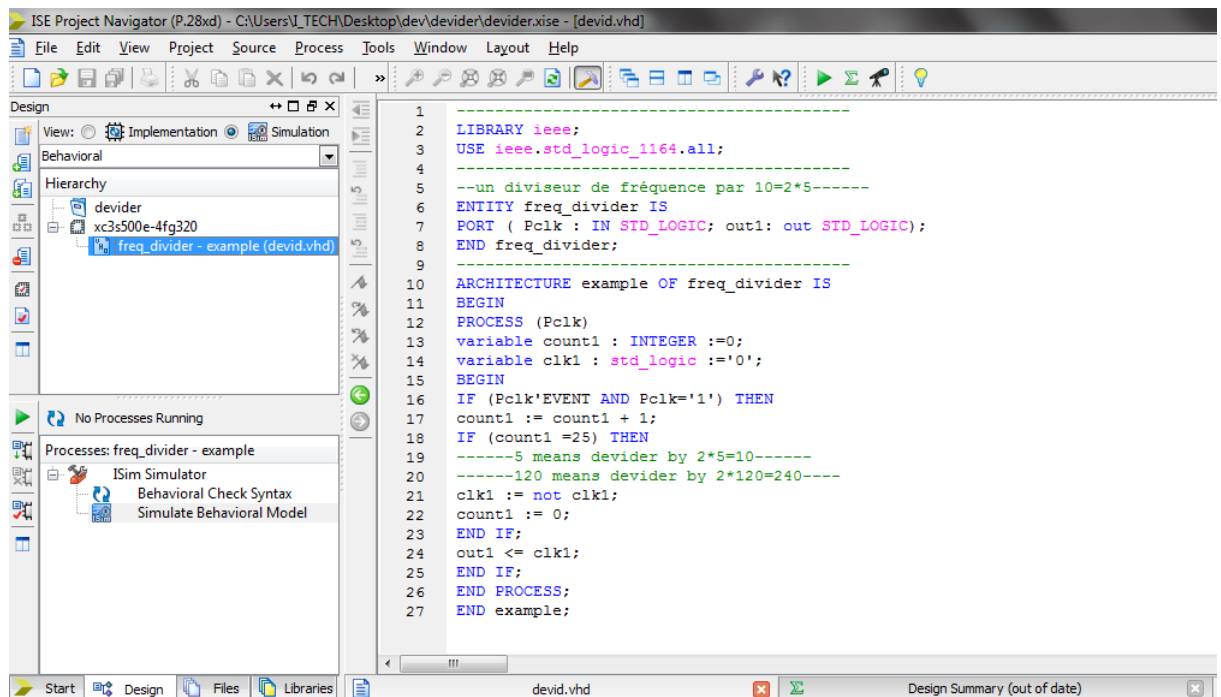
Figure III.14 – Organigramme du diviseur de fréquence

### III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA

à la sortie du programme, la fréquence résultante clk est donnée par :

$$f_r = \frac{f_q}{2 \times V \times N} \quad (\text{III.9})$$

pour vérifier la validité du programme on fait une simulation en utilisant le Xilinx ISE 14 et son simulateur Isim. La figure III.15 représente le code VHDL du programme dans Xilinx ISE 14, alors que la figure III.16 représente les résultats de la simulation.



```
1  -----
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  -----
5  --un diviseur de fréquence par 10=2*5-----
6  ENTITY freq_divider IS
7  PORT ( Pclk : IN STD_LOGIC; out1: out STD_LOGIC);
8  END freq_divider;
9  -----
10 ARCHITECTURE example OF freq_divider IS
11 BEGIN
12 PROCESS (Pclk)
13 variable count1 : INTEGER :=0;
14 variable clk1 : std_logic :='0';
15 BEGIN
16 IF (Pclk'EVENT AND Pclk='1') THEN
17 count1 := count1 + 1;
18 IF (count1 =25) THEN
19 -----5 means divider by 2*5=10-----
20 -----120 means divider by 2*120=240----
21 clk1 := not clk1;
22 count1 := 0;
23 END IF;
24 out1 <= clk1;
25 END IF;
26 END PROCESS;
27 END example;
```

