Centre Universitaire BOUSSOOUF Abdelhafid -Mila
Institut des Sciences et Technologie
Département de Génie Mécanique et Électromécanique

Nº Réf : .....

Projet de Fin d'Etude préparé En vue de l'obtention du diplôme de MASTER

Spécialité : Électromécanique

Implémentation de la commande directe du couple (DTC) sur FPGA de type ALTERA

Réalisé par :
- DJARIT Yassine
- ZEMIECHE Houssam

Soutenu devant le jury :

M.	**BAAZI Smail**	**Président**
M.	**GUENTRI Hocine**	**Examinateur**
M.	**SMAANI Billel**	**Promoteur**

Année universitaire : 2021/2022

Remerciement

Grâce à dieu miséricordieux tout puissant, qui nous a éclairé le chemin de réussite, et de nous avoir donné la santé, la patience, la puissance et la volonté pour réaliser ce travail.

Au terme de ce travail, Nous tenons également à exprimer nos sincères remerciements à tous ceux qui ont contribués de loin au de près à l'élaboration de ce projet de fin d'étude surtout ***Mr Smaani Billel,*** *notre promoteur de nous avoir conseillé judicieusement, orienté, encouragé et de nous avoir apporté son attention tout au long de la réalisation de ce travail.*

Nos vifs remerciements vont également à l'ensemble des membres du jury, ***Mr Guentri Hocine*** *et* ***Mr Baazi Smail.*** *D'avoir accepté de juger notre travail et pour l'intérêt qu'ils y ont manifesté.*

Nos vifs remerciements à toute personne ayant contribué, de près ou de loin, dans ce travail

Dédicace

Avant tout et avec l'aide et la protection d'ALLAH S'est réalisé se modeste travail, merci Allah pour m'avoir donné la santé, la force et le courage pour mener a réalisé ce travail.

Je dédie ce modeste travail à :

A mes parents qui m'ont inculqué un esprit de combativité et de persévérance et qui m'ont toujours poussé et motivé dans mes études.

A ma mère ***Akila*** *et mon père* ***Amor ,*** *qui m'a soutenu durant tous les années d'études et pour leurs aides, encouragement, sacrifices et leur patience, merci pour tout ce que vous fait pour moi, que j'espère les rendre fière par ce travail. Puisse Allah vous garder Toujours la source de bonheur de ma vie.*

A mes chères sœurs ***Amira et Bariza*** *et ma princesse* ***Samira*** *tous mes chères frères* ***Aissam ,Nouri ,Salah ,Mounir ,Abelekrim et Ahmed*** *je vous souhaite une vie pleine de bonheur et de réussite.*

A mon binôme et ami ***Houssam*** *merci pour l'excellent travail. Qu'avec lesquelles nous passeront des beaux moments et des bons souvenirs.*

A toute personne ayant contribué, de près ou de loin, dans ce travail, Enfin je dédie toute la promotion de ***MASTER 2 Electromécanique*** *je vous souhaite de bonne continuation.*

Yassine

Dédicace

Avant tout et avec l'aide et la protection d'ALLAH S'est réalisé se modeste travail, merci Allah pour m'avoir donné la santé, la force et le courage pour mener a réalisé ce travail.

Je dédie ce modeste travail à :

A mes parents qui m'ont inculqué un esprit de combativité et de persévérance et qui m'ont toujours poussé et motivé dans mes études.

A ma mère ***Akila*** *et mon père* ***Mouloud*** *qui m'a soutenu durant tous les années d'études et pour leurs aides, encouragement, sacrifices et leur patience, merci pour tout ce que vous fait pour moi, que j'espère les rendre fière par ce travail. Puisse Allah vous garder Toujours la source de bonheur de ma vie.*

A mes chère frères ***Sofiane, Khaled et Aziz*** *et mes chère sœurs* ***Samira et Ahlem*** *et ma petite princesse* ***Ritaj,*** *je vous souhaite une vie pleine de bonheur et de réussite.*

A mon binôme et ami ***Yassine*** *merci pour l'excellent travail. Qu'avec lesquelles nous passeront des beaux moments et des bons souvenirs.*

A toute personne ayant contribué, de près ou de loin, dans ce travail, Enfin je dédie toute la promotion de ***MASTER 2 Electromécanique*** *je vous souhaite de bonne continuation.*

Houssam

Table des matières

Chapitre II : *Les circuits FPGA et le langage VHDL*

Chapitre III : *Implémentation et simulation de la commande DTC sur FPGA*

Notations utilisées

Symbole	Signification
Rs	Résistance statorique par phase
Rr	Résistance rotorique par phase.
Ns	Vitesse de synchronisme
f	La fréquence des courants statoriques
P	Nombre de pair de pôles.
g	Le glissement
isabc	Les courants triphasés statoriques
irabc	Les courants triphasés rotoriques.
Ms	Inductance mutuelle entre deux phases statoriques.
Lss	Inductance cyclique du stator
Lrr	Inductance cyclique du rotor
Ls	Inductance propre d'une phase statorique.
Lr	Inductance propre d'une phase rotorique.
Ɵ	Écart angulaire mécanique telle que $\alpha = P.\Theta$
Ɵr	Angle électrique formé par l'axe direct d et l'axe de la phase a du rotor
Ɵs	Angle électrique formé par l'axe direct d et l'axe de la phase a du stator
Mss	Inductance mutuelle stator par apport au rotor et
Mr	Inductance mutuelle entre deux phases rotoriques.
Mrs	L'inductance mutuelle rotor par apport au stator.
Vsabc	Les tensions triphasées statoriques
Vrabc	Les tensions triphasées rotoriques.
Cem	Couple électromagnétique
ϕrabc	Les flux triphasés à travers les enroulements rotoriques
ϕsabc	Les flux triphasés à travers les enroulements statoriques
ϕsα, ϕsβ	Composantes directes et quadratures de flux statorique dans le
Tr	Constante de temps rotorique
Ɓ	Coefficient de dispersion
Cflx	Contrôleur du flux
Ccpl	Contrôleur du couple
ϕsref	La consigne de flux statorique
Cref	La consigne de couple

Abréviation utilisée

Abréviation	Signification
ALM	Adaptative Logique Module
ASIC	Aplication-Specific Sntegrated Circuit
CAN	Convertisseur Analogique Numérique
CLB	Configurable Logic Blocks
CMOS	Somplementary Metal Oxide Semiconductor
CPLD	Complex Programmable Logic Device
DSP	Digital Signal Processing
DTC	Direct Torque Control
DTNC	Direct Torque Neuronal Control
DTFC	Direct Torque Fuzzy Control
EPLD	Erasable Programmable Logic Device
FOC	Field oriented control
FPGA	Field Programmable Gate Array
GAL	Generique Array Logic
GPIO	General Purpose Input/Output
GTL	Gunning Transceiver Logic
GTO	Gate Turn Off.
HSTL	High-Speed Transceiver Logic
IEEE	Institute of Electrical and Electronics Engineers
IGBT	Insulated Gate Bipolar Transistor.
IOB	Input Output Blocks
JTAG	Joint Test Action Group
LCA	Logic Cell Array
LVCMOS	Low-Voltage CMOS
LVDS	Low Voltage Differential Signaling
LVPECL	Low Voltage Positive Emitter Coupled Logic
LVTTL	Low-Voltage Transistor-Transistor Logic
LUT	Look Up Table
MAS	Moteur Asynchrone Symétrique
PAL	Programmable Array Logic
PCI	Peripheral Component Interconnect
PLD	Programmable Logic Device
VHDL	Very High Speed Integrated Circuits Hardware Description Language

Liste des figures

Liste des tableaux

Introduction générale

Introduction générale

La machine à induction connait un succès croissant depuis deux décennies en remplaçant progressivement les machines à courant continu et synchrones dans de nombreuses applications industrielles et dans les transports. Ce succès acquis par la machine à induction s'explique par sa conception robuste réduisant les frais de maintenance, par son coût relativement moindre par rapport aux autres machines électriques et également par l'augmentation des capacités de calcul des microprocesseurs permettant de réaliser une commande performante. L'évolution conjointe de l'électronique de puissance et de l'électronique numérique a contribué à l'élaboration des algorithmes de commande plus avancés améliorant les performances statiques et dynamiques de cette machine et assurant ainsi un découplage du flux et du couple.

Un FPGA, soit en Anglais « Field Programmable Gate Array » est un circuit logique reprogrammable selon les exigences d'application et les fonctionnalités souhaitées. Il peut être utilisé pour implémenter des équations et des systèmes logiques simples ou complexes, ainsi que pour la conception des systèmes de contrôle de différentes machines. Le VHDL est un langage HDL utilisé pour décrire le comportement des systèmes, il s'agit donc d'un langage de description de matériel de type VHSIC (circuit intégré à très grande vitesse), il est largement utilisé par l'industrie pour le prototypage de circuits ASICS ainsi que dans le milieu universitaire afin simuler et de synthétiser des architectures nouvelles

Ce mémoire est organisé en trois chapitres, comme suit :

Nous commençons le premier chapitre par une modélisation de la machine asynchrone accompagnée par sa représentation mathématique dans un référentiel triphasé. Le reste de ce chapitre sera consacré à l'essentiel de l'état de l'art la commande DTC classique.

Dans le deuxième chapitre, nous allons faire une présentation des différentes familles de circuits logiques programmables en particulier le circuit FPGA, sa structure interne, ses blocs logiques de base, ses domaines d'application et ses avantages. Ensuite, nous parlerons de langage de description matérielle VHDL, la structure d'un programme de ce langage ainsi que ses éléments.

Dans le troisième chapitre, on a utilisé l'outil de conception et de simulation QUARTUS II, ainsi que les caractéristiques et les différents bank d E/S de la carte FPGA cible Altera EP2C5T144C8, avec la présentation consacrée de la structure de commande DTC numérique suivie des résultats et synthèse de chaque bloc de l'algorithme de commande DTC conçu, à la

fin de ce chapitre, nous présenterons les résultats de simulation de l'algorithme général de la commande DTC d'une machine asynchrone.

On termine avec une conclusion et quelques perspectives pour donner une étendue future de notre modeste travail.

Chapitre I :

Commande directe du couple (DTC)

I.1. Introduction

Le contrôle direct du couple, venu du terme anglais " Direct Torque Control (DTC)", des machines asynchrones proposé par Takachachi et Depenbrok est apparu dans la deuxième moitié des années 1980 comme concurrentielles des méthodes classiques Dans un repère lié au stator, les valeurs instantanées du flux statorique du couple électromagnétique sont estimées à partir des grandeurs statoriques. En utilisant des comparateurs à hystérésis, le flux et le couple sont contrôlés directement et indépendamment avec une sélection appropriée du vecteur de tension imposé par l'onduleur [1].

Les méthodes de commande directe du couple DTC sont moins sensible aux variations paramétriques de la machine rapport la commande vectorielle et permettent d'obtenir une dynamique précise et rapide du couple, elles consistent à commander directement la fermeture ou l'ouverture des interrupteurs de l'onduleur à partir des valeurs pré-calculées du flux statorique et du couple. Les changements d'état des interrupteurs sont liés à l'évolution des états électromagnétiques du moteur. Ils ne sont plus commandés à partir des consignes de tension et de fréquence donnée à la commande rapprochée d'un onduleur à modulation de largeur d'impulsion. La commande des interrupteurs a pour but de donner au vecteur représentant le flux statorique, la direction déterminée par les valeurs de consigne. Dans ce chapitre, on présentera le principe du contrôle direct du couple pour une MAS [2,3].

Dans ce chapitre, nous allons décrire globalement les machines asynchrones. Puis, nous allons étudier théoriquement la commande directe du couple (DTC).

I.2. La machine asynchrone

I.2.1. Définition

On appelle machine asynchrone toute machine, qui, ayant (2 p) pôles et étant reliée à un réseau de fréquence f_s , ne tourne pas exactement à la vitesse asynchrone ($\frac{60f_s}{\pi}$). On parle généralement de moteurs asynchrones car ces machines sont destinées à fournir de la puissance mécanique à partir du réseau électrique. Parmi les machines asynchrones, on peut distinguer deux types [4]:

- Les machines d'induction.
- Les machines à collecteur.

Le moteur d'induction est tellement plus utilisé que les autres que lorsqu'on parle de moteur asynchrone on sous-entend d'induction. La machine d'induction est caractérisée par une armature non alimentée (rotor), parcourue par des courants induits par l'autre armature qui est alimentée à partir d'un réseau de fréquence f_s (stator) [5].

I.2.2. Constitution

La machine asynchrone est constituée des principaux éléments suivants: Le stator, l'entrefer, le rotor et les organes mécaniques

I.2.2.1. Le stator

Il est constitué des enroulements bobinés répartis dans les encoches du circuit magnétique, ce circuit magnétique est constitué d'un empilage de tôles dans lesquelles sont découpées des encoches parallèles à l'axe de la machine (voir la figure (I-1)).

Le bobinage statorique peut se décomposer en deux parties : les conducteurs d'encoches et les têtes de bobines. Les conducteurs d'encoches permettent de créer dans l'entrefer le champ magnétique à l'origine de la conversion électromagnétique. Les têtes des bobines permettent, quant à elles, la fermeture des courants en organisant la circulation judicieuse des courants d'un conducteur d'encoche à l'autre [6].

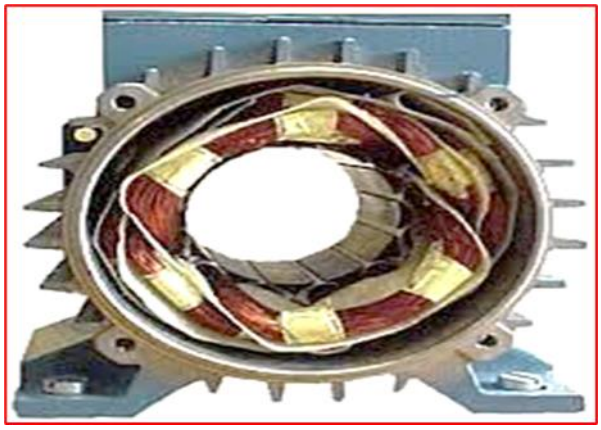

Figure I-1 : Stator de la machine asynchrone

I.2.2.3. L'entrefer

C'est l'intervalle d'air entre le stator et le rotor, son épaisseur est de l'ordre de dixième de millimètre, il varie entre 0.2 et 1.2 millimètre

I.2.2.4. Rotor

C'est l'élément mobile du moteur, son circuit magnétique est similaire à celui du stator, il est constitué d'un empilage de tôles minces isolées entre elles et formant un cylindre claveté sur l'arbre du moteur et il contient des encoches destinées à loger l'enroulement.

Il existe deux types de moteurs asynchrones, qui se distinguent par la forme de leur rotor qui est soit bobiné ou à cage, dans tous les cas le stator reste au moins dans son principe, le même. [7]

-Rotor bobiné

Les tôles de ce rotor sont munies d'encoches où sont placés des conducteurs formant un bobinage de structure généralement semblable à celle des enroulements statoriques.

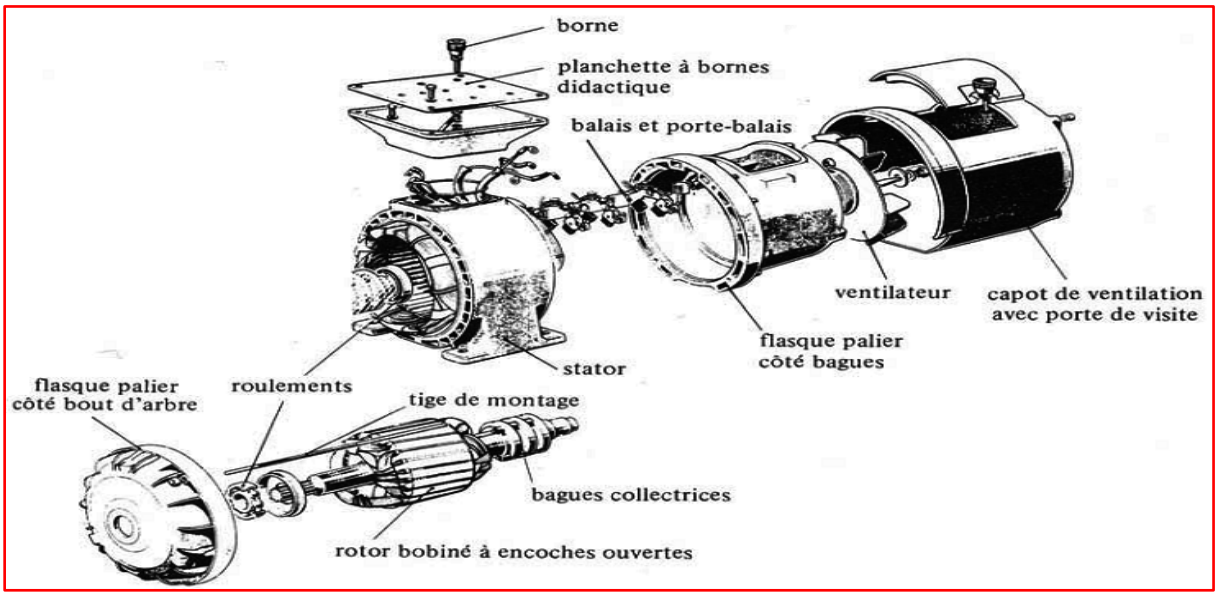

Figure I-2 : Vue éclatée d'une machine asynchrone triphasée à bagues.

Dans le cas très fréquent où le bobinage du rotor est triphasé, trois bagues et trois balais sont prévus pour accéder à ces enroulements. Ce dispositif permet de modifier certaines caractéristiques électriques du circuit rotorique et par là, les propriétés électromagnétiques du moteur asynchrone.

Ce type de moteur est utilisé essentiellement dans des applications où les démarrages sont difficiles et/ou nombreux. [7]

-Rotor à cage

Un rotor à cage d'écureuil porte un système de barres conductrices faites en cuivre, en bronze ou en aluminium logées dans un empilement de tôles.

Les extrémités de ces barres sont réunies par deux couronnes également conductrices.

Ce type de moteur est plus aisé à construire que le moteur à rotor bobiné et par conséquent d'un prix de revient inférieur et à une robustesse intrinsèquement plus grande. Il n'est donc pas étonnant qu'il constitue la plus grande partie du parc des moteurs asynchrones actuellement en service.

