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

المدرسة الوطنية العليا المتعددة التقنيات
Ecole Nationale Supérieure Polytechnique

Département d'Electronique
Laboratoire des Dispositifs de Communication et de Conversion Photovoltaïque
(LDCCP)

Mémoire de Magister en Electronique
Option : Electricité Solaire

Présenté par :

CHEKIRED Fathya
Ingénieur d'Etat en Electronique, Université de Jijel

Thème

Etude et implémentation d'une commande MPPT neuro-floue sur FPGA

Membres du Jury

M^{me}. M. GUERTI, Professeur, ENSP	Présidente
M. C. LARBES, Maître de Conférences, ENSP	Rapporteur
M. M. HADDADI, Professeur, ENSP	Examinateur
M. M. S.AIT-CHEIKH, Maître de conférences, ENSP	Examinateur
M. A. MALEK, Maître de Recherches, CDER	Examinateur

-2008-

Ecole Nationale Supérieure Polytechnique 10, Avenue Hassen Badi, El-Harrach, ALGER

ملخص

تهدف دراستنا هذه إلى التحكم في تتبع نقطة الاستطاعة العظمى« MPPT » للألواح الكهروضوئية ، اهتمامنا بوجه خصوصا نحو طريقة تحكم جديدة من بين طرق تحكم الذكاء الاصطناعي والتي تعرف بتقنية شبكة العصبونات بالمنطق الغامض و التي يتم استعمالها لاستغلال أحسن للطاقة المستمدة من الألواح الشمسية.

هذا العمل يهدف إلى دراسة مفصلة لمختلف مراحل تطبيق برمجة تقنية التحكم "شبكة العصبونات بالمنطق الغامض" على الدارات المبرمجة « FPGA » و ذلك باستخدام بطاقة التطوير « Memec design *Virtex-II V2MB1000* » لتتبع نقطة الاستطاعة العظمى و من خلال هذا يبين المزايا التي تقدمها هذه الدارات بفضل اختصار وقت برمجتها و مرونة استعمالها .

كلمات مفتاحية : كهروضوئي, تتبع نقطة الاستطاعة العظمى, شبكة العصبونات الاصطناعية, المنطق الغامض, شبكة العصبونات بالمنطق الغامض, المراقب العصبوني الغامض, دارة« FPGA » ,لغة البرمجة «VHDL »

Résumé

Ce travail s'intéresse à la commande de poursuite du point de puissance maximale (MPPT) des générateurs photovoltaïques. Notre intérêt s'est particulièrement dirigé vers une commande d'actualité qui fait partie de l'intelligence artificielle. C'est la commande par l'approche neuro-floue qui est définie comme étant un réseau neuronal multicouche avec des paramètres flous, ou comme un système flou mis en application sous une forme distribuée parallèle. Le contrôleur neuro-flou sert à exploiter au mieux la puissance délivrée par les panneaux solaires.

Ce mémoire présente le développement d'une telle commande et son implémentation sur un circuit " FPGA Virtex II de Xilinx " en se basant sur les différentes étapes nécessaires à la programmation de ce circuit et en utilisant la carte de développement « Memec Design *Virtex-II V2MB1000* ». On démontre ainsi l'avantage apporté par les circuits FPGA grâce à leur faible temps de développement, leur puissance de calcul et leur flexibilité de fonctionnement.

Mots clés : Photovoltaïque, MPPT, Réseaux de neurones, Logique floue, Réseaux neuro-flous, Contrôleur neuro-flou, Circuit FPGA, Langage VHDL.

Abstract

In this work, we present the principle of maximum power point tracking (MPPT) control of photovoltaic generators. However, our interest was particularly directed to a new kind of control which is a part of the artificial intelligence. This is the neuro-fuzzy approach, which is defined as a multilayer neural network with fuzzy parameters, or as a fuzzy system implemented, in a distributed parallel form. The neuro-fuzzy controller is used to exploit effectively the power delivered by solar panels.

This thesis presents the design and the implementation of a such controller on a "Xilinx's FPGA Virtex II" circuit, relying on the various steps needed to program this circuit using the development card "Memec Design *Virtex-IIV2MB1000*" .It demonstrates the advantages provided by the FPGA circuits due to their low development time and flexibility of operation.

Key words: Photovoltaic, MPPT, Artificial neural network, Fuzzy logic, Neuro-fuzzy, Neuro-fuzzy controller, FPGA circuit, VHDL language.

sommaire

Chapitre III: Techniques de poursuite du point de puissance maximale

Chapitre IV: Les circuits logiques programmables FPGA

Chapitre V: Application, résultats et discussion

Liste des figures

Introduction générale

L'électricité solaire est vue comme étant une importante énergie renouvelable qui peut être une alternative aux autres sources classiques d'énergie afin de satisfaire les larges besoins d'énergie dans le futur. Cette énergie trouve tout son avantage dans des applications de petite et moyenne consommation dans des régions isolées et loin des lignes de distribution électrique. Elle trouve aussi ses applications dans le domaine de l'espace (satellites, sondes…).

L'électricité solaire est en train de s'imposer depuis que les panneaux solaires sont devenus très disponibles et ont un rendement acceptable. En parallèle, la technologie des composants semi-conducteurs de grande puissance a nettement évolué par l'introduction de composants de puissance très performants du point de vue rendement et puissance de fonctionnement.

Une caractéristique importante des panneaux solaires est que la puissance maximale disponible est fournie seulement en un seul point de fonctionnement donné, localisé par une tension et un courant connus, appelé point de puissance maximale, en anglais Maximum Power Point (MPP). Autre problème est que la position de ce point n'est pas fixe mais elle se déplace en fonction de l'ensoleillement et de la température des cellules solaires ainsi que de la charge utilisée. A cause du coût relativement onéreux de ce genre d'énergie on doit extraire le maximum de watts des panneaux solaires. Cela nécessite un mécanisme de poursuite (*Tracking*) de l'MPP appelé 'maximum power point tracking' (*MPPT*) afin que la puissance maximale soit générée en permanence.

Dans ce mémoire, nous nous intéressons au problème de poursuite du point de puissance maximale d'un générateur photovoltaïque suivant deux méthodes: la méthode floue et la méthode neuro-floue.

Le but essentiel de notre travail est l'étude et l'implémentation d'un contrôleur MPPT par la méthode neuro-floue, en d'autre terme la méthode des réseaux de neurones adaptatifs par la logique floue, sur un circuit FPGA. Le présent mémoire est subdivisé en cinq chapitres à savoir:

Dans le premier chapitre, nous présentons les principales caractéristiques d'un générateur photovoltaïque (GVP) ainsi que la problématique de l'énergie solaire photovoltaïque.

Le deuxième chapitre présente une étude théorique sur les notions de la logique floue, les réseaux de neurones et neuro-flous ainsi que les concepts nécessaires à leurs conceptions, l'apprentissage et la validation.

Le troisième chapitre sera consacré aux principaux algorithmes de recherche du point de puissance maximale (MPPT) d'un GPV développés et les plus répandus, en particulier ceux émergents (les techniques intelligentes telle que la logique floue, les réseaux de neurones, les réseaux neuro-flous).

Les circuits logiques programmables « FPGA », particulièrement les circuits FPGA de la famille Virtex-II de Xilinx, sur lesquels on va implémenter notre application, sont le sujet du quatrième chapitre, on va s'intéresser à l'étude de ces circuits et leur principe de fonctionnement.

Le dernier chapitre sera consacré à l'implémentation du contrôleur MPPT neuro-flou sur la carte FPGA en commençant par la programmation de l'application jusqu'à son implémentation sur la carte de développement FPGA.

Nous terminerons notre mémoire par une conclusion générale dans laquelle nous donnerons les perspectives et les améliorations qui peuvent être prises en compte pour la continuité de ce travail.

Concepts de base des systèmes photovoltaïques

I.1. Introduction

Un système photovoltaïque est un système d'alimentation électrique, constitué principalement d'un générateur photovoltaïque composé d'un seul ou plusieurs modules solaires, d'un ensemble de batteries pour le stockage d'énergie électrique, d'un ou de plusieurs convertisseurs continu-continu pour fournir les tensions d'alimentation adéquates pour les batteries et les charges continues et éventuellement un convertisseur continu-alternatif pour l'alimentation des appareils à courant alternatif (figure I.1).

Ce chapitre décrit les concepts de base des systèmes photovoltaïques et de production d'électricité grâce à l'effet photovoltaïque.

I.2. Le générateur Photovoltaïque

Le générateur photovoltaïque est un ensemble d'équipements mis en place pour exploiter l'énergie photovoltaïque afin de satisfaire les besoins en charge. En fonction de la puissance désirée, les modules peuvent être assemblés en panneaux pour constituer un "champ photovoltaïque". Relié au récepteur sans autre élément, le panneau solaire fonctionne "au fil du soleil", c'est-à-dire que la puissance électrique fournie au récepteur est fonction de la puissance d'ensoleillement. Elle est donc à son maximum lorsque le soleil est au zénith et nulle la nuit.

Mais, très souvent, les besoins en électricité ne correspondent pas aux heures d'ensoleillement et nécessitent une intensité régulière (éclairage ou alimentation de réfrigérateurs, par exemple). On équipe alors le système de batteries d'accumulateurs qui permettent de stocker l'électricité et de la restituer en temps voulu.

Un régulateur est alors indispensable pour protéger les batteries contre les surcharges ou les décharges profondes nocives à sa durée de vie.

Pour un certain nombre d'applications, le courant continu produit, par le générateur photovoltaïque, est convertit à l'aide d'un onduleur en courant alternatif.

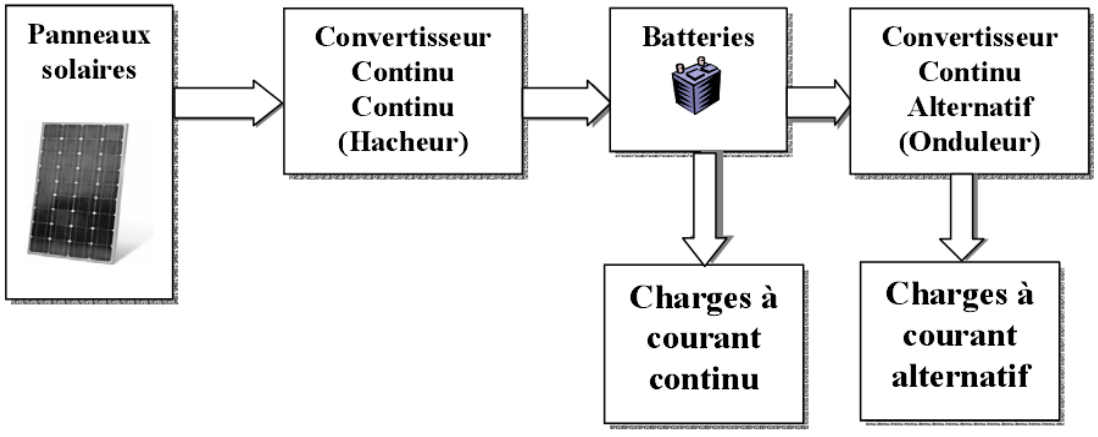


Figure I.1: Composantes principales d'une chaîne photovoltaïque.

I.2.1. L'effet photovoltaïque

Lorsqu'un matériau est exposé à la lumière du soleil, les atomes exposés au rayonnement sont " bombardés " par les photons constituants la lumière; sous l'action de ce bombardement, les électrons des couches électroniques supérieures (appelés électrons des couches de valence) ont tendance à être " arrachés ":

Si l'électron revient à son état initial, l'agitation de l'électron se traduit par un échauffement du matériau. L'énergie cinétique du photon est transformée en énergie thermique.

Par contre, dans les cellules photovoltaïques, une partie des électrons ne revient pas à son état initial. Les électrons " arrachés " créent une tension électrique continue faible. Une partie de l'énergie cinétique des photons est ainsi directement transformée en énergie électrique: c'est l'effet photovoltaïque.

L'effet photovoltaïque constitue la conversion directe de l'énergie du rayonnement solaire en énergie électrique au moyen de cellules généralement à base de silicium. Pour obtenir une puissance suffisante, les cellules sont reliées entre elles et constituent le module solaire.

L'effet photovoltaïque, c'est-à-dire la production d'électricité directement de la lumière, fut observée la première fois, en 1839, par le physicien français Edmond Becquerel. Toute fois, ce n'est qu'au cours des années 1950 que les chercheurs de la compagnie Bell Telephone, aux Etats-Unis, parvinrent à fabriquer la première photopile, l'élément primaire d'un système photovoltaïque.

I.2.2. La cellule photovoltaïque ou la photopile

Le fonctionnement de la photopile est basé sur les propriétés électroniques acquises par le silicium quand des atomes étrangers en petit nombre (des impuretés) sont substitués dans un réseau cristallin. Cette action est appelée dopage.

• Si l'atome d'impureté contient plus d'électrons que le silicium, le matériau contiendra des électrons libres en excès : il sera dit de type N (exemple: silicium dopé au phosphore).

• Si au contraire, l'atome d'impureté contient moins d'électrons que le silicium, le matériau sera déficitaire en électrons: il sera dit de type P (exemple: silicium dopé au bore).

La fabrication des cellules s'effectue à partir de lingots de silicium. Ces lingots sont découpés en fines couches de type P ou N en y diffusant du brome ou du phosphore. Une cellule solaire est alors obtenue en constituant une jonction de deux zones de type opposé (jonction PN). Au voisinage de la jonction apparaît un champ électrique qui maintient la

séparation des charges positives et négatives. Des contacts métalliques en formes de grille, contacts avant et arrière, sont déposés (Figure I.2).

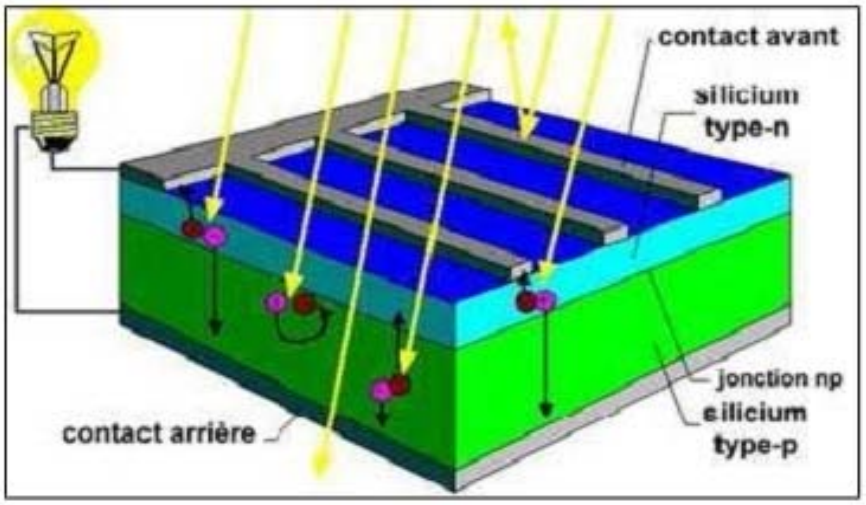


Figure I.2: Représentation schématique d'une pile solaire à jonction PN standard [1].

Une cellule photovoltaïque donc est un dispositif qui permet de transformer l'énergie solaire en énergie électrique. Cette transformation est basée sur les trois mécanismes suivants :

- Absorption des photons (dont l'énergie est supérieure au gap) par le matériau constituant le dispositif ;
- Conversion de l'énergie du photon en énergie électrique, ce qui correspond à la création des pairs électrons/trous dans le matériau semi-conducteur ;
- Collecte des particules générées dans le dispositif.

Le matériau constituant la cellule photovoltaïque doit donc posséder deux niveaux d'énergie et être assez conducteur pour permettre l'écoulement du courant: d'où l'intérêt des semi-conducteurs pour l'industrie photovoltaïque.

Afin de collecter les particules générées, un champ électrique permettant de dissocier les pairs électrons / trous créés est nécessaire. Pour cela on utilise le plus souvent une jonction p-n.

I.2.3. Caractéristiques électriques d'une cellule photovoltaïque :

Le circuit équivalent d'une cellule photovoltaïque peut être schématisé comme suit [2], [3]:

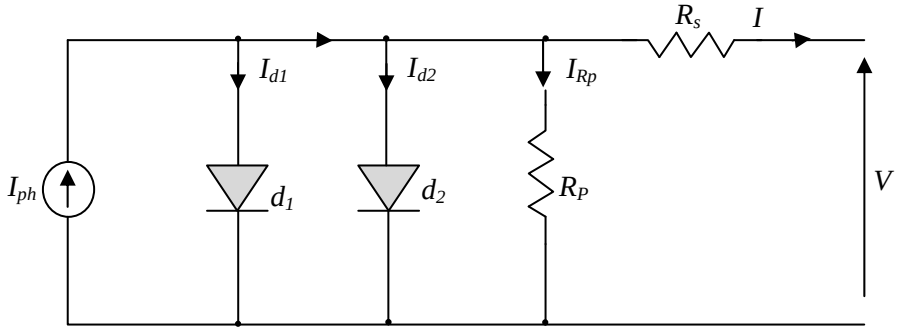


Figure I.3: Circuit équivalent d'une cellule solaire.

Comme le montre la figure I.3 une cellule solaire comporte en réalité une résistance série *Rs* et une résistance en dérivation ou shunt *Rp*. Ces résistances ont une certaine influence sur la caractéristique ***I =f (V)*** de la cellule:

- La résistance série est la résistance interne de la cellule ; elle dépend principalement de la résistance du semi-conducteur utilisé, de la résistance de contact des grilles collectrices et de la résistivité de ces grilles;
- La résistance shunt est due à un courant de fuite au niveau de la jonction; elle dépend de la façon dont celle-ci a été réalisée.

D'après la figure I.3 le modèle mathématique pour la caractéristique courant-tension est donné par [4]:

$$I = I_{ph} - I_{d1}\left[e^{\frac{q(V+I.R_s)}{n_1 kT}} - 1\right] - I_{d2}\left[e^{\frac{q(V+I.R_s)}{n_2 kT}} - 1\right] - \frac{V + IR_s}{R_P} \qquad \text{(I.1)}$$

I_{d1} et I_{d2} sont les courants de saturation des diodes; n_1 et n_2 les facteurs de pureté de la diode ; R_s et R_p sont respectivement la résistance série et la résistance parallèle, et *T* est la température absolue en Kelvin. L'équation contient également la charge élémentaire constante $q\,(1{,}602\cdot 10^{-19}.C)$ et la constante de Boltzmann k $(1{,}380\cdot 10^{-23}\ J/K)$. Le photo courant $I_{ph.\max}$ est atteint à une insolation maximum, souvent nous avons ($I_{ph} = S \cdot I_{ph.\max}$) avec S : pourcentage d'insolation.

Il est évident, de l'équation (I.1), que la caractéristique courant-tension dépend fortement de l'insolation et de la température. La dépendance, vis-à-vis de la température, est encore amplifiée par les propriétés du photo-courant I_{ph} et les courants de saturation inverse des diodes qui sont donnés par [3]:

$$I_{ph}(T) = I_{ph}\big|_{(T=298\cdot K)}\left[1 + (T - 298\cdot K)\cdot(5\cdot 10^{-4})\right], \qquad \text{(I.2)}$$

$$I_{d1} = K_1 T^3 e^{-\frac{E_g}{kT}}, \qquad \text{(I.3)}$$

$$I_{d2} = K_2 T^{\frac{5}{2}} e^{-\frac{E_g}{kT}}, \qquad \text{(I.4)}$$

$$K_1 = 1{,}2\ \ A/cm^2.K^3 \qquad \text{(I.5)}$$

$$K_2 = 2{,}9\cdot 10^5\ A/cm^2.K^{5/2}. \qquad \text{(I.6)}$$

I.2.4. Modèle mathématique d'un module solaire

L'équation suivante montre le modèle mathématique d'un module photovoltaïque, avec N_r cellules photovoltaïques raccordées en série [3].

$$I = I_{ph} - I_{d1}\left[e^{\frac{q(V+IN_rR_s)}{N_r n_1 kT}} - 1\right] - I_{d2}\left[e^{\frac{q(V+INrR_s)}{N_r n_2 kT}} - 1\right] - \frac{V + IN_rR_s}{N_rR_p}. \quad \text{(I.7)}$$

Ces modules peuvent être alors arrangés en série ou en parallèle pour obtenir la tension et les valeurs de courant désirées pour le système.

I.3. Les performances du générateur photovoltaïque

Un module photovoltaïque est constitué d'un ensemble de cellules photovoltaïques élémentaires montées en série et/ou en parallèle afin d'obtenir des caractéristiques électriques désirées tels que: la puissance, le courant de court-circuit *Icc* ou la tension en circuit ouvert *Vco*. Un générateur photovoltaïque est constitué d'un ou plusieurs modules PV en série et / ou en parallèle pour obtenir une puissance, un *Icc* et un *Vco* désirés [5].

I.3.1. Caractéristique Courant-Tension

La figure I.4 représente la courbe ***I =f (V)*** d'un module photovoltaïque typique dans des conditions constantes d'irradiation et de température.

L' irradiation standard adoptée pour mesurer la réponse des modules photovoltaïques est une intensité rayonnante de 1000 W/m^2 et une température de 25^{0}C.