Figure III.15 – Le code VHDL du programme du diviseur de fréquence

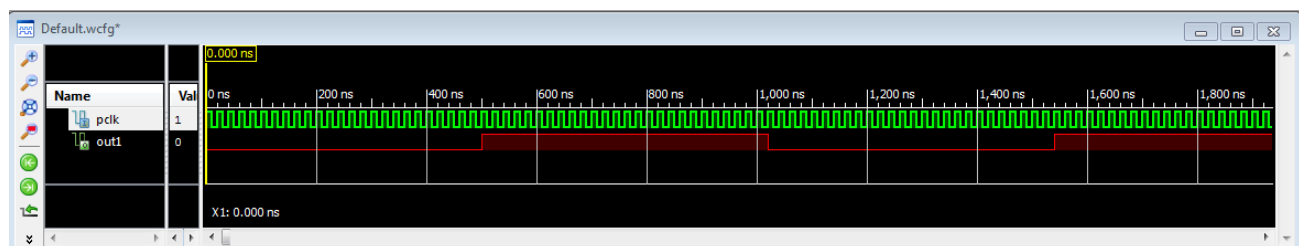
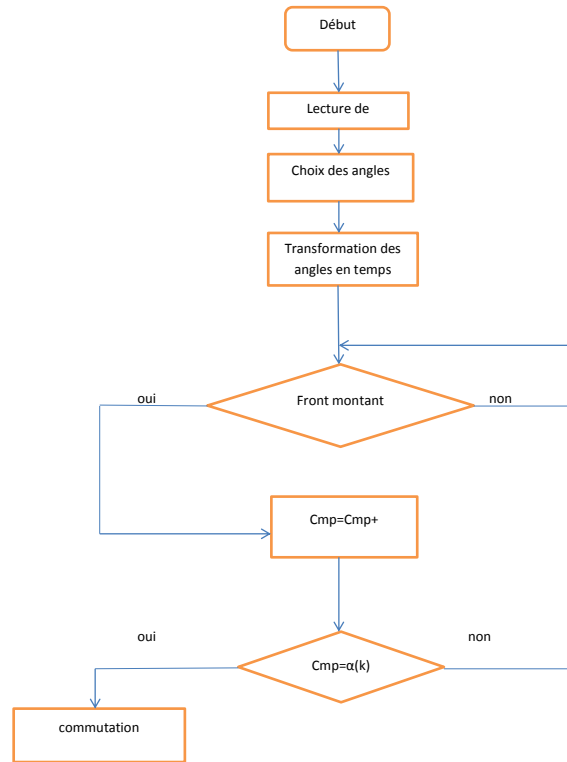


Figure III.16 – Le résultat de la simulation du diviseur

On voit bien que la fréquence du signal d'entrée a été divisée de façon à avoir la fréquence de sortie égale à 1MHZ. On utilise ce signal pour générer les six signaux de commande.

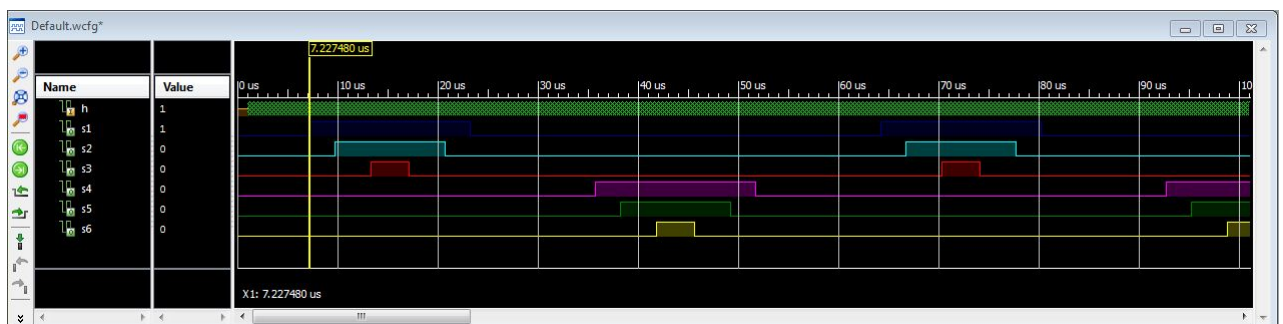
L'organigramme complet du programme de génération des signaux est illustré dans la figure III.17.

### III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA



**Figure III.17** – Organigramme du programme de génération des signaux

les résultats de la simulation sont présentés dans les figures qui suivent : pour les taux de modulation  $r=0.35$ ,  $r=0.61$ ,  $r=0.8$ , et  $r=1$ .



**Figure III.18** – Résultat de la simulation pour  $r=0.35$  ( $f=17.5$  Hz)

## III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA

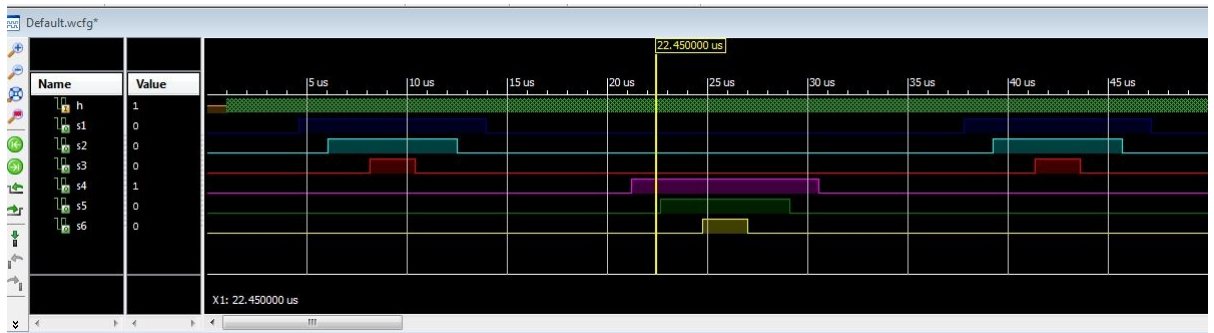


Figure III.19 – résultat de la simulation pour  $r=0.61$  ( $f= 30.5$  Hz)