Son inconvénient majeur est qu'il a au démarrage de mauvaises performances, courant élevé et faible couple. C'est pour remédier à cette situation qu'ont été développés deux autres types de cages, rotor à double cage et rotor à encoches profondes. [13]

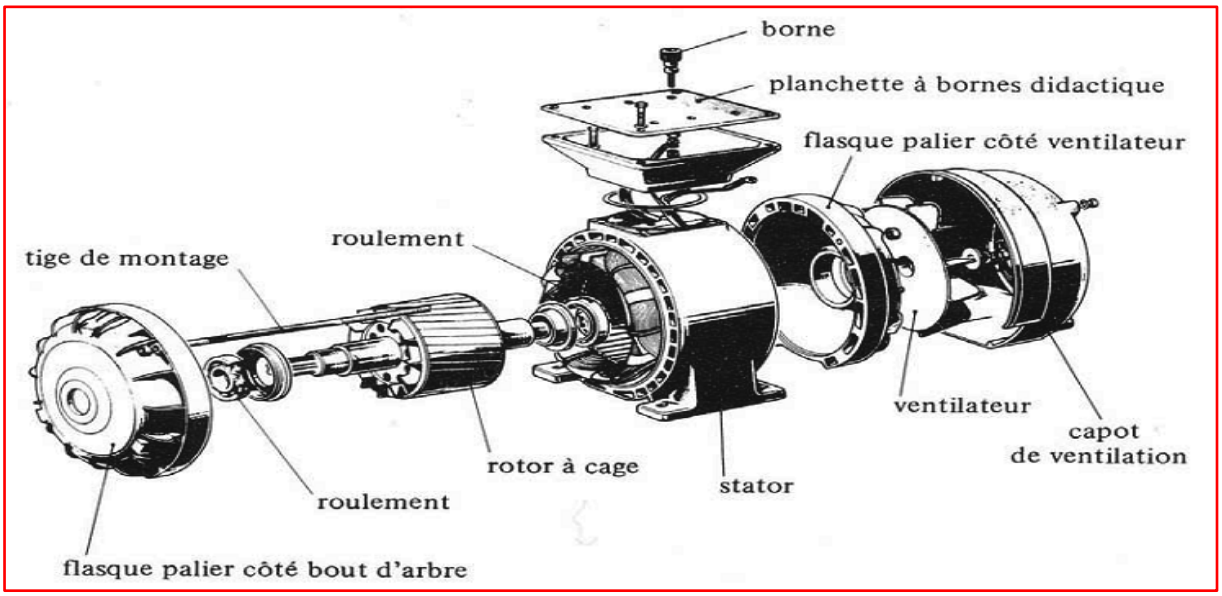

Figure I-3 : Vue éclatée d'une machine asynchrone triphasée à cage

I.2.3. Les organes mécaniques

Le stator auto-porteur reçoit de chaque côté un flasque sur lequel le rotor sera positionné grâce à des roulements à billes ou à rouleaux suivant le type de charge (axiale ou radiale). Un ventilateur est placé en bout d'arbre sur le rotor pour le refroidissement de la machine. Il peut être remplacé par une ventilation forcée motorisée pour le refroidissement aux vitesses lentes.[8]

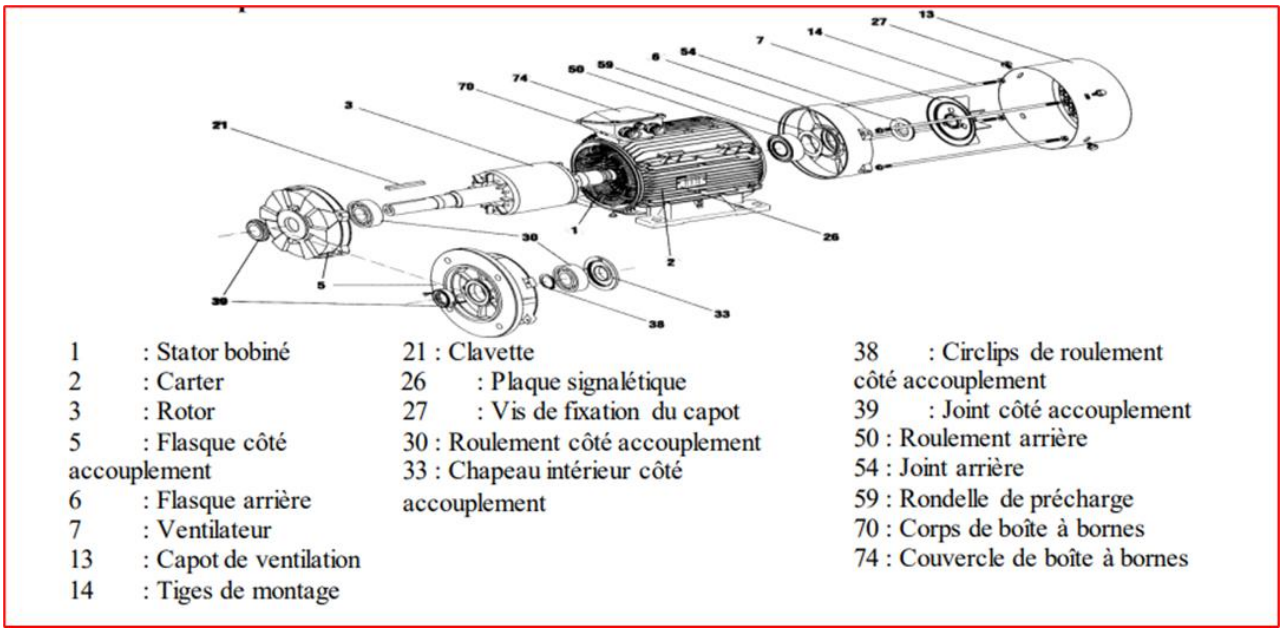

Figure I-4 : vues en coupe d'une machine asynchrone

I.2.4. Bobinage de la machine asynchrone

On peut effectuer le bobinage d'une machine asynchrone de plusieurs façons, la disposition des bobines dans les encoches différencie ces types d'enroulements d'où en trouve habituellement trois types, l'enroulement imbriqué, concentrique et ondulé. Chaque type présente des avantages dans certaines applications.

L'enroulement du stator peut être à une seule ou à deux couches, ce dernier nous permet de raccourcir le pas d'enroulement.

Le bobinage statorique peut se décomposer en deux parties, les conducteurs d'encoches et les têtes de bobines. Les conducteurs d'encoches permettent de créer dans l'entrefer le champ magnétique à l'origine de la conversion électromagnétique. Les têtes de bobines permettent, la fermeture des courants en organisant la circulation judicieuse des courants d'un conducteur d'encoche à l'autre ; l'objectif est d'obtenir à la surface de l'entrefer une distribution de courant la plus sinusoïdale possible, afin de limiter les ondulations du couple électromagnétique. [7]

I.2.5. Principe de fonctionnement

Le principe de fonctionnement de la machine repose sur les lois fondamentales de l'induction électromagnétique. En effet, le stator alimenté par un système de tensions triphasées équilibrées crée un champ magnétique tournant. La vitesse de rotation du champ tournant statorique Ns,

appelée vitesse synchrone, est rigidement liée à la fréquence de la tension d'alimentation f (Hz) et au nombre de pair de pôles P de chacun des enroulements par :

$$\mathbf{Ns(tr\ /mn)}=\frac{60f}{p} \qquad (1)$$

Soumis au champ tournant crée par le stator, les enroulements rotoriques sont le siège d'un système de forces électromotrices triphasées engendrant elles-mêmes trois courants rotoriques. Ces courants rotoriques par leurs effets vont s'opposer à la cause qui leur a donné naissance.

Ainsi, les effets de l'induction statorique sur les courants induits rotoriques se manifeste par un couple électromagnétique qui lance le rotor à une vitesse N en essayant d'atteindre la vitesse synchrone mais en vain. Il est évident que le couple s'annule si le rotor arrive à tourner à la vitesse synchrone.

Le fonctionnement du moteur est donc caractérisé par le glissement défini ainsi :

$$g = \frac{\mathbf{Ns-N}}{Ns} \qquad (2)$$

Contrairement à la machine à courant continu et la machine synchrone, seuls les enroulements statoriques sont généralement reliés au réseau de tensions créent cette induction motrice.

Les enroulements rotoriques ne sont pas reliés à aucune source de tension, mais court-circuités sur eux-mêmes c'est la raison pour laquelle, on l'appelle aussi la machine d'induction [9,10]

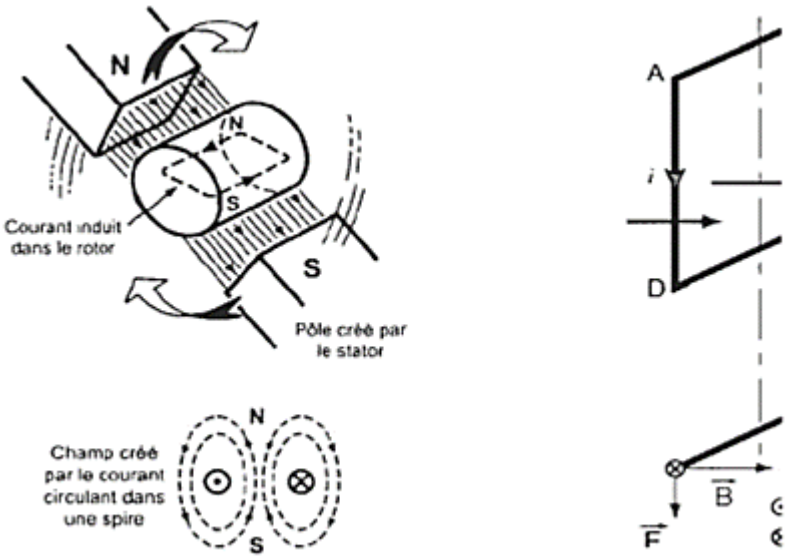

Figure I-5 : Schéma de principe de fonctionnement

I.2.6. Equations des tensions et de flux

I.2.6.1. Equations des tensions

Compte-tenu des lois des phénomènes d'induction électromagnétique, les équations de tensions s'écrivent :

Au stator : $$\begin{bmatrix} Vsa \\ Vsb \\ Vsc \end{bmatrix} = \begin{bmatrix} Rs & 0 & 0 \\ 0 & Rs & 0 \\ 0 & 0 & Rs \end{bmatrix} \begin{bmatrix} isa \\ isb \\ isc \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \emptyset sa \\ \emptyset sb \\ \emptyset sc \end{bmatrix} \quad (3)$$

Ou : $[Vsabc] = [Rs][isabc] + \frac{d}{dt}[\emptyset sabc]$

Au rotor : $$\begin{bmatrix} Vra \\ Vrb \\ Vrc \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} Rr & 0 & 0 \\ 0 & Rr & 0 \\ 0 & 0 & Rr \end{bmatrix} \begin{bmatrix} ira \\ irb \\ irc \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \emptyset ra \\ \emptyset rb \\ \emptyset rc \end{bmatrix} \quad (4)$$

Ou : $\mathbf{[0] = [Rr][irabc] + \frac{d}{dt}[\Phi rabc]}$

I.2.6.2. Equations de flux

Les équations des flux sous forme matricielle s'écrivent : ∅

Au stator :

$$\begin{bmatrix} \emptyset sa \\ \emptyset sb \\ \emptyset sc \end{bmatrix} = \begin{bmatrix} ls & Ms & Ms \\ Ms & ls & Ms \\ Ms & Ms & Ms \end{bmatrix} \begin{bmatrix} isa \\ isb \\ isc \end{bmatrix} + [Msr] \begin{bmatrix} ira \\ irb \\ irc \end{bmatrix} \quad (5)$$

Ou : $[\emptyset sabc] = [ls][isabc] + [Msr][irabc]$

Au stator : $$\begin{bmatrix} \emptyset ra \\ \emptyset rb \\ \emptyset rc \end{bmatrix} = \begin{bmatrix} lr & Mr & Mr \\ Mr & lr & Mr \\ Mr & Mr & Mr \end{bmatrix} \begin{bmatrix} ira \\ irb \\ irc \end{bmatrix} + [Mrs] \begin{bmatrix} isa \\ isb \\ isc \end{bmatrix} \quad (6)$$

Ou : $[\emptyset rabc] = [lr][irabc] + [Mrs][isabc]$

Par raison de réciprocité mutuelle $[Mrs] = [Msr]^T$ (7)

I.3. Commande directe du couple (DTC)

La technique DTC est un principe de commande des moteurs à courant alternatif [11].

Elle permet de calculer les grandeurs de contrôle que sont le flux statorique et le couple électromagnétique, uniquement à partir des mesures de tensions et de courants statoriques, sans disposer de mesures de vitesse, de flux ou de couple.

Ces deux derniers sont entièrement estimés en utilisant des régulateurs à hystérésis qui comparent les valeurs actuelles calculées du couple et du flux aux valeurs de référence. Dépendamment du signal du contrôleur à hystérésis ou comparateur, l'onduleur est commandé par un "sélecteur d'interrupteurs optimisé". Les interrupteurs de l'onduleur imposent au moteur les tensions nécessaires qui contrôleront le couple et le flux en même temps dans une commande à boucle fermée [12]. La technique de la DTC est ainsi représentée par la figure (I-6)

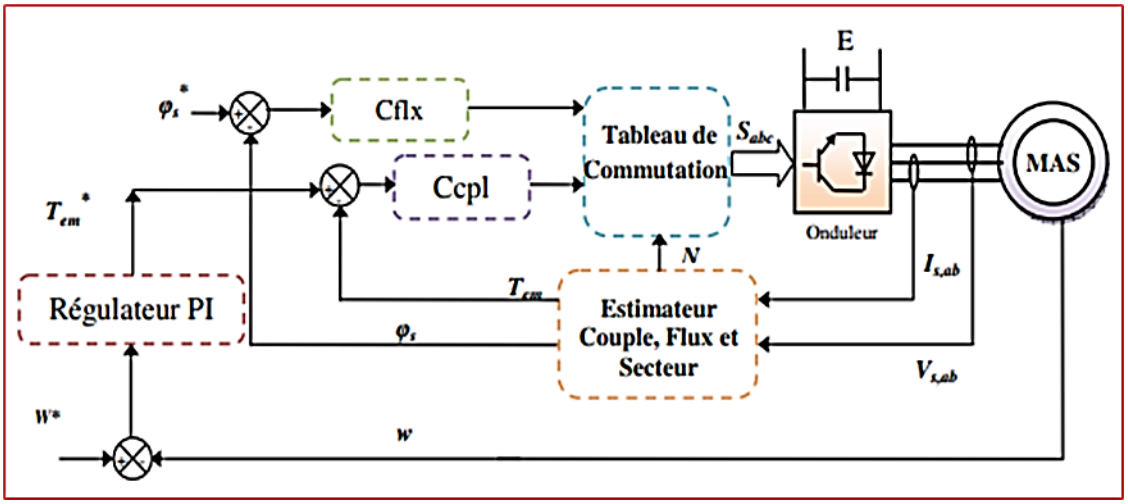

Figure I-6 : Structure de base de la commande directe du couple

I.3.1. Estimation du flux et du couple

Le couple électromagnétique et le flux statorique peuvent s'écrire respectivement

$$C_{em} = \frac{3}{2}.\mathrm{P}.(\varphi_s.I_s) \qquad (8)$$

$$\varphi_s = \int_0^t (V_s - R_s.I_s)dt \qquad (9)$$

Dans le repère(αβ) on obtient respectivement : V_s, I_s , ω_r^*

$$C_{em} = \frac{3}{2}.\mathrm{P}.(\varphi_{s\alpha}.I_{s\beta} - \varphi_{s\beta}.I_{s\alpha}) \qquad (10)$$

$$\begin{cases} \varphi_{s\alpha} = \int_0^t (V_{s\alpha} - R_s.I_{s\alpha})dt \\ \varphi_{s\beta} = \int_0^t (V_{s\beta} - R_s.I_{s\beta})dt \end{cases} \qquad (11)$$

I.3.2. Stratégie de commande

I.3.2.1. Contrôle du vecteur de flux statorique

Le contrôle direct du couple est basé sur l'orientation du flux statorique. L'expression du flux statorique dans le référentiel lié au stator de la machine est obtenue par l'équation suivante [14,15] :

$$\phi_s(t) = \int_0^t (V_s - R_s I_s)\, dt + \phi_{s0} \qquad (12)$$

Dans le cas où ont appliqué un vecteur de tension non nul pendant un intervalle de temps $[0, T_e]$ On aura : $V_s >> R_s I_s$. Donc (13) peut s'écrire :

$$\phi_s(t) = \phi_s(0) + V_s C_{em} \qquad (13)$$

Donc : $$\Delta\phi_s = \phi_s - \phi_s(0) = V_s T_e \qquad (14)$$

L'équation (14) implique que l'extrémité du vecteur flux statorique $\boldsymbol{\phi_s(t)}$ se déplace sur une droite dont la direction est donnée par le vecteur tension appliquée V_s, comme il est illustré par la Figure (I-7).

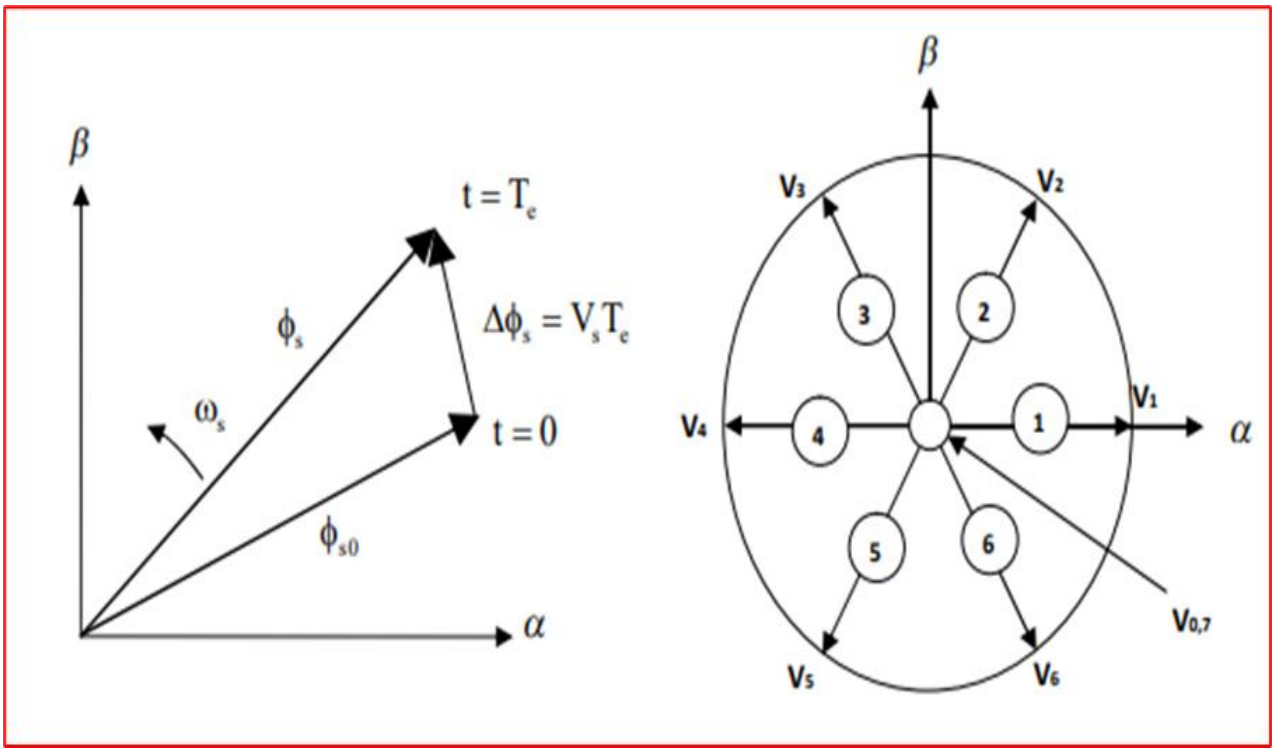

Figure I-7 : Evolution du flux statorique et séquences de fonctionnement

La "composante du flux" du vecteur tension (composante radiale) fait varier l'amplitude de Φs et sa "composante du couple" (composante tangentielle) fait varier la position Φs .

En choisissant une séquence adéquate des vecteurs Vs , sur les périodes de commande Te, il est, donc, possible de fonctionner avec un module de flux Φs pratiquement constant, en faisant suivre à l'extrémité de Φs une trajectoire presque circulaire, si la période Te est très faible devant la période de rotation du flux statorique.

Lorsque le vecteur de tension Vs sélectionné est non nul, la direction du déplacement de l'extrémité de Φs est donnée par sa dérivée $\frac{d\Phi s}{dt}$, Ainsi la "vitesse" de déplacement de l'extrémité de Φs .

Lorsqu'on néglige le terme RsIs , est donnée par Vs =$\frac{d\Phi s}{dt}$. La vitesse de rotation de Φsdépend fortement du choix de Vs ; elle est maximale pour un vecteur Vs perpendiculaire à la direction deΦs , et nul si on appliqué un vecteur nul. Elle peut aussi être négative.