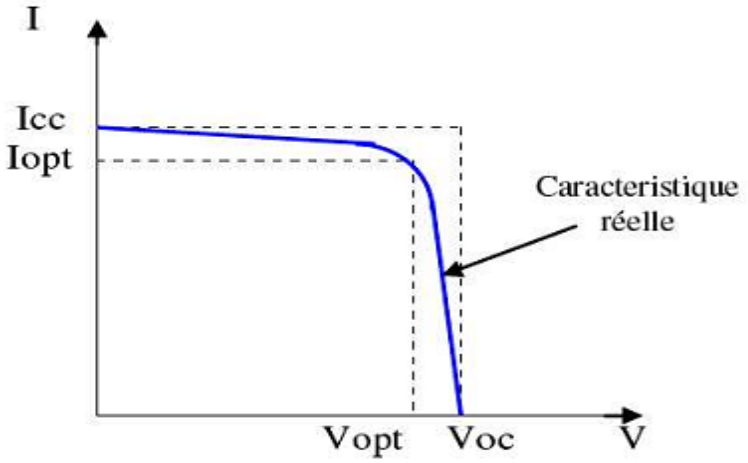


Figure I.4: Caractéristique I =f (V) d'un module photovoltaïque.

Il est difficile de donner un caractère source de courant ou de tension à un module photovoltaïque sur toute l'étendue de la caractéristique courant-tension. Par conséquent, le module photovoltaïque est considéré comme une source de puissance avec un point *Pm* où la puissance se trouve être maximale. Il est donc intéressant de se placer sur ce point pour tirer le maximum d'énergie et ainsi exploiter au mieux la puissance crête installée. Il est important de noter que certains régulateurs solaires réalisent une adaptation d'impédance afin qu' à chaque instant on se trouve proche de ce point *Pm*.

I.3.2. Influence de l'éclairement sur les courbes I(V), P(V)

La figure I.5 présente un exemple de courbes pour différents niveaux de rayonnement à une température constante.

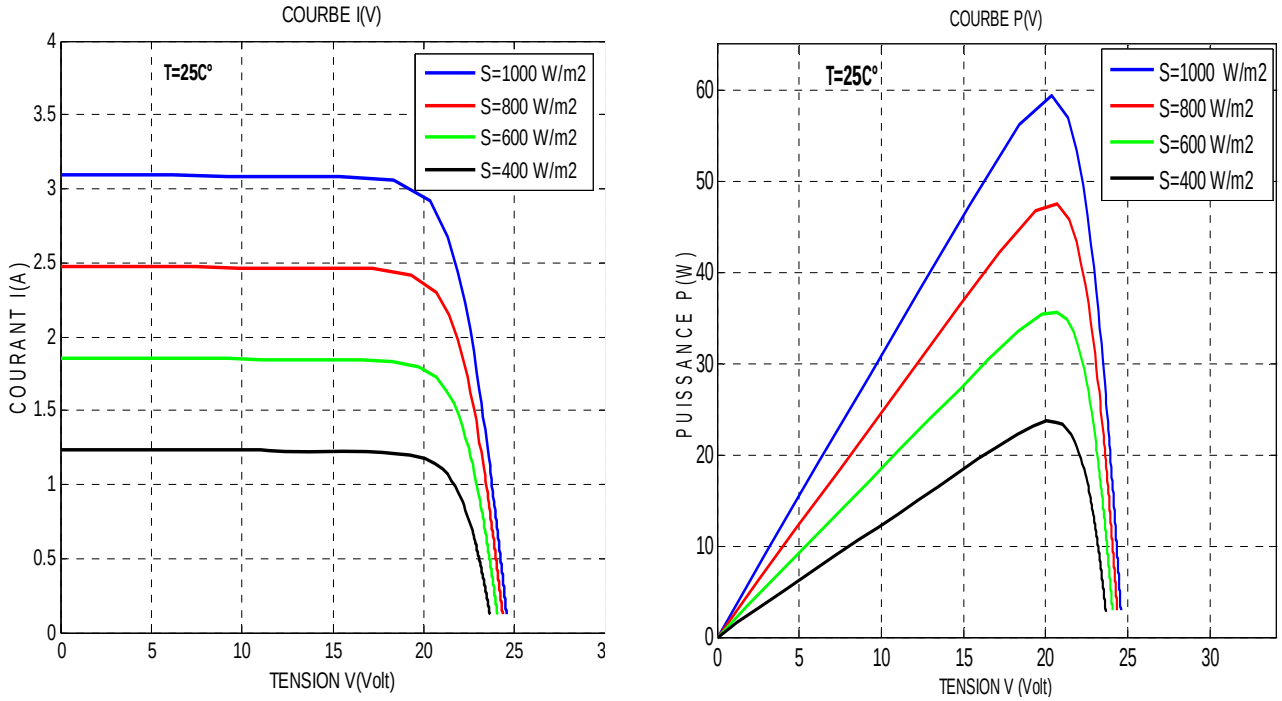


Figure I.5: Résultats de simulation des caractéristiques I(V) et P(V) d'un générateur PV en fonction de différentes irradiations à T=25°C.

Il est clair que la valeur du courant de court-circuit est directement proportionnelle à l'intensité du rayonnement. Par contre, la tension en circuit ouvert ne varie pas dans les mêmes proportions, mais reste quasiment identique même à faible éclairement.

Ceci implique donc que :

- La puissance optimale de la cellule (P_M) est pratiquement proportionnelle à l'éclairement;
- Les points de puissance maximale se situent à peu près à la même tension.

I.3.3. Influence de la température sur les courbes I(V), P(V)

La figure I.6 présente des courbes I(V) et P(V) pour différentes températures de fonctionnement de générateur photovoltaïque à une irradiation constante.

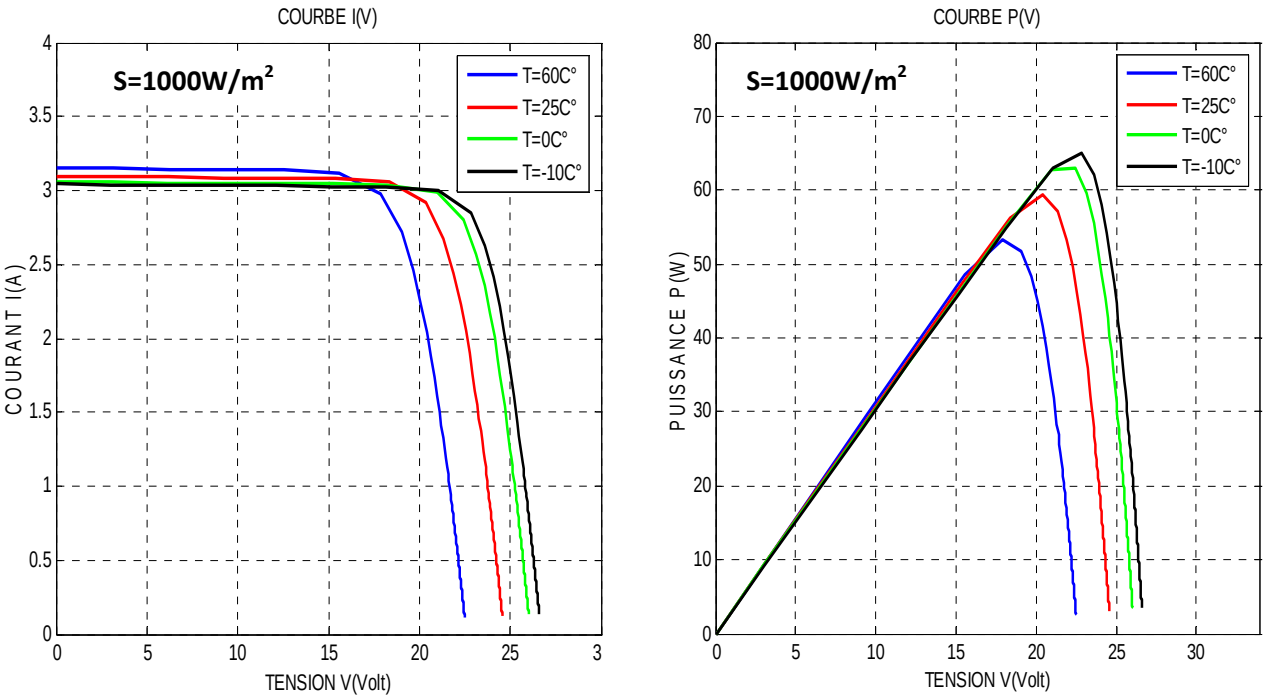


Figure I.6: Résultats de simulation des caractéristiques I(V) et P(V) d'un générateur PV en fonction de différentes températures à S=1000W/m^2.

Nous remarquons que la température a une influence négligeable sur la valeur du courant de court-circuit. Par contre, la tension en circuit ouvert baisse assez fortement lorsque la température augmente, par conséquent la puissance extractible diminue. Lors du dimensionnement d'une installation, la variation de la température du site sera impérativement prise en compte. Il est important de savoir que la puissance du module diminue environ de 0,5% par chaque degré d'augmentation de la température de la cellule au dessus de 25^{0}C [5].

I.4. Les Convertisseurs DC-DC (Les Hacheurs)

Les hacheurs présentent la partie essentielle dans le dispositif de commande d'un générateur photovoltaïque, ils sont des convertisseurs statiques continu-continu permettant de contrôler la puissance électrique dans les circuits fonctionnant en courant continu avec une très grande souplesse et un rendement élevé.

D'un point de vue circuit, le hacheur apparaît comme un quadripôle (figure I.7), jouant le rôle d'organe de liaison entre deux parties d'un réseau. On peut le considérer comme un transformateur de grandeurs électriques continues [6].

La figure (I.7) rappelle le schéma de principe d'un convertisseur DC-DC.

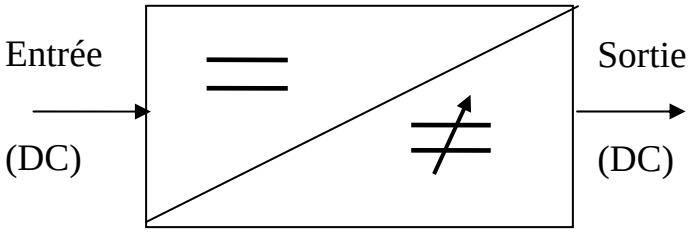


Figure I.7: Schéma de principe du hacheur.

Le hacheur se compose de condensateurs, d'inductances et de commutateurs. Dans le cas idéal, tous ces dispositifs ne consomment aucune puissance active, c'est la raison pour laquelle on a de bons rendements dans les hacheurs.

Le commutateur est un dispositif semi-conducteur en mode (bloqué -saturé), habituellement un transistor MOSFET.

Si le dispositif semi-conducteur est bloqué, son courant est zéro et par conséquent sa dissipation de puissance est nulle. Si le dispositif est dans l'état saturé, la chute de tension à ses bornes sera presque zéro et par conséquent la puissance perdue sera très petite [7].

I.4.1. Hacheur série (Buck)

La figure I.8 donne le schéma électrique d'un hacheur série. Il est composé d'un transistor MOSFET qui fonctionne en régime de commutation avec une période *Ts*. Dans la première fraction *DTs* le transistor est dans un état de saturation alors l'inductance *L* se charge d'énergie avec augmentation du courant I_L. Dans la deuxième fraction de temps *(1-D)Ts* l'inductance *L* libère cette énergie à la charge *Z* avec une diminution de courant I_L. Aussi le circuit peut être décomposé en deux circuits linéaires qui correspondent chacun à une position du transistor *S*.

Les figures I.9, I.10 donnent les schémas équivalents du hacheur dévolteur ''Buck'' dans les deux intervalles de temps.

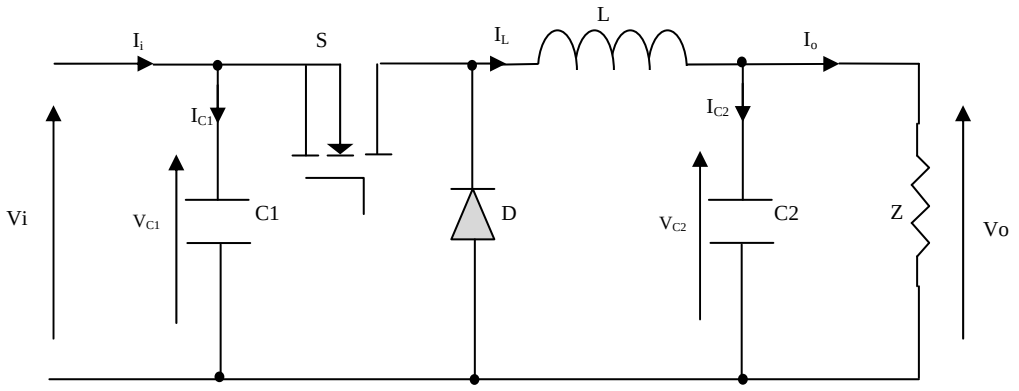


Figure I.8: Circuit électrique d'un hacheur série

I.4.1.1. Modèle mathématique équivalent

Pour obtenir le modèle mathématique du convertisseur, il faut l'étudier dans les deux phases de fonctionnement (*S* fermé, et *S* ouvert). Puis, faire un modèle « moyen » (*Averaged Model*) qui relie les différentes grandeurs moyennes d'entrée et de sortie du convertisseur.

Les variables dynamiques du circuit sont i_L, v_{C1}, v_{C2}, associés aux composants dynamiques L, C_1, C_2. Les équations qui relient les dérivées $\frac{di_L}{dt}$ et $\frac{dv_c}{dt}$, avec les variables d'entrée et de sortie ainsi que les composantes du convertisseur et les variables dynamiques i_L, v_c sont de la forme :

$$\frac{dv_c}{dt} = f(i_L, v_c, L, R_L, C) \qquad \text{(I.8)}$$

$$\frac{di_L}{dt} = g(i_L, vc, L, R_L, C) \qquad \text{(I.9)}$$

Les grandeurs temporelles sont représentées par des lettres minuscules alors que les grandeurs moyennes sont représentées par des majuscules.

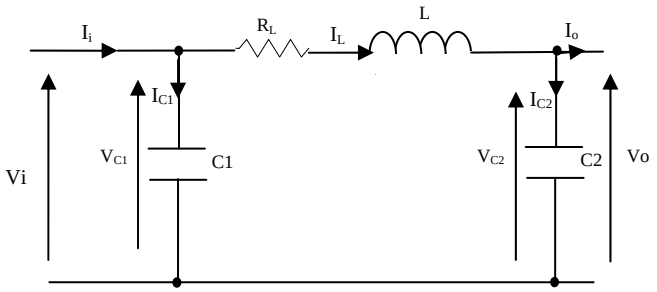


Figure I.9: Circuit équivalent de Buck lorsque S fermé.

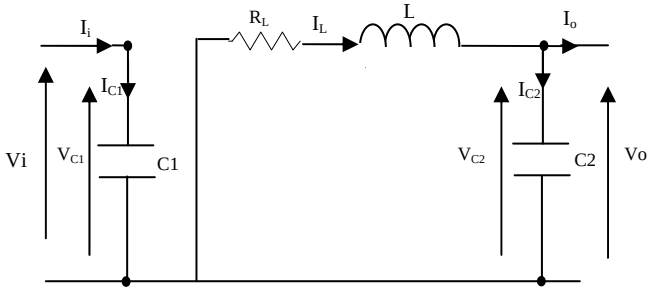


Figure I.10: Circuit équivalent de Buck lorsque S ouvert.

En appliquant les lois de Kirchhoff sur les deux circuits, on obtient les systèmes d'équations suivants :

***0 < t < DTs* :**

$$\begin{cases} i_{C1} = C_1 \dfrac{dvi}{dt} = i_i - i_L \\ i_{C2} = C_2 \dfrac{dv_O}{dt} = i_L - i_O \\ v_L = L \dfrac{di_L}{dt} + R_L i_L = v_i - v_O \end{cases} \qquad \text{(I .10)}$$

***DTs < t < Ts* :**

$$\begin{cases} i_{C1} = C_1 \dfrac{dvi}{dt} = i_i \\ i_{C2} = C_2 \dfrac{dv_O}{dt} = i_L - i_O \\ v_L = L \dfrac{di_L}{dt} + R_L i_L = -v_O \end{cases} \qquad \text{(I .11)}$$

I.4.1.2. Modèle 'Moyen' d'un hacheur Buck

Les systèmes d'équations de base I.10, I.11 représentent le hacheur Buck pour une période *DTs et (1-D)Ts* respectivement. Le convertisseur s'alterne entre ces deux états avec une fréquence élevée, nous devons trouver une représentation dynamique valable pour les deux

intervalles de temps appelée modèle moyen (*Averaged model*). Pour cela nous considérons que la variation des variables dynamiques I_C, V_L est de forme linéaire. En d'autre termes nous pouvons faire une approche linéaire pour l'exponentielle ($e^{\varepsilon} \approx \varepsilon$), ainsi la dérivé de ces grandeurs sera constante.

Cette approche nous permet de décomposer l'expression de la valeur moyenne de la dérivé de la variable dynamique *x* sur les deux laps de temps *DTs* et *(1-D) Ts* :

$$<\frac{dx}{dt}>Ts=\frac{dx}{dt}_{(DTs)}.DTs+\frac{dx}{dt}_{((1-D)Ts)}.(1-D)Ts \qquad \text{(I.12)}$$

Où $<\frac{dx}{dt}>$ est la valeur moyenne de la dérivée de *x* sur une période *Ts*. Cette relation est valide si $\frac{dx}{dt}_{(DTs)}$ et $\frac{dx}{dt}_{((1-D)Ts)}$ sont constantes sur les périodes *DTs* et *(1-D)Ts* respectivement. En d'autres termes, cette approximation est valable si les périodes *DTs* et *(1-D)Ts* sont très faibles devant la constante de temps du circuit C_1Rg, C_2Z, LR_L [8].

Dans ce cas la forme exponentielle du courant qui parcourt la self et la tension aux bornes de la capacité sont de forme linéaire comme le montre la figure I.11.

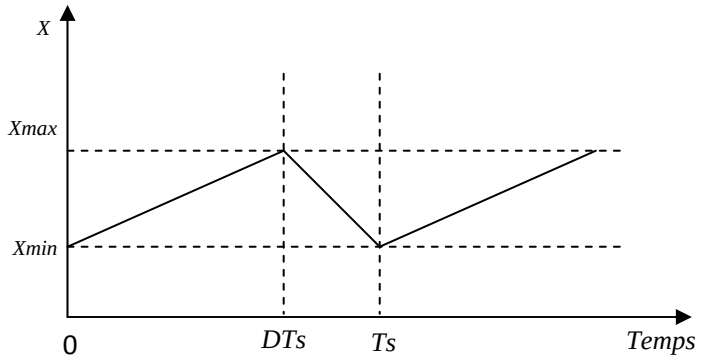


Figure I.11: Variations des variables dynamiques I_L, V_{c1}, V_{c2}.

En appliquant la relation I.12 sur les systèmes d'équations I.10 et I.11, on obtient les équations qui régissent le système sur une période entière :

$$\begin{cases} C_1\dfrac{dvi}{dt}Ts = DTs(i_i - i_L) + (1-D)Tsi_i \\ C_2\dfrac{dv_O}{dt}Ts = DTs(i_L - i_O) + (1-D)Ts(i_L - i_O) \\ L\dfrac{di_L}{dt}Ts = DTs(v_i - v_O - R_L i_L) + (1-D)Ts(-v_O - R_L i_L) \end{cases} \qquad \text{(I .13)}$$

En arrangeant les termes des équations précédentes pour qu'on puisse interconnecter le convertisseur avec les autres blocs de simulation, on obtient la modélisation dynamique du hacheur Buck :

$$\begin{cases} i_O = i_L - C_2 \dfrac{dv_O}{dt} \\ i_L = \dfrac{1}{D}(i_i - C_1 \dfrac{dv_i}{dt}) \\ v_i = \dfrac{1}{D}(v_O + R_L i_L + L \dfrac{di_L}{dt}) \end{cases} \quad \text{(I .14)}$$

I.4.1.3. Les ondulations des courants et des tensions

Pour le dimensionnement des différentes composantes du circuit, afin de minimiser les ondulations des courants et des tensions sans faire un surdimensionnement des composants, ce qui accroîtrait le poids et le prix du circuit, un calcul de ces composants en fonction des ondulations voulues est nécessaire. Cette remarque est très importante pour le dimensionnement de l'inductance *L* afin de respecter le courant admissible par le transistor MOSFET *S.*

En appliquant la relation $v_L = L \frac{di_L}{dt}$, et par l'approximation des segments d'exponentielles par des droites, la pente de courant I_L pendant la première période de fonctionnement est donnée par :

$$\frac{di_L}{dt} = \frac{v_L}{L} \approx \frac{V_i - V_O - R_L I_L}{L} \quad \text{(I .15)}$$

A partir de la relation I.15, la valeur crête à crête de courant I_L est :

$$I_{Lcc} = 2\Delta I_L = \frac{V_i - V_O - R_L I_L}{L} DTs \quad \text{(I .16)}$$

La valeur de l'inductance L à choisir pour certaine ondulation ΔI_L est :

$$L = \frac{Vi - Vo - R_L I_L}{2\Delta I_L} DTs \quad \text{(I .17)}$$

Pour le calcul des capacités C_1 et C_2, et avec les mêmes démarches précédentes nous avons :

$$\frac{dV_{c1}}{dt} = \frac{i_{C1}}{C_1} \approx \frac{I_i - I_L}{C_1} \quad \text{(I.18)}$$

$$\frac{dV_{C2}}{dt} = \frac{i_{C2}}{C2} \approx \frac{I_L - I_O}{C_2} \quad \text{(I .19)}$$

Les valeurs des ondulations crête à crête des tensions d'entrée et de sortie sont :

$$V_{icc} = \frac{I_i - I_L}{C_1} DTs \quad \text{(I .20)}$$

$$V_{occ} = \frac{I_L - I_o}{C_2} DTs \quad \text{(I .21)}$$

Les valeurs des condensateurs C_1 et C_2 sont :

$$C_1=\frac{I_i-I_L}{2\Delta Vi}DTs \quad \text{(I .22)}$$

$$C_2=\frac{I_L-I_o}{2\Delta V_o}DTs \quad \text{(I .23)}$$

I.4.1.4. Etude du régime continu

Le régime continu est obtenu en éliminant les dérivées des variables dynamiques, et en remplaçant ces signaux par leurs valeurs moyennes.