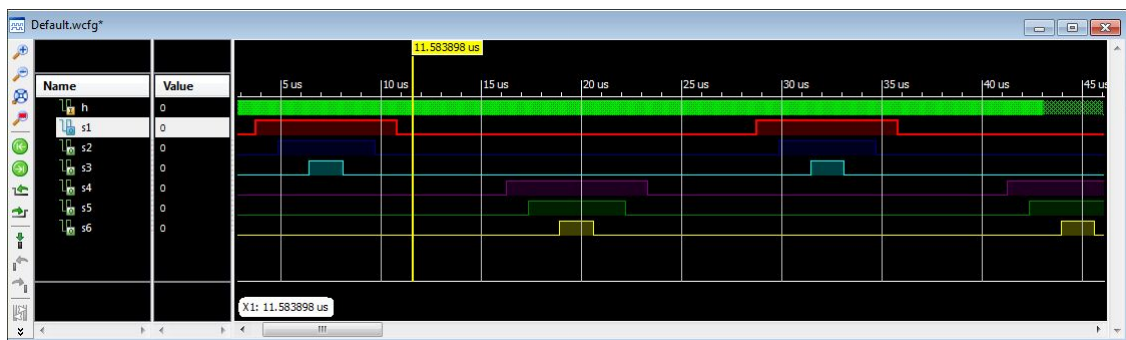


Figure III.20 – Résultat de la simulation pour  $r=0.8$  ( $f= 40$  Hz)

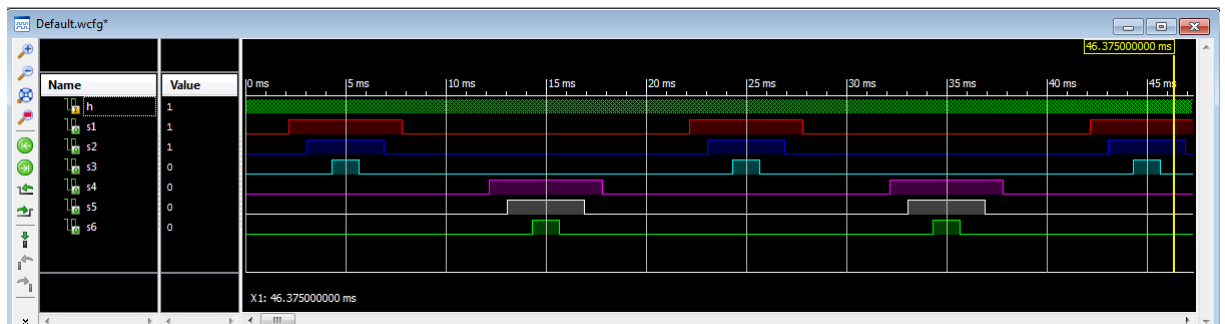


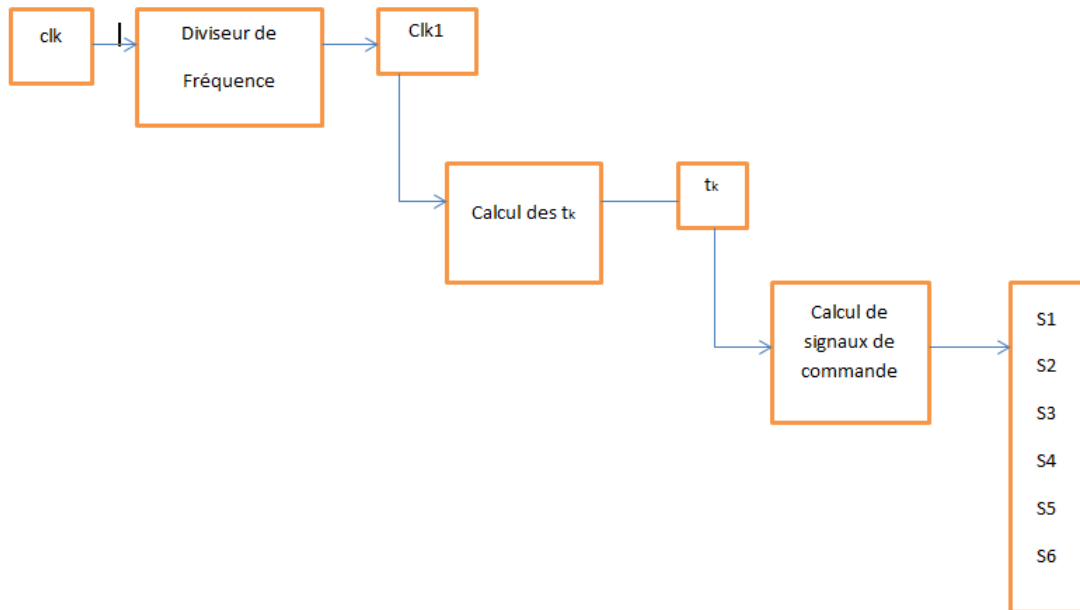
Figure III.21 – Résultat de la simulation pour  $r=1$  ( $f= 50$  Hz)

### III.7.3 L'impémentation de la commande sur la carte Spartan-3E

#### Intorduction

Pour implémenter et tester notre commande sur la carte de développement Spartan-3E, on propose le programme qui est structuré dans la figure III.22. Il est composé de trois blocks, un diviseur de fréquence, un block pour calculer les valeurs temporelles et un block qui donne les six signaux de commande.