I.3.2.2. Contrôle du couple électromagnétique

Parmi les différentes formes utilisées pour représenter la machine asynchrone, celle qui utilise le flux et le courant statorique, et la vitesse de rotation, comme variable d'état ; sa présentation dans le référentiel statorique (α,β), est généralement celle qui est retenue pour implanter la DTC. Ce modèle est donné par le système d'équations suivant :

$$\begin{cases} \overline{V_s} = R_s\overline{I_s} + \frac{d\overline{\Phi_s}}{dt} \\ \overline{V_r} = \overline{O} = R_r\overline{I_r} + \frac{d\overline{\Phi_r}}{dt} - j\omega\overline{\phi_r} \end{cases} \quad (15)$$

$$\begin{cases} \overline{\Phi_s} = L_s\overline{I_s} + M\overline{I_r} \\ \overline{\Phi_r} = L_r\overline{I_r} + M\overline{I_s} \end{cases} \quad (16)$$

A partir des expressions des flux, on peut écrire :

$$\overline{I_r} = \frac{1}{\sigma}\left(\frac{\overline{\Phi_r}}{L_r} - \frac{M}{L_r L_s}\overline{\Phi_s}\right) \quad (17)$$

avec$:\boldsymbol{\sigma = 1 - \frac{M^2}{L_s L_r}}$étant le coefficient de dispersion, d'où (17) déviant

$$\begin{cases} \overline{V_S} = R_S\overline{I_S} + \frac{d\overline{\Phi_S}}{dt} \\ \frac{d\phi_R}{dt} + \left(\frac{1}{\sigma T_r} - j\omega\right)\overline{\Phi_R} = \frac{M}{L_S}\frac{1}{\sigma T_r}\overline{\Phi_S} \end{cases} \quad (18)$$

Avec la constante de temps rotorique de la machine définie comme :$\mathbf{T_r = \frac{L_r}{R_r}}$

Ces relations montrent que :

– On peut contrôler le vecteur $\overline{\Phi_s}$ à partir du vecteur $\overline{V_S}$, aux chutes de tension $R_S\overline{I_S}$près,

– Le flux $\overline{\Phi_r}$ suit les variations de $\overline{\Phi_s}$ avec une constante de temps σTr. Le rotor agit comme un filtre de constante de temps σTr entre les flux $\overline{\Phi_s}$ et$\overline{\Phi_r}$.

Ceci traduit l'action d'un filtre passe-bas qui existe entre les deux flux. Cette constante de temps détermine aussi la rapidité de variation de l'angle θ_{sr} entre les deux flux statorique et rotorique. $\overline{\Phi_r}$ S'exprime par :

$$\overline{\Phi_r} = \frac{M}{L_s}\frac{\overline{\Phi_s}}{1+j\omega\sigma T_r} \qquad (19)$$

Si on reporte dans l'expression du couple électromagnétique, en posant l'angle$\theta_{sr} = (\overline{\Phi_s}\overline{\Phi_r})$le couple s'exprime par :

$$Cem = K\,(\Phi_s * \Phi_r) = K\|\Phi_s\| * \|\Phi_r\|\sin\theta_{sr} \qquad (20)$$

Avec :

$$K = \frac{pM}{L_r L_s} = p(\frac{1-\sigma}{\sigma M})$$

$\|\overline{\Phi_s}\|$: module du vecteur flux stator,

$\|\overline{\Phi_r}\|$: module du vecteur flux rotor,

θ_{sr} : angle entre les vecteurs flux stator et flux rotor.

Le couple dépend, donc, de l'amplitude des deux vecteurs $\overline{\Phi_s}$et $\overline{\Phi_r}$ et de leur positionrelative, si l'on parvient à contrôler parfaitement le flux $\overline{\Phi_s}$ (à partir de $\overline{V_s}$) en module et enposition, on peut donc contrôler l'amplitude et la position relative de $\overline{\Phi_s}$ et $\overline{\Phi_r}$, donc lecouple. Ceci est bien sur possible si la période de commande T_e de la tension $\overline{V_s}$est telle que$T_e << \sigma T_r$ [14].

I.3.2.3. Sélection du vecteur de tension

Une sélection appropriée du vecteur tension V_s par les interrupteurs (Sa, Sb, Sc) permet de déplacer le vecteur flux$\overline{\Phi_s}$de manière à maintenir son amplitude constante. Le choix de Vs dépend alors de la variation souhaitée pour le module du flux statorique$\overline{\Phi_s}$, du sens de rotation de$\overline{\Phi_s}$, mais également de l'évolution souhaitée pour le couple. Ainsi, nous pouvons délimiter l'espace d'évolution de$\overline{\Phi_s}$dans le référentiel (S) en le décomposant en six zones (N= i), avec (i=1,…., 6), déterminées à partir des composantes du flux dans le plan (α, β) comme indiqué sur figure (I-8) [16].

Lorsque le vecteur flux$\overline{\Phi_s}$se trouve dans une zone numérotée N = i, le contrôle du flux et du couple peut être assuré en sélectionnant l'un des huit vecteurs tensions suivantes : [17]

- Si V_2 est sélectionné alors$\overline{\Phi_s}$croit et Cem croît.
- Si V_6 est sélectionné alors $\overline{\Phi_s}$croît et Cem décroît.
- Si V_3 est sélectionné alors $\overline{\Phi_s}$décroît et Cem croît.
- Si V_5 est sélectionné alors $\overline{\Phi_s}$décroît et Cem décroît.

- Si V_0 et V_7 est sélectionné alors la rotation du flux $\overline{\Phi_s}$est arrêtée, d'ou une décroissance du couple alors que le module du vecteur $\overline{\Phi_s}$reste inchangé

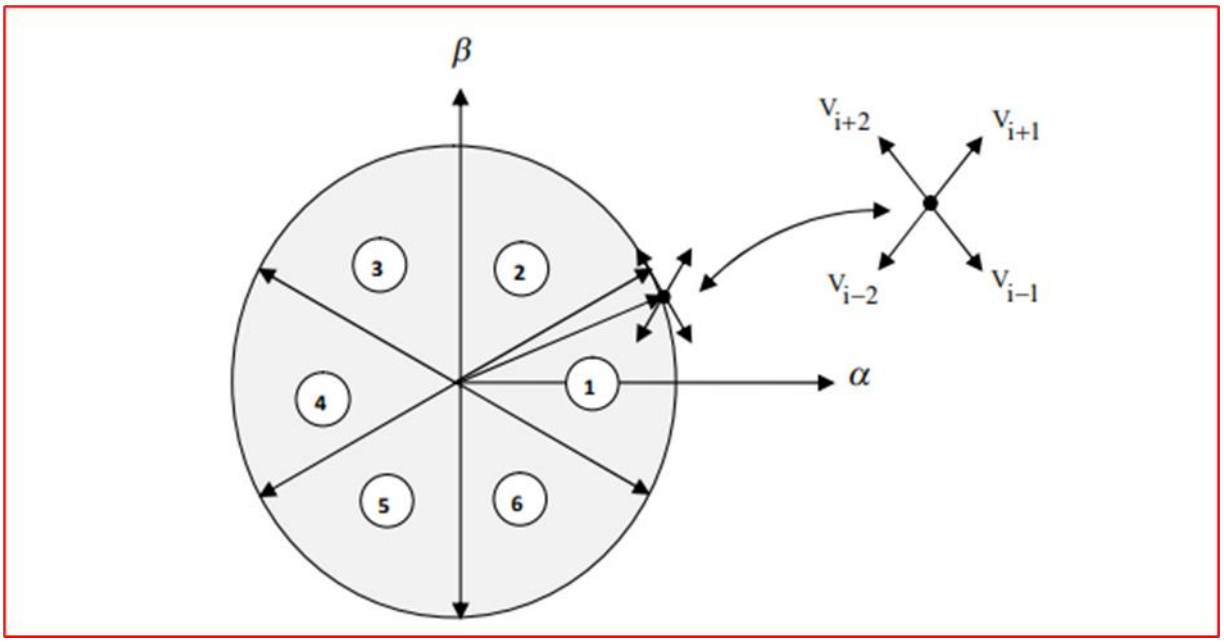

Figure I-8 : Sélection du vecteur tension Vs selon la zone de fonctionnement.

Cependant l'application des vecteurs tensions V_s dépend de la position du vecteur flux de la zone N, en début de la zone N=1, les vecteurs V_2et V_5, sont perpendiculaires au vecteur flux $\overline{\Phi_s}$, ce qui signifie que le changement du couple est très rapide, mais l'amplitude du flux ne change pas considérablement et pour les deux vecteurs V_6et V_3, correspond un composant du couple très petit. Les variations des flux sont importantes. Le vecteur de tension V_sà la sortie de l'onduleur est déduit des écarts du couple et du flux estimés par rapport à leur référence, ce que nécessite un estimateur de module et de position du flux statorique et un estimateur de couple [17].

I.3.3. Elaboration de la table de commutation

La table de commande est construite en fonction de l'état des variables Cflx, Ccpl et la zone Nqui définit la position du vecteur flux statorique Φs . Plusieurs tables de commutation peuvent être utilisées pour contrôler le couple et le flux statorique [17].

Tableau I-1 : Table de commande tenant compte des deux cas du contrôleur de couple

N		**1**	**2**	**3**	**4**	**5**	**6**	Comparateur
Cflx = 1	**Ccpl = 1**	V_2	V_3	V_4	V_5	V_6	V_1	**2 Niveaux**
	Ccpl =0	V_7	V_0	V_7	V_0	V_7	V_0	
	Ccpl = - 1	V_6	V_1	V_2	V_3	V_4	V_5	**3 Niveaux**
Cflx = 0	**Ccpl =1**	V_3	V_4	V_5	V_6	V_1	V_2	**2 Niveaux**
	Ccpl = 0	V_0	V_7	V_0	V_7	V_0	V_7	
	Ccpl = - 1	V_5	V_6	V_1	V_2	V_3	V_4	**3 Niveaux**

On sélectionne l'un des vecteurs nulsV_0ouV_7. La rotation du flux est arrêtée est cela entraîne une décroissance du couple Ce. On choisit alternativement V_0etV_7 de manière à diminuer le nombre de commutations de l'onduleur.

La figure (I-9) représente la stratégie de commande directe de couple d'une machine asynchrone. Le choix judicieux des largeurs de bande des hystérésis des comparateurs de flux et du couple reste l'essentiel pour la réalisation des performances de la DTC.

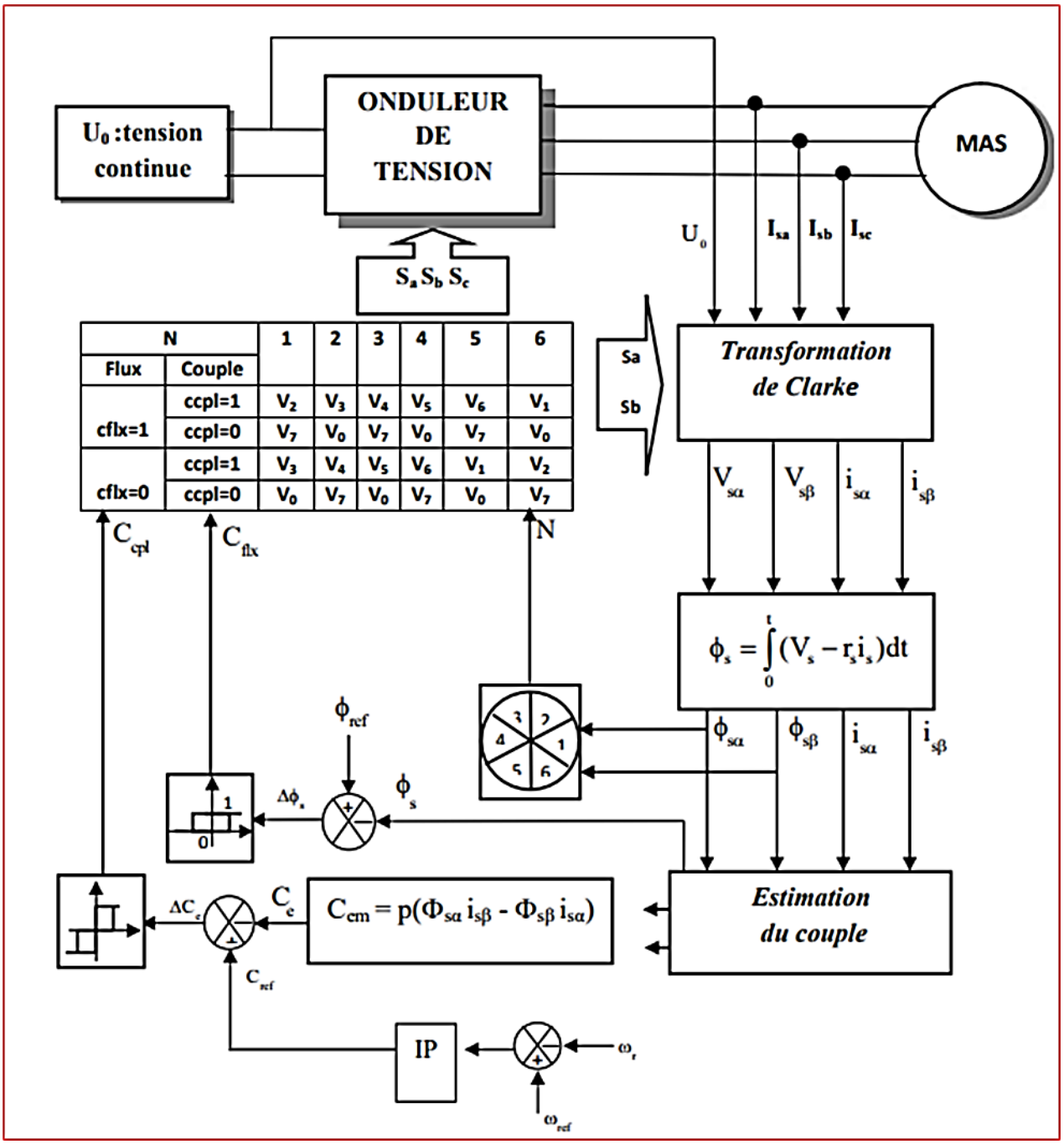

Figure I-9 : Schéma de la structure générale du contrôle directe du couple.

I.4. Conclusion

Dans ce chapitre, nous avons présenté la théorie de la commande directe du couple (DTC) des machines asynchrones (MAS). En effet, cette technique de commande offre beaucoup d'avantage, comme la réponse rapide et précise du flux statorique et du couple électromagnétique. Par contre, les inconvénients majeurs de cette commande sont l'ondulation importante du couple et du flux, la distorsion élevée du courant et la fréquence de commutation est variable due à l'utilisation des comparateurs à hystérésis.

Chapitre II :

Les circuits FPGA et le langage VHDL

II.1. Introduction

Au niveau le plus élevé, un FPGA est un circuit en silicium reprogrammable. A l'aide de blocs logiques préconstruits et de ressources de routage programmables, On peut configurer ce circuit afin de mettre en œuvre des fonctionnalités matérielles personnalisées, sans avoir jamais besoin d'utiliser une maquette ou un fer à souder. Il nous suffit de développer des tâches de traitement numérique par logiciel et de les compiler sous forme de fichier de configuration ou de flux de bits contenant des informations sur la manière dont les composants doivent être reliés. En outre, les FPGAs sont totalement reconfigurables et peuvent adopter instantanément une nouvelle « personnalité » si on recompile une nouvelle configuration de circuits. Jusqu'à présent, seuls des ingénieurs particulièrement expérimentés en matière de conception de matériel numérique pouvaient utiliser la technologie FPGA.

Toutefois, la généralisation des outils de conception de haut niveau est en train de modifier les règles de la programmation de l'FPGA, grâce à de nouvelles technologies permettant de convertir des diagrammes graphiques ou même du code C ANSI 'American National Standards Institute' en circuits matériels numériques. Si les FPGAs rencontrent un tel succès dans tous les secteurs, c'est parce qu'ils réunissent le meilleur des ASIC et des systèmes basés processeur. Ainsi, ils offrent un cadencement par matériel qui leur assure vitesse et fiabilité, mais sont plus rentables que les ASIC personnalisés. Les circuits logiques reprogrammables jouissent également de la même souplesse d'exécution logicielle qu'un système basé processeur, mais ils ne sont pas limités par le nombre de cœurs de traitement disponibles. Contrairement aux processeurs, les FPGAs sont vraiment parallèles par nature, de sorte que plusieurs opérations de traitement différentes ne se trouvent pas en concurrence pour l'utilisation des ressources. Chaque tâche de traitement indépendante est affectée à une sectionne spécifique du circuit, et peut donc s'exécuter en toute autonomie sans dépendre aucunement des autres blocs logiques. En conséquence, on peut accroître le volume de traitement effectué sans que les performances d'une partie de l'application n'en soient affectées pour autant [18]. Pour terminer, soulignons que la conception d'un circuit intégré numérique (ASIC) se fait également à l'aide d'un langage de description matérielle, qui peut être le VHDL [19,20].

Dans ce chapitre, nous allons décrire les circuits FPGA ainsi que le langage de description matériel VHDL, qui est survente utilisée pour reconfigurer de ce type de composants.

II.2. Généralités sur les FPGAs

Un FPGA électronique est un composant constitué de millions voir milliers de transistors connectés ensembles pour réaliser des fonctions logiques. Des fonctions logiques simples peuvent être réalisées telles que des additions ou soustractions tout comme des fonctions complexes peuvent être réalisées telles que le filtrage numérique du signal ou détection d'erreurs et correction. Aérospatial, aviation, automobile, radar, missiles, ordinateur, ne sont que quelques exemples des domaines ayant recours aux FPGAs.

L'évolution des circuits logiques programmable, depuis la création des PALs qui présentaient l'avantage de réduire l'encombrement et de créer des fonctions logiques personnalisé. Puis vient l'étape des EPLDs qui présentent l'avantage de l'écriture électrique mais en ayant recours à un programmateur (un appareil qui permet d'injecter le routage du circuit via un fichier de programmation) mais effaçable à l'UV (ultra-violet) pour évoluer vers les CPLDs qui sont effaçable électriquement. Puis l'avènement des FPGAs qui représentent une technologie qui permet de reprogrammer le circuit in situ (soit sur circuit ou sur site) [21,22].

Xilinx, Altera et Quicklogic sont les pionniers dans le domaine des FPGAs, et plusieurs autres compagnies produisent les FPGAs. Toutes ces compagnies se partagent le même concept architectural4. Il se divise en trois parties : Interfaces d'entrées/Sorties (I/O interface), les blocs logiques de base (Basic Logic Building Blocks) et les interconnexions (voir figure II-1) [19,23].

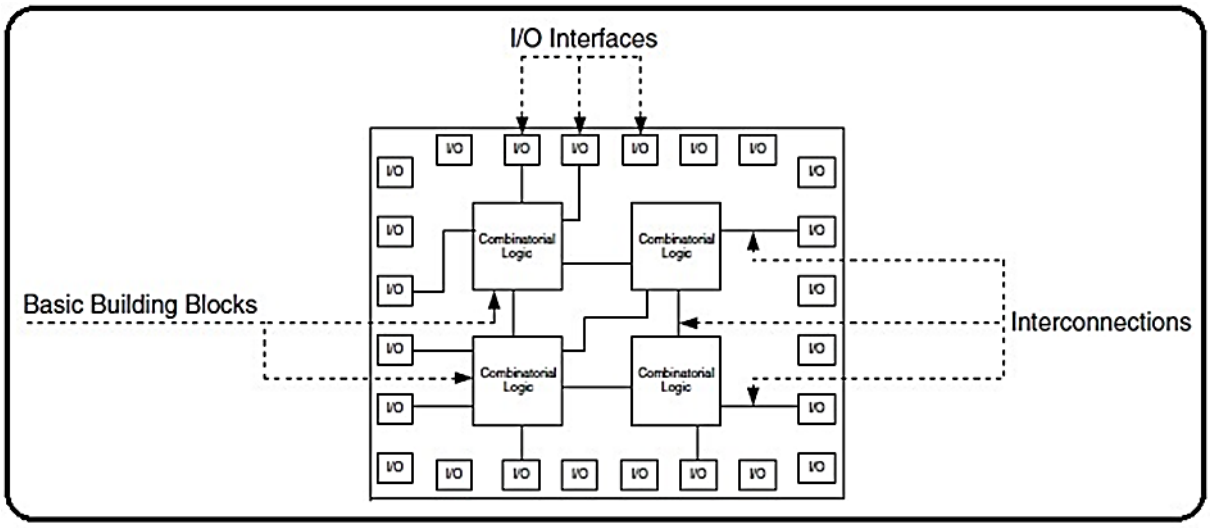

Figure II-1 : Structure de base des FPGAs.