Le système d'équations I.14 donne :

$$\begin{cases} I_i - DI_L = 0 \\ I_O - I_L = 0 \\ DV_i - V_O - R_L I_L = 0 \end{cases} \quad \text{(I .24)}$$

I .4.1.5. Rapport de conversion et rendement

Le rapport de conversion *M* est défini comme étant le rapport entre la tension de sortie et la tension d'entrée comme

$$M(D)=\frac{V_o}{V_i}=\eta.D \quad \text{(I .25)}$$

Où η est le rendement de convertisseur défini comme étant le rapport entre la puissance de sortie sur la puissance d'entrée :

$$\eta=\frac{Po}{Pi}=\frac{V_oI_o}{V_iI_i} \quad \text{(I .26)}$$

La relation I.25 donne :

$$M(D)=\frac{V_O}{V_i}=\frac{1}{1+\frac{R_L I_O}{V_O}}D=\frac{1}{1+\frac{R_L}{Z}}D=\eta D \quad \text{(I .27)}$$

Avec

$$\eta=\frac{1}{1+\frac{R_L}{Z}} \quad \text{(I .28)}$$

Avec *Z* l'impédance complexe de la charge.

A partir des relations (I.27) et (I.28) on conclut que le rapport de conversion *M* reste linéaire en fonction de *D* et reste confiné entre zéro et la valeur du rendement, et que des charges Z importantes causent une grande perte dans le transfert de puissance à travers le convertisseur ainsi qu'une tension de sortie faible.

I.4.2. Hacheur parallèle (Boost)

Le convertisseur Boost est connu par le nom d'élévateur de tension, abaisseur de courant. Le schéma ci-dessous représente le circuit électrique du Boost. Durant le temps *DTs*, le transistor S est fermé, le courant dans l'inductance croit progressivement, au fur et à mesure elle emmagasine de l'énergie, jusqu'à la fin du premier intervalle. Le transistor S s'ouvre et l'inductance *L* délivre le courant I_L et ainsi génère une tension qui s'ajoute à la tension de source, qui s'applique sur la charge *Z* à travers la diode D.

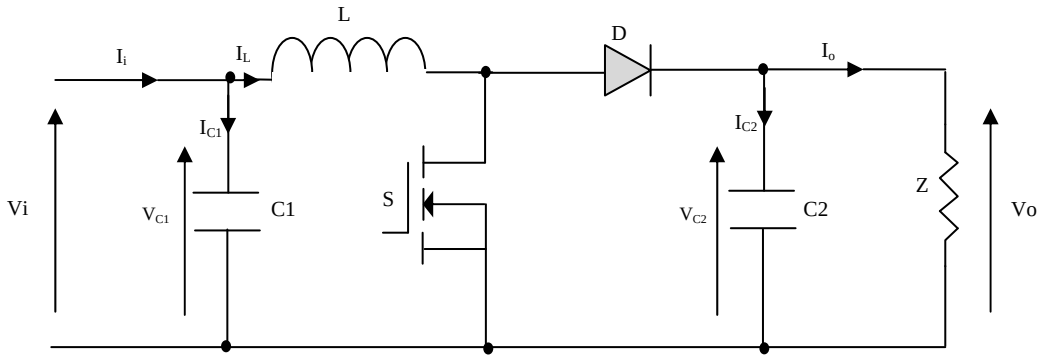


Figure I.12: Circuit électrique du hacheur Boost

I.4.2.1. Modèle mathématique équivalent

Comme pour le circuit Buck, l'application des lois de Kirchhoff sur les deux circuits équivalents des deux phases de fonctionnement, les figures I.13, I.14, donnent les équations suivantes :

0 < t < DTs :

$$\begin{cases} i_{C1} = C_1 \dfrac{dvi}{dt} = i_i - i_L \\ i_{C2} = C_2 \dfrac{dv_O}{dt} = -i_O \\ v_L = L \dfrac{di_L}{dt} = v_i - R_L i_L \end{cases} \qquad \text{(I.29)}$$

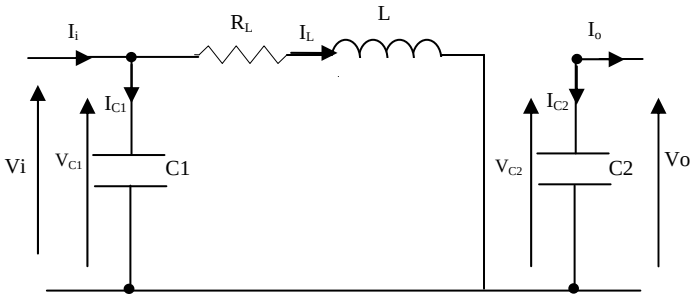


Figure I.13: Circuit équivalent du Boost lorsque S fermé.

DTs < t < Ts :

$$\begin{cases} i_{C1} = C_1 \dfrac{dvi}{dt} = i_i - i_L \\ i_{C2} = C_2 \dfrac{dv_O}{dt} = i_L - i_O \\ v_L = L \dfrac{di_L}{dt} = v_i - v_O - R_L i_L \end{cases} \qquad \text{(I.30)}$$

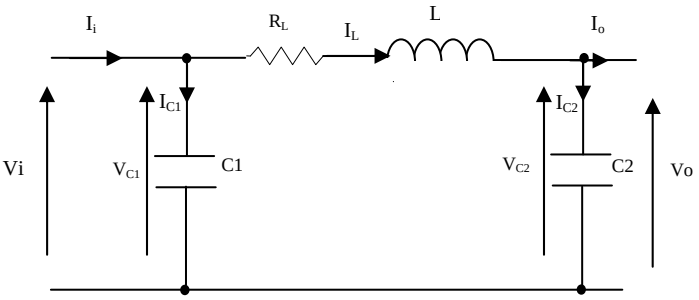


Figure I.14: Circuit équivalent du Boost lorsque S ouvert.

I.4.2.2. Modèle Moyen du hacheur Boost

Comme pour le Buck, en appliquant la relation I.12 sur les systèmes d'équations I.29 et I.30, on trouve le modèle moyen:

$$\begin{cases} C_1 \dfrac{dvi}{dt} Ts = DTs\,(i_i - i_L) + (1-D)Ts\,(i_i - i_L) \\ C_2 \dfrac{dvi}{dt} Ts = -DTs\, i_O + (1-D)Ts\,(i_L - i_O) \\ L \dfrac{di_L}{dt} Ts = DTs\,(v_i - R_L i_L) + (1-D)Ts\,(v_i - v_O - R_L i_L) \end{cases} \quad \text{(I.31)}$$

En arrangeant les termes des équations précédentes, pour qu'on puisse interconnecter le Boost avec les autres blocs de simulation, on obtient la modélisation dynamique du hacheur Boost :

$$\begin{cases} i_L = i_i - C_1 \dfrac{dv_i}{dt} \\ i_O = (1-D) i_L - C_2 \dfrac{dv_O}{dt} \\ v_i = (1-D) v_O + R_L i_L + L \dfrac{di_L}{dt} \end{cases} \quad \text{(I.32)}$$

I.4.2.3. Les ondulations des courants et tensions

Comme pour le circuit Buck la pente du courant I_L et des tensions V_{c1} et V_{c2} pendant la première période de fonctionnement est donnée par :

$$\begin{cases} \dfrac{di_L}{dt} \approx \dfrac{v_L}{L} = \dfrac{V_i - R_L I_L}{L} \\ \dfrac{dv_{C1}}{dt} \approx \dfrac{i_{C1}}{C_1} = \dfrac{I_i - I_L}{C_1} \\ \dfrac{dv_{C2}}{dt} \approx \dfrac{i_{C2}}{C_2} = \dfrac{-I_O}{C_2} \end{cases} \quad \text{(I.33)}$$

Les valeurs crête à crête des courants et des tensions sont :

$$\begin{cases} I_{Lcc} = 2\Delta I_L = \dfrac{V_i - R_L I_L}{L} DTs \\ V_{icc} = 2\Delta V_i = \dfrac{I_i - I_L}{C_1} DTs \\ V_{Occ} = 2\Delta V_O = \dfrac{-I_O}{C_2} DTs \end{cases} \quad \text{(I.34)}$$

Les valeurs des composants à choisir pour des ondulations données sont :

$$\begin{cases} L = \dfrac{V_i}{2\Delta I_L} DTs \\ C_1 = \dfrac{I_i - I_L}{2\Delta V_i} DTs \\ C_2 = \dfrac{-I_O}{2\Delta V_O} DTs \end{cases} \quad \text{(I.35)}$$

I.4.2.4. Etude du régime continu

Comme pour le circuit Buck, en remplaçant les dérivées des signaux par des zéros, ainsi on peut remplacer les signaux du convertisseur par leurs grandeurs moyennes, cela simplifie les systèmes d'équations précédents comme suit :

$$\begin{cases} I_i = I_L \\ I_O = (1-D)I_L \\ V_i = (1-D)V_O + R_L I_L \end{cases} \quad \text{(I.36)}$$

I.4.2.5. Rapport de conversion et rendement

En utilisant les relations I.36, on peut calculer le rapport de conversion V_O/Vi:

$$M(D)=\frac{Vo}{Vi}=\frac{1}{(1-D)+\frac{R_L I_L}{Vo}}=\frac{1}{1+\frac{R_L Io}{(1-D)^2 Vo}}\frac{1}{1-D}=\frac{1}{1+\frac{R_L}{(1-D)^2 Z}}\frac{1}{1-D}=\eta\frac{1}{1-D} \quad \text{(I.37)}$$

On remarque que le rendement η ne dépend pas seulement de la charge complexe *Z* du convertisseur et des résistances parasites des composants, mais il dépend aussi du rapport cyclique *D*. Ainsi il est recommandé, pour que le Boost fournisse un bon rendement, de ne pas dépasser des rapports cycliques *D* supérieurs à une certaine valeur, fixée par la qualité de l'inductance et la charge utilisée.

I.5. Conclusion

Le générateur photovoltaïque est l'ensemble des modules photovoltaïques couplés aux éléments de contrôle.

La technologie photovoltaïque présente un grand nombre d'avantages.

- D'abord, une haute fiabilité (elle ne comporte pas de pièces mobiles) qui la rend particulièrement appropriée aux régions isolées. C'est la raison de son utilisation sur les engins spatiaux.
- Ensuite, le caractère modulaire des panneaux photovoltaïques permet un montage simple et adaptable à des besoins énergétiques divers. Les systèmes peuvent être dimensionnés pour des applications de puissance allant du milliwatt au Mégawatt.
- Leurs coûts de fonctionnement sont très faibles vu les entretiens réduits et ils ne nécessitent ni combustible, ni transport, ni personnel hautement spécialisé.
- Enfin, la technologie photovoltaïque présente des qualités sur le plan écologique car le produit fini est non polluant, silencieux et n'entraîne aucune perturbation du milieu, si ce n'est par l'occupation de l'espace pour les installations de grandes dimensions.

Le système photovoltaïque présente toutefois des inconvénients:

- La fabrication du module photovoltaïque relève de la haute technologie et requiert des investissements d'un coût élevé.
- Le rendement réel de conversion d'un module est faible (la limite théorique pour une cellule au silicium cristallin est de 28%).
- Les générateurs photovoltaïques ne sont compétitifs par rapport aux générateurs Diesel que pour des faibles demandes d'énergie en région isolée.
- lorsque le stockage de l'énergie électrique sous forme chimique (batterie) est nécessaire, le coût du générateur photovoltaïque est accru. La fiabilité et les performances du système restent cependant équivalentes pour autant que la batterie et les composants de régulation associés soient judicieusement choisis.
- Tributaire des conditions météorologiques.
- Le faible rendement des panneaux photovoltaïques s'explique par le fonctionnement même des cellules. Pour arriver à déplacer un électron, il faut que l'énergie du rayonnement soit au moins égale à 1 eV. Tous les rayons incidents ayant une énergie plus faible ne seront donc pas transformés en électricité. De même, les rayons lumineux dont l'énergie est supérieure à 1eV perdront cette énergie, le reste sera dissipé sous forme de chaleur [9].

Nous avons présenté dans ce chapitre le principe de la génération photovoltaïque, la modélisation du GPV. Un aperçu sur les hacheurs les plus utilisés dans ce genre de dispositifs a été exposé.

Les réseaux de neurones et la logique floue

II.1. Introduction

Ce chapitre est consacré aux concepts de base des réseaux de neurones, de la logique floue et l'approche *hybride* neuro-floue.

II.2. Les réseaux de neurones

Avec l'apparition des ordinateurs l'homme a découvert un moyen d'effectuer diverses tâches avec deux capacités non négligeables que lui ne possède pas: la rapidité et la précision.

Cependant l'exécution d'une tâche pour l'ordinateur nécessite sa programmation préalable par l'homme. Cette caractéristique fait apparaître les ordinateurs comme des machines exécutant des ordres « aveuglement » et l'homme n'a pas désespéré de voir un jour construire une machine à son image, c'est à dire intelligente, capable d'apprendre, de raisonner, de réfléchir sans son intervention.

Ce sont des recherches basées sur le fonctionnement du cerveau qui ont constitué le point de départ de cette gigantesque recherche.

Des travaux de neurobiologistes ont, en effet, révélé que le cerveau est constitué d'un nombre extrêmement élevé d'unités de traitement élémentaire de l'information (les neurones biologiques) fortement interconnectées.

L'information contenue dans le cerveau est stockée dans les connexions entre les neurones et c'est la coopération entre les neurones, qui effectuent un traitement fortement parallèle et distribué, qui donne sa puissance au cerveau.

II.2.1. Présentation générale d'un réseau de neurones biologiques

Le neurone est une cellule composée d'un corps cellulaire et d'un noyau. Le corps cellulaire se ramifie pour former ce que l'on nomme les dendrites. Celles-ci sont parfois si nombreuses que l'on parle alors de chevelure dendritique ou d'arborisation dendritique. C'est par les dendrites que l'information est acheminée de l'extérieur vers le soma, corps du neurone [10].

L'information traitée par le neurone chemine ensuite le long de l'axone pour être transmise aux autres neurones. La transmission entre deux neurones n'est pas directe. En fait, il existe un espace intercellulaire de quelques dizaines d'Angströms (10^{-9}m) entre l'axone du neurone afférent et les dendrites du neurone efférent. La jonction entre deux neurones est appelée la synapse (figure II.1).

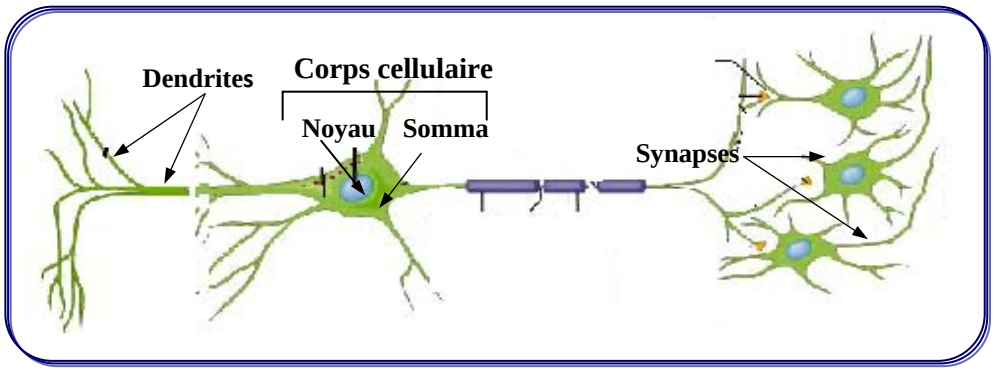


Figure II.1: Structure d'un réseau de neurones biologiques.

II.2.2. Le modèle mathématique d'un réseau de neurones

Les réseaux de neurones biologiques réalisent facilement un certain nombre d'applications telles que la reconnaissance de formes, le traitement du signal, l'apprentissage par l'exemple, la mémorisation, la généralisation. Ces applications sont pourtant, malgré tous les efforts déployés en algorithmique et en intelligence artificielle, à la limite des possibilités actuelles. C'est à partir de l'hypothèse que le comportement intelligent émerge de la structure et du comportement des éléments de base du cerveau que les réseaux de neurones artificiels se sont développés. Les réseaux de neurones artificiels sont des modèles, à ce titre ils peuvent être décrits par leurs composants, leurs variables descriptives et les interactions des composants.

Cet héritage de la neurobiologie forme une composante importante de l'étude des réseaux connexionnistes, et le souci de maintenir une certaine correspondance avec le système nerveux humain à animer une part importante des recherches dans ce domaine. Malgré cet héritage, l'essentiel des travaux réalisés aujourd'hui ont pour objet, les réseaux de neurones formels et non leur corrélât neurobiologique. Par ailleurs, et considérés comme des systèmes de calcul, les réseaux de neurones possèdent plusieurs propriétés qui les rendent intéressants d'un point de vue théorique et fort utile en pratique.

Un réseau connexionniste est constitué par un graphe orienté et pondéré, dans lequel les noeuds sont des automates simples nommés neurones formels ou tout simplement unités du réseau. Ces unités sont dotées d'un état interne, que l'on appelle état d'activation dont le rôle est de propager son état d'activation aux autres unités du graphe en passant par des arcs pondérés appelés connexions, liens synaptiques ou bien poids synaptiques. La règle qui détermine l'activation d'un neurone en fonction de l'influence venue de ses entrées et de leurs poids respectifs s'appelle règle d'activation ou fonction d'activation. Les changements apportés aux valeurs des poids synaptiques ou dans la structure d'interconnexion des unités du réseau sont responsables des changements de comportement d'activation du réseau, lui permettant

d'apprendre un nouveau comportement. En effet, le réseau est capable d'établir des associations entrée-sortie (stimulus et réponse) afin de bien résoudre un problème, par ce qu'on appelle règle d'apprentissage [11].

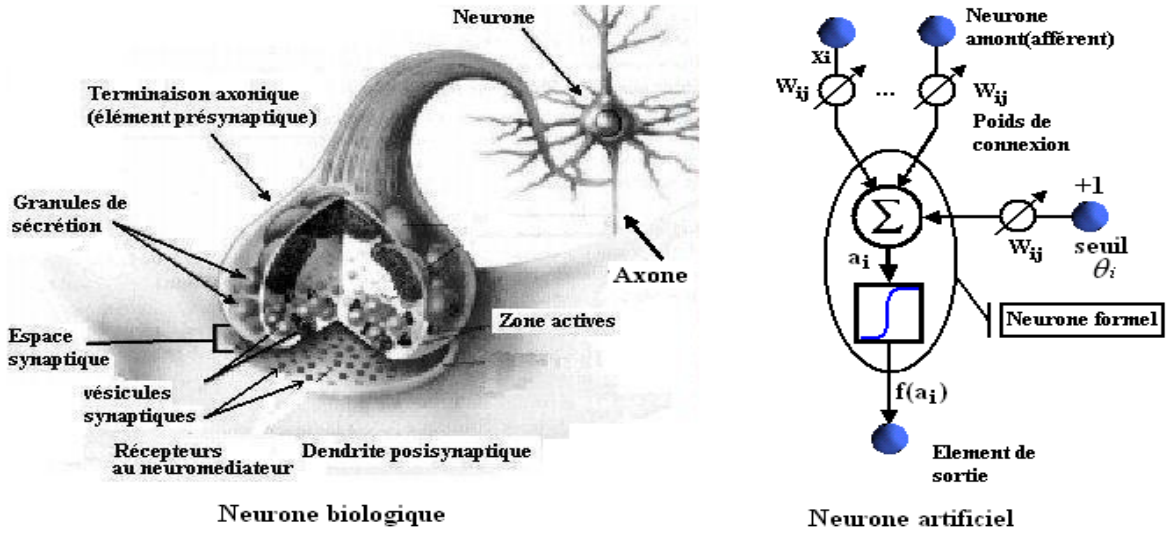


Figure II.2: Mise en correspondance neurone biologique / neurone artificiel.

D'après la définition de Kohonen [11], " Les réseaux de neurones artificiels sont des réseaux fortement connectés de processeurs élémentaires fonctionnant en parallèle. Chaque processeur élémentaire calcule une sortie unique sur la base des informations qu'il reçoit. Toute structure hiérarchique de réseaux est évidemment un réseau. Ils sont censés interagir avec les objets du monde réel de la même manière que les systèmes nerveux biologiques"**.**

Aujourd'hui, une manière simple de concevoir un réseau de neurones est de considérer qu'il s'agit d'un système de traitement de l'information composé d'un grand nombre de processeurs interconnectés (cellules). Chaque cellule calcule sa sortie sur la base des informations reçues des autres cellules qui lui sont connectées et des poids de ces connexions.

En étudiant le cerveau humain et son comportement, on constate que le comportement intelligent émerge de sa structure (cerveau) et du comportement de ses éléments de base.
C'est à partir de cette hypothèse que les réseaux de neurones artificiels se sont développés.

A. Structure :

Les réseaux de neurones artificiels sont des modèles à base de neurones artificiels, chaque neurone est un processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones amonts. A chacune de ces entrées est associé un poids **(W)** abréviation de poids synaptiques représentatifs de la force de la connexion. Chaque processeur élémentaire est doté d'une sortie unique, qui se ramifie ensuite pour alimenter un nombre variable de neurones avals.

B. Comportement :

On distingue deux phases pour décrire le comportement d'un neurone (i):

1/ Le calcul de la somme pondérée des entrées, A_i, selon l'expression suivante

$$A_i = \sum (W_{ij} X_j) \qquad \text{(II .1)}$$

2/ A partir de cette valeur, une fonction de transfert calcule la valeur de l'état du neurone.

$$Y_i = f(A_i) \qquad \text{(II .2)}$$

Où :

X_j : les entrées, qui peuvent être les états des neurones en amant.