### III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA

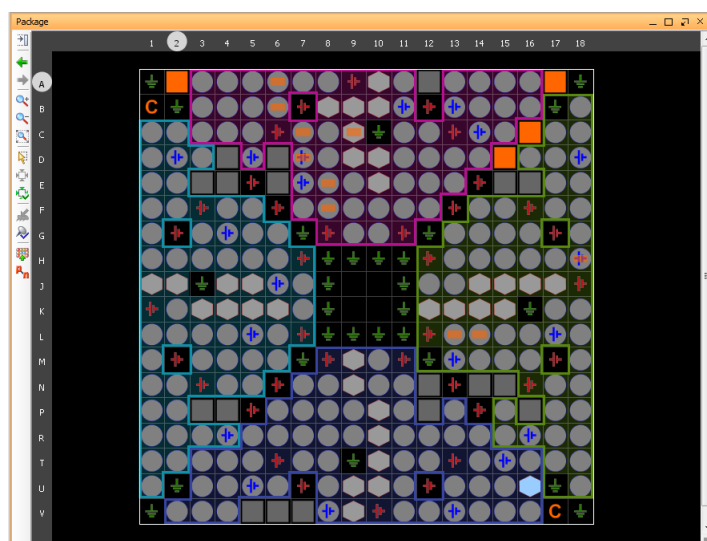


**Figure III.22** – La structure du programme qui donne les signaux de commande

#### Implémentation

Avant l'implémentation du programme dans la carte cible, il faut choisir les contraintes. Pour notre commande on choisit sur la carte comme horloge de synchronisation l'horloge qui est de fréquence 50 MHz, et comme entrée 'm' l'interface (DIP switch) qui contient 8 entrées exploitables par l'utilisateur correspondant à huit valeurs de l'indice de modulation, et on dirige les six signaux de sortie vers l'interface LDVS de transmission pour qu'ils puissent être visualisés sur un oscilloscope

les figures III.23 III.24 III.25 montre les fenêtres l'outil 'I/O Pin planning(Plan Ahead)' qui permet de choisir les contraintes d'implémentation, alors que La figure III.31 montre le fichier de contrainte édité (.ucf).



**Figure III.23** – Plan ahead des pins

## III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA

Name	Direction	Neg Diff Pair	Site	Fixed	Bank	I/O Std	Vcco	Vref	Drive Stre...	Slew Type	Pull Type
All ports (42)											
im (3)											
im[0]	Input		L13	<input checked="" type="checkbox"/>		1 default (LVCMOS25)	2,500				NONE
im[1]	Input		L14	<input checked="" type="checkbox"/>		1 default (LVCMOS25)	2,500				NONE
im[2]	Input		H18	<input checked="" type="checkbox"/>		1 default (LVCMOS25)	2,500				NONE
sort (32)	Output					default (LVCMOS25)	2,500		12 SLOW		NONE
Scalar ports (7)											
h	Input		C9	<input checked="" type="checkbox"/>		0 default (LVCMOS25)	2,500				NONE
s1	Output		D7	<input checked="" type="checkbox"/>		0 default (LVCMOS25)	2,500		12 SLOW		NONE
s2	Output		C7	<input checked="" type="checkbox"/>		0 default (LVCMOS25)	2,500		12 SLOW		NONE
s3	Output		F8	<input checked="" type="checkbox"/>		0 default (LVCMOS25)	2,500		12 SLOW		NONE
s4	Output		E8	<input checked="" type="checkbox"/>		0 default (LVCMOS25)	2,500		12 SLOW		NONE
s5	Output		A6	<input checked="" type="checkbox"/>		0 default (LVCMOS25)	2,500		12 SLOW		NONE
s6	Output		B6	<input checked="" type="checkbox"/>		0 default (LVCMOS25)	2,500		12 SLOW		NONE

Figure III.24 – Le tableau des contraintes

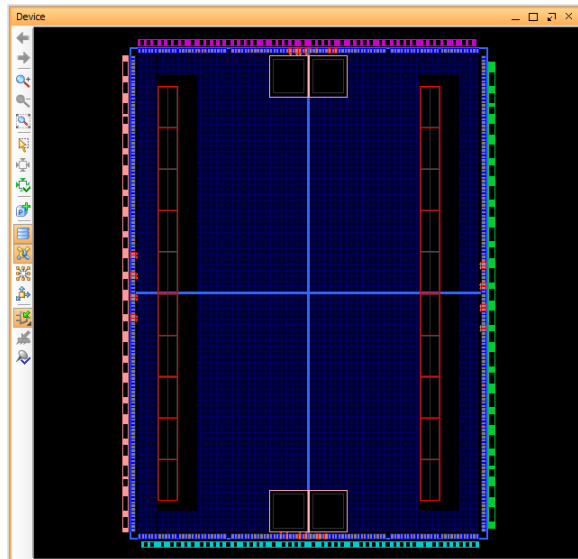


Figure III.25 – Présentation du circuit

```
mli_cal.ucf
1 |
2 |
3 NET "im[0]" LOC = L13;
4 NET "im[1]" LOC = L14;
5 NET "im[2]" LOC = H18;
6 NET "h" LOC = C9;
7 NET "s1" LOC = D7;
8 NET "s2" LOC = C7;
9 NET "s3" LOC = F8;
10 NET "s4" LOC = E8;
11 NET "s5" LOC = A6;
12 NET "s6" LOC = B6;
13
```