II.3. Architecture interne des FPGA

Un FPGA est un circuit à densité d'intégration très élevée dont l'architecture est formée d'un ensemble de blocs logiques programmables que l'utilisateur peut les interconnecter afin de réaliser les fonctions désirées.

Les circuits FPGAs se composent de plusieurs types de ressources matérielles. Les principaux éléments formant un FPGA sont [25]:

- Les blocs logiques programmables ;
- Les blocs d'entrées/sorties ;
- Le réseau d'interconnexion ;

La figure II-2 illustre l'architecture général d'un FPGA

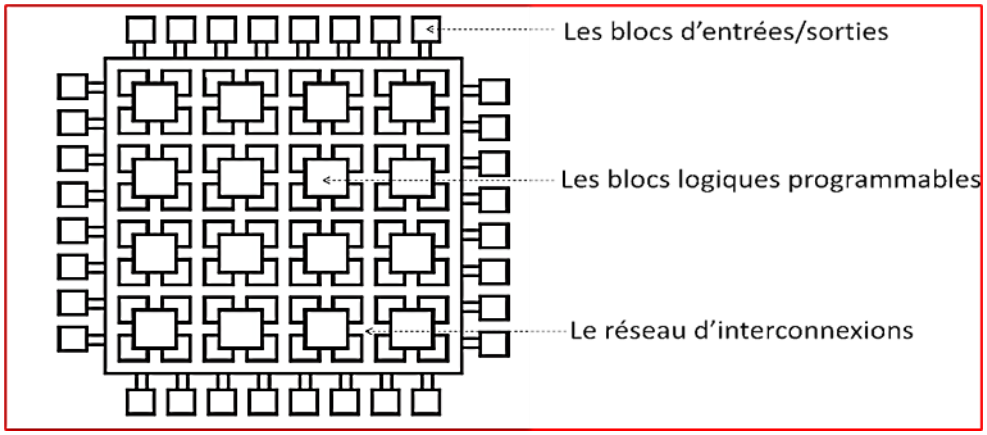

Figure II-2 : Architecture générale d'un FPGA.

Les éléments formant le circuit FPGA sont disposés par groupes et par collons, comme présente la figure ci-dessus.

On note que l'architecture des circuits FPGA se diverse selon le fabricant (exemples : Xilinx ou Altera), mais pas dans l'architecture générale. Cela d'une part. Et d'autre part, selon la technologie des composants utilisés.

II.3.1. Structure des blocs CLB

Les CLB représentent la ressource principale à implémenter pour la logique séquentielle tout comme pour la logique combinatoire, chaque CLB est constitué de SLICES interconnectées, ces slices sont disposés en paires et chaque paire est disposée en colonne. La paire de gauche est appelé SLICE et celle de droite SLICEO. [24].

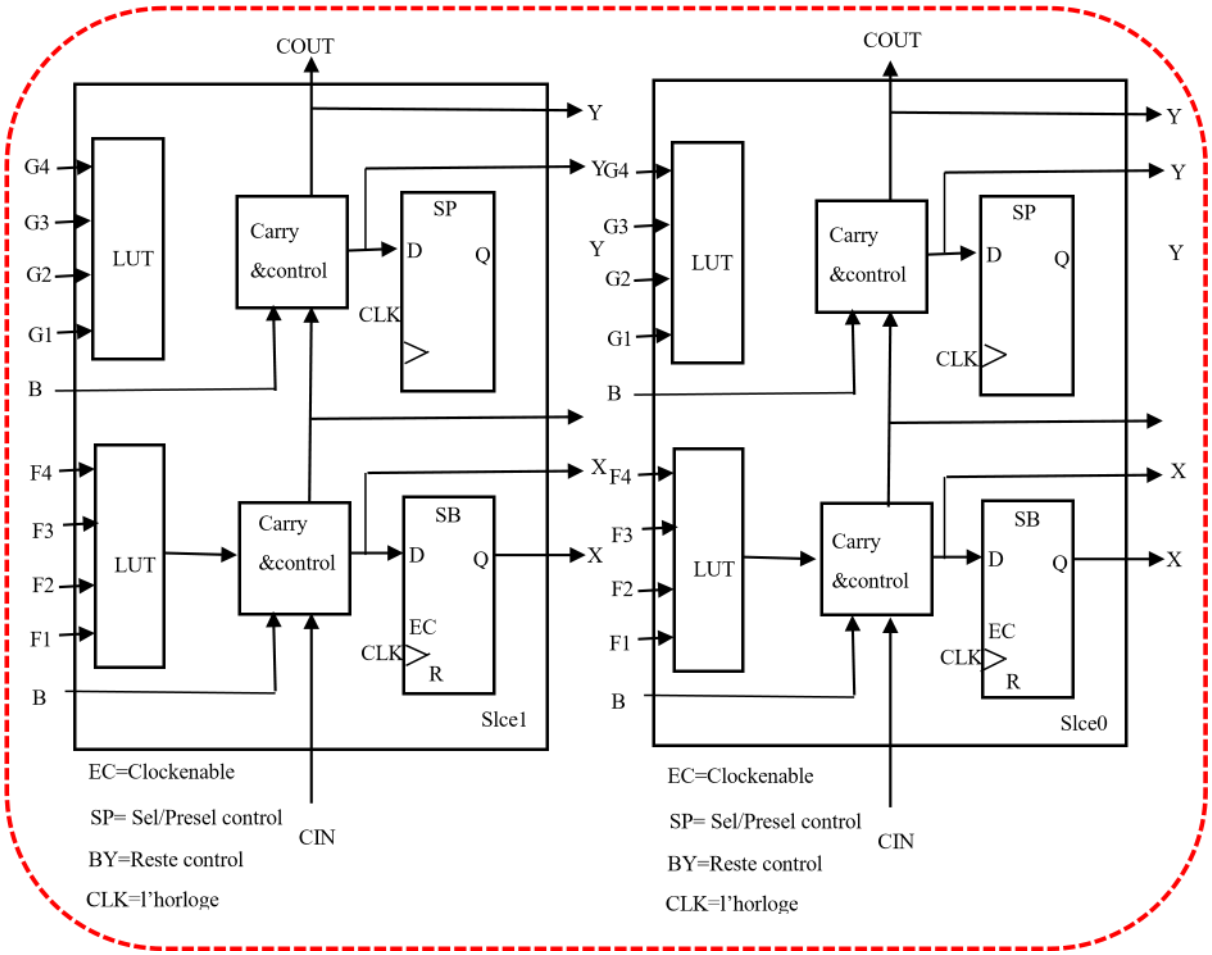

Figure II-3 : Architecture d'un CLB de famille Virtex**.**

L'architecture d'un SLICE est comme suit :

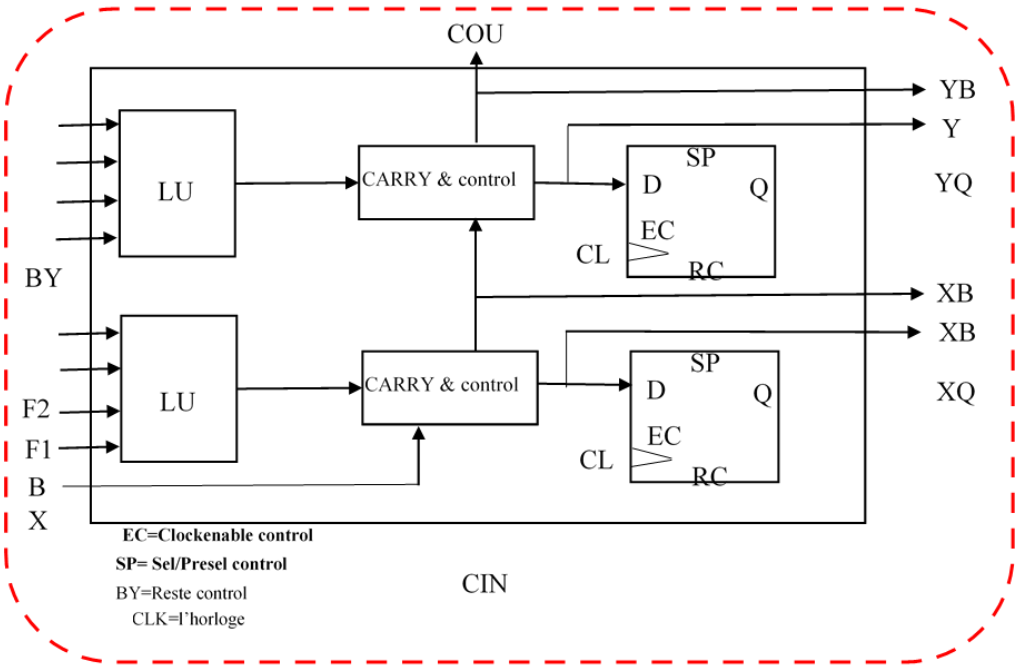

Figure II-4 : Architecture d'un SLICE de famille Virtex.

II.3.2. Structure des blocs IOB

Les blocs d'entrées/sorties disposent aussi de bascules de contrôle à trois états :

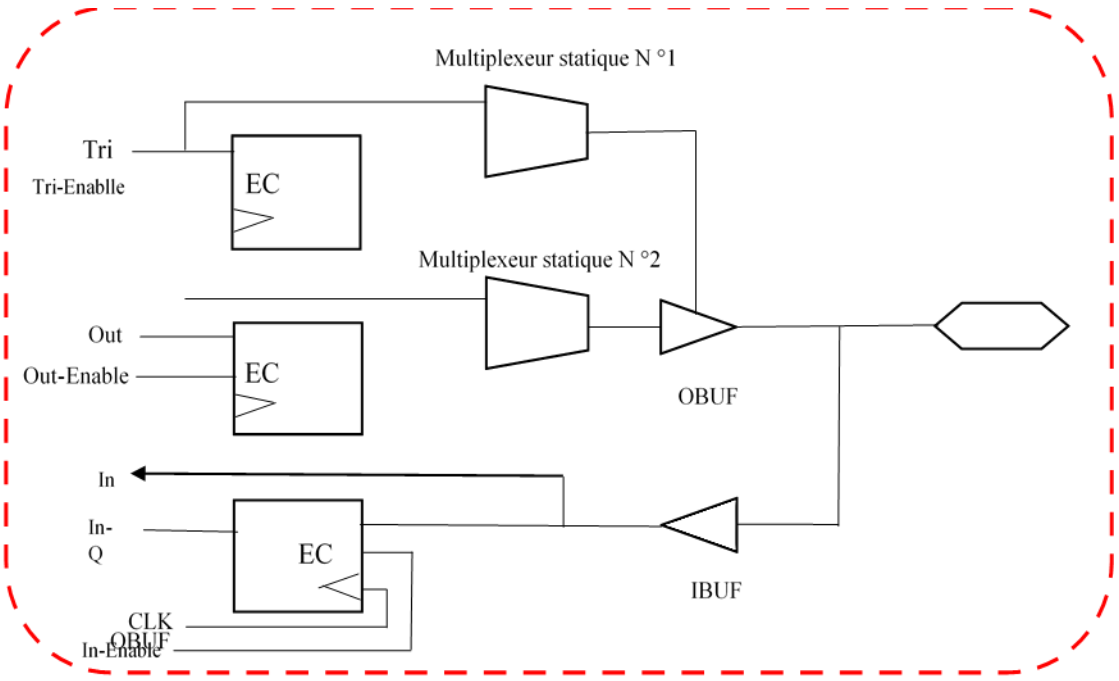

Figure II-5 : Architecture d'un IOB de famille Virtex.

II.3.3. Interfaces d'entrées/Sorties (I/O interfaces)

Les interfaces d'entrées/Sorties se présentent comme les intermédiaires par lesquelles les données transitent depuis les blocs logiques internes jusqu'aux ressources externes et vice versa. L'interfaçage des signaux peut être : unidirectionnel, bidirectionnel, à deux états ou trois états (0, 1 ou haute impédance) voir même obéir à un standard d'entrées/sorties. Voici quelques-uns de ces standards:

_ GTL (gunning transceiver logic).

_ HSTL (high-speed transceiver logic).

_ LVCMOS (low-voltage CMOS).

_ LVTTL (low-voltage transistor-transistor logic).

_ PCI (peripheral component interconnect)…

Le rôle principal des interfaces d'entrées/sorties est de transmettre et de recevoir des données. Néanmoins l'interface d'entré/sorties peut être dotée d'options telles que des registres, impédances et buffers.

Chaque fabricant a sa propre appellation pour désigner l'interface d'entrées/sorties mais la fonction reste toujours la même. Altera, les nomme IOE Input Output Element. L'IOE remplie toujours son rôle d'interface d'entrées/sorties, elle dispose d'une résistance de rappel pull-up et un temporisateur du signal [27].

La Figure II-6 est un exemple d'IOE de la série Cyclone 2

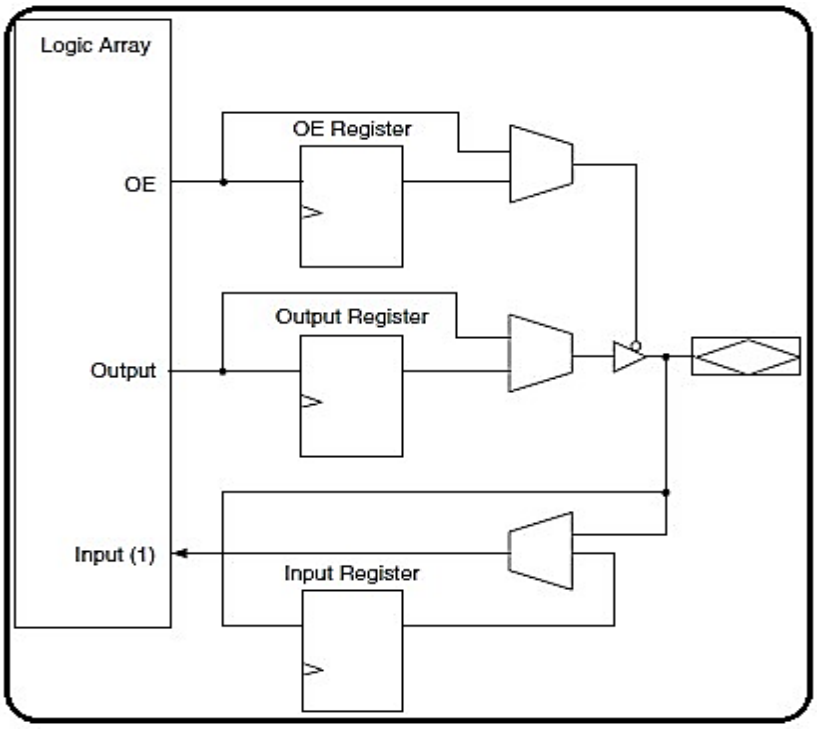

Figure II-6 : structure interne d'IOE de la série Cyclone 2.

II.4. Avantages et inconvénients des FPGA

II.4.1. Avantages

- Un circuit reprogrammable : L'avantage du FPGA est de pouvoir être reprogrammable contrairement aux circuits intégrés de type ASIC. Ce qui rend cette solution modulable et donne la possibilité de modifier le programme générique de base afin de le rendre spécifique au circuit utilisé.

- Une solution de validation utilisant le FPGA peut alors convenir à beaucoup de projets et donc diffusée à plusieurs équipes.

- Un investissement rentable dans la durée : Cela est dû à sa reprogrammation, ce qui implique une réutilisation à destination d'autres projets, malgré un prix à l'achat supérieur à un circuit ASIC. - Une Reprogrammation quasi-instantanée du circuit. Une fois le programme validé cela ne prend que quelques minutes à l'implémenter. A titre de comparaison, la fabrication d'un circuit ASIC peut prendre plusieurs semaines.

Cependant, le FPGA n'est pas le seul composant reprogrammable du marché. Le DSP « Digital Signal Processor », processeur de signal numérique, permet également d'émuler un montage numérique. Le DSP est programmable grâce au langage C, le FPGA utilise quant à lui le VHDL. Le format de description machine est généré automatiquement par les logiciels de développement des concepteurs. Il est possible alors de vérifier les circuits sans avoir à les concevoir par nous-mêmes.
L'importation de leur fichier dans notre programme est suffisante. Par ailleurs, le FPGA peut disposer d'un DSP sous forme d'IP incluse dans le système du circuit. [24].

II.4.2. Inconvénients

- Performances non optimisées.
- Temps de réponse long par rapport aux ASIC

II.5. Les Interconnexions

Les interconnexions sont très importantes car ce sont elles qui transmettent le signal d'un point à un autre. Il existe plusieurs types de lignes :

II.5.1. Les interconnexions à usage général

Ils sont composés de segments verticaux et horizontaux qui entourent chaque bloc logique de base et qui peuvent être reliés entre eux par une matrice de communication.

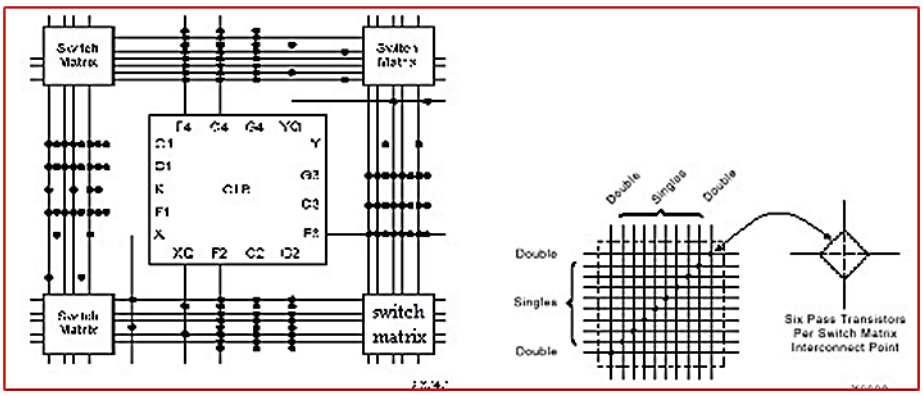

Figure II-7 : Connexions à usage général et détail d'une matrice de commutation.

II.5.2. Les lignes directes

Fournissant des chemins entre les blocs logiques de base adjacents et entre les blocs logiques de base et les interfaces d'entrées/sorties.

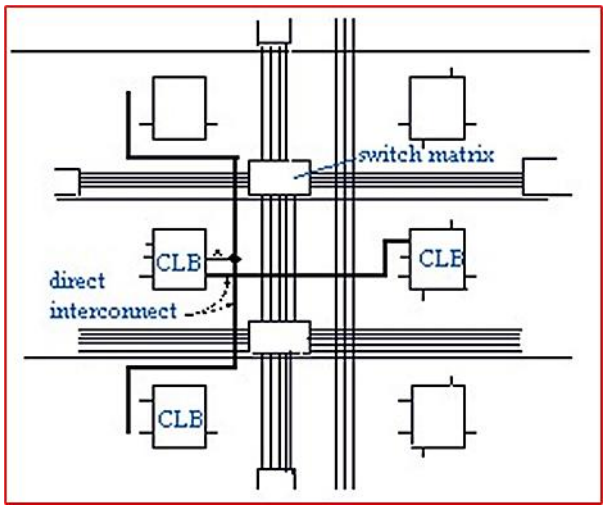

Figure II-8 : Les interconnexions directes.