Y_i : l'état d'un neurone i.

A_i : l'activité du neurone i.

W_{ij} : le poids synaptiques de la connexion entre les neurones j et i.

Le modèle de neurone formel donc est un modèle mathématique très simple dérivé de l'analyse de la réalité biologique dont la structure est donnée par la figure II.3.

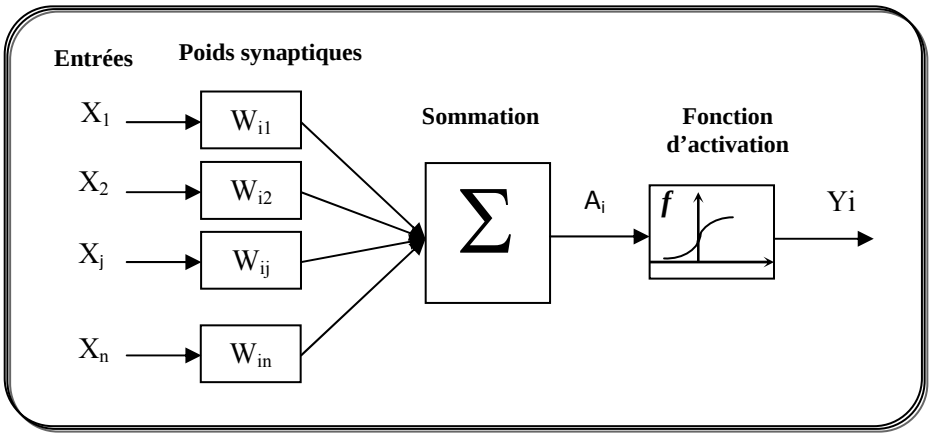


Figure II.3: Modèle mathématique du neurone.

C'est cette valeur qui sera transmise aux neurones avals. Il existe de nombreuses formes possibles pour la fonction de transfert. Les plus courantes sont présentées sur la Figure II.4. On remarquera qu'à la différence des neurones biologiques dont l'état est binaire, la plupart des fonctions de transfert sont continues, offrant une infinité de valeurs possibles comprises dans l'intervalle [0, +1] ou [-1, +1].

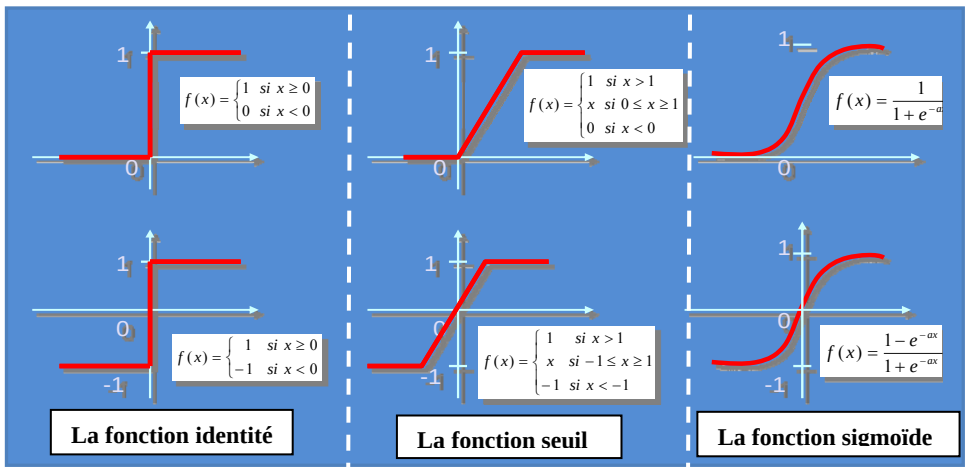


Figure II.4: Les différentes fonctions d'activation.

II.2.3. Apprentissage des réseaux de neurones

II.2.3.1. Définition

L'apprentissage est une phase de développement d'un réseau de neurones durant laquelle le comportement du réseau est modifié jusqu'à l'obtention du comportement désiré. Pour cela il nécessite :

1- Un ensemble d'exemples d'apprentissage: en effet les réseaux de neurones sont des fonctions paramétrées, utilisées pour réaliser des modèles statistiques à partir d'exemples (dans le cas de la classification) ou de mesures (dans le cas de la modélisation) ; leurs paramètres sont calculés à partir de ces exemples ou couples {entrée, sortie}.

2- La définition d'une fonction de coût qui mesure l'écart entre les sorties du réseau de neurones et les sorties désirées.

3- Un algorithme de minimisation de la fonction de coût par rapport aux paramètres.

II.2.3.2. Types d'apprentissage

Il existe trois types d'apprentissage :

A. Apprentissage supervisé

Dans l'apprentissage supervisé, un professeur qui connaît parfaitement la sortie désirée ou correcte, guide le réseau en lui apprenant à chaque étape le bon résultat. Donc l'apprentissage ici, consiste à comparer le résultat obtenu avec le résultat désiré, puis à ajuster les poids des connexions pour minimiser la différence entre les deux résultats.

B. Apprentissage renforcé

L'utilisateur ne possède que des indications imprécises sur le comportement final désiré (par exemple, échec/succès du réseau). En effet, souvent nous ne disposons que d'une évaluation qualitative du comportement du système réel.

C. Apprentissage non supervisé

L'apprentissage supervisé s'effectue sous le contrôle d'un expert, alors que l'apprentissage non supervisé est autodidacte. Les paramètres internes du réseau ne sont modifiés qu'avec les seuls stimuli, aucune réponse désirée n'est prise en considération.

II.2.4. Topologies de réseaux (structure d'interconnexion)

Les connexions entre les neurones qui composent le réseau décrivent la topologie du modèle. Elle peut être quelconque, mais le plus souvent il est possible de distinguer une certaine régularité.

II.2.4.1. Réseaux acycliques ("feedforward")

1. **Réseaux à une seule couche :**

Dans ce type d'architecture, les neurones sont tous sur le même niveau et sont connectées directement aux entrées et sont aussi les sorties du réseau. Les réseaux à une seule couche ont normalement des connexions latérales (entre les neurones d'une même couche). Un exemple de ce type d'architecture est les réseaux du type "Kohonen Feature Map".

2. **Réseaux multicouche (MLP):**

Dans les réseaux MLP « Multi Layer Perceptron », Les neurones sont arrangés par couche. Il n'y a pas de connexion entre neurones d'une même couche et les connexions ne se font qu'avec les neurones des couches avales (figure II.5). Habituellement, chaque neurone d'une couche est connecté à tous les neurones de la couche suivante et celle-ci seulement. Ceci nous permet d'introduire la notion de sens de parcours de l'information (de l'activation) au sein d'un réseau et donc définir les concepts de neurone d'entrée, neurone de sortie. Par extension, on appelle couche d'entrée l'ensemble des neurones d'entrée, couche de sortie l'ensemble des neurones de sortie. Les couches intermédiaires n'ayant aucun contact avec l'extérieur sont appelés couches cachées.

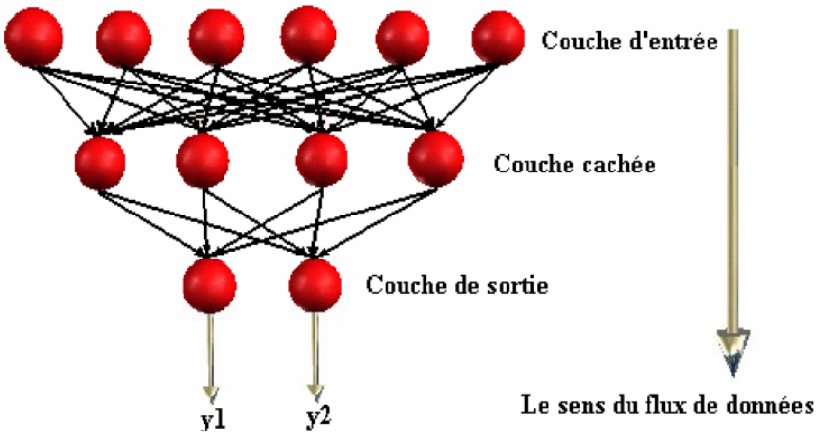


Figure II.5: Topologie d'un réseau multicouche (MLP).

II.2.4.2. Réseaux à connexions récurrentes :

Les réseaux récurrents peuvent avoir une ou plusieurs couches, mais leur particularité réside dans la présence d'interconnexions depuis la sortie d'un neurone vers un autre de la même couche ou d'une couche inférieure. Ce type d'interconnexions permet de modéliser des aspects temporaux et des comportements dynamiques dans lesquels la sortie de chaque neurone dépend de son état antérieur. Les boucles internes rendent ce type de réseaux instable, ce qui nous oblige à utiliser des algorithmes plus spécifiques (et usuellement plus complexes) pour l'apprentissage. Un type particulier de réseau récurrent est les réseaux totalement connectés, tels que le modèle de Hopfield.

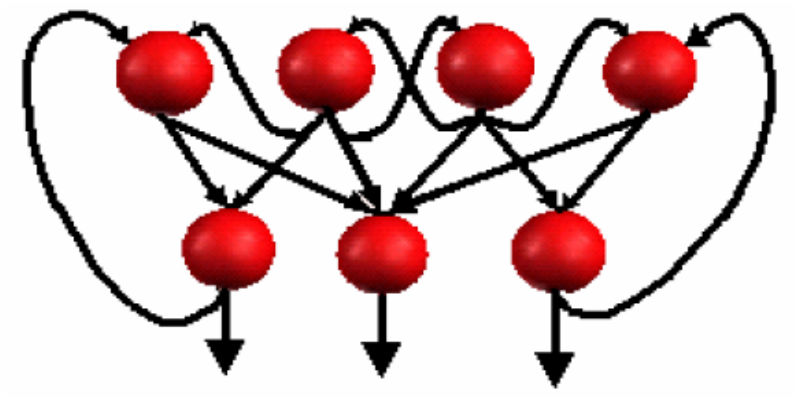


Figure II.6: Réseau à connexions récurrentes.

II.2.4.3. Réseaux à connexion complète :

C'est la structure d'interconnexion la plus générale (figure .II .7).

Chaque neurone est connecté à lui-même et à tous les neurones du réseau.

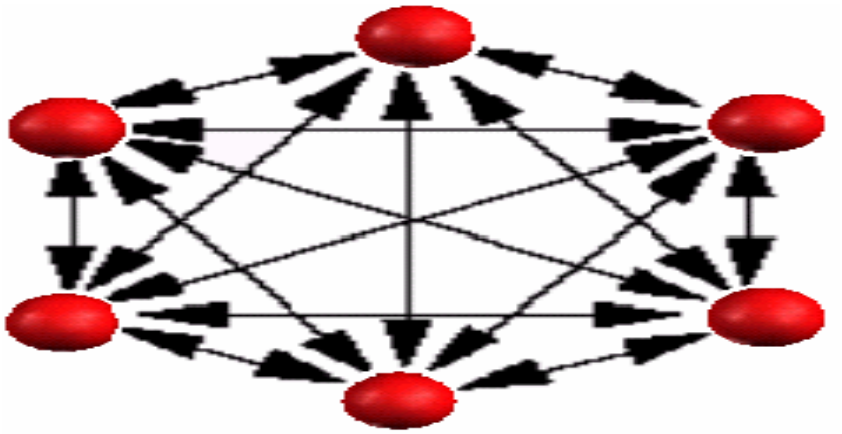


Figure II.7: Réseau à connexions complètes.

Il existe de nombreuses autres topologies possibles, mais elles n'ont pas eu à ce jour la notoriété des quelques unes que nous avons décrites ici.

II.2.5. Le perceptron multicouche (PMC)

C'est une extension du perceptron monocouche avec une ou plusieurs couches cachées entre l'entrée et la sortie. Les fonctions d'activation utilisées dans ce type de réseaux sont principalement les fonctions à seuil ou sigmoïdes. Ce type de réseau peut résoudre des problèmes non linéairement séparables et des problèmes logiques plus compliqués, et notamment le fameux problème du XOR.

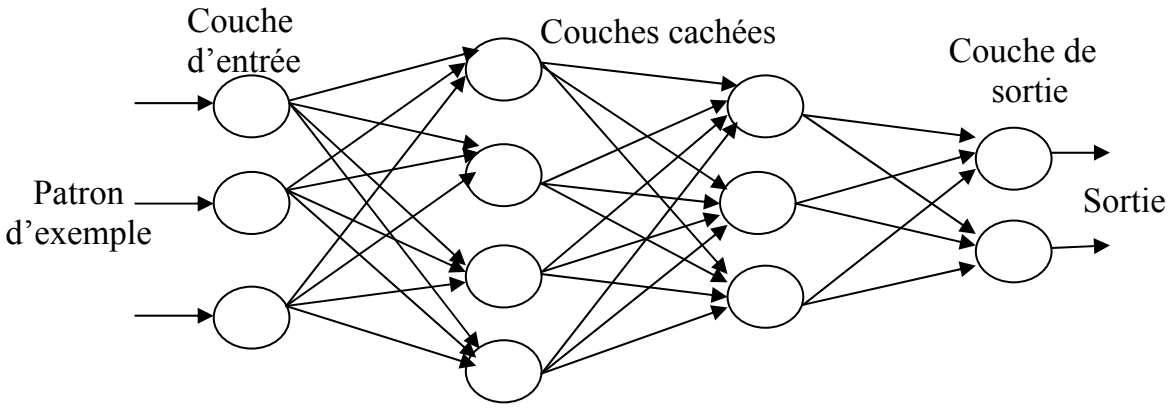


Figure II.8: Architecture du perceptron multicouche.

Les réseaux PMC sont constitués par des neurones organisés en couches à savoir: une couche d'entrée, une ou plusieurs couches cachées, et une couche de sortie. Les unités de la couche d'entrée reçoivent les signaux provenant de l'environnement. Ces signaux représentent les informations traitées par le réseau, telles que : la description de l'état du système, les valeurs des attributs associés aux variables d'entrées, les mesures obtenues à partir de capteurs externes ou les faits qui composent les prémisses d'un système expert. La couche de sortie rend disponible le résultat de l'activation du réseau à l'environnement extérieur.

Enfin, les unités cachées, comme leur nom l'indique, n'ont pas d'interaction directe avec l'environnement. Dans ce cas les unités sont connectées selon une architecture sans boucles (connexions unidirectionnelles), mais on peut avoir des connexions inter-couches du type "raccourcis".

Les réseaux PMC utilisent une méthode d'apprentissage de type supervisé réalisé par un algorithme de minimisation sur l'erreur estimée à partir des sorties du réseau. L'emploi de cette méthode permet d'entraîner les unités de sortie du réseau, où l'erreur peut être obtenue directement par simple différence entre la valeur de sortie désirée et la valeur obtenue par le réseau; mais il permet aussi d'entraîner les unités cachées grâce à une technique de propagation en arrière de l'erreur à travers les couches du réseau, d'où le nom de Rétro-Propagation" (Back-Propagation Learning) [12].

II.2.6. Domaines d'application et mise en œuvre [12]

II.2.6.1. La régression non linéaire ou modélisation de données statiques

Il existe une immense variété de phénomènes statiques qui peuvent être caractérisés par une relation déterministe entre des causes et des effets ; les réseaux de neurones sont de bons candidats pour modéliser de telles relations à partir d'observations expérimentales, sous réserve que celles-ci soient suffisamment nombreuses et représentatives.

II.2.6.2. La modélisation de processus dynamiques non linéaires

Modéliser un processus, c'est trouver un ensemble d'équations mathématiques qui décrivent le comportement dynamique du processus, c'est-à-dire l'évolution de ses sorties en fonction des entrées ; c'est donc typiquement un problème qui peut être avantageusement résolu par un réseau de neurones, si le phénomène que l'on désire modéliser est non linéaire

II.2.6.3. La commande de processus

Commander un processus, c'est imposer à celui ci un comportement défini à l'avance en fonction des signaux de commande, l'ensemble (commande/processus) peut donc être considéré comme un système qui réalise une fonction (non linéaire) qu'un réseau de neurones peut approcher.

II.2.6.4. La classification

Supposons que l'on désire classer des formes en deux catégories, A ou B, en fonction de certaines caractéristiques de ces formes ; on peut définir une fonction qui vaut (+1) pour toutes les formes de la classe A et (-1) pour toutes les formes de la classe B. Les réseaux de neurones sont de bons candidats pour réaliser une approximation de cette fonction, et l'on peut démontrer que cette approximation constitue une estimation de la probabilité d'appartenance de la forme inconnue à la classe A. Les réseaux de neurones fournissent donc une information très riche, qui est loin d'être une simple réponse binaire. Cette propriété remarquable (que les réseaux de neurones partagent avec d'autres classificateurs) n'est malheureusement pas mise à profit dans la plupart des applications.

- Pour utiliser les RN dans l'une de ces applications il faut suivre certaines procédures :

- Il faut tout d'abord choisir l'architecture du réseau, c'est-à-dire les entrées externes, le nombre de neurones cachés, et l'agencement des neurones entre eux, de telle manière que le réseau soit en mesure de reproduire ce qui est déterminé dans les données, le nombre de poids ajustables est un des facteurs fondamentaux de la réussite d'une application : si le réseau possède un trop grand nombre de poids, c'est-à-dire si le réseau est trop "souple", il risque de s'ajuster au bruit qui est présent dans les données de l'ensemble d'apprentissage, et, même en l'absence de bruit, il risque de présenter des oscillations non significatives entre les points

d'apprentissage, donc de posséder de mauvaises propriétés d'interpolation ou de "généralisation». Si ce nombre est trop petit, le réseau est trop "rigide" et ne peut reproduire la partie déterministe de la fonction. Le problème de la détermination de l'architecture optimale est resté pendant longtemps un problème ouvert, mais il existe actuellement diverses méthodes, mettant notamment en jeu des tests statistiques, qui permettent de déterminer cette architecture pour une vaste classe de réseaux.

- Il faut calculer les poids du réseau ou, en d'autres termes, estimer les paramètres du réseau à partir des exemples, en minimisant l'erreur d'approximation sur les points de l'ensemble d'apprentissage, de telle manière que le réseau réalise la tâche désirée.
- Il faut enfin estimer la qualité du réseau obtenu en lui présentant des exemples qui ne font pas partie de l'ensemble d'apprentissage.

II.2.7. Les limitations d'un réseau de neurones [13]

II.2.7.1. Avantages

- Implémentation du parallélisme.
- Apprentissage.
- Robustesse: données bruitées ou incomplètes.
- Généralisation à des Modèles similaires.
- Trouve des solutions aux problèmes non linéaires.
- Trouve des solutions aux problèmes qui n'ont pas une modélisation.

II.2.7.2. Inconvénients

- N'ont pas encore expliqué le fonctionnement du cerveau.
- Les poids ne sont pas interprétables.
- L'apprentissage n'est pas toujours évident.
- Ne sont pas extensibles (l'ajout d'un neurone).

II.3. La logique floue

La logique floue est un axe de recherche important sur lequel sont focalisés de nombreux scientifiques, car elle utilise la méthodologie du raisonnement humain, à la résolution des problèmes sans avoir besoin d'une modélisation mathématique.

En 1965 le professeur L. Zadeh a fondé les bases théoriques de la logique floue. L'application de cette théorie à la régulation n'a eu lieu qu'a partir des années soixante dix [14]. A partir de 1985 la logique floue est devenue une actualité scientifique pour un large public de chercheurs.

Dans la théorie classique des ensembles, un élément ne peut avoir que deux états différents {0,1} alors qu'en logique floue, il peut admettre les deux propositions avec un degré de vérité pour chacune.

L'idée est de remplacer l'ensemble binaire {0,1} par un intervalle [0 1] ceci permet des graduations dans l'appartenance d'un élément à une situation, ce qui permet la modélisation de l'observation humaine exprimée par des expressions linguistiques [14].

II.3.1. Terminologie de la logique floue

II.3.1.1. Variable linguistique

La description d'une certaine situation imprécise ou incertaine peut contenir des expressions floues comme: très grand, grand, moyen, petit, …etc.

Ces expressions forment les valeurs d'une variable "x", appelée "linguistique", soumise à des fonctions appelées les fonctions d'appartenance.

II.3.1.2. Fonctions d'appartenance

La variable "x" varie dans un domaine appelé univers de discours, ce dernier est partagé en sous-ensembles flous à l'aide de la théorie des ensembles flous, et l'expertise humaine, de façon que dans chaque zone il y à une situation dominante. Ces zones sont décrites par des fonctions convexes sous forme triangulaire, trapézoïdale, …etc., admettant comme argument la position de la variable "x" dans l'univers de discours, et comme sortie le degré d'appartenance de "x" a une situation décrite par la fonction ; notée :

μE(x): degré d'appartenance de "x" à l'ensemble E.

Pour un choix judicieux, le chevauchement entre deux fonctions d'appartenance, doit être entre zéro et la mi-hauteur [15].

Quelques formes des fonctions d'appartenance sont illustrées sur la Figure II.9.

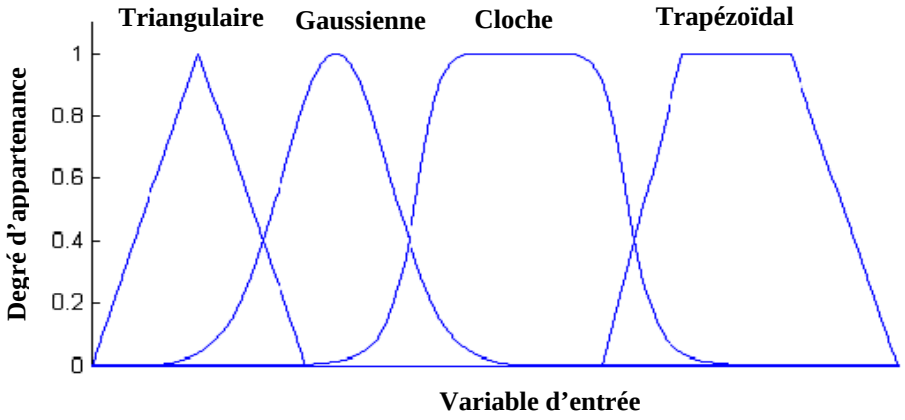


Figure II.9: Exemple de fonctions d'appartenance.