Figure III.26 – Le fichier .ucf



## III.7. IMPLÉMENTATION DE LA COMMANDE SUR UN CIRCUIT FPGA

### III.7.4 Test des signaux de sorties

Pour vérifier le fonctionnement de notre commande on la fait visualiser sur l'oscilloscope présenté sur la figure III.27 pour plusieurs valeurs de l'indice de modulation.

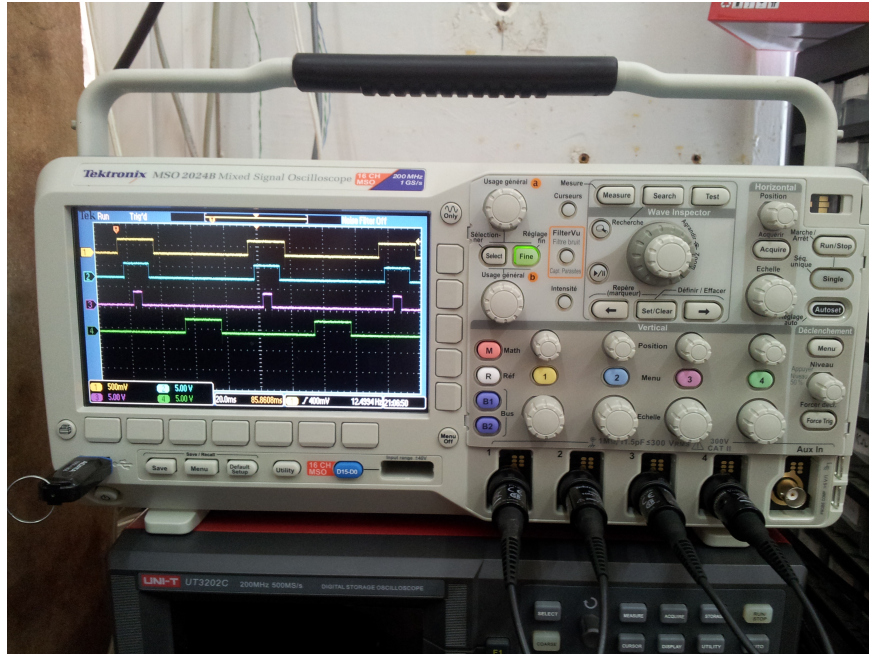


Figure III.27 – Oscilloscope utilisé pour la visualisation

Les figures suivantes montrent la visualisation de notre commande pour  $r=0.35$ ,  $r=0.61$ ,  $r=0.8$ , et  $r=1$ .