II.5.3. Les lignes longues

Qui sont des lignes verticales et horizontales qui n'utilisent pas de matrices de communication. Elles parcourent toutes les zones d'interconnexion. Elles sont utilisées pour véhiculer les signaux qui doivent parcourir de long trajet. Ces lignes conviennent pour véhiculer les signaux d'horloge [28].

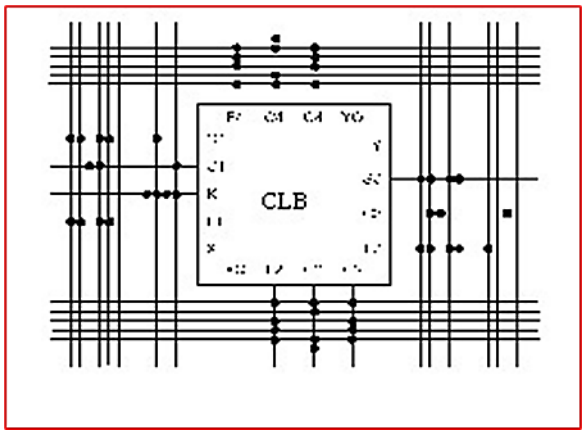

Figure II-9 : Les lignes longues

En plus de la diversité architecturale il existe deux types de FPGAs :

- Les FPGAs de type SRAM, appelé LCA chez XILINX, FLEX chez ALTERA.
- Les FPGAs à antifusibles proposé entre autre, par Texas Instrument et ACTEL.

Ces deux technologies différentes essentiellement dans la façon dont leurs interconnections sont réalisées.

- Les FPGAs de type SRAM :

Introduit dès 1985 par Xilinx, la configuration du composant réside en deux points : la définition des interconnexions définies par des cellules SRAM et les fonctions booléennes définies dans des mémoires de type SRAM.

La mémorisation de la configuration est donc volatile. Ces boîtiers sont programmés après chaque mise sous tension. Les données de programmation sont issues soit directement d'une ROM (mémoire à lecture seul), soit d'un système intelligent (microprocesseur ou microcontrôleur).

- Les FPGAs à antifusibles :

Une technologie apparue en 1990 et elle est différente des autres FPGAs car ceux-là ne sont pas reprogrammables, voir même l'architecture interne qui diffère. Ils sont proposés pour l'essentiel par Texas Instrument.

L'architecture globale de ces FPGAs est analogue à celle des LCA, avec un certain nombre de blocs d'entrées/sorties répartis tout autour de la puce. Des blocs logiques placés en matrice au centre de celle-ci et des lignes d'interconnexions, mais la similitude s'arrête là.

En effet, alors que les LCAs faisaient appel à des cellules logiques de base relativement complexes, les CLB, des FPGAs à antifusibles utilisent des cellules très simples. Comme référence, la famille TCP 10XX et TCP 12XX de TEXAS INSTRUMENT [19,23].

II.6. Applications

Les circuits FPGAs sont utilisés dans divers domaines d'applications comme le domaine militaire, le domaine médical, domaine de la recherche et de développement, etc. [26]

II.6.1. Domaine militaire

Les circuits FPGA de Xilinx ou Altera permettent aux concepteurs de répondre aux exigences de l'industrie aérospatiale et militaires. Cela afin de développer des systèmes de haute performance, d'une large plage de fonctionnement et d'une longue durée de vie du système.

II.6.2. Domaine médical

Les performances technologies des circuits FPGAs permette l'utilisation de ces types de composants dans les applications d'acquisition et du traitement d'images.

Dans le domaine industriel, on trouve comme exemple le fabricant Altera qui fournit des circuits FPGAs avec des outils pour optimiser les performances technologiques des systèmes en vue de leurs utilisations dans des applications médicales.

II.6.3. Domaine de la communication

Beaucoup d'efforts de la part de Altera et de Xilinx pour le développement des systèmes et des outils pratiques et flexibles utiles dans le domaine de la communication cellulaire mobile.

II.6.4. Domaine de l'automobile

Les circuits FPGAs pouvant être utilisés dans la technologie des automobiles évoluées tels que les véhicules hybrides et les véhicules électriques.

II.6.5. Domaine de la recherche & développement (R&D)

Actuellement, les circuits FPGAs sont très utilisés dans divers domaines de la R&D. Citant comme exemples : la commande des systèmes à temps réel, le traitement d'images (du

son et de la vidéo), l'acquisition des données ainsi que le test/mesure. Ce type de composant joue un rôle incontournable dans la conception et le prototypage des circuits ASICs. [26]

II.7. Le langage de description matériel VHDL

Le VHDL est un langage de description matériel HDL. Il répond à tous les critères établis pour un langage HDL

Les plus importants points forts de ce langage sont :

Simulation, synthèse : le VHDL permet d'avoir des lignes de code pouvant être simulé (simulation fonctionnelle) dans le but de vérifier si le code obéit correctement à la fonction souhaitée mais pour parvenir à une maquette réalisable ce n'est pas suffisant. D'où la nécessité de pouvoir synthétiser le même code. Lors de la synthèse, le compilateur traduit le premier code en un autre équivalent mais à un niveau d'abstraction plus bas.

À ce niveau de synthèse, le compilateur traduit le code de haut niveau en portes logiques ceci en fonction du circuit logique programmable qui est ciblé. Le fichier synthétisé lui aussi, se doit d'être simulé (simulation événementielle) et vérifier si le compilateur (synthétiseur) est resté fidèle aux exigences de départ. Puis le fichier est compilé une deuxième fois pour aboutir au circuit logique qui sera implémenté.

Cette fois encore, une troisième simulation (facultative) (simulation temporelle) pour être certain que le circuit est resté inchangé.

Portabilité : avec deux objectifs, portable vis-à-vis du circuit logique programmable, c'est-àdire peut-être implémenté sur l'importe quel circuit logique programmable à condition d'avoir la capacité logique requise. Portable vis-à-vis du compilateur, pouvoir passer d'un compilateur à un autre et obtenir un même circuit en fin de processus.

Une construction hiérarchique : cette approche permet de simplifier la conception puis ce que les taches peuvent être divisées pour aboutir au point le plus simple que possible puis faire un assemblage des blocs.

Une description fonctionnelle : complémentaire de la précédente, la vision fonctionnelle apporte la puissance des langages de programmation. Tout algorithme est la description interne d'un bloc situé quelque part dans la hiérarchie du schéma complet. La vision structurelle est concurrente, la vision algorithmique est séquentielle

Un programme VHDL doit être compris comme l'assemblage en parallèle des taches indépendantes qui s'exécutent concurremment.[29].

II.7.1. La Structure et l'architecture d'un programme VHDL

Un opérateur élémentaire, un circuit intégré, une carte électronique ou un système complet, est complètement défini par des signaux d'entrées et de sorties et par la fonction réalisée de façon interne.

Les concepteurs du VHDL ont adopté l'approche suivante : l'importe quel système est considéré comme une boite noire. Cette boite noire a des entrées et des sorties. Ils ont appelé la boîte noire « ENTITY ». L'importe quel système électronique effectue des opérations sur le signal d'entré pour donner le résultat du traitement en sortie. Ces opérations sont le contenu de la boîte noire ce contenu est appelé « ARCHITECTURE ».

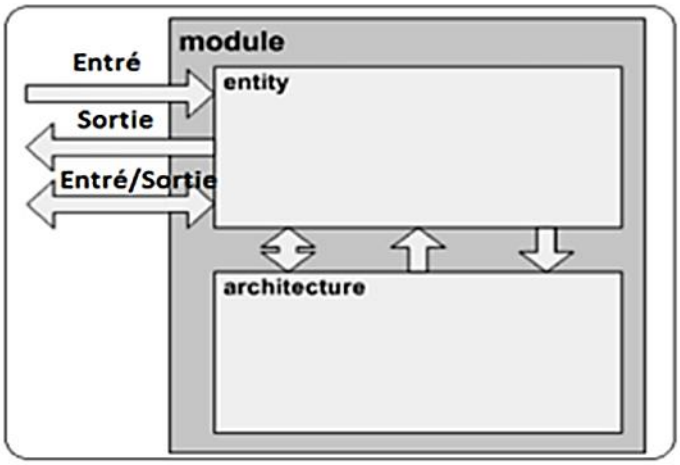

Figure II-10 : structure de base d'un module sous VHDL

Entité : Dans la partie déclarative de l'entité le circuit est décrit comment il est vu par l'extérieur ceci à travers les entrées, sorties et entrées/sorties.

```
Entity <entity name> Is Port(
   <signal name   : <signal direction> <data type>);
End <entity name>;
```

Figure II-11 : Syntaxe déclarative de l'entité.

Architecture : l'architecture décrit le comportement que doit avoir le circuit ou les opérations qu'il doit effectuer. Une architecture se doit toujours d'être attachée à une entité (ils vont de pair)

```
Architecture <architecture name> Of <entity name> Is
<Define signals and constants>
Begin
End <architecture name>;
```

Figure II-12 : Syntaxe déclarative de l'architecture.

C'est dans cette section que le programme est rédigé. Un programme comporte essentiellement les éléments suivants : les signaux internes, opérateurs logiques (synchrone ou concurrent), La description d'une architecture peut prendre trois formes :

Description comportementale : Ce type de description spécifie le comportement du composant ou du circuit à réaliser au moyen d'instructions séquentielles ou sous forme de flot de données (constituant un process).

Description structurelle : Dans ce type de description, les interconnexions des composants préalablement décrits sont énoncées. Cette description est la transcription directe d'un schéma.

Description mixte : Elle regroupe les deux descriptions décrites précédemment. À chaque entité peut être associée à une ou plusieurs architectures mais au moment de l'exécution (simulation, synthèse…) seulement une architecture et une seule est utilisée [29].

II.7.2. Les Éléments du langage VHDL

II.7.2.1. Le signal

Est représenté par certains types de données. Il est assigné par un nom et un type de donnée comme suit :

Signal « nom_signal » : « type_donné» ;

Le terme Signal indique implicitement au compilateur de convertir le signal en un canal d'interconnexion.

II.7.2.2. Les constantes

La constante est un objet qui porte une valeur à l'initialisation l'enregistre tout au long du déroulement du programme. Elle est déclarée par le terme **Constant**.

II.7.2.3. Les variables

La valeur d'une variable est immédiatement mise à jour lors ce qu'elle est assignée. Une variable peut être déclarée soit à l'intérieur d'un processus ou d'un sous-programme et elle est

local à son bloc de déclaration. Une variable n'est utilisée que dans le domaine de la programmation séquentielle. Une variable est déclarée comme suit :

Variable « nom_variable » : type « expression initiale » ;

II.7.2.4. Les types de données

Le VHDL est un langage très typé. Chaque élément utilisé (signal, variable) doit avoir un type bien défini. Toute opération doit se faire avec des objets d'un même type. Sous VHDL les types de données sont standardisés et on y réfère par std_logic, std_logic_vector, bit, bit_vector… Ils sont regroupés dans les bibliothèques par défaut. L'utilisateur peut créer lui-même un type de donnée et l'intégré dans une bibliothèque. Le tableau suivant montre les valeurs admises par le type standard logicvector.

II.7.2.5. Le Process

Il permet de décrire les instructions parallèles qui utilisent les mêmes signaux. Un process peut être vu comme un sous-programme en boucle infinie qui ne s'arrête qu'à l'instruction de synchronisation **wait** (attendre). L'énoncé process est habituellement accompagné d'une liste de sensibilité (horloge, bit d'activation ou de sélection…) et il est exécuté lors ce qu'un des signaux de la liste de sensibilité change d'état.

II.7.2.6. Les instructions concurrentes

Les instructions concurrentes sont décrites directement dans le corps de l'architecture. L'objectif de ces relations étant d'affecter des valeurs à des composants ou de réaliser des connexions, alors elles n'ont pas d'ordre d'exécution. Elles sont exécutées en parallèle.

II.7.2.7. Les instructions séquentielles

Les instructions séquentielles sont internes aux processus, aux procédures et aux fonctions (sous-programmes). Elles constituent les outils de base des descriptions comportementales. Ces instructions, pour la plupart, sont inspirées des langages classiques de programmation. Parmi elles, on retrouve **if-then-else** qui permet de réaliser les boucles conditionnelles. **Case-then** pour tester des valeurs et choisir l'opération à effectuer.

L'instruction **wait**, qui permet de suspendre l'exécution d'un processus jusqu'à ce que la durée spécifiée soit évaluée. Les boucles telque : **for-loop, while-loop** …

II.7.2.8. Les attributs

Ce sont des propriétés ou des caractéristiques associées à un objet ou un type.

Ils sont soit prédéfinis ou définis par l'utilisateur. Ils sont déclarés comme suit :

Nom_var' nom_attribut ;

II.7.2.9. Les Librairies et les Paquetages

Ce sont les librairies qui font qu'un programme et transportable d'un programme à un autre et permettent aussi leurs réutilisations. Une librairie est constituée d'un ou plusieurs paquetages et ce sont ces derniers qui renferment les fonctions, les procédures, les constantes, les types… elles sont fournies soit par l'IEEE en tant que partisan du standard VHDL ou bien créé par le concepteur lui-même. La syntaxe déclarative se fait comme suit :

Library IEEE ; pour déclarer la librairie

Use IEEE.STD_LOGIC_1164.ALL ; pour choisir le paquetage Standard Logic 1164

II.7.3. Structure d'un programme sous VHDL :

Un programme écrit sous VHDL obéit à la structure suivante :

II.7.3.1. Entête

C'est une partie facultative, elle referme des informations concernant le programmeur, la description du programme en général, la date de rédaction et toute information qui semblera importante pour celui qui rédige le programme.

II.7.3.2. Déclaration des librairies

En deuxième lieu vient la déclaration des librairies et des paquetages que le programmeur juge nécessaire pour son programme.

II.7.3.3. Déclaration d'entité

En troisième lieu vient la déclaration d'entité avec les signaux à utiliser dans tout le programme avec leur direction.

II.7.3.4. Déclaration d'architecture

Juste après l'entité vient la déclaration de l'architecture qui décrit l'entité. Dans cette partie les signaux internes sont déclarés et les variables propres à cette même architecture. Puis le programme est rédigé [29]. Voir la Figure (II-13).

```
Library IEEE;
Use IEEE.std_logic_1164.all;

Entity <entity name> Is Port
  (<list of ports or design inputs and outputs>);
End <entity name>;

Architecture <architecture name> Of <entity name> Is
  <in this section define signals and constants>

Signal <signal name>        : Data Type;

Begin
<concurrent statements>

<process name>: Process (sensitivity list)
Begin
  <sequential statements>
End;

End <architecture name>;
```

Figure II-13 : Structure d'un programme sous VHDL

II.8. Intégration et implémentation

L'implémentation est la réalisation proprement dite qui consiste à mettre en œuvre l'algorithme sur l'architecture du circuit configurable cible, c'est-à-dire à compiler, charger, puis lancer l'exécution sur un ordinateur ou calculateur.

C'est une étape de programmation physique et de tests électriques qui clôture la réalisation du circuit. La figure (II-14) résume un peut l'ensemble de ces étapes [24].

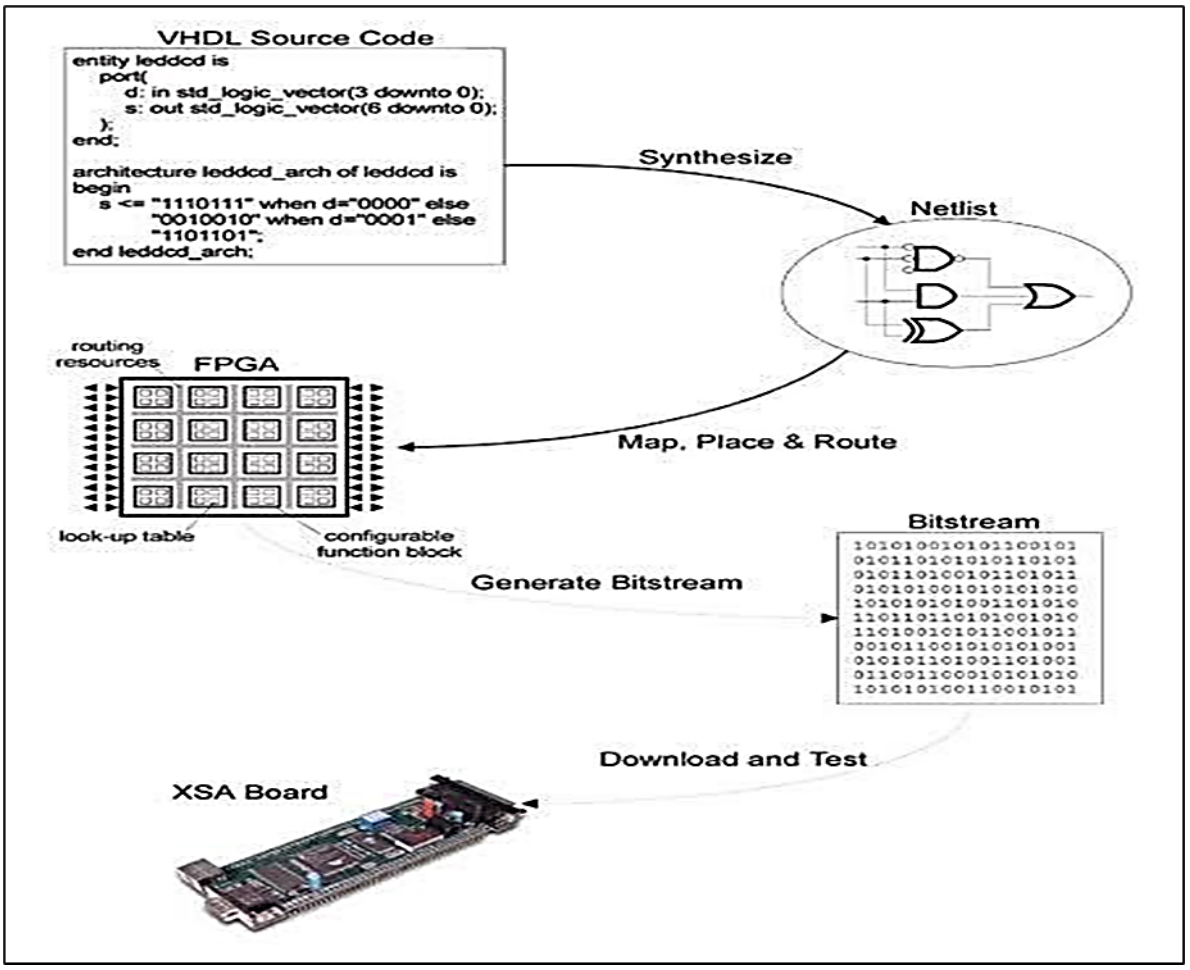

Figure II-14 : Etapes de conception sur FPGA

II.9. Conclusion

Dans ce chapitre, nous avons présenté l'architecture des réseaux FPGA ainsi que le langage VHDL. En effet, le couple FPGA/VHDL est largement utilisé dans le cadre de ce projet.

Dans la première partie de ce chapitre, nous avons présenté l'architecture des circuits FPGA, et nous avons décrit les principaux blocs formant un composant FPGA : Les blocs d'entrées/sorties, aussi ces avantages et inconvénients, les différents types des réseaux d'interconnexion et les blocs logiques configurables ainsi que les différentes technologies utilisées dans les circuits FPGA et les domaines d'application.

Dans la deuxième partie, nous avons passé au langage de description matériel VHDL. Dans ce cas, nous avons décrire la structure et l'architecture d'un programme VHDL, les différents types de description et les éléments de langage VHDL aussi on a présenté la structure d'un programme sous VHDL, les instructions concourantes et séquentielles et leurs syntaxes ainsi les différents attributs.

Dans le chapitre suivant nous allons aussi décrit les détails de plateforme de prototypage rapides de compagnie Altéra pour les utiliser ensuite dans nos implémentations. Nous allons présenter aussi le logiciel de Altéra (Quartus2), ce qui donne la possibilité de comparaison concernant le nombre des ressources utilisées.