II.3.1.3. Opérateurs de la logique floue

La description d'une situation où il y a plus qu'une variable qui interviennent, nécessite l'utilisation des opérateurs logiques tels que: "ET", "OU", et "NON".

Dans la théorie de la logique floue l'opérateur "ET" correspond à l'opération "Minimum", "OU" à l'opération "Maximum" et "NON" au complément à un. L'analogie d'utilisation de ces opérateurs dans les deux logiques classique et floue est sur le Tableau II.1.

	Logique classique	Logique floue
$C = A\ ET\ B$	$C = A \cap B$	$\mu_C(x) = Min(\mu_A(x), \mu_B(x))$
$C = A\ OU\ B$	$C = A \cup B$	$\mu_C(x) = Max(\mu_A(x), \mu_B(x))$
$C = NON\ A$	$C = \overline{A}$	$\mu_C(x) = 1 - \mu_A(x)$

Tableau II.1. Application des opérateurs dans les deux logiques classique et floue. Avec : A, B, C : ensembles.

II.3.1.4. Règles floues

La logique floue a pour objectif de formaliser et de mettre en œuvre la façon de raisonnement d'un être humain. En cela, elle peut être classée dans le domaine de l'intelligence artificielle. L'outil le plus utilisé dans les applications de logique floue est la base de règles floues. Une base de règles floues est composée de règles qui sont généralement utilisées en parallèle, mais peuvent également être enchaînées dans certaines applications.

Une règle est du type: *SI* « prédicat » *ALORS* « conclusion ».

Par exemple: «Si température élevée et pression forte *ALORS* ventilation forte et soupape grande ouverte».

Les bases de règles floues, tout comme les systèmes experts classiques, fonctionnent en s'appuyant sur une base de connaissance issue de l'expertise humaine. Il y a néanmoins de grandes différences dans les caractéristiques et le traitement de cette connaissance.

- **Prédicat**

Un prédicat (encore appelé prémisse ou condition) est une combinaison de propositions par des opérateurs ET, OU, NON. Les propositions « température élevée » et «pression forte » de l'exemple précédent sont combinées par l'opérateur ET pour former le prédicat de la règle.

- **Inférence**

Les inférences lient les grandeurs mesurées et les variables de sortie par des règles linguistiques. Ces règles sont combinées en utilisant les liaisons *ET* et *OU*.

- **Conclusion**

La conclusion d'une règle floue est une combinaison de propositions liées par des opérateurs ET. Dans l'exemple précédent, « ventilation forte » et « soupape grande ouverte » sont la conclusion de la règle.

Les bases de règles floues, dans leur cas général, sont définies par des fonctions d'appartenance sur les variables du système, et par des règles qui peuvent être écrites textuellement, où chaque règle fait appel à des entrées et des sorties. Cependant, beaucoup d'applications définissent des tableaux de règles. Dans cette optique, l'espace est quadrillé, et à chaque case correspond une règle. Cette approche a l'avantage d'être systématique, mais elle ne permet pas toujours de traduire simplement (en un minimum de règles) l'expertise existante, et elle n'est applicable que pour deux ou trois entrées, alors que des bases de règles « libres » peuvent être bâties avec un nombre important de variables.

II.3.1.5. Mécanisme d'inférence de Mamdani

Une base de règles floues de Mamdani comprend des règles linguistiques faisant appel à des fonctions d'appartenance pour décrire les concepts utilisés. Le mécanisme d'inférence comprend les étapes suivantes :

1. Fuzzification

La fuzzification consiste à évaluer les fonctions d'appartenance utilisées dans les prédicats des règles. La figure II.10 illustre ceci par un exemple.

Exemple : Si «pression forte» ET «température élevée» ALORS «ouverture vanne grande»

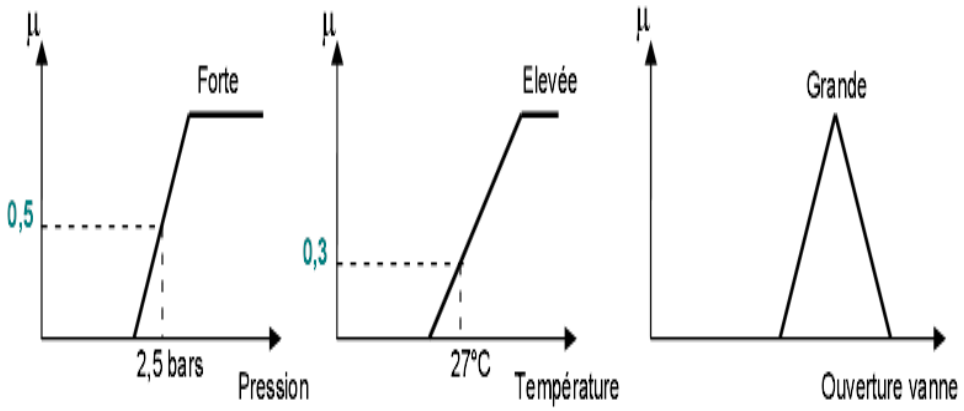


Figure II.10: Exemple du principe de fuzzification.

2. Degré d'activation et implication

Le degré d'activation d'une règle est l'évaluation du prédicat de chaque règle par combinaison logique des propositions du prédicat. Le «ET» est réalisé en effectuant le minimum entre les degrés de vérité des propositions.

Le degré d'activation de la règle permet de déterminer la conclusion de la règle, c'est l'implication. Il existe plusieurs opérateurs d'implication, mais le plus utilisé est le «minimum». L'ensemble flou de conclusion est construit en réalisant le minimum entre le degré d'activation et la fonction d'appartenance, sorte d'«écrêtage» de la fonction d'appartenance de conclusion.

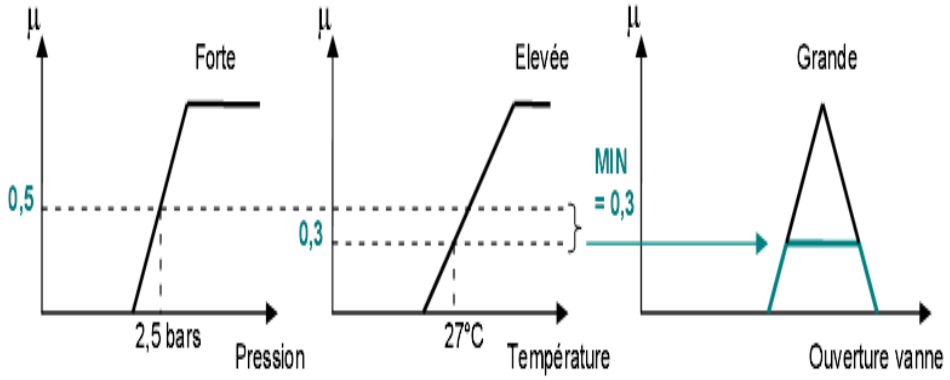


Figure II.11: Exemple du principe d'implication.

3. Agrégation

L'ensemble flou global de sortie est construit par agrégation des ensembles flous obtenus par chacune des règles concernant cette sortie. L'exemple suivant présente le cas où deux règles agissent sur une sortie. On considère que les règles sont liées par un «OU» logique, et on calcule donc le maximum entre les fonctions d'appartenance résultantes pour chaque règle.

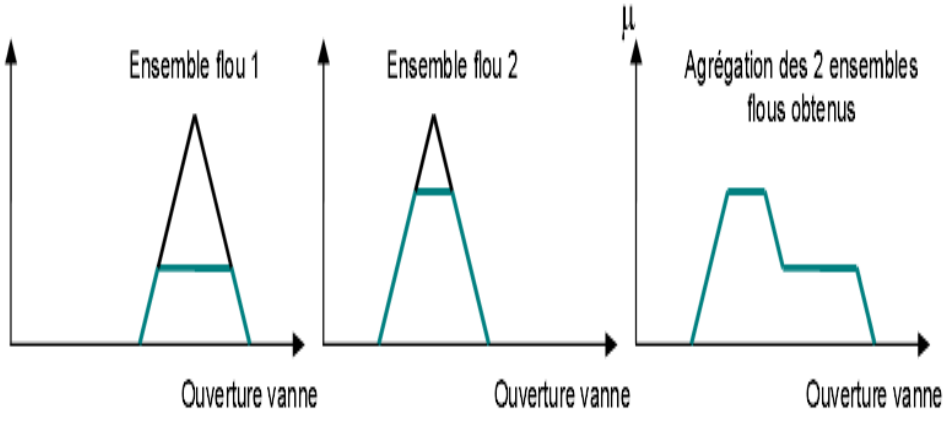


Figure II.12: Exemple du principe d'agrégation.

4. Défuzzification

A la fin de l'inférence, l'ensemble flou de sortie est déterminé mais il n'est pas directement utilisable pour donner une information précise à l'opérateur ou commander un actionneur. Il est nécessaire de passer du « monde flou» au «monde réel», c'est la défuzzification. Pour réaliser cette opération il existe plusieurs méthodes, la plus souvent rencontrée étant le calcul du «centre de gravité» de l'ensemble flou (figure II.13)

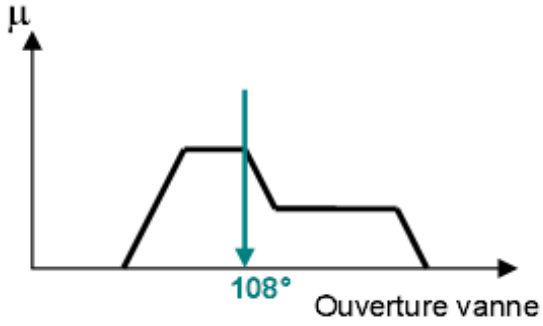


Figure II.13: Défuzzification par centre de gravité.

La formule qui permet d'obtenir le centre de gravité à partir de l'ensemble flou de sortie est la suivante :

$$x_0 = \frac{\int x \cdot \mu(x)\, dx}{\int \mu(x)\, dx} \qquad \text{(II .3)}$$

II.3.1.6. Mécanisme d'inférence de Takagi Sugeno [16,17]

La première partie d'une règle de type Takagi Sugeno est similaire à celle de Mamdani tandis que la deuxième est une fonctionnelle.

La forme typique de cette règle s'écrit donc :

Si x_1 *est* E_1 *(et)* x_2 *est* E_2 *(et)* x_n *est* E_n ***ALORS :***

$$\mu_1 = f_1(x_1,..,x_n),\ \mu_2 = f_2(x_1,..,x_n),...,\ \mu_n = f_n(x_1,..,x_n) \qquad \text{(II .4)}$$

Où $f_1,...,f_n$: fonctions réelles, théoriquement peuvent être linéaires ou non linéaires mais l'implémentation de la méthode exige qu'elles soient des fonctions linéaires.

Il faut noter que ces deux modèles portent des avantages et des inconvénients croisés.

–Les SIF de type Mamdani sont bien adaptés au raisonnement qualitatif et leurs règles sont faciles à interpréter. Par contre, leur optimisation automatique à partir de données reste assez délicate et leur précision est limitée.

– Les SIF de type Takagi-Sugeno sont très utilisés pour des problèmes de régression dans lesquels il faut modéliser avec précision une relation d'entrée/sortie donnée. Ils admettent une expression analytique très simple et leur optimisation automatique est facile.

II.3.2. Avantages et désavantages du réglage par logique floue

1. Avantages

Le réglage par logique floue présente les avantages suivants :

- La non nécessité de la modélisation du système.
- La possibilité d'implémenter des connaissances linguistiques de l'opérateur du processus.
- La maîtrise du système à régler avec un comportement complexe (systèmes non linéaires et difficiles à modéliser).
- La disponibilité de système de développement efficace, soit par microprocesseur ou PC, soit par circuits intégrés.

2. Désavantages

- Le manque de directives précises pour la conception d'un réglage (choix de grandeur mesurée, détermination de la fuzzification, des inférences et de la défuzzification).
- L'approche artisanale et asymptotique (implantation des connaissances de l'opérateur souvent difficile).
- L'impossibilité de la démonstration de la stabilité du circuit de réglage en toute généralité (en l'absence d'un modèle valable).
- La cohérence des inférences non garanties (possibilité d'apparition de règles d'inférence contradictoires).

II.4. L'approche neuro-floue

L'utilisation conjointe des réseaux de neurones et de la logique floue permet de tirer les avantages des deux méthodes; les capacités d'apprentissage de la première et la lisibilité et la souplesse de la seconde. Diverses combinaisons de ces deux méthodes ont été développées depuis 1988. Elles ont donné naissance aux ***systèmes neuro-flous***, qui sont le plus souvent orientés vers la commande de systèmes complexes et les problèmes de classification [18].

II.4.1. Motivations pour une approche hybride

Les dernières années ont vu se développer de nombreux travaux sur l'étude conjointe des réseaux de neurones et de la logique floue. Les raisons de cet engouement sont sans doute à rechercher dans les nombreuses similarités qui existent entre les deux approches. Tout d'abord les deux techniques présentent des propriétés d'approximateurs universels de fonction. En effet il a été montré qu'il est possible d'approcher n'importe quelle fonction continue à l'aide d'un système flou ou d'un réseau neuronal classique à couches. On peut aussi trouver des similitudes au niveau de la structure. Il existe en premier lieu une analogie entre les données des règles (IF-THEN) de la logique floue et les couples (entrée-sortie) des réseaux de

neurones. De même on peut trouver un point commun entre les fonctions réalisées par les neurones formels et les fonctions d'appartenance caractérisant les différentes variables d'un système flou. Enfin on peut établir une corrélation entre les opérations de multiplication et d'addition induites par la structure neuronale et celles de maximisation et de minimisation correspondant aux conjonctions et aux disjonctions des règles [19], [20].

De nombreux auteurs ont cherché à tirer une partie de cette complémentarité en présentant des modèles hybrides empruntant des caractéristiques aux deux approches.

II.4.2. Systèmes hybrides neuro-flous [19], [20]

Les réseaux de neurones artificiels et la logique floue peuvent être complémentaires sur plusieurs points. La logique floue permet une spécification rapide des tâches à accomplir à partir de la connaissance symbolique disponible. Le réglage précis du système obtenu et l'optimisation de ses différents paramètres reste néanmoins beaucoup plus difficile dans de nombreux cas. Les modèles les plus courants de RNA, au contraire, n'autorisent pas l'incorporation de connaissance a priori mais permettent de régler par apprentissage le comportement précis du système.

De nombreux auteurs ont donc tout naturellement chercher à combiner ces deux paradigmes depuis le début des années 90 et ceci de plusieurs manières.

Nous portons ici notre attention sur les approches permettant de représenter sous forme de RNA les règles d'un système d'inférence floue. Cette technique consiste à utiliser des fonctions d'activation particulières pour les unités du réseau, ainsi qu'une organisation spécifique de ces dernières et ce afin de reproduire les différents éléments constitutifs d'un système d'inférence floue.

La structure du réseau est ainsi choisie en fonction de la forme de la fonction que l'on cherche à approcher. Les dépendances entre les variables d'entrée et de sortie sont en effet spécifiées par le choix des règles floues. Les différents paramètres de ces règles (c à d forme et position des sous-ensembles flous, sortie et poids des règles), peuvent ensuite être modifiés par un algorithme d'apprentissage supervisé. Nous pouvons noter que cela ne restreint en rien l'ensemble des fonctions approchables puisqu'il a été démontré que la logique floue présente comme les RNA des propriétés d'approximation universelles de fonction.

Si l'on adopte le point de vue des systèmes flous, cette approche permet de régler de manière précise, par apprentissage, le comportement du système réalisé. Si l'on se place au contraire du coté des RNA, la connaissance sous forme de règles floues permet de choisir l'architecture du réseau en fonction de la tâche à accomplir [19], [20].

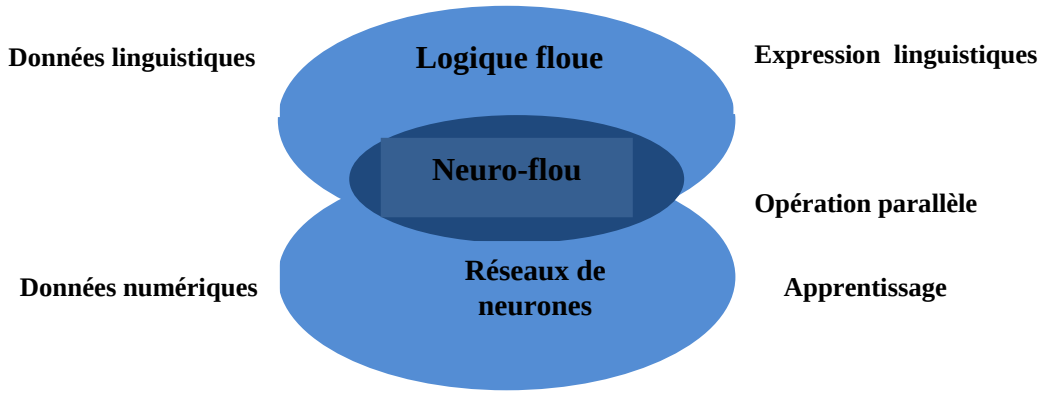


Figure II.14: Principe du système neuro-flou.

La figure II.14 résume le principe du système neuro-flou qui représente l'intersection entre la logique floue et les réseaux de neurones.

Un système neuro-flou est donc défini comme étant un réseau neuronal multicouche avec des paramètres flous, ou comme un système flou mis en application sous une forme distribuée parallèle. Il a été notamment répertorié trois familles différentes :

1. Le modèle FALCON et le modèle GARIC [18]:

Ces deux structures à 5 couches utilisent la fuzzification en entrée et la défuzzification en sortie, correspondant à l'interprétation juste de la technique de Mamdani. Ce sont des structures très précises mais très lentes à l'exécution. Ces modèles sont généralement utilisés pour la commande.

2. Le modèle NEFCLASS [18]:

Modèle utilisé principalement en classification, il est constitué de 3 couches; une couche d'entrée avec des fonctions d'appartenance, une couche cachée représentée par des règles et une couche de sortie définissant les classes. Modèle facile à mettre en application car il évite l'étape de défuzzification tout en étant précis.

3. Le modèle ANFIS [18], [21], [22] :

Modèle le plus utilisé en pratique. C'est une structure à 5 couches qui affine les règles floues déjà établies par des experts humains et réajuste le chevauchement entre les différents sous-ensembles flous. Des applications dans le traitement du signal et le filtrage adaptatif ont été réalisées avec cette architecture.

II.4.3. Système d'inférence floue basé sur les réseaux de neurones adaptatifs

Les réseaux de neurones (RN) multicouches constituent des approximateurs universels. L'atout principal de ces réseaux réside dans leur capacité d'apprentissage. Par contre, leur structure et leurs paramètres n'ont pas toujours des justifications physiques.

De plus, la connaissance humaine ne peut pas être exploitée pour les construire. Les systèmes d'inférence floue sont également des approximateurs universels .Ces systèmes possèdent deux points forts par rapport aux RN. D' une part, ils sont généralement construits à partir de la connaissance humaine, d'autre part, ils ont une capacité descriptive élevée due à l'utilisation de variables linguistiques. Il est donc apparu naturel de construire des systèmes hybrides qui combinent les concepts des systèmes d'inférence floue et des RN. Ainsi l'approche Neuro-floue est apparue [19].

Plusieurs auteurs ont utilisé cette approche. Nous présentons ici un type spécial de réseaux neuro-flous. Il s'agit de l'approche ANFIS (Système d'inférence floue organisé en réseau adaptatif) proposée par Jang [17]. La modélisation découverte par Takagi & Sugeno a trouvé plusieurs applications dans les domaines de contrôle, de prédiction et d'inférence.

Cependant, des aspects de base qui renforcent l'appréhension manquent, à savoir :

- La non-existence des méthodes standards pour transformer l'expérience humaine ou la connaissance humaine en une base de règles et une base de données d'un système d'inférence floue (S.I.F).
- Le besoin de méthodes effectives pour ajuster les fonctions d'appartenance en minimisant l'erreur mesurée en sortie ou en maximisant l'indice de performance.

Dans cette perspective, J.S.R. JANG proposa une nouvelle architecture appelée système d'inférence floue basé sur les réseaux de neurones adaptatifs S.I.F.R.N.A (A.N.F.I.S : ***Adaptative Network Fuzzy Inference System)*** qui peuvent servir comme une base pour construire l'ensemble des règles "Si-Alors" avec des fonctions d'appartenance appropriées en partant d'un ensemble de paires de données (entrées/sorties) du système.

- **Structure d'ANFIS (Adaptative Network Fuzzy Inference System)**

ANFIS met en application un SIF du type Takagi Sugeno et a une architecture composée de cinq couches comme représentée sur la figure II.15.

Pour simplifier la compréhension et sans perte de généralité, nous considérons un système à deux entrées x_1 et x_2 et une sortie y. Considérons aussi un modèle flou de type TSK de ce système, composé des deux règles suivantes:

Si x_1 **est** A_1 **est** x_2 **est** B_1 **alors** $f_1(x_1, x_2) = a_1 x_1 + b_1 x_2 + c_1$

Si x_1 **est** A_2 **est** x_2 **est** B_2 **alors** $f_2(x_1, x_2) = a_2 x_1 + b_2 x_2 + c_2$ (II .5)

Jang a proposé de représenter cette base de règles par le réseau adaptatif de la figure II.15.