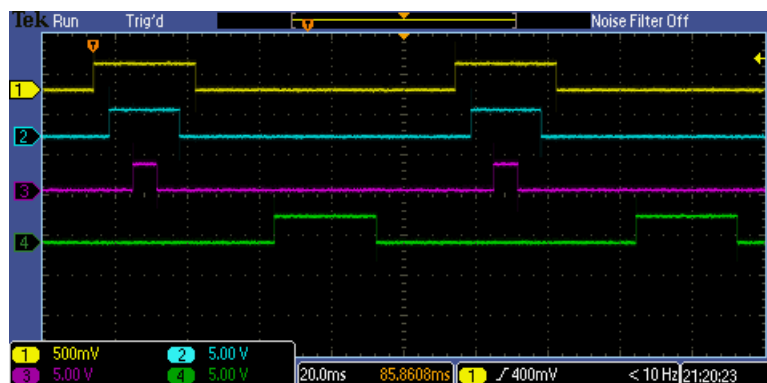


Figure III.28 – Les signaux de commande pour MLI  $r=0.35$

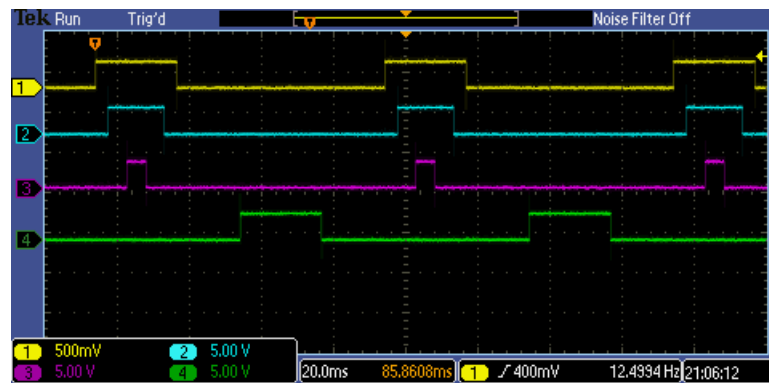


Figure III.29 – Les signaux de commande MLI pour  $r=0.61$

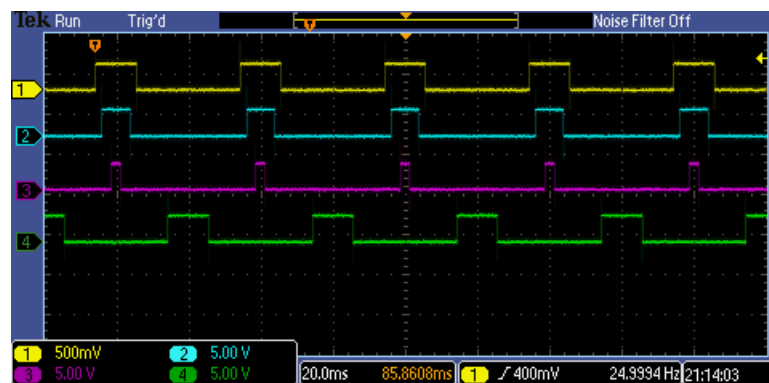


Figure III.30 – Les signaux de commande MLI pour  $r=0.8$

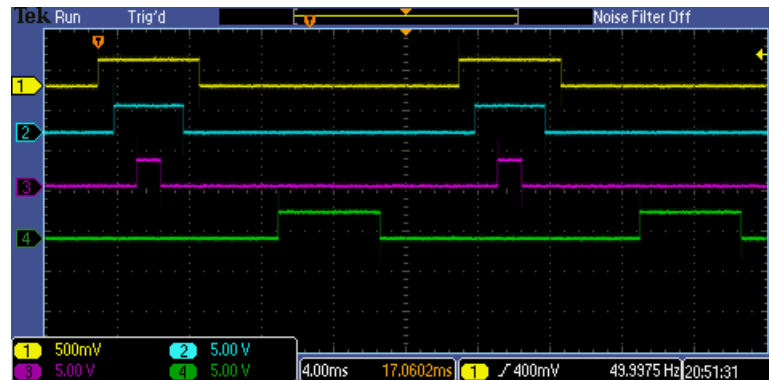


Figure III.31 – Les signaux de commande MLI pour  $r=1$

## III.8 Conclusion

Dans ce chapitre on a proposé un programme pour implémenter la commande MLI 'off-line' sur un circuit FPGA.

Avant d'implémenter ce programme sur la carte on l'a simulé par le ISIM de Xilinx pour vérifier le fonctionnement de notre commande. Après la vérification de programme on l'a implémenté sur notre circuit FPGA Spartan-3E et on a visualisé les résultats sur un oscilloscope.

D'après les figures de simulation et de visualisation, on voit que les résultats sont très proches.

# Conclusion générale

L'objectif de notre étude est la commande d'un onduleur multi-niveaux afin de varier la vitesse d'un moteur asynchrone triphasé dans toute la gamme de vitesses, de zéro à la vitesse nominale du moteur.

Pour obtenir une tension sinusoïdale variable en tension et en fréquence nécessaire à la commande de vitesse d'un moteur asynchrone on a choisi d'utiliser un onduleur multi-niveaux en pont H cascadié.

L'utilisation de l'onduleur multi-niveaux nous a permis d'une part d'atteindre des puissances élevées, et de se rapprocher mieux vers le sinusoïde, et d'autre part elle a permis de minimiser le nombre de commutations par quart de période pour chaque signal ce qui minimise énormément les pertes.

Pour obtenir un signal de sortie sinusoïdal on a utilisé la commande MLI calculée, avec élimination sélective des harmoniques et asservissement du fondamental de Patel et Hoft pour un onduleur multi-niveaux.

Le problème de la commande MLI calculée, avec élimination sélective d'harmoniques pour un onduleur multi-niveaux est la résolution du système d'équations non linéaires.

Il fallait donc trouver une méthode de résolution avec une grande précision. La méthode Résultante était la solution idéale à ce problème, seulement cette méthode nécessite un calcul énorme.

Pour cela on a utilisé les logiciels mathématiques telle que MATLAB, MATHEMATICA, MAPLE pour faire ces calculs. Le résultat était excellent, la méthode nous a donné des valeurs très précises. En plus la méthode a permis de trouver toutes les solutions existantes, ce qui n'est pas le cas avec les autres méthodes telle que Newton-Raphson. Après le calcul exact des angles on ait passé à l'implémentation de notre algorithme.

Lors du passage à l'implémentation nous avons choisi d'utiliser des architectures reconfigurables FPGA, qui donnent de bonnes performances pour les algorithmes avec des contraintes temporelles.

La difficulté qu'on a rencontré lors de la programmation est l'absence des bibliothèques traitants les nombres réels et les fonctions mathématiques élémentaires, telles que les fonctions division et la puissance donc on a été obligé de travailler avec les nombres entiers et faire quelques approximations. On a trouvé que le circuit FPGA nous a permis de faire et de développer des applications importantes

avec des vitesses rapides. L'interface XILINX nous a permis de vérifier nos programmes et de générer le fichier « .bit » qui sera implémenté dans le circuit FPGA.