Chapitre III :

Implémentation et simulation de la commande DTC sur FPGA

III.1. Introduction

Actuellement les techniques de l'intelligence artificielle sont très utilisées dans le cas des problèmes liés à l'automatisation des processus industriels, tels que le contrôle, la commande et l'estimation des paramètres des systèmes électriques.

Les appareils Cyclone II comprennent un ensemble de fonctionnalités FPGA puissantes optimisées pour les applications à faible coût, y compris une large gamme d'options de densité, de mémoire, de multiplicateur intégré et de conditionnement. Les périphériques Cyclone II prennent en charge une large gamme d'interfaces de mémoire externe et de protocoles d'E / S courants requis dans les applications à faible coût [30]. Quant au le logiciel QUARTUS, il est dédié à la programmation des CPLD et FPGA du fabricant Altéra ; il permet la description d'un projet, sa compilation, sa simulation logique et temporelle, son analyse temporelle et la programmation d'un circuit cible [31].

Dans ce chapitre, nous allons présenter les résultats de l'implémentation de la commande directe du couple (DTC) d'une machine asynchrone (MAS) sur un FPGA Altera à laide du langage VHDL. Ce chapitre se composera de deux parties. Dans la première partie, nous allons présenter les caractéristiques des appareils Cyclone II et Le logiciel QUARTUS que nous avons utilisé pour la présentation et la simulation et Dans la deuxième partie, nous allons implémenter et simuler l'algorithme général de la commande DTC en considérant un ALTERA EP2C5T144C8.

III.2. Caractéristiques des appareils Cyclone II

Les principales caractéristiques des appareilles Cyclone II sont [32]:

• Une architecture haute densité avec 4608 et 68416 LE (Eléments logiques).

• Ils intègrent jusqu'à 1.1 Mbits de RAM (M4K Blocks) avec :

- Une largeur de bus de données configurable (x1, x2, x4, x8, x 16, x32 et x36).
- Un mode double accès,
- Une vitesse de fonctionnement pouvant atteindre 260 MHz.

• Ils comprennent une zone où sont gravés dès la fabrication jusqu'à 150 Multiplexeurs 18x18 (Embeded Multipliers).

• Ils possèdent des blocs d'entrées/sorties avancées (IOE : Input/Output Element) avec :

- Des entrées/sorties différentielles (LVDS, LVPECL . . .),
- Des entrées/sorties simples (3.3v, 2.5v, 1.8v et 1.5v LVCMOS, 3.3v, 2.5v et 1.8v LVTTL …),
- Une compatibilité avec les signaux PCI et PCI Express.

• Il y a une circuiterie dédiée aux horloges (PLL) :

- 2 à 4 PLL pour multiplier ou diviser les fréquences d'horloge entrantes, les retarder ...
- Jusqu'à 16 lignes d'horloges en interne
- Une gestion jusqu'à la fréquence maximale de 402.5 MHz

• Un dispositif de configuration avec :

- Un mode rapide pour une configuration en moins de 100 ms
- Mode série ou JTAG possibles avec des mémoires de configuration série de bas coût.

III.3. Présentation du logiciel Quartus

Dans le cadre de ce projet, nous avons utilisé le Software Quartus. L'outil Quartus est un logiciel développé par la société Altéra, permettant la gestion complète d'un flot de conception CPLD -Complex Programmable Logic Device- ou FPGA -Field Programmable Gate Array-. Ce logiciel permet de faire une saisie graphique ou une description HDL -Hardware Description Language- (VHDL – Very Hardware Description Language- ou AHDL - Altera Hardware Description Language-) d'architecture numérique, d'en réaliser une simulation, une synthèse et une implémentation sur cible reprogrammable.

Il comprend une suite de fonctions de conception au niveau système, permettant d'accéder à la large bibliothèque d'IP d'Altera et un moteur de placement routage intégrant la technologie d'optimisation de la synthèse physique et des solutions de vérification. De manière générale, un flot de conception ayant pour but la configuration de composants programmables.

La version utilisée pour ce tutorial est QUARTUS II Version 13.0 elle est gratuite est téléchargeable sur le site d'INTEL.

III.4. Implémentation et résultats de simulation

III.4.1. Décomposition algorithmique

Afin de réaliser le travail demandé, notre stratégie consiste à décomposer la DTC en quatre blocs réalisant les fonctions principales assurées par le circuit FPGA de la famille Altera.

La figure III.1 montre les quatre blocs relisant les fonctions principales de la commande DTC d'une machine asynchrone.

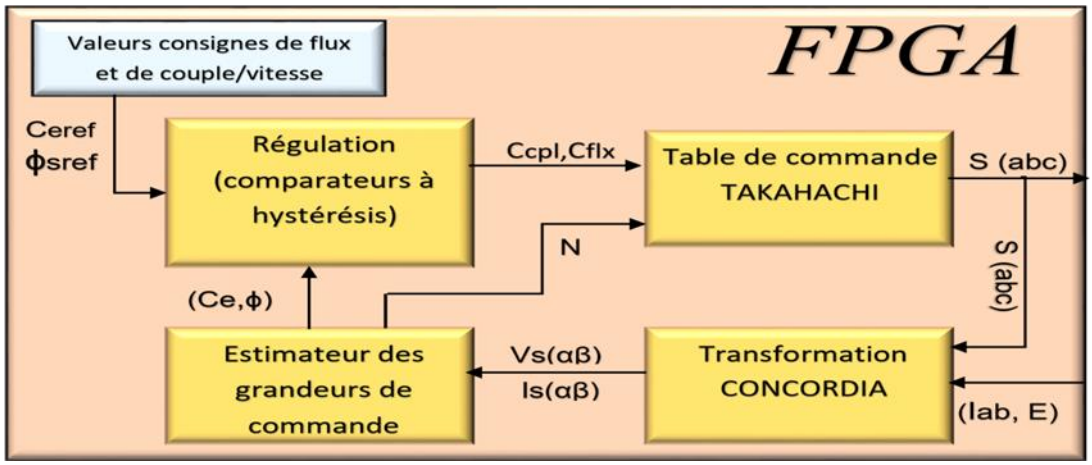

Figure III-1 : Les fonctions principales d'un système de commande DTC d'une MAS.

L'algorithme global de la commande DTC se composera ainsi de quatre principaux blocs: traitement, estimation, régulation et commande. Ceci comme montre la figure III.2. Chaque bloc assure une des quatre fonctions de la commande DTC de la MAS.

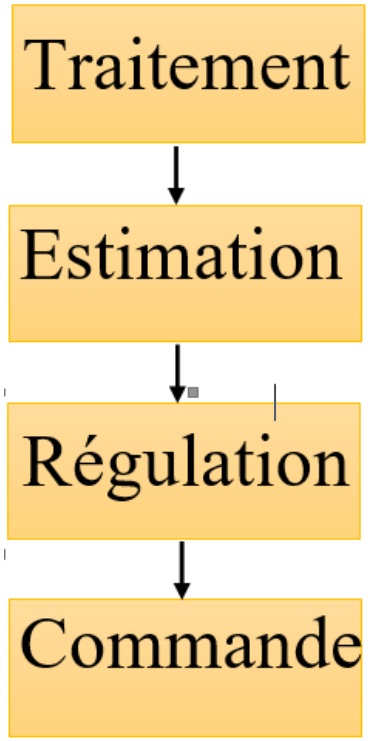

Figure III-2 : Algorithme général de la commande DTC.

III.4.2. Bloc de traitement (Transformation CONCORDIA TC)

Ce module réalise les opérations arithmétiques qui ramènent les grandeurs statoriques issues des capteurs au repère diphasé (α,β) lié au stator en utilisant une description VHDL flot de données. La figure III.2 présente la description VHDL du bloc de traitement, sachant que nous avons utilisé de nouveaux symboles parce que le logiciel QUARTUS II ne reconnait pas les symboles ϕ, β et α utilisés dans les équations de traitement et d'estimation des grandeurs de la commande.

Les symboles adaptés sont présentés dans le tableau III.1.

Tableau III-1 : Les symboles utilisés

Symbole mathématique	Symbole utilisé
A	Alpha
B	Beta
Φs	Qs

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity TC is
6  Port (S:inout std_logic_vector(2 downto 0);
7        Ia,Ib:in unsigned(7 downto 0);
8        Vsa:out signed(25 downto 0);
9        Vsb:out signed(25 downto 0);
10       Isa,Isb:out signed(15 downto 0));
11       end TC;
12
13 architecture descTC of TC is
14
15 signal Sa,Sb,Sc:signed(1 downto 0):="00";
16 constant V:unsigned(7 downto 0):="11111111";
17 signal E,I1,I2:signed(7 downto 0);
18 constant a:signed(7 downto 0):="00000101";
19 constant b:signed(7 downto 0):="00000100";
20 constant c:signed(7 downto 0):="00000011";
21 signal Ic: signed(7 downto 0);
22
23 begin
24
25 I1<=signed(Ia);
26 I2<=signed(Ib);
27 E<=signed(V);
28 Sa(0)<=S(0);
29 Sb(0)<=S(1);
30 Sc(0)<=S(2);
31
32 Vsa<=a*E*(Sa - c*(Sb + Sc));
33 Vsb<="00000000"&a*E*(Sb - Sc);
34
35 Ic<= -I1-I2;
36 Isa<=a*I1;
37 Isb<=b*(I2-Ic);
38
39 end descTC;
```

Figure III-3 : Description VHDL du bloc de traitement.

La figure III.4 montre le schéma RTL du bloc de traitement obtenu.

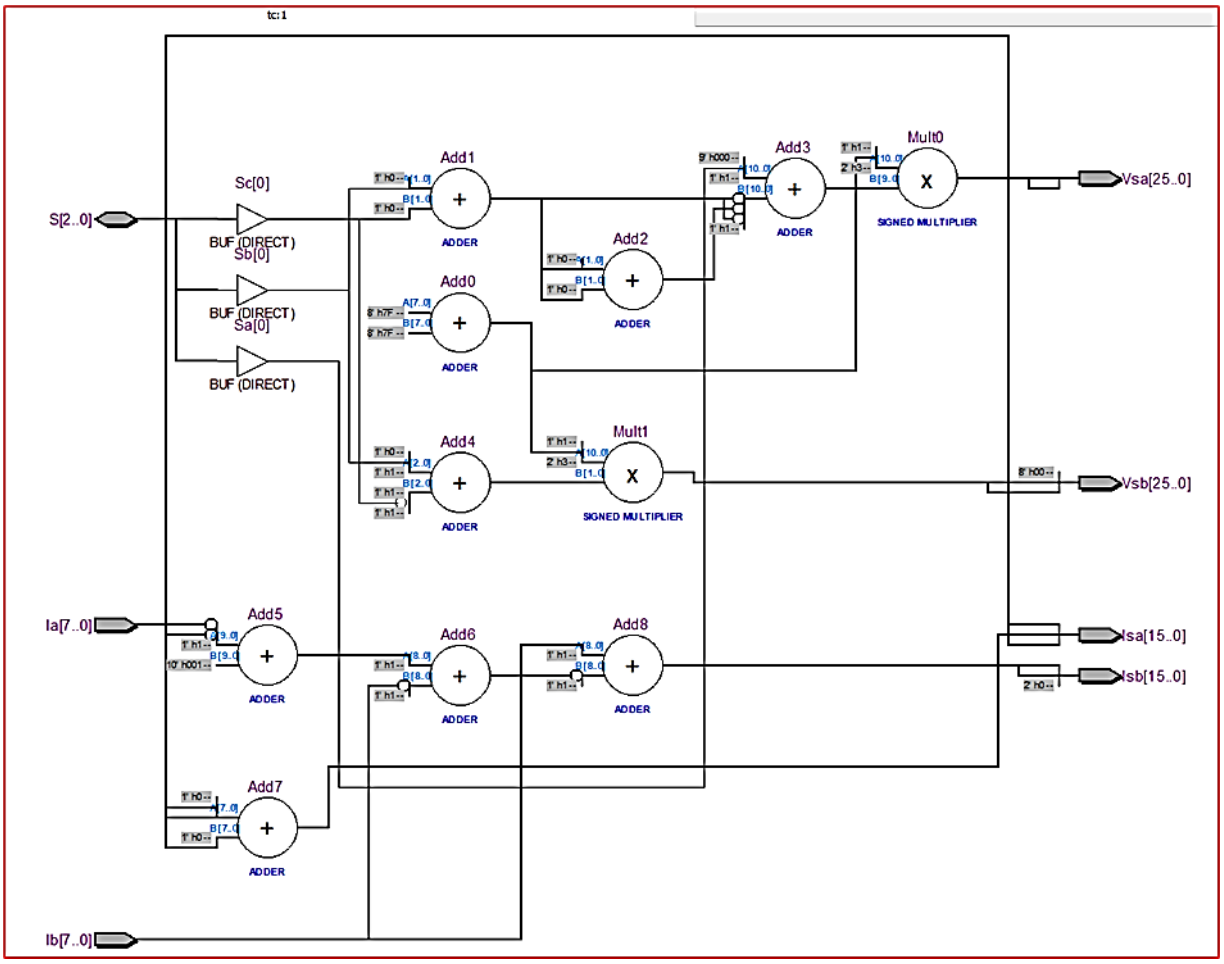

Figure III-4 : Schéma RTL du bloc de traitement.

Le tableau III.2 démontre les ressources du FPGA consommées par ce bloc (le traitement).

Tableau III-2 : Ressources utilisées sur le FPGA par l'algorithme de traitement.

Top levelentity : TC	
Ressources	**Utilisées**
Pins	103
Éléments logiques (LE)	46
Fonctions combinatoires	46
Registres	0
Multiplicateur intégré d'élément 9 bits	0

Le nombre de broches (Pin) consommé (103) représente 16 broches physiques (Ia, Ib) et les autres 87 pins sont des signaux internes issus du bloc précédant et orientés vers les autres blocs.

Notre carte Altera EP2C5T144C8 se dispose de 144 broches parmi lesquelles il n'y a que 89 pins qui peuvent être utilisées comme entrées/sorties (E/S). Pour éviter toute sorte de conflit nous avons choisi la carte EP2C15F484C6 (484 pins) lors de l'implémentation de chaque algorithme indépendamment.

III.4.3. Bloc d'estimation

Ce bloc regroupe les sous algorithmes spécifiques liés à l'estimation de grandeurs impliquées dans le processus de commande (le couple électromagnétique ***Ce*** et le flux statorique **ϕs**). La description VHDL utilisée comporte deux parties, un flot de données et l'autre est structurelle.

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity Esti is
6  Port (Vsa:in signed(25 downto 0);
7        Vsb:in signed(25 downto 0);
8        Isa,Isb:in signed(15 downto 0);
9        Qs:buffer signed(33 downto 0);
10       Ce:out signed(65 downto 0);
11       cos,sin:out signed(33 downto 0));
12       end Esti;
13
14 architecture descesti of esti is
15
16       constant Rs:signed(7 downto 0):="00000110";
17
18       component sqrt is
19       port (data_in:in signed(67 downto 0);
20             data_out:out signed(33 downto 0));
21             end component;
22       for unit: sqrt use entity work.sqrt(behaviour);
23
24       constant d:signed(7 downto 0):="00000110";
25       signal Qsa,Qsb:signed(33 downto 0);
26       signal Q:signed(67 downto 0);
27       constant p:signed(7 downto 0):="00000011";
28
29       begin
30
31       Qsa<=Qsa + Vsa - Rs*Isa;
32       Qsb<=Qsb + Vsb - Rs*Isb;
33
34       Q<= Qsa*Qsa + Qsb*Qsb;
35       unit: sqrt port map(Q,Qs);
36
37       Ce<= d*p*(Qsa*Isb - Qsb*Isa);
38
39       cos<=Qsa/Qs;
40       sin<=Qsb/Qs;
41
42       end descesti;
```

Figure III-5 : Description VHDL du bloc d'estimation.

Pour modéliser la fonction mathématique racine carré, on a exploité l'algorithme de Sqrt (que nous avons présenté dans l'annexe).

Le schéma RTL (figure III.6) et le tableau de ressources consommées du bloc d'estimation donnés ci-dessous montrent le nombre important de pin consommés dans cet algorithme, cela est dû au quantité importante de multiplieurs utilisés, sachant que le multiplieur est plus consommé en ressources matérielles.

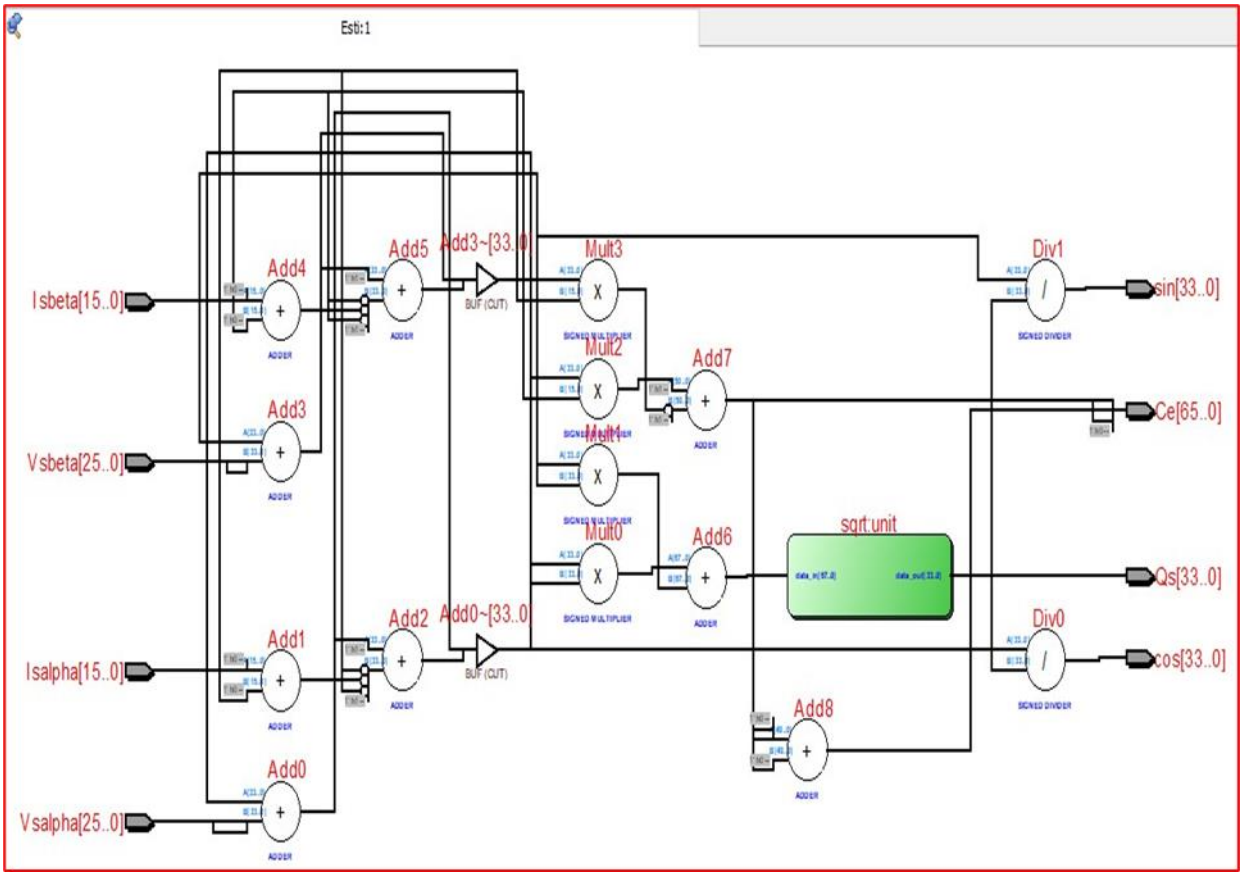

Figure III-6 : Schéma RTL du bloc d'estimation.