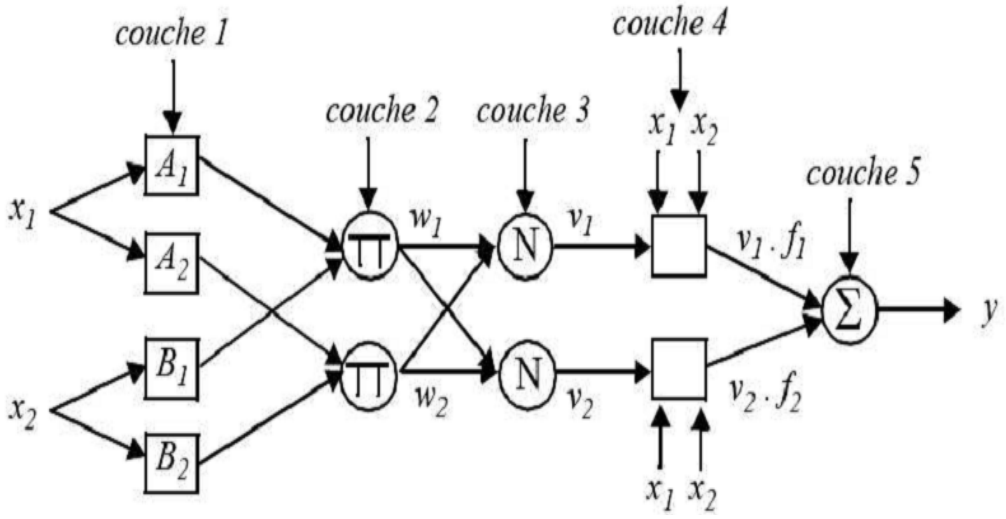


Figure II.15: Architecture équivalente d'ANFIS pour deux entrées à deux règles.

Le réseau adaptatif ANFIS est un réseau multi-couches dont les connexions ne sont pas pondérées, ou ont toutes un poids de 1. Les noeuds sont de deux types différents selon leur fonctionnalité: les nœuds carrés (adaptatifs) contiennent des paramètres, et les nœuds circulaires (fixes) n'ont pas de paramètres. Toutefois, chaque nœud (carré ou circulaire) k applique une fonction sur ses signaux d'entrées.

La sortie O_i^k du nœud i de la couche k (appelée nœud (i, k)) dépend des signaux provenant de la couche k-1 et des paramètres du nœud (i, k) c'est-à -dire,

$$O_i^k = f(o_l^{k-1} \ldots O_{nk-1}^{k-1}), a, b, c \ldots) \quad \text{(II .6)}$$

Où n_{K-1} est le nombre de noeuds dans la couche k-1, et a, b, c ... sont les paramètres du nœud (i, k) . Pour un nœud circulaire ces paramètres n'existent pas.

Dans le réseau de la figure (II.16), les noeuds d'une même couche ont des fonctions issues d'une même famille que nous explicitons ci-dessous.

Couche 1 :

La première couche représente les fonctions d'appartenance floues. Chaque nœud de cette couche est un nœud carré avec une fonction:

$$O_i^l = \mu_{Ai}(x) \quad \text{(II .7)}$$

Où x est l'entrée du nœud i, et A_i le terme linguistique associé à sa fonction. En d'autres termes, O_i^l est le degré d'appartenance de x à A_i . Les paramètres d'un nœud de cette couche sont ceux de la fonction d'appartenance correspondante.

Couche 2 :

Chaque nœud i de cette couche est un nœud circulaire appelé π qui engendre en sortie le produit de ses entrées. Ce produit représente le degré d'activation d'une règle:

$$w_i = \mu_{Ai}(x_1).\mu_{Bi}(x_2) \qquad ,i=1,2 \qquad \text{(II .8)}$$

Couche 3 :

Chaque nœud de cette couche est un nœud circulaire appelé N.

La sortie du nœud i est le degré d'activation normalisé de la règle i:

$$v_i = \frac{w_i}{w_1 + w_2} \qquad \text{(II .9)}$$

- La seconde et la troisième couche contiennent les noeuds qui forment la partie antécédente dans chaque règle

Couche 4 :

La quatrième couche calcule les coefficients de l'équation du premier ordre d'une règle de type Takagi-Sugeno et cela pour chaque règle floue.

Chaque nœud de cette couche est un nœud carré avec une fonction réalisant le calcul:

$$O_i^4 = v_i f_i = v_i (a_i x + b_i x_2 + c_i) \quad , i=1,2 \qquad \text{(II .10)}$$

Où vi est la sortie de la couche 3, et {ai, bi, ci} est l'ensemble des paramètres de sortie de la règle i.

Couche 5 :

La cinquième couche qui représente la couche de sortie, calcule la sortie globale Pesée (pondérée) du système.

Le seul nœud de cette couche est un nœud circulaire qui effectue la somme des signaux provenant de la couche 4, c'est-à -dire,

$$O_l^5 = y = \sum_i v_i f_i \qquad \text{(II .11)}$$

La généralisation du réseau à un système à r entrées ne pose aucun problème particulier. Le nombre de noeuds de la couche 1 est toujours égal au nombre total de termes linguistiques définis.

L'apprentissage à partir d'un ensemble de données concerne l'identification des paramètres des prémisses et des conséquences, la structure du réseau étant fixée.
L'algorithme d'apprentissage commence par construire un réseau initial, ensuite on applique une méthode d'apprentissage par rétro-propagation de l'erreur. Jang a proposé d'utiliser une règle hybride d'apprentissage qui combine un algorithme de descente de gradient avec une estimation par moindres carrées.

En ce qui concerne les paramètres des prémisses (couche 1), Jang a suggéré de n'appliquer une technique d'identification à partir des données que si l'ensemble de données est suffisamment grand. Dans ce cas, l'identification permet d'affiner les fonctions d'appartenance proposées par l'expert humain. Dans le cas contraire, il vaut mieux conserver les fonctions d'appartenance proposées par l'expert car elles reflètent les connaissances de l'expert.
Jang a montré l'équivalence entre les réseaux ANFIS et des réseaux neuronaux classiques avec des connexions pondérées [23].

II.5. Conclusion

Comme on a déjà vu, Les réseaux de neurones sont donc une façon d'élaborer un modèle de connaissance à partir d'une base de données. La mise au point de ce modèle de connaissance se fait par une phase d'apprentissage qui dépend de l'architecture de ce réseau et d'une règle d'apprentissage des poids.

- Le modèle de réseau doit être adapté à la complexité de la tâche à apprendre : doit disposer de paramètres suffisants lui assurant un degré de flexibilité.
- La règle d'apprentissage des poids découle d'un critère d'erreur, sa minimisation itérative conduit à une solution optimale des poids.

Dans cette partie, nous avons introduit aussi les notions de base de la théorie des ensembles flous et de la logique floue. Alors nous pouvons dire que la logique floue ouvre des possibilités remarquables de codification des connaissances des experts. Cependant les applications utilisant la logique floue ne sont pas fondamentalement plus performantes. Elles sont tout simplement plus faciles à réaliser et à utiliser. L'utilisation faite par la logique floue d'expressions du langage courant permet aux systèmes flous de rester compréhensibles pour les personnes non expertes.

Enfin, nous avons présenté la méthode hybride neuro-floue. On peut dire que :

- Cette méthode combine les traitements parallèles et les capacités d'apprentissage des réseaux de neurones avec les raisonnements anthropomorphiques et les capacités d'explication des systèmes flous : Les RNA deviennent plus transparents, les systèmes flous acquièrent la capacité d'apprendre.
- Les réseaux de neurones fonctionnellement équivalents à un modèle d'inférence floue ; on peut l'entraîner à développer des règles floues « SI-ALORS » et à trouver les fonctions d'appartenance de variables d'entrées/sorties en partant d'un ensemble de données représentatives.

Ces réseaux sont bien adaptés pour une implémentation sur des circuits FPGA.

Techniques de poursuite du point de puissance maximale MPPT

III.1. Introduction

Comme on a vu dans le chapitre (I) les caractéristiques *I=f(V)* et *P=f(V)* d'un module solaire montrent bien que la puissance maximale générée dépend fortement de l'intensité des radiations solaires ainsi que de la température. En plus de ces dépendances, le module solaire ne peut générer sa puissance maximale que pour une certaine tension et courant de fonctionnement, la position de ce point de fonctionnement dépendant à son tour de l'ensoleillement et de la température ainsi que de la charge. Pour que le module fournisse sa puissance maximale disponible il faut une adaptation permanente de la charge avec le générateur solaire.

Cette adaptation pourra être réalisée par l'insertion d'un convertisseur DC-DC (hacheur) contrôlé par un mécanisme de poursuite (Maximum Power Point Tracking) pour extraire le maximum de la puissance générée par le module solaire. Ne pas confondre le tracking de puissance avec la notion de tracking du module qui s'occupe de la poursuite de soleil.

Le but de ce chapitre est l'étude des différents mécanismes du tracking pour l'extraction de la puissance maximale sous différentes conditions de fonctionnement. On trouve dans la littérature plusieurs variétés de mécanismes et d'approches allant des méthodes analogiques simples jusqu'aux des méthodes nettement plus efficaces utilisant des algorithmes de contrôle plus sophistiqués.

La figure III.1 montre le diagramme synoptique d'un système photovoltaïque, avec un module MPPT qui a pour entrées la tension et le courant de sortie du module. Dans la plupart des cas on fait une maximisation de la puissance fournie par le module solaire et non la puissance fournie à la charge. Car dans le cas d'un moteur, comme charge, la maximisation de la puissance de charge conduit à la maximisation de la puissance perdue dans les bobines et non seulement la puissance mécanique générée par le moteur, ainsi pour que le mécanisme du MPPT soit indépendant de la charge, il est préférable de maximiser la puissance de sortie du module.

Dans ce qui suit nous allons décrire les différentes techniques MPPT.

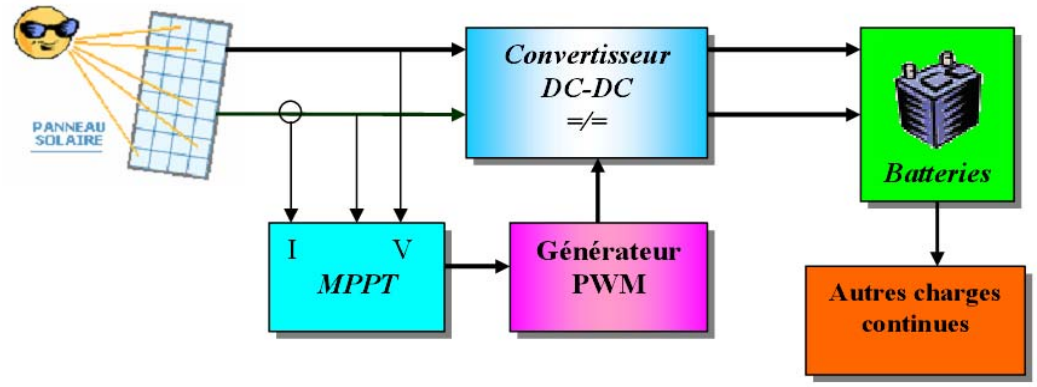


***Figure III.1: Schéma synoptique d'un système photovoltaïque avec MPPT* [24].**

La figure III.2 représente l'influence du MPPT sur les performances d'un module solaire.

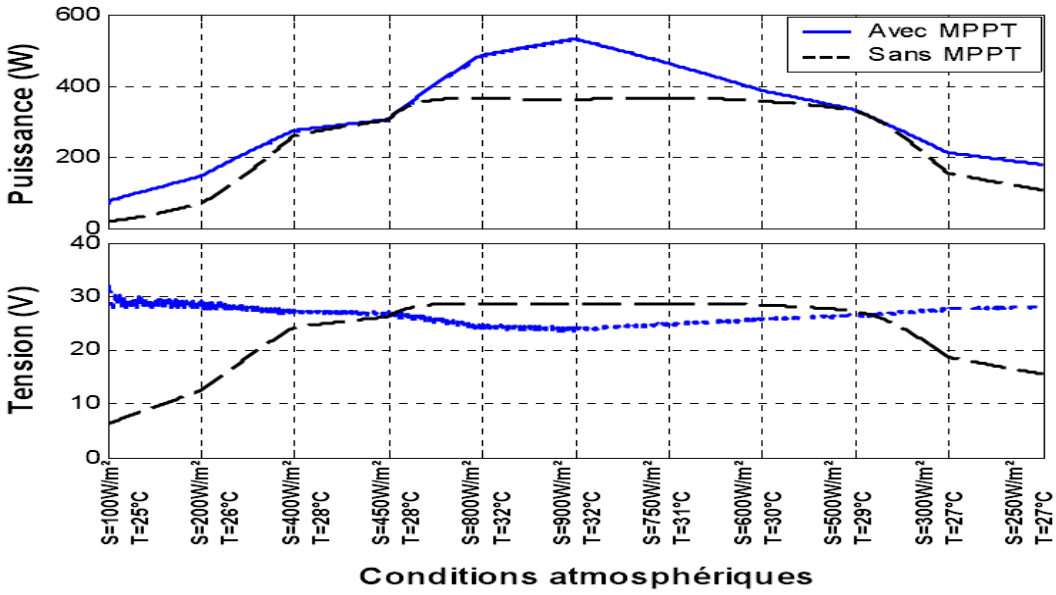


Figure III.2: Puissance et tension de sortie du module solaire avec et sans MPPT (rapport cyclique fixé à 0.85), pour des ensoleillements et des températures variables [24].

III.2. Techniques et méthodes de poursuite du point de puissance maximale

III.2.1. Méthodes avec contre réaction de tension [25,26, 27]

Ce genre de mécanismes repose sur le contrôle de la tension de fonctionnement des panneaux par comparaison de cette tension avec une référence. Cela génère une tension d'erreur qui fait varier le rapport cyclique de la MLI (Modulation Large Impulsion), en anglais PWM (Pulse Width Modulation), de commande afin d'annuler cette erreur comme le montre la figure III.3.

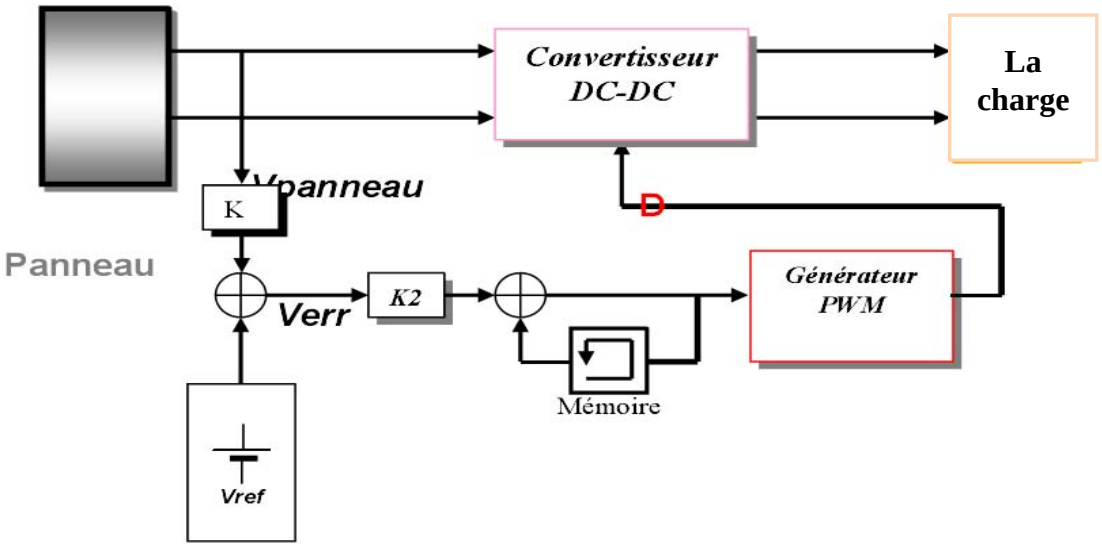


Figure III.3: Méthode avec Contre-réaction de tension avec tension de référence [28].

Cette méthode permet d'utiliser un panneau solaire avec des conditions de charge inconnues ou variables et d'avoir toujours un point souhaitable de fonctionnement pour le panneau.
Son inconvénient est que, le système n'est pas capable de s'adapter dans le cas de changement des conditions environnementales telles que l'insolation et la température.

III.2.1.1. Méthode à tension de référence fixe

C'est la méthode basée sur une simple comparaison entre la tension de sortie du panneau avec une tension de référence prédéfinie, pour ajuster continuellement le rapport cyclique «D» du convertisseur DC-DC. Cette tension correspond à la tension moyenne de l'intervalle des points des puissances maximales, relevées par des tests sous différentes conditions d'ensoleillement et de température en agissant simplement sur les différents facteurs de pondération lors de la mise au point afin de générer le maximum de puissance. A cause de la dépendance de la tension du panneau avec l'ensoleillement et la température, la tension de puissance maximale est déviée, alors la tension de référence doit être corrigée pour différents ensoleillements et températures le long des périodes de l'année.

III.2.1.2. Méthode MPPT avec mesure de V_{OC} du panneau [25,26, 27]

La tension à vide du panneau V_{OC} est mesurée en interrompant le fonctionnement normal du système avec une certaine fréquence, stockant la valeur mesurée, et puis ajustant la tension de référence sur une certaine fraction de la tension du circuit ouvert qui a été expérimentalement déterminée pour être 76% de Voc. Cette tension de référence est comparée à la tension de fonctionnement du panneau et le signal d'erreur résultant est employé comme entrée dans la commande MLI du convertisseur DC-DC.

Cette augmentation permet la commande de la tension de fonctionnement du panneau photovoltaïque avec la considération de facteurs importants comme l'insolation et la température. En plus, le vieillissement et l'accumulation de la poussière sur la surface de cellules sont aussi pris en compte.

L'interruption du fonctionnement du système avec une certaine fréquence engendre des pertes qui sont estimées à 0,05% de la puissance maximale disponible. On estime que des pertes provoquées par l'inexactitude de localisation du point de puissance maximale (MPP) réel sont autour 0,5%. L'interruption fréquente du système provoque une augmentation du bruit électrique.

III.2.1.3. Méthode MPPT avec cellule pilote

Pour éviter les inconvénients liés à l'interruption fréquente de l'exploitation du système, on propose l'utilisation d'une cellule pilote. C'est une cellule photovoltaïque simple qui est électriquement indépendante du reste de la rangée photovoltaïque. Sa tension V_{OC} est constamment mesurée, elle fournit donc l'information implicite du reste des conditions de fonctionnement courantes du panneau. La tension de la cellule pilote est multipliée par un certain facteur constant pour être comparée à la tension de référence de la boucle de contre réaction.

Puisque cette méthode emploie toujours un facteur fixe pour estimer la tension optimale Vop d'une valeur mesurée de V_{OC}, le point de puissance maximale n'est pas toujours vraiment obtenu. Cette technique est fondée sur la connaissance des caractéristiques du panneau photovoltaïque pour estimer le lieu du point de puissance maximale.

III.2.2. Méthodes à contre réaction de puissance [25,26, 27]

Ces méthodes sont basées sur des algorithmes de recherche itérative pour trouver le point de fonctionnement du panneau pour que la puissance générée soit maximale sans interruption de fonctionnement du système. Elles ne sont pas basées sur des valeurs de références prédéfinies ou à partir des paramètres opérationnels, mais sur la maximisation permanente de la puissance générée par les panneaux solaires. Ainsi pour un point donné on fait la recherche dans un certain sens, si on a une augmentation de la puissance du panneau alors on maintient cette direction de recherche, sinon on cherche dans le sens opposé.

La puissance extraite du panneau est calculée à partir des mesures de courant ***I*** et de tension ***V*** du panneau et la multiplication de ces deux grandeurs $P = I * V$. Le sens de variation de la puissance P est connu par le calcul de manière approximative de la dérivée

$dp \approx \Delta p(k) = p(k) - p(k-1)$ sur un temps d'échantillonnage qui représente la vitesse d'exécution du microprocesseur ou du microcontrôleur.

III.2.2.1. Méthode de Perturbation et Observation (P&O) [25, 26,27]

Cette méthode à la particularité d'avoir une structure de régulation simple, et peu de paramètre de mesure. Elle opère en perturbant périodiquement la tension du panneau, et en comparant l'énergie précédemment délivrée avec la nouvelle après perturbation. Si la perturbation (l'ajout d'un dV de tension) implique une augmentation de la puissance alors nous nous trouvons dans la phase ascendante de la courbe de puissance, donc la tension de sortie devra être augmentée et vise versa. Dans ces conditions le ''tracker '' cherche en permanence le maximum de puissance.

III.2.2.2. Algorithme d'incrémentation de l'inductance (Incremental Conductance) [25, 26,27]

Dans cet algorithme la dérivée de la puissance de sortie du panneau est calculée d'une autre manière. Elle est calculée en fonction de la tension V et sa différence dV et du courant I et sa différence dI. Cette dérivée est nulle au point de puissance maximale, positive à gauche du point MPP et négative à droite.