Le simulateur de ISE ( I Sim) était aussi très utile car il nous a permis de simuler nos programmes avant de les implémenter afin de vérifier la fonctionnalité de notre algorithme global. En ce qui concerne la carte de développement SAPRTAN-3E, elle nous a permis d'implémenter notre application et de prouver son bon fonctionnement. Cette carte est bien développée et contient beaucoup de périphériques ce qui nous a facilité la tâche.

Pour terminer nous avons visualisé les résultats de l'implémentation sur un oscilloscope et nous avons constaté que les résultats de l'implémentation étaient excellents et similaires à ceux obtenus par simulation sous le logiciel PSIM.

Avec la génération des signaux de la commande MLI calculée, à élimination sélective des harmoniques et asservissement du fondamental pour un onduleur multi-niveaux on a atteint le but de notre travail.

En perspectives, nous proposons l'amélioration de cet algorithme pour qu'il soit utilisé en temps réel en développant une commande MLI on-line et ceci en utilisant les réseaux de neurones par exemple.

# Bibliographie

- [1] F. Badin, J. Scordia, R. Trigui, E. Vinot et B. Jeanneret (December 2006). Hybrid electric vehicles energy consumption decrease according to drive train architecture, energy management and vehicle use, Dans Hybrid Vehicle Conference, IET The Institution of Engineering and Technology, pp 213 – 224. I
- [2] O.Bouakaz, contribution à l'analyse des onduleurs multiniveau, mémoire de magister de l'université de Batna, 2005. (document), I.8.4
- [3] A.Nouh, contribution au développement d'un simulateur pour les véhicules électriques routiers ,thèse de doctorat délivré conjointement par l'Université de Technologie de Belfort-Montbéliard 2008. I.2, I.3
- [4] D. Fodorean, Conception et réalisation d'une machine synchrone à double excitation : Application à l'entraînement direct, Thèse de doctorat, Université de Technologie de Belfort-Montbéliard, Université Technique de Cluj-Napoca, juillet 2005. I.2.1
- [5] B.Multon(2001), Motorisation des véhicules électriques, Dans Techniques de l'ingénieur, numéro E3996. I.2.1
- [6] W.Lajnaf, Modélisation des supercondensateurs et évaluation de leur vieillissement en cyclage actif à forts niveaux de courant pour des applications véhicules électriques et hybrides thèse Présentée à L'université BORDEAUX 2006. I.2.2
- [7] M. H. Westbrook (2001). The electric car : development and future of battery, hybrid and fuel-cell cars, série38.The Institution of Electrical Engineers, London, United Kingdom and Society of Automotive Engineers, USA, numéro ISBN 0852960131. I.3
- [8] M. Zeraoulia, M. E. H. Benbouzid et D. Diallo (November 2006). Electric motor drive selection issues for HEV propulsion systems : a comparative study, IEEE Transactions On Vehicular Technology, volume 55, numéro 6, pp. 1756-1764. I.3
- [9] P.Mayé, Aide mémoire électrotechnique, version Dunod. I.4.1
- [10] INTERSECTION, magazine technique juin 1998. (document), I.7
- [11] Freescale Semiconductor, 3-Phase AC Motor Control with V/Hz Speed Closed Loop Using the 56F800/E, Freescale Semiconductor Application Note 2005. (document), I.9, I.5
- [12] D.Bendib, Etude et réalisation d'une commande MLI on-line sur circuit FPGA, mémoire de magister à l'école nationale polytechnique ,juin 2009. (document), I.6.1, I.6.2, III.1, III.1, III.6
- [13] Zhong Du, Active Harmonic Elimination in Multilevel Converters, Université de Tennessee, Knoxville, Mai 2005. (document), I.7.2, I.8.3, I.8.4, I.9.3, I.16, I.17, I.18, II.4.1, II.4.1, II.4.3, II.4.4
- [14] J. Rodríguez, J. Lai, and F. Peng, "Multilevel converters : a survey of topologies, controls and applications," IEEE Transaction on Industrial Electronics, vol. 49, no. 4, Aug. 2002, pp. 724-738. I.8.1, I.9.2, I.9.5
- [15] A. Nabae, I. Takahashi, and H. Akagi, "A new neutral-point clamped PWM converter," IEEE Transactions on Industry Applications, vol. IA-17, pp. 518–523, Sept./Oct. 1981. I.8.2, I.8.2
- [16] P. N. Enjeti, P. D. Ziogas, J. F. Lindsay, "Programmed PWM Techniques to eliminate Harmonics : A Critical Evaluation," IEEE Transactions on Industry Applications, vol. 26, no. 2, March/April. 1990. pp. 302 – 316. (document), I.12
- [17] C. Hochgraf, R. Lasseter, D. Divan, and T. A. Lipo, "Comparison of multilevel converters for static var compensation," in Conf. Rec. IEEE IAS Annu. Meeting, Oct. 1994, pp. 921–928. I.8.3