Tableau III-3 : Ressources utilisées par le FPGA dans le cas de l'algorithme d'estimation.

Top levelentity : ESTI	
Ressources	**Utilisées**
Pins	252
Éléments logiques (LE)	3622
Fonctions combinatoires	3622
Registres	0
Multiplicateur intégré d'élément 9 bits	24

III.4.4. Bloc de régulation (COMP)

C'est le module de calcul des lois de commande adoptées pour commander la machine asynchrone en fonction des grandeurs de consigne et des grandeurs mesurées et estimées, il regroupe les sous algorithmes spécifiques de correction des grandeurs de contrôle.

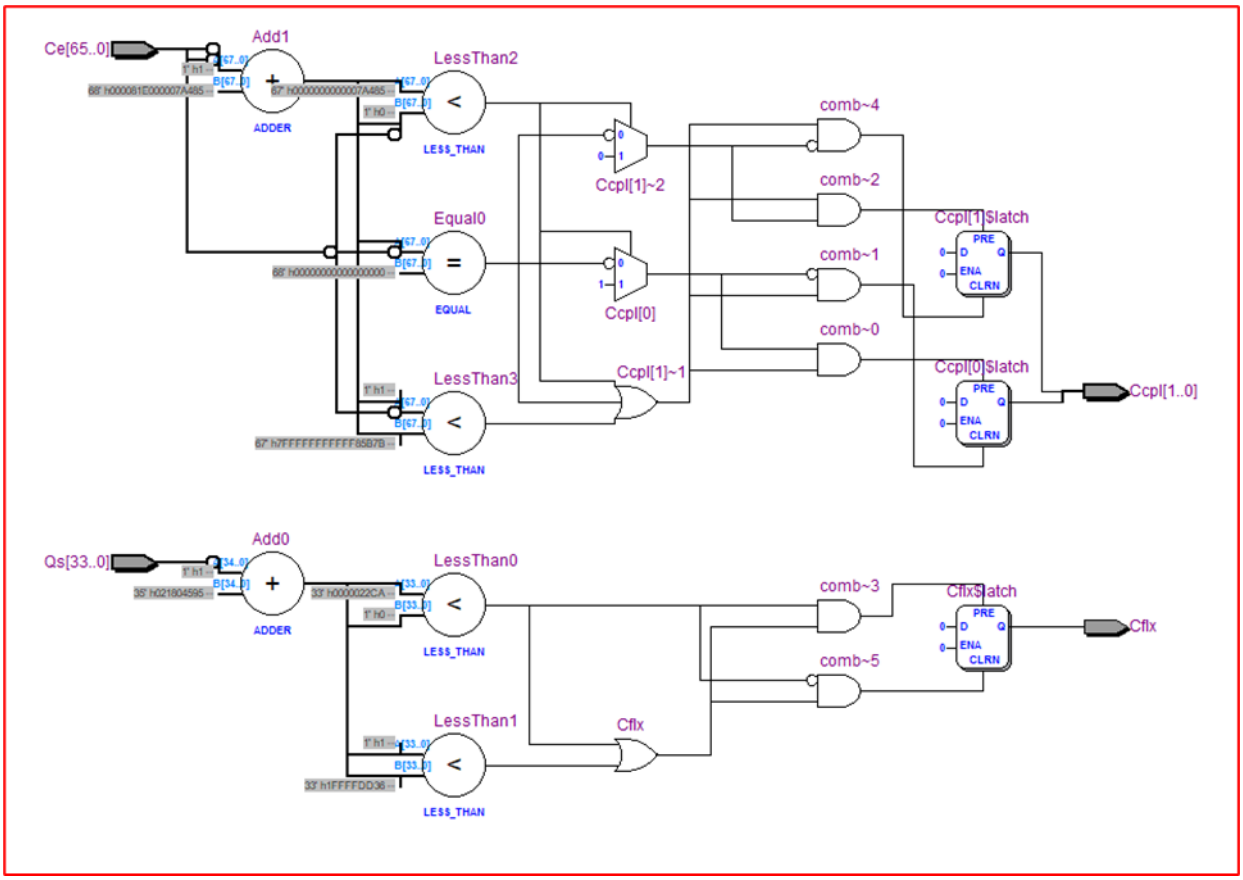

Figure III-7 : Schéma RTL du bloc de régulation

Le tableau III.4 présente les ressources consommées lors la conception de l'algorithme de régulation.

Tableau III-4 : Ressources utilisées sur le FPGA par l'algorithme de régulation

Top levelentity : COMP	
Ressources	**Utilisées**
Pins	103
Éléments logiques (LE)	180
Fonctions combinatoires	180
Registres	0
Multiplicateur intégré d'élément 9 bits	0

III.4.5. Bloc de commande (Table de commande TAKAHASHI)

La commande de la machine asynchrone est assurée à travers ce bloc qui génère les signaux de commande en fonction des résultats de comparateurs à hystérésis du bloc de régulation, la description utilisée est de type comportementale.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity cmd is
port (clk,reset:in std_logic;
      Ccpl:in signed(1 downto 0);
      Cflx:in std_logic;
      cos,sin:in signed(33 downto 0);
      s:inout std_logic_vector(2 downto 0)
      );
      end cmd;

architecture desccmd of cmd is

   constant x:signed(33 downto 0):="0110111000010100011110101110000101";
   constant y:signed(33 downto 0):="0011111111111111111111111111111111";
   constant z:signed(33 downto 0):="0111111111111111111111111111111111";

begin

   process(Cflx,Ccpl,cos,sin,clk,reset)
   begin
   if reset='1' then
   S<="000";
   else
   if clk 'event and clk='1' then
      if (Ccpl="01" and Cflx='1') then
         if (cos>=x and sin<=y and sin>-y) then
            s<="110";
         elsif (cos<x and cos>=0 and sin<=z and sin>y) then
            s<="010";
         elsif (cos<=0 and cos>=-x and sin<z and sin>y) then
            s<="011";
         elsif (cos<=-x and sin<=y and sin>-y) then
            s<="001";
         elsif (cos<=0 and cos>-x and sin<=-y and sin>-z) then
            s<="101";
         else s<="100";
         end if;

      elsif (Ccpl="00" and Cflx='1') then
         if (cos>=x and sin<=y and sin>-y) then
            s<="111";
         elsif (cos<x and cos>=0 and sin<=z and sin>y) then
            s<="000";
         elsif (cos<=0 and cos>=-x and sin<z and sin>y) then
            s<="111";
         elsif (cos<=-x and sin<=y and sin>-y) then
            s<="000";
         elsif (cos<=0 and cos>-x and sin<=-y and sin>-z) then
            s<="111";
         else s<="000";
         end if;

      elsif (Ccpl="11" and Cflx='1') then
         if (cos>=x and sin<=y and sin>-y) then
            s<="101";
         elsif (cos<x and cos>=0 and sin<=z and sin>y) then
            s<="100";
```