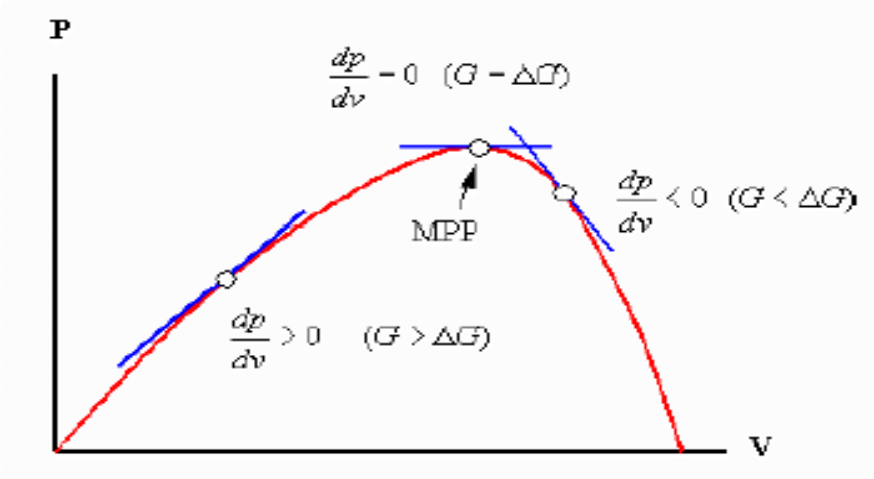


Figure III.4: Signe de dP/dV pour différentes zones de fonctionnement

La puissance du panneau solaire est donnée par :

$$P = I * V \qquad \text{(III .1)}$$

La dérivé partielle $\frac{dP}{dV}$ est donnée par :

$$\frac{dP}{dV} = I + V \frac{dI}{dV} \qquad \text{(III .2)}$$

$$\frac{1}{V}\frac{dP}{dV} = \frac{I}{V} + \frac{dI}{dV} \qquad \text{(III .3)}$$

On définit la conductance de la source $G = \frac{I}{V}$ et l'incrémentale conductance $\Delta G = \frac{\Delta I}{\Delta V}$.

Puisque la tension V du panneau est toujours positive, la relation (III.3) explique que le point de puissance maximale MPP est atteint si la conductance de la source G égale l'incrémentale conductance ΔG de la source avec un signe moins, et qu'elle est à gauche de ce point lorsque la conductance G est supérieure à l'incrémentale conductance ΔG et vice-versa, comme suit :

$$\begin{cases} \frac{dP}{dV} \succ 0 & \text{SI} \quad \frac{I}{V} \succ -\frac{dI}{dV} \\ \frac{dP}{dV} = 0 & \text{SI} \quad \frac{I}{V} = -\frac{dI}{dV} \\ \frac{dP}{dV} \prec 0 & \text{SI} \quad \frac{I}{V} \prec -\frac{dI}{dV} \end{cases} \qquad \text{(III .4)}$$

Cette méthode est plus efficace que la méthode de perturbation, et indépendante des caractéristiques des différents composants utilisés.

Les tensions et courants du panneau sont monitorés, de telle manière que le contrôleur peut calculer la conductance et la conductance incrémentale, et décider de son comportement. Cet algorithme implique un nombre important de calculs de dérivées.

III.2.3. Méthode avec contre réaction du courant [25,26, 27]

Dans toutes les méthodes MPPT présentées jusqu'ici, une mesure de I et V a été employée pour obtenir des informations sur la puissance actuelle de sortie du panneau. Basé sur cette information le mécanisme MPPT ajuste la tension V de sortie du panneau pour déplacer le point de fonctionnement des piles solaires plus près de leur MPP.

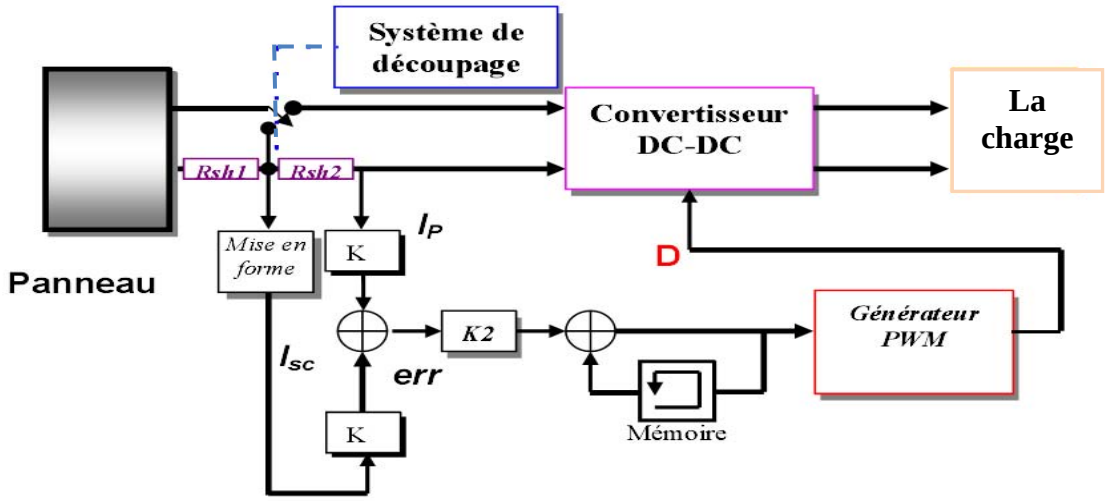


Figure III.5: Méthode de la contre réaction du courant [28].

Cette nouvelle méthode MPPT utilise seulement une mesure de courant pour obtenir l'information sur le point opérationnel actuel du système.

L'avantage de cette méthode est qu'elle est simple, contrairement aux méthodes mentionnées précédemment.

III.2.4.Technique de balayage du courant (current sweep)[29]

Cette méthode utilise des formes d'onde du courant de panneau PV, sachant que la caractéristique I-V du panneau est obtenue et fixée à des intervalles du temps bien précis , on peut donc calculer la tension V_{MPP} de la caractéristique de la courbe pendant les mêmes intervalles du temps.

La fonction choisie pour ces formes d'onde est directement proportionnelle à sa dérivée

$$f(t) = k \frac{df(t)}{dt} \quad \text{(III .5)}$$

k est la constante de proportionnalité, la puissance du panneau PV est donnée par :

$$p(t) = v(t)i(t) = v(t)\, f(t) \quad \text{(III .6)}$$

Au point MPPT :

$$\frac{dp}{dt} = v(t)\frac{df(t)}{dt} + f(t)\frac{dv(t)}{dt} = 0 \quad \text{(III .7)}$$

substituant (III.5) de (III.7) :

$$\frac{dp}{dt} = \left[v(t) + k \frac{dv(t)}{dt} \right] \frac{df(t)}{dt} = 0 \quad \text{(III .8)}$$

L'équation différentielle (III.5) a comme solution :

$$f(t) = C \exp\left[t / k \right] \quad \text{(III .9)}$$

'C' est choisi pour être égal au courant maximal I_{MAX} du panneau PV et k négative, le résultat est la diminution de la fonction exponentielle avec un temps constant $\tau = -k$

$$f(t) = I_{max} \exp\left[-t / \tau \right] \quad \text{(III .10)}$$

Le courant en (III.10) peut être facilement obtenu en employant une décharge du courant par un condensateur ; pour que la dérivée de (III.10) ne soit pas nulle, l'équation (III .8) peut être devisé par $df(t)/dt$ est la même équation donne alors :

$$\frac{dp(t)}{di(t)} = v(t) + k \frac{dv(t)}{dt} = 0 \quad \text{(III .11)}$$

Cette technique de MPPT est seulement faisable si la consommation de puissance pendant le tracking du MPP contrôle est inférieure à l'augmentation de la puissance qu'elle peut apporter à tout le système PV [28].

III.2.5.Techniques intelligentes pour la commande MPPT

Dans la partie précédente, les méthodes classiques et les mécanismes de poursuite de puissance maximale les plus rencontrées dans la littérature ont été présentés. Cette partie décrit des techniques intelligentes nouvellement introduites dans le monde de contrôle des MPPT. Il s'agit principalement des réseaux de neurones artificiels et les différentes structures de

commande qui leurs sont associées, la logique floue, les systèmes neuro-flous et, à un degré moindre, les algorithmes génétiques(AGs) lesquels font actuellement leur apparition dans la commande électrique.

III.2.5.1. MPPT par les algorithmes génétiques [15], [30]

Les algorithmes génétiques sont des algorithmes d'optimisation stochastiques fondés sur les mécanismes de la sélection naturelle et de la génétique, inspirés de la théorie de survie de Darwin. En raison des performances remarquables des AGs, ces derniers ont envahi plusieurs domaines de recherche dans lesquels ils ont apporté des satisfactions appréciables en raison de leurs avantages à savoir: la rapidité et la possibilité de résoudre des équations non linéaires à plusieurs variables.

Les algorithmes génétiques comptent parmi les approches intelligentes utilisées pour la poursuite du point de puissance maximale (PPM) des cellules photovoltaïques pour pouvoir ainsi profiter au maximum de l'énergie solaire et suivre rapidement ce point.

Beaucoup de travaux ont montrés que les AGs donnent des résultats très intéressants dans lesquels la convergence est assurée, avec des temps de calculs et des opérations simples.

Néanmoins, les algorithmes génétiques présentent un inconvénient qui réside dans l'utilisation des relations de probabilité rendant ainsi le résultat parfois imprévisible et les performances moins contrôlées [15], [30].

III.2.5.2. MPPT par la logique floue [14], [31]

Cette méthode utilise la logique floue afin de rendre plus rapide la réponse du contrôleur et d'augmenter la stabilité du système une fois le point MPP atteint.

La poursuite du point MPP sera divisée en deux phases : la première sera la phase de recherche rude, avec un pas de recherche important pour améliorer la réponse du contrôleur MPPT, la seconde sera une phase fine où le pas sera très petit, ce qui garantira une stabilité du système et la diminution au maximum des oscillations autours du point MPP. Cette caractéristique du contrôleur flou illustre son efficacité et le place parmi les meilleurs dispositifs de poursuite MPPT.

Le contrôleur flou se compose de trois blocs : l'opération de ***fuzzification*** des variables d'entrée est effectuée dans le premier bloc, elle permet le passage du domaine réel au domaine flou, le deuxième bloc est consacré aux ***règles d'inférence***, alors que le dernier bloc est le siège de l'opération de ***défuzzification*** permettant le retour vers le domaine réel. Cette dernière opération utilise la méthode du centre de masse pour déterminer la valeur de la sortie.

La figure suivante donne la structure de base du contrôleur flou utilisé.

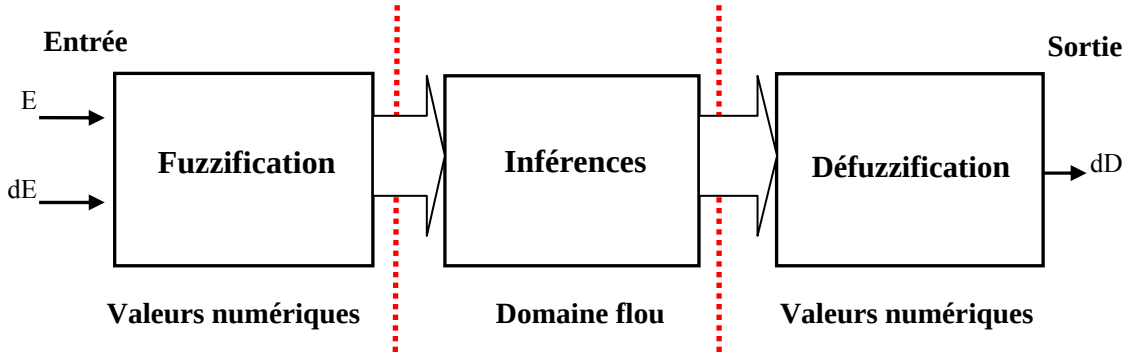


Figure III.6: Structure de base du contrôleur MPPT flou.

En utilisant la boite à outil Fuzzy sous Simulink, le contrôleur flou est représenté globalement par la figure suivante :

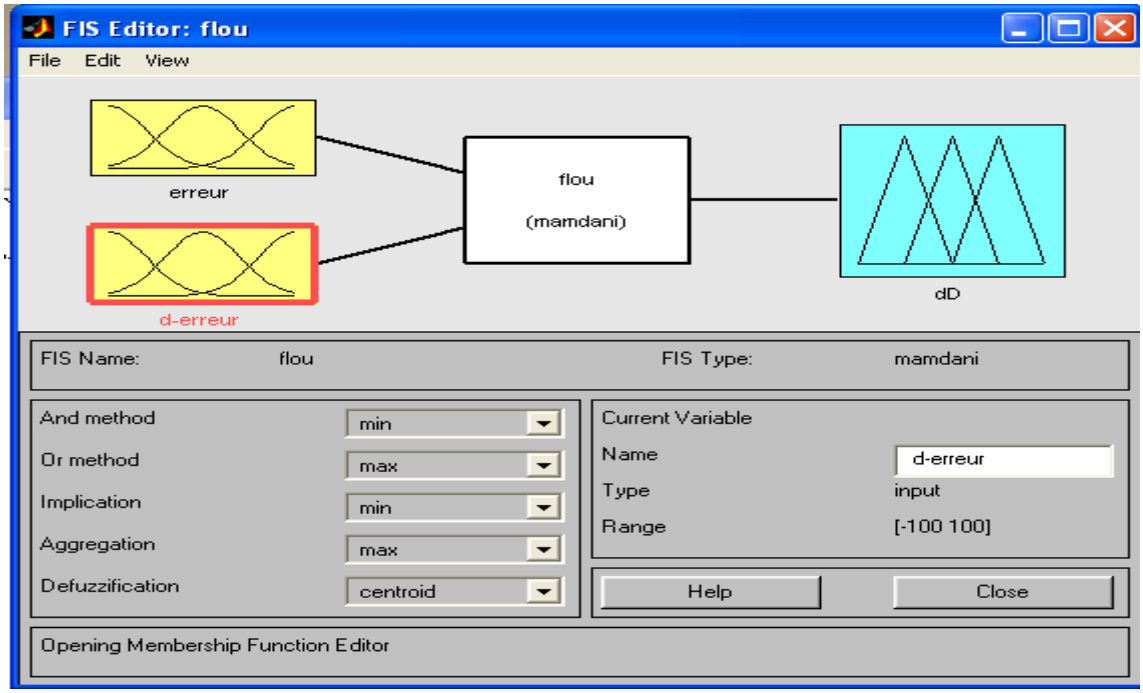


Figure III.7: Schéma général du contrôleur MPPT flou sous MATLAB.

1. Fuzzification

Tout d'abord, les valeurs instantanées du courant et de la tension de sortie du générateur photovoltaïque sont mesurées par un convertisseur analogique numérique (CAN), la puissance à l'instant actuel est obtenue par la multiplication de ces deux grandeurs ***:***

$$P(k) = V(k).I(k) \qquad \text{(III .12)}$$

Les variables d'entrées sont l'erreur sur la puissance (E) et la dynamique de l'erreur (dE), on donne leurs expressions respectives :

$$E(k) = \frac{p(k) - p(k-1)}{V(k) - V(k-1)} \quad \text{(III .13)}$$

$$dE(k) = E(k) - E(k-1) \quad \text{(III .14)}$$

La valeur de l'entrée $E(k)$ nous indique de quel coté est situé le point de fonctionnement par rapport au point de puissance maximale MPP (figure III.4).

- Si *E(k)* est positif, le point de fonctionnement est a gauche du point de puissance maximale.
- Si *E(k)* est négatif, le point de fonctionnement est a droite du point de puissance maximale.
- *Si E(k)* est Zéro au point de puissance maximale (figure III.4).

L'entrée $dE(k)$ montre la direction du point de fonctionnement.

Dans ce cas, pour atteindre le point de puissance maximale on doit:

- Pousser vers l'adroite le point de fonctionnement qui se situe à gauche du point maximal ce qui veut dire qu'on doit augmenter la tension.
- Pousser vers la gauche le point de fonctionnement qui se situe à droite du point maximal ce qui veut dire qu'on doit diminuer la tension.

Ces deux entrées nous permettent de prendre la décision appropriée sur la valeur de la sortie $dD(k)$ du pas du rapport cyclique. L'augmentation du rapport cyclique du convertisseur $DC-DC$ permet d'augmenter la tension du point de fonctionnement et inversement.

On applique la théorie des ensembles flous aux entrées-sorties, et on exprime chacune d'entre elles par cinq fonctions d'appartenance qui sont :

NG: négatif grand, **NP**: négatif petit, **ZE**: zéro, **PP**: positif petit, **PG**: positif grand.

Le choix de cette classification est basé sur le raisonnement suivant :

On travaille sur plusieurs phases de poursuite, la première est rude où on utilise un pas de recherche important pour diminuer le temps de réponse.

Une fois le point est proche du MPP, on utilise un pas plus faible pour diminuer les amplitudes des ondulations, c'est la phase fine.

Donc on choisit des variables linguistiques pour savoir si le point de fonctionnement est loin ou proche du MPP, afin d'arriver le plus vite possible à ce point.

Les figures III.8, III.9 et III.10 montrent les fonctions d'appartenance des cinq sous ensembles flous des variables d'entrées et de la sortie.

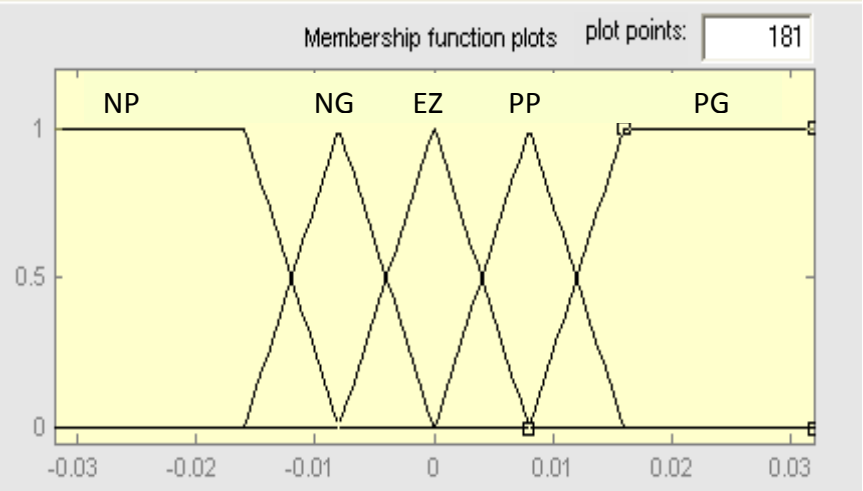


Figure III.8: Fonctions d'appartenance des variables d'entrée E.

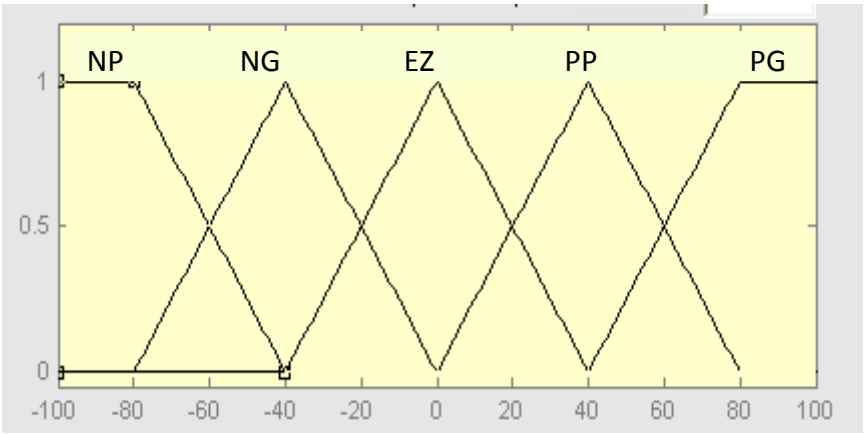


Figure III.9: Fonctions d'appartenance des variables d'entrée dE.

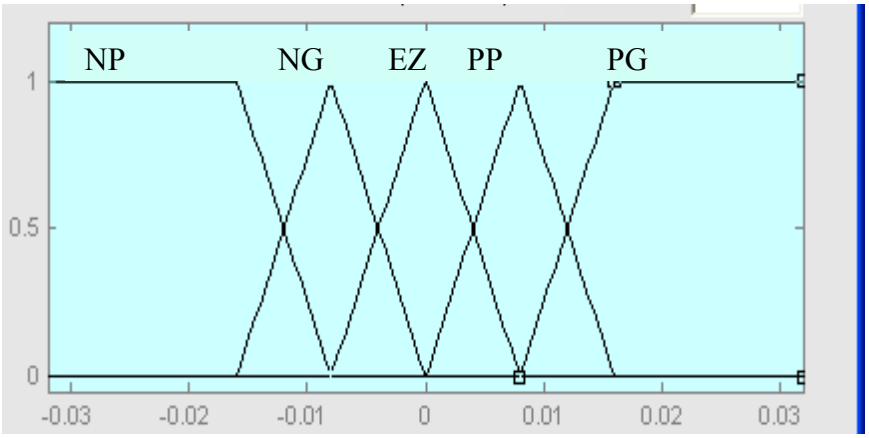


Figure III.10: Fonctions d'appartenance de variable de sortie dD.

2. Inférences

La méthode d'inférence utilisée pour ce contrôleur est celle de «Mamdani», c'est généralement la méthode la plus utilisée. Elle utilise l'opérateur MIN pour le «ET» et l'opérateur MAX pour le «OU».

Les règles d'inférence permettent de prendre la bonne décision sur la sortie dD à partir des valeurs des entrées E et dE.

Le tableau III.1 résume les règles d'inférence du contrôleur flou MPPT utilisé [14].

E ↓ dE →	NP	NG	ZE	PP	PG
NP	ZE	ZE	PG	PG	PG
NG	ZE	ZE	PP	PP	PP
ZE	PP	ZE	ZE	ZE	NP
PP	NP	NP	NP	ZE	ZE
PG	NG	NG	NG	ZE	ZE

Tableau III.1: Table d'inférence du régulateur flou.

On prend comme exemple la règle suivante : Si E est NG et dE est PP alors dD est PG.