- [18] L. M. Tolbert, F. Z. Peng, T. G. Habetler, "Multilevel Converters for Large Electric Drives," IEEE Transactions on Industry Applications, vol. 35, no. 1, Jan./Feb. 1999, pp. 36-44. I.8.4
- [19] P. Hammond, "A new approach to enhance power quality for medium voltage ac drives," IEEE Transactions on Industry Applications, vol. 33, pp. 202-208, Jan./Feb. 1997. I.8.4
- [20] E. Cengelci, S. U. Sulistijo, B. O. Woom, P. Enjeti, R. Teodorescu, and F. Blaabjerge, "A new medium voltage PWM converter topology for adjustable speed drives," in Conf. Rec. IEEE-IAS Annu. Meeting, St. Louis, MO, Oct. 1998, pp. 1416-1423. I.8.4, I.9.3
- [21] N.DJAAFRI, et A.REBAI, Implémentation des techniques MLI sur un circuit FPGA, mémoire de PFE à l'école nationale polytechnique, juin 2012. I.9.1, III.2.2
- [22] N. Celanovic and D. Boroyevic, "A fast space vector modulation algorithm for multilevel three-phase converters," IEEE Industry Applications Society Annual Meeting, Phoenix, AZ, Oct. 1999, pp. 1173-1177. I.9.2
- [23] J. Rodríguez, P. Correa, and L. Morán, "A vector control technique for medium voltage multilevel converters," IEEE APEC, Anaheim, CA, Mar. 2001, pp. 173-178. I.9.2, I.9.5
- [24] Y. H. Lee, R. Y. Kim, and D. S. Hyun, "A novel SVPWM strategy considering DC-link balancing for a multi-level voltage source converter," IEEE APEC, 1998, pp. 509-514. I.9.4
- [25] D. W. Kang et al., "Improved carrier wave-based SVPWM method using phase voltage redundancies for generalized cascaded multilevel converter topology," IEEE APEC, New Orleans, LA, Feb. 2000, pp. 542-548. I.9.4
- [26] A.Guellal, Implémentation sur FPGA d'une commande MLI on-line basée sur le principe des réseaux de neurones, mémoire de magister à l'école nationale polytechnique 2009. I.9.6
- [27] H. S. Patel, R. G.Hoft, "Generalized technique of harmonics elimination and voltage control in thyristor inverter", IEEE Tran. Ind. Appl., 1973, pp. 310-317. I.9.6, II.2
- [28] J. N. Chiasson, L. M. Tolbert, K. J. McKenzie, Z. Du, "Control of a Multilevel Converter Using Resultant Theory," IEEE Transactions on Control System Theory, vol. 11, no. 3, May 2003, pp. 345-354. II.4.1
- [29] J. N. Chiasson, L. M. Tolbert, K. J. McKenzie, Z. Du, Elimination of Harmonics in a Multi-level Converter using the Theory of Symmetric Polynomials and Resultants. II.4.2
- [30] Kurosh.A, " Cours d'Algèbre supérieure ", p.332-345 édition MIR,1973, Moscou. II.4.2
- [31] K.Imarazene, H.Chekireb2 et E.M.Berkouk, Application de la théorie de résultante à la commande par élimination des harmoniques d'un onduleur à 7 niveaux : International Conference On Industrial Engineering and Manufacturing ICIEM'10, May, 9-10, 2010, Batna, Algeria (document), II.4.3, II.8
- [32] Claude Guex Introduction au VHDL Ecole d'ingénieurs de Canton de Vaud Juin 98. (document), III.1, III.2.4, III.2, III.3
- [33] S.Kilts. "Advanced FPGA design". Edition WILEY, 2007.
- [34] S. C. Chan, H.O. Ngai and K.L. Ho "A programmable image processing system using FPGA" International journal of electronics, vol 75, N°4 pp 725-730, 1993. III.2
- [35] M. Alves de Barros "Traitement bas niveau d'images en temps réel et circuits reconfigurables" Thèse de doctorat, Université de Paris-Sud, 1994. III.2
- [36] J.G. Eldrerge, B.L. Hutchings "Density enhancement of neural network using FPGAs and run-time reconfiguration" FCCM, 1994. III.2
- [37] E. Lemoine, J. Quinqueton and J. Salantin "High speed pattern matching in genetic data base with reconfigurable hardware" Proceeding of th 2nd INT. Conf. o, Intelligent systems for molecular biology, pp 269-276. AAAI, 1994. III.2
- [38] B.Heeb and C. Pfister "Chameleon, a workstation of a different colour" 2nd International Workshop on Field-Programmable Logic Applications, paper 5.6, Vienna, Austria, 1992. III.2
- [39] S.Kilts. "Advanced FPGA design". Edition WILEY, 2007. III.2.1
- [40] M. Philippe LETENNEUR, LES CIRCUITS LOGIQUES PROGRAMMABLES, Lycée Juliot de la Morandière III.4
- [41] S.Berto, A.Paccagnella, M.Ceschia, "Potentials and Pitfalls of FPGA Application in Inverter Drives - a Case Study" in Proc. IEEE ICIT 2003, pp.500-505. III.6.3

- [42] PONG P. CHU. "RTL Hardware Design Using VHDL" USA :John Wiley et Sons 2006, 669p. III.6.3
- [43] Bob Zeidman,"Designing with FPGAs and CPLDs"ELSEVIER 2002. III.6
- [44] Volnei A Pedroni, "Circuit Design With VHDL" MIT Press 2004. III.6.5
- [45] Etienne Messerli ,Manuel VHDL synthèse et simulationn ,version 6a septembre 2007.