```
61          elsif (cos<=0 and cos>=-x and sin<z and sin>y) then
62             s<="110";
63          elsif (cos<=-x and sin<=y and sin>-y) then
64             s<="010";
65          elsif (cos<=0 and cos>-x and sin<=-y and sin>-z) then
66             s<="011";
67          else s<="001";
68          end if;
69
70       elsif (Ccpl="01" and Cflx='0') then
71          if (cos>=x and sin<=y and sin>-y) then
72             s<="010";
73          elsif (cos<x and cos>=0 and sin<=z and sin>y) then
74             s<="011";
75          elsif (cos<=0 and cos>=-x and sin<z and sin>y) then
76             s<="001";
77          elsif (cos<=-x and sin<=y and sin>-y) then
78             s<="101";
79          elsif (cos<=0 and cos>-x and sin<=-y and sin>-z) then
80             s<="100";
81          else s<="010";
82          end if;
83
84       elsif (Ccpl="11" and Cflx='0') then
85          if (cos>=x and sin<=y and sin>-y) then
86             s<="001";
87          elsif (cos<x and cos>=0 and sin<=z and sin>y) then
88             s<="101";
89          elsif (cos<=0 and cos>=-x and sin<z and sin>y) then
90             s<="100";
91          elsif (cos<=-x and sin<=y and sin>-y) then
92             s<="110";
93          elsif (cos<=0 and cos>-x and sin<=-y and sin>-z) then
94             s<="010";
95          else s<="011";
96          end if;
97
98       else
99          if (cos>=x and sin<=y and sin>-y) then
100            s<="000";
101         elsif (cos<x and cos>=0 and sin<=z and sin>y) then
102            s<="111";
103         elsif (cos<=0 and cos>=-x and sin<z and sin>y) then
104            s<="000";
105         elsif (cos<=-x and sin<=y and sin>-y) then
106            s<="111";
107         elsif (cos<=0 and cos>-x and sin<=-y and sin>-z) then
108            s<="000";
109         else s<="111";
110         end if;
111      end if;
112      else S<=S;
113      end if;
114      end if;
115   end process;
116end descomd;
```

Figure III-8 : Description VHDL du bloc de commande

Tableau III-5 : Ressources utilisées sur le FPGA par l'algorithme de commande.

Top levelentity : CMD	
Ressources	**Utilisées**
Pins	76
Éléments logiques (LE)	91
Fonctions combinatoires	91
Registres	3
Multiplicateur intégré d'élément 9 bits	0

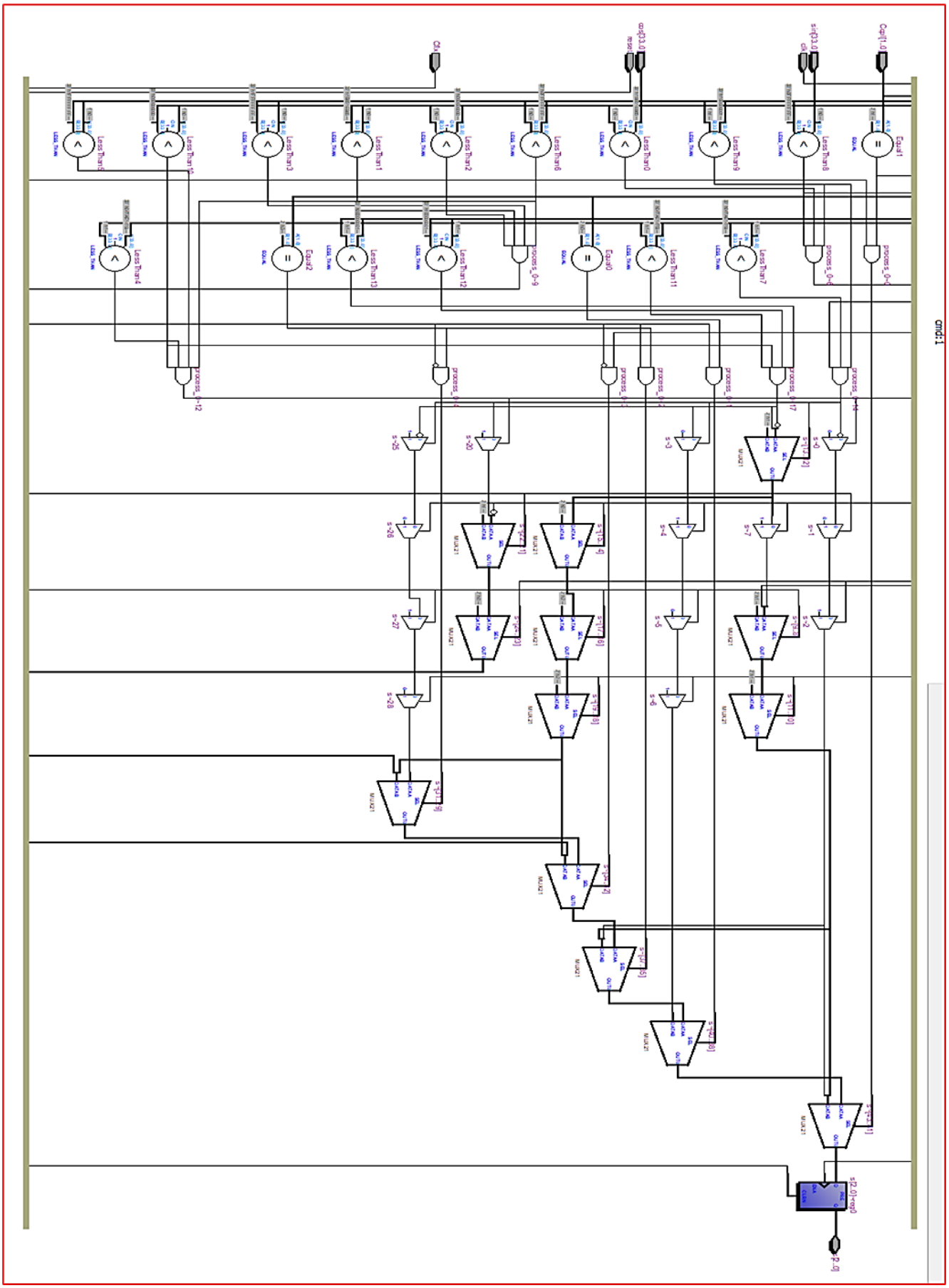

Figure III-9 : Schéma RTL du bloc de commande

III.4.6. Résultats de synthèse

La figure III-10 présente le résultat de synthèse de l'algorithme de la commande DTC à implémenter sur FPGA. Le fichier VHDL synthétisé dans cette figure est le Top Level, qui fait appel à tous les modules qui forment l'algorithme de la commande DTC.

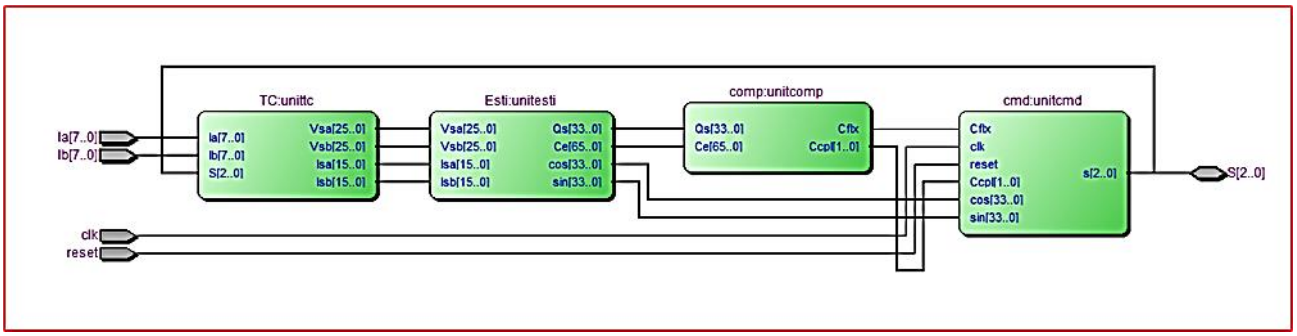

Figure III-10 : Résultat de synthèse de l'algorithme général de la commande DTC

Tableau III-6 : Ressources utilisées sur le FPGA par l'algorithme de la commande DTC

Flow Summary	
Flow Status	Successful - Thu Aug 04 16:08:48 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	dtc
Top-level Entity Name	dtc
Family	Cyclone II
Device	EP2C5T144C8
Timing Models	Final
Total logic elements	3,914 / 4,608 (85 %)
Total combinational functions	3,914 / 4,608 (85 %)
Dedicated logic registers	3 / 4,608 (< 1 %)
Total registers	3
Total pins	21 / 89 (24 %)
Total virtual pins	0
Total memory bits	0 / 119,808 (0 %)
Embedded Multiplier 9-bit elements	24 / 26 (92 %)
Total PLLs	0 / 2 (0 %)

A base du tableau III.6, on constate que la conception de l'algorithme général a consommé 85% des ressources de la carte en éléments logiques et 92% des ressources en multiplieurs 9bits éléments cela est dû au nombre important des opérations arithmétiques et des comparaisons.

III.4.7. Résultats de simulation

Dans cette partie, on présente les résultats de simulation relatifs à la commande DTC d'une MAS alimentée par un onduleur de tension triphasé. Le programme VHDL de simulation établi sous l'environnement QUARTUS II, nous a permis de capter fidèlement le chronogramme de l'évolution des signaux de sortie présentés dans la figure III.11. Les simulations sont effectuées en temps discret pour une période d'échantillonnage de 10 *ns* pendant un intervalle de 200 *ns*.

Le tableau III.7 donne les grandeurs nominales et les paramètres physiques d'une machine asynchrone d'un Laboratoire.

Tableau III-7 : Paramètres de la machine asynchrone.

Puissance nominale	**1.5 Kw**
Couple nominal	10 Nm
Vitesse nominale	1435 tr/min
Nombre de pair de pôles	2
Résistance du stator Rs	5.717 Ω
Résistance du rotor Rr	4.282 Ω
Inductance du stator Ls	0.464 H
Inductance du rotor Lr	0.464 H
Mutuelle inductance M	0.441 H
Moment d'inertie J	0.0049 kg.m

La simulation concerne le fonctionnement de la MAS en régime nominal, pour des consignes du flux et du couple égaux respectivement à 0.91 Wb et 10 Nm.

On a introduit un comportement de type ***reset*** pour les initialisations des variables et des signaux.

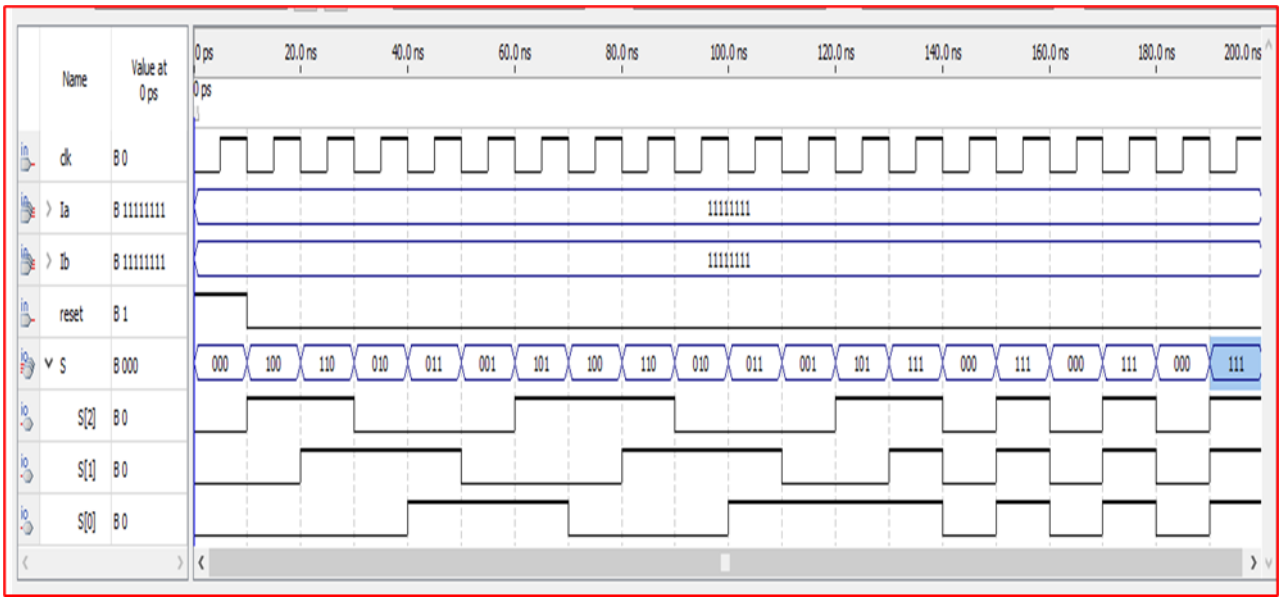

Figure III-11 : Chronogramme des signaux de sortie

A partir de ce chronogramme on peut déduire la table des vecteurs de commande correspondants aux signaux générés :

Tableau III-8 : Vecteurs de sortie générés pendant la simulation.

000	**100**	**110**	**010**	**011**	**001**	**101**	**100**	**110**	**010**	**011**	**001**	**101**	**111**	**000**	**111 000 111**		
V0	V1	V2	V3	V4	V5	V6	V1	V2	V3	V4	V5	V6	V7	V0	V7	V0	V7

On peut remarquer, un démarrage avec une valeur nulle (Sa Sb Sc = 000) imposer par le signal *reset* suivi par une augmentation de couple (la vitesse de rotation) et de flux statorique pour une durée de temps égale à 120 *ns* qui correspond aux dix premières périodes qui suivent le premier signal de sortie.

Une fois les valeurs consignes de couple et de flux atteintes (Ccpl=0, Cflx=0) le bloc de commande commence à générer les vecteurs nuls V0(000) et V(111) qui mènent à une décroissance du couple alors que le module de flux ϕ reste inchangé.

Cette simulation nous monte une bonne conformité des résultats avec le tableau de commande de TAKAHASHI.

III.4.8. Affectation des broches

La figure III.12 présente l'affectation des signaux d'entrée/sortie sur les différentes broches de la carte Altera EP2C5T144C8, on a réservé tout le *bank 1* pour les entrées. Les trois signaux bidirectionnels de sortie ont été placés sur les pins 40 ,41 et 42 du *bank 4*, l'entrée d'horloge est placée sur la broche N°73 du *bank*3.

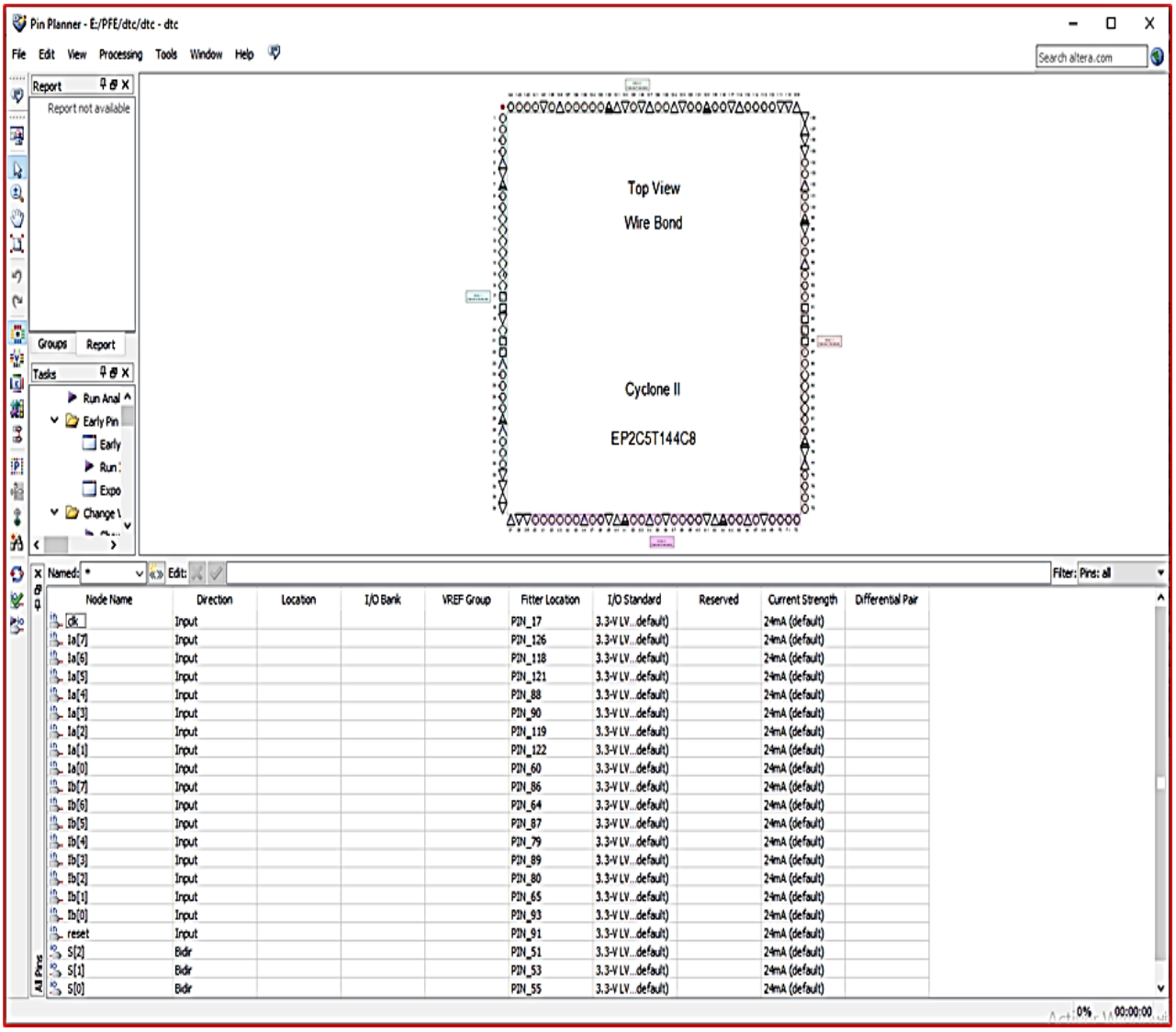

Figure III-12 : Affectation des broches.

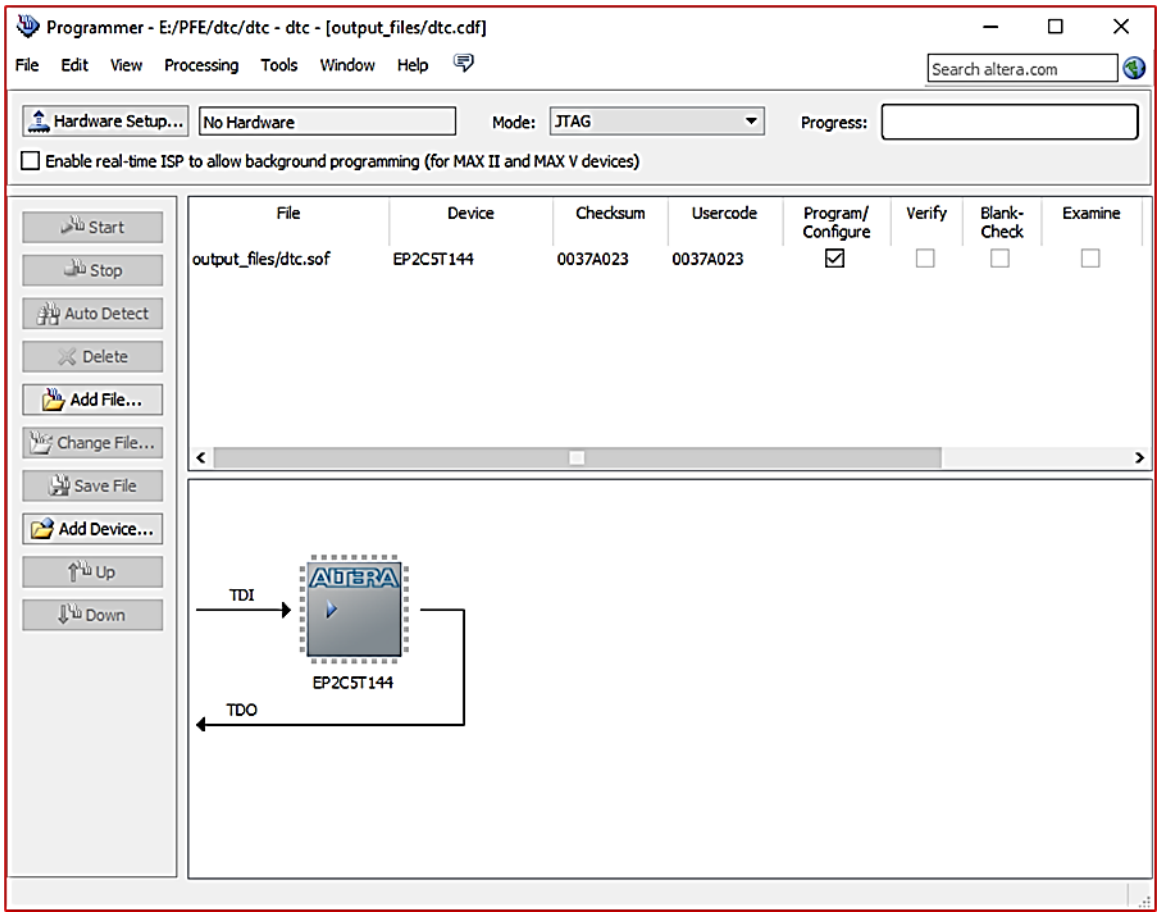

Figure III-13 : Implémentation de la commande sur carte.

III.5. Conclusion

Dans ce chapitre, nous avons présenté les caractéristiques des dispositifs Cyclone II et le logiciel QUARTUS II qu'on utilisé, ainsi que les résultats de l'implémentation et de la simulation de la commande DTC de la machine asynchrone sur une cible Altera EP2C5T144C8. Cela, en commençant par la conception séparée de chacune des quatre blocs formant cette commande : le traitement, l'estimation, la régulation et la commande.

Nous avons, ainsi programmé en VHDL la fonction de chaque bloc (séparément) de la technique DTC, puis nous avons élaboré une description principale en VHDL qui fait appel aux quatre descriptions élémentaires.

Les bons résultats obtenus montrent la fiabilité de l'implémentation de la technique DTC en considérant un FPGA - Altera.

Conclusion générale

Conclusion générale

Dans le cadre de ce projet, on a instauré une solution technologique dans la commande des machines asynchrones. Plus particulièrement, on a intégré une technique de l'intelligence artificielle basée sur un circuit FPGA dans l'élaboration de la commande directe du couple (DTC) de la machine asynchrone. Pour cet objectif, on a présenté une démarche méthodologique pour l'implémentation matérielle de cette commande sur une cible FPGA de type Altera. Les principaux volets de ce travail sont : on a commencé par présenter le modèle électrique en régime dynamique de la machine asynchrone. Ensuite, on a présenté la stratégie de commande DTC basée sur le principe de maintien des grandeurs du flux statorique et du couple électromagnétique à l'intérieur des bandes d'hystérésis de deux régulateurs avec l'architecture des circuits FPGA, ses interfaces d'entrées/ sorties et ses blocs logiques de base. Aussi, on a parlé de l'utilité et la structure du langage de description matérielle VHDL qui considère n'importe quel système comme cellule de base et déclare ses signaux d'entrées/sorties dans une partie appelée *ENTITY* puis décrit les différentes opérations effectuées sur ces signaux dans sa partie qui est appelée *ARCHITRCTURE.*

Nous avons effectué une description du circuit cible « Altera EP2C5T144C8 », suivie d'une présentation de l'environnement de conception QUARTUS II et la méthodologie pour concevoir, simuler, synthétiser, et enfin implémenter un algorithme issu d'une description VHDL. Aussi, on a présenté tout d'abord les différents blocs constituant la commande numérique d'une machine asynchrone : le bloc de mesure, le bloc de traitement et bloc de conversion de l'énergie.

Ensuite, nous avons décomposé notre algorithme général de commande DTC en quatre composantes élémentaires :

- Composante de traitement qui contient une description VHDL flot de données des opérations nécessaires pour réaliser la transformation CONCORDIA afin de mener les valeurs statoriques de la MAS du repère triphasé à un repère diphasé.
- Composante d'estimation contenant une description VHDL mixte pour le calcul de flux statorique et de couple électromagnétique.

- Composante de régulation conçue avec une description VHDL comportementale qui décrit les deux comparateurs à hystérésis.
- Composante de commande qui utilise une description VHDL comportementale pour générer les signaux de commande en fonction des erreurs issues de comparaison.

Enfin, ces quatre composantes (ou modules) ont été réunis dans une seule description VHDL structurelle qui a été synthétisée, simulée puis implémentée dans le circuit cible Altera EP2C5T144C8. La simulation de la description VHDL développé, nous a montré l'apport de l'intelligence artificielle dans la Commande DTC de la MAS en termes de réduction de temps d'exécution en adoptant un traitement parallèle et de prototypage rapide de la commande numérique sur FPGA.

Les travaux présentés dans ce mémoire ouvrent un certain nombre de perspectives dans des voies différentes : A court terme on peut :

- Implémenter le mécanisme de création du temps mort sur FPGA. De cette manière la carte de commande FPGA génère directement les trois signaux de commande Sa, Sb et Sc, et leurs compléments avec un temps mort entre les signaux d'un même bras de l'onduleur.
- Intégrer la commande Neuro-Floue dans la commande DTC de la machine asynchrone, en profitant des avantages de la logique floue et les avantages des réseaux de neurones.
- Aborder la thématique de recherche afférente au domaine touchant le pilotage des véhicules électriques.

Références bibliographies

[1] A. Rahal, M.F Edjiri, « Contrôle direct du couple de la machine asynchrone », Mémoire d'ingéniere, université Msila, 2004.

[2] I. Takahashi, S. Asakawa, « Ultra-Wide Speed Control of Induction Motor Covered 10A6 range», IEEE Trans. Ind. Applicat., IA-25: 227-232, 1987.

[3] O. Bouchelegheme, S. Sahraoui (2020), « Implémentation de la technique de commande DTC intelligente sur FPGA-ALTERA », Mémoire d'ingéniere Université M'HAMED BOUGARA-BOUMERDES

[4] H. S. Allah, (2016). " Commande par retour d'état linéarisation d'une machine asynchrone avec et sans défaut (Doctoral dissertation, Université de Mohamed boudiaf M'sila).

[5] R. Abdelli, « Perturbations singulières appliquées au modèle de machine asynchrone avec défauts au stator et au rotor », mémoire de Magister, Ecole nationale polytechnique d'Alger, 2007.

[6] O. Ondel, «Diagnostic par reconnaissance des formes: application à un ensemble convertisseur-machine asynchrone», thèse de Doctorat, Ecole centrale de Lyon, 2006.

[7] Wildi, T. (2005). Électrotechnique (4e édition). Presses de l'université Laval

[8] M. Nacera. « Découverte Génie Electrique», 2016.

[9] J. Lessenne, F. Notelet, et al., « Introduction à l'électrotechnique approfondie », Editions Techniques & Documentations, Paris, 1981.

[10] C. C. d. Wit « Modélisation contrôle vectoriel et DTC » tome 1, Editions Sciences Hermes, 2000.

[11] J.-W. Jung, "Space Vector PWM Inverter" The Ohio State University, Feb. 2005.

[12] H. Boubacar, « Prototypage rapide à base de FPGA d'un algorithme de contrôle avancé pour le moteur à induction. Diss. Université du Québec à Trois-Rivières, 2010.

[13] Assem, O. O., & Karim, A. (2013). Commande d'un moteur asynchrone destiné à l'entrainement d'une électropompe (Doctoral dissertation, Université Mouloud Mammeri).

[14] T. Bakhti et S. Bendaas, « commande par DTC d'un machine asynchrone sans capteur de vitesse en utilisant un observateur adaptatif », mémoire d'ingénierie, université de Batna, 2008.

[15] F. Bensmaine, N. Ameghchouche, « commande par DTC d'un Moteur à induction sans capteur de vitesse en utilisant un observateur adaptatif » , mémoire d'ingénierie , université de Batna 2010.

[16] B. Messaoudi, « Utilisation du contrôle direct du flux statorique et du filtre de kalman en vue du contrôle direct du couple d'un moteur asynchrone», mémoire de Magister, université de Biskra, 2007.

[17] M. Abdelkebir, N. Merzoug, « contrôle direct de couple d'une machine a induction », mémoire d'ingénieur, université de Msila, 2005.

[18] Khitas, M. (2018). *Implantation d'application de traitement de signal sur système mono-puce reconfigurable SoPc* (Doctoral dissertation).

[19] http://www.xilinx.com/(date de consultation : 03/07/2022)

[20]http://comelec.enst.fr/hdl.html(date de consultation : 16/06/2022)

[21] G. Asch, « Les capteurs en instrumentation industrielle », 6eme Edition. PARIS: DUNOD, 2006.

[22] Brown, S. (2010). Fundamentals of digital logic design with VHDL.

[23]www.altera.com(date de consultation : 20/06/2022)

[24] Brahim, F. A. D. L. I., & Choayb, S. A. H. N. O. U. N. E. (2019). Implémentation des stratégies de commande sur FPGA pour le contrôle de l'ensemble MAS convertisseurs (Doctoral dissertation, UNIVERSITE MOHAMED BOUDIAF-M'SILA).

[25] B. Zeidman, "DesigningwithFPGAs and CPLDs", CMP Books, 2002.

[26] B. Smaani. 2019. « Electronique numérique avancée : FPGA et VHDL. Support de cours », université de Boumerdes, Algérie, 70 p.

Références bibliographies

[27] L. Dutrieux, « Logique Programmable » Eyrolles, 1997.

[28] Kafig, W. (2011). VHDL 101: Everything you need to know to get started. Elsevier.

[29] Nachef, T. (2011). Implémentation d'une instrumentation sur un FPGA (Doctoral dissertation, Université Mouloud Mammeri).

[30] Altera Corporation February 2008.Cyclone II Device Family Data Sheet. Distribued by JAMECO Electronics.

[31] denis.rabaste.free.fr(date de consultation : 06/08/2022)

[32] Djamel, H. S. (2013). Implémentation d'une application de tracking sur FPGA (Doctoral dissertation, Université Mouloud Mammeri).

Résumé

Nous avons implémenté la technique de commande directe du couple (DTC) d'une machine asynchrone (MAS) sur un FPGA de la famille Altera. Ceci, en utilisant le langage de description matérielle VHDL sous l'outil Quartus II développé par Altera. Nous avons donc élaboré un programme en VHDL décrivant les quatre blocs des principales fonctions dans la commande numérique DTC : la transformation Concordia, l'estimation des grandeurs de commande, la régulation en utilisant des comparateurs à hystérésis et la génération de table de commande (Takahachi). Les résultats obtenus montrent la fiabilité de l'implémentation de la technique de commande DTC en considérant le FPGA EP2C5T144C8 - Altera.

Mots clés : technique de commande directe du couple (DTC), transformation Concordia, comparateurs à hystérésis, FPGA EP2C5T144C8, VHDL, Quartus II.

ملخص

لقد طبقنا تقنية التحكم المباشر في عزم الدوران (DTC) لآلات غير متزامنة (MAS) على FPGA من عائلة Altera. هذا، باستخدام لغة وصف الأجهزة VHDL ضمن أداة Quartus II التي طورتها Altera. لذلك قمنا بتطوير برنامج في VHDL يصف الكتل الأربعة للوظائف الرئيسية في التحكم الرقمي DTC: تحويل كونكورديا ، وتقدير عناصر التحكم ، والتنظيم باستخدام مقارنات التباطؤ وإنشاء جدول التحكم) تاكاهاتشي(. تظهر النتائج التي تم الحصول عليها موثوقية تنفيذ تقنية التحكم DTC مع مراعاة FPGA Altera
.EP2C5T144C8

الكلمات الأساسية: تقنية التحكم المباشر في عزم الدوران (FPGA)، تحويل كونكورديا، مقارنات التباطؤ، DTC
EP2C5T144C8 ، Quartus II ،VHDL

Abstract

We have implemented the direct torque control technical (DTC) of asynchronous machines (MAS) on an FPGA of the Altera family. This, using the hardware description language VHDL under Quartus II tool developed by Altera. So we have developed a program in VHDL describing the four blocks of the main functions in the digital control DTC: the Concordia transformation, the estimation of the ordre quantity, regulation using hysteresis comparators and the generation of command tables (Takahachi). The results obtained show the reliability of the implementation of the control technical DTC considering the FPGA Altera EP2C5T144C8.

Key words: direct torque control (DTC) technique, Concordia transformation, hysteresis comparators, FPGA EP2C5T144C8, VHDL, Quartus II.