On explique dans ce qui suit cette règle : si E(k) est négatif grand (NG) et dE(k) est positif petit (PP), cela veut dire que le point de fonctionnement est à droite du point MPP, et en plus il est éloigné de ce dernier. La décision à prendre sera alors d'incrémenter le rapport cyclique d'un grand pas, c'est-à-dire dD sera positif grand (PG).

Les règles sont liées entre elles par un « OU » logique, et on calcule donc le maximum entre les fonctions d'appartenance résultantes pour chaque règle. L'ensemble flou global de la sortie dD est construit par agrégation des ensembles flous obtenus par chacune de ces règles.

3. Défuzzification

A partir de l'ensemble flou obtenu par les règles d'inférences, on doit obtenir une seule valeur de la variable de commande dD. L'opération de défuzzification est réalisée par la méthode du centroïde (Figure III.11). Elle permet de calculer le centre de gravité de la surface obtenue par l'expression suivante :

$$dD_0 = \frac{\sum_{i=1}^{n} D_i \cdot \mu(D_i)}{\sum_{i=1}^{n} \mu(Di)} \qquad \text{(III .15)}$$

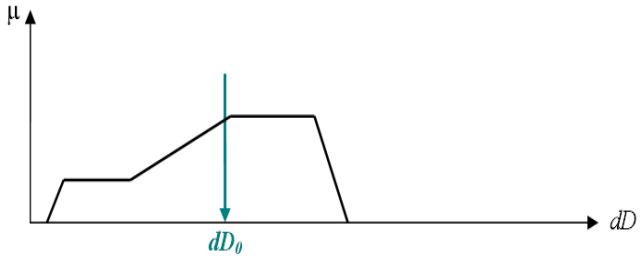


Figure III.11: Défuzzification de la sortie dD par la méthode du centre de masse

III.2.5.3. MPPT par les réseaux de neurones

Avec la venue des contrôleurs par la logique floue, une autre technique MPPT par les réseaux de neurones est apparue.

Les réseaux de neurones ont généralement trois couches : couche d'entrée, couches cachées et couche de sortie comme montré dans le chapitre II. Les variables d'entrées peuvent être des paramètres du panneau photovoltaïque tel que V_{OC} et I_{SC}, données atmosphériques comme l'irradiation et la température, ou toute combinaison de ces dernières.

La sortie est habituellement le signal du rapport cyclique utilisé pour commander le hacheur MPPT.

La figure III.12 représente un exemple d'une architecture d'un réseau de neurones proposé pour la commande d'un convertisseur MPPT [32].

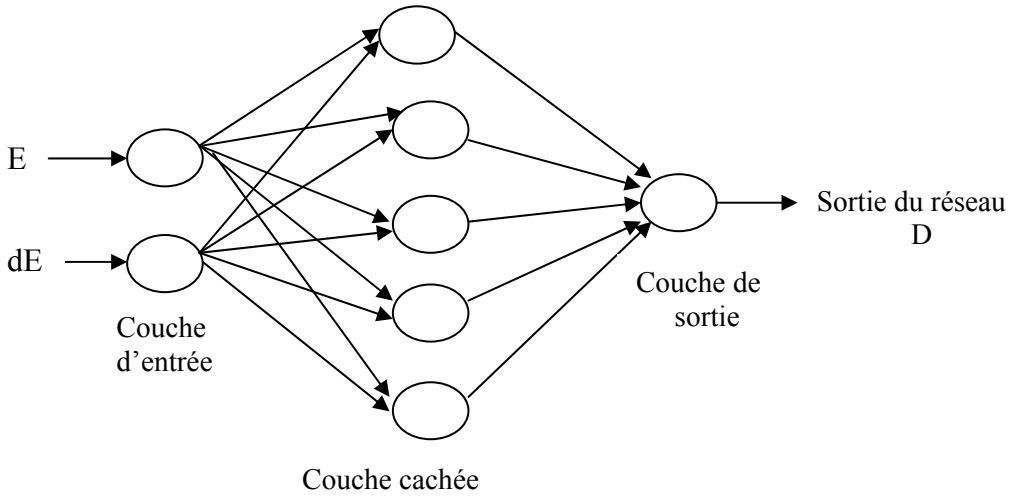


Figure III.12: Architecture d'un réseau de neurone proposé pour une commande MPPT [32].

Sachant que les variables « E» et « dE » des entrées et la variable de sortie « D » sont déjà expliquées dans la section précédente, ce réseau de neurones est constitué de :

- Une couche d'entrée composée de deux neurones, dont le rôle est de transmettre les valeurs des entrées qui correspondent aux variables (E, dE) vers la couche suivante appelée couche cachée.
- Une couche cachée dotée de cinq neurones avec des fonctions d'activations choisies de type sigmoïde tangentielle .
- Une couche de sortie avec un seul neurone représentant la commande "D"**.**

Le nombre de neurones de la couche cachée a été optimisé empiriquement durant la phase d'apprentissage. En effet, les essais ont montré que la structure la plus stable est celle composée de cinq neurones [32].

Il est également à noter que le choix de la fonction d'activation de la couche cachée n'a pas été adopté de manière arbitraire, mais a été choisie après plusieurs tests qui ont montré que la fonction de type sigmoïde tangentielle converge plus rapidement par rapport à la fonction sigmoïde exponentielle durant la phase d'apprentissage.

En plus, deux autres paramètres doivent être également optimisés à savoir : le nombre d'époques qui représente la durée d'apprentissage, ainsi que la performance de l'apprentissage qui est l'écart minimum entre les valeurs présentées au réseau , ou 'Patrons entrées-sorties' et les valeurs obtenues. Cet entraînement a été fait en utilisant l'algorithme de rétro-propagation qui sera détaillé ultérieurement.

La figure III.13 représente le schéma synoptique d'un système photovoltaïque doté d'une commande MPPT à base des réseaux de neurones artificiels [32].

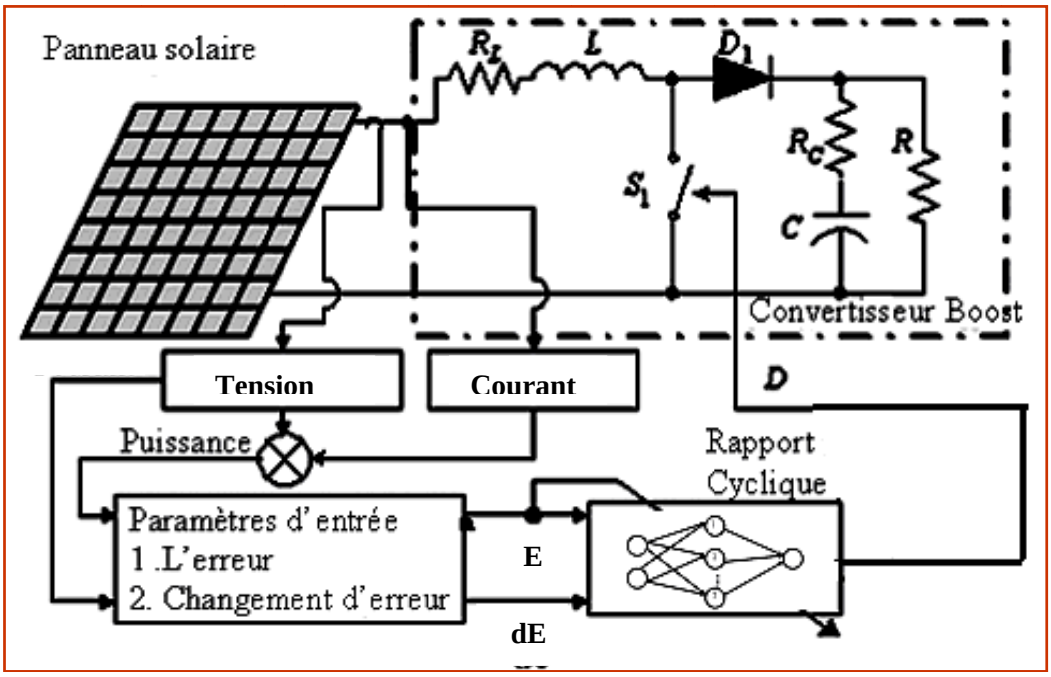


Figure III.13: Schéma synoptique d'un système photovoltaïque avec une commande MPPT par réseaux de neurones.

III.2.5.4. MPPT par l'approche neuro-floue

III.2.5.4.1. Conception d'un contrôleur MPPT à base de réseau neuro-flou

La figure III.14 représente le schéma synoptique d'un système photovoltaïque doté d'une commande MPPT à base de réseaux neuro-flous.

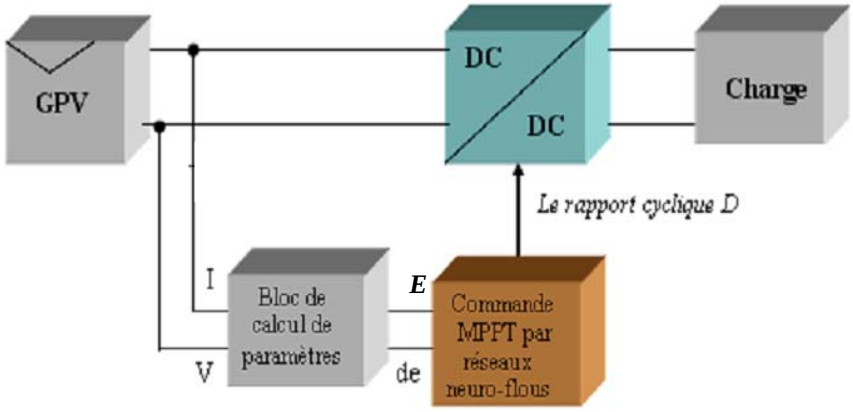


Figure III.14: Schéma synoptique d'un système photovoltaïque avec une commande MPPT par réseau neuro-flou (ANFIS).

III.2.5.4.2. Description et structure du contrôleur MPPT neuro-flou proposé

Le contrôleur neuro-flou développé comprend deux entrées ' E' et 'dE' et une seule sortie ' D' qui représentent respectivement l'erreur, la variation de l'erreur et la commande. Les deux variables d'entrées génèrent l'action de contrôle 'D' qui sera appliquée au hacheur, afin d'ajuster le rapport cyclique de ce dernier de telle manière à assurer l'adaptation de la puissance fournie par le GPV.

Le contrôleur neuro-flou est un contrôleur ***ANFIS*** qui met en application un système d'inférence flou (SIF) du type Takagi Sugeno et a une architecture composée de cinq couches comme représentée sur la figure III.15. Ce contrôleur permet une génération automatique de règles floues basées sur le modèle d'inférence de Sugeno :

Si E est A_1 et dE est B_1 Alors $d_1=f(E, dE)$
Si E est A_2 et dE est B_2 Alors $d_2= f(E, dE)$
.
.
.
Si E est A_{25} et dE est B_{25} Alors $d_{25}=f(E, dE)$

Où ***E, dE*** sont des variables d'entrées, ***$A_1, A_2, \ldots, A_5$ et $B_1, B_2, \ldots B_5$*** sont des ensembles flous.

Lorsque ***d_i= constante***, on obtient un modèle de Sugeno d'ordre zéro.

Le conséquent d'une règle est un singleton.

Lorsque d est une combinaison linéaire des entrées ***: $d= a E + b dE + c$***, on obtient un modèle de Sugeno de premier ordre.

La figure III.15 montre un exemple d'application de deux règles en utilisant la méthode de défuzzification " moyennes pondérées " ou encore "weighted average».

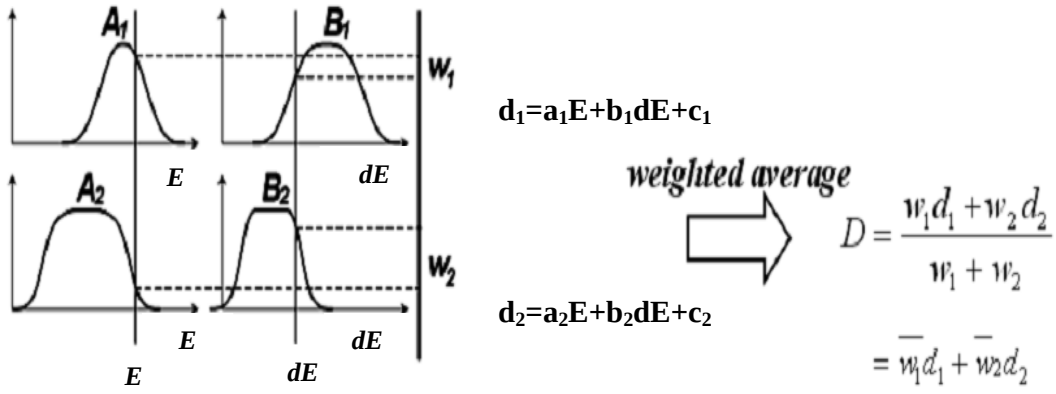


Figure III.15: Exemple de défuzzification par la méthode weighted average.

La formule du rapport cyclique D est donnée donc par l'équation III.16.

$$D = \frac{w_1 d_1 + w_2 d_2 + \ldots\ldots w_{25} d_{25}}{w_1 + w_2 + \ldots\ldots w_{25}} \qquad \text{(III.16)}$$

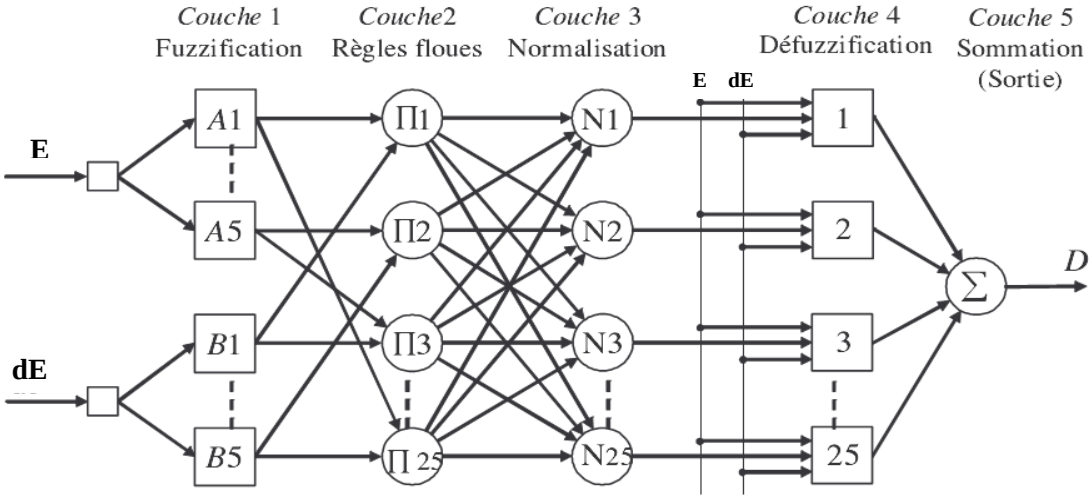


Figure III.16: Architecture du modèle ANFIS proposé.

Les nœuds d'entrées « E » et « dE » transmettent simplement les données d'entrée à la couche de fuzzification.

Les cinq couches du réseau neuro-flou ont pour rôle :

Couche 1 :

Les neurones de cette couche réalisent les ensembles flous qui serviront dans les antécédents des règles. Dans le modèle de Jang, les fonctions d'appartenance sont des gaussiennes.

Couche 2 :

Chaque neurone dans cette couche correspond à une règle floue Sugeno. Il reçoit les sorties des neurones de fuzzification et calcule son activation.

La conjonction des antécédents est réalisée avec l'opérateur produit.

Couche 3 :

Chaque neurone calcule le degré de vérité normalisé d'une règle floue donnée. La valeur obtenue représente la contribution de la règle floue au résultat final.

Couche 4 :

Chaque neurone i de cette couche est relié à un neurone de normalisation correspondant et aux entrées initiales du réseau. Il calcule le conséquent.

Couche 5 :

Comprend un seul neurone qui fournit la sortie de ANFIS en calculant la somme des sorties de tous les neurones de défuzzification

La structure neuronale équivalente proposée sous Matlab est représentée à la figure III.17.

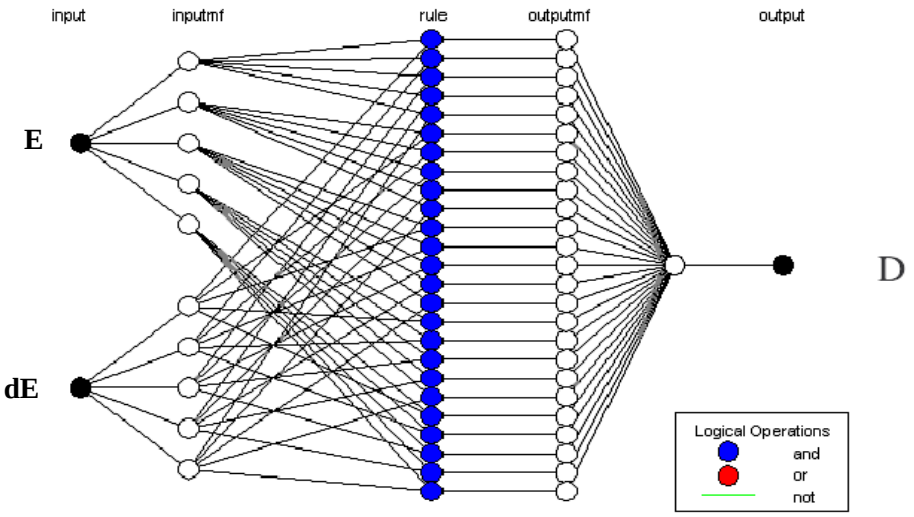


Figure III.17: Structure neuronale du modèle proposé sous Matlab.

III.2.5.4.3. Apprentissage du contrôleur "Entraînement d'un réseau ANFIS "

L'apprentissage du contrôleur est effectué par le biais de l'algorithme de rétro propagation, afin de déterminer les paramètres des prémisses (ajustement des paramètres liés aux fonctions d'appartenance) et l'estimation des paramètres conséquents par la méthode des moindres carrés. Ce qui a pour appellation "apprentissage hybride".

L'entraînement du réseau ANFIS est donc effectué par un algorithme à deux temps où on estime d' abord les paramètres des conséquents par une technique de moindres carrés et ensuite les poids du réseau par une descente de gradient.

Chaque époque d'entraînement comprend une passe avant et une passe arrière.

Durant la passe avant, les patrons d'entrée servent à déterminer les sorties des neurones couche par couche, permettant de déterminer les valeurs des paramètres de conséquents en bout de compte. Durant la passe arrière, l'algorithme de rétro-propagation d'erreur est appliqué pour régler les poids des différentes couches.

Lors de la passe arrière, l'algorithme de rétro propagation d'erreur est appliqué pour mettre à jour les poids des antécédents des règles.

Dans l'algorithme ANFIS de Jang, on optimise aussi bien les paramètres des antécédents que ceux des conséquents. Durant la passe avant, les paramètres des conséquents sont adaptés alors que les paramètres des antécédents sont maintenus constants ; durant la passe arrière, les rôles sont échangés.

- **Utilisation de l'ANFIS comme modèle de base pour la poursuite du point de puissance maximale**

Une fois que le vecteur (d'entrée-sortie) des données mesurées est obtenu, et comme ANFIS fait partie de la boite à outil "logique floue" du logiciel Matlab, il a été utilisé efficacement pour construire l'ensemble des règles floues « ***Si Alors »*** avec leurs fonctions d'appartenance appropriées. La manière dont ANFIS fonctionne est décrite dans l'organigramme de la figure III.18.

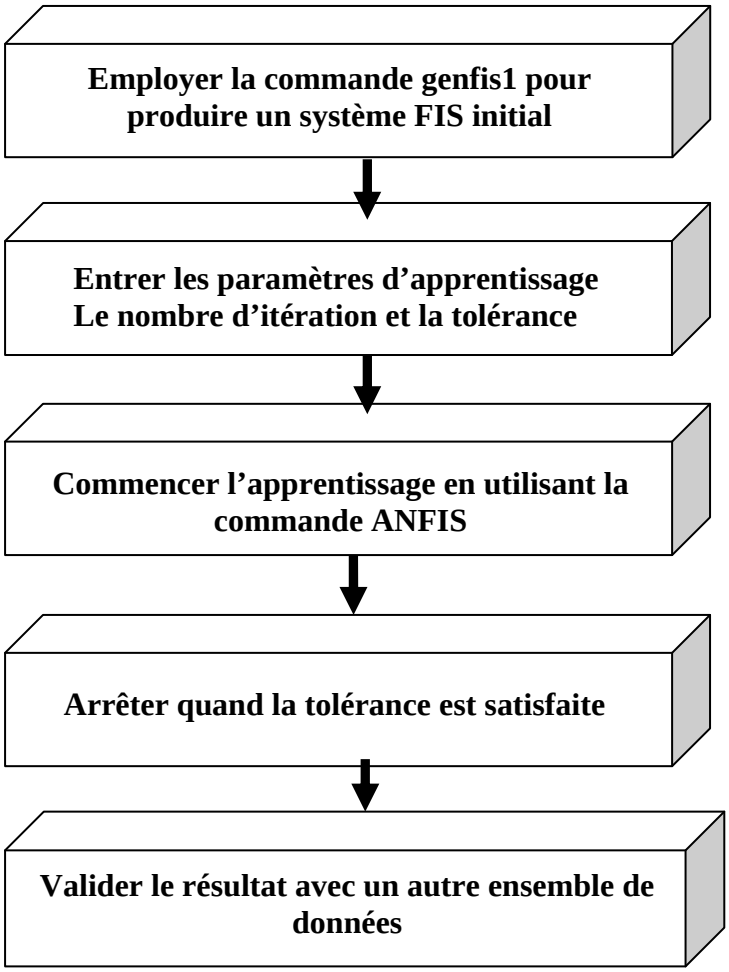


Figure III.18: Organigramme du processus de calcul d'ANFIS dans Matlab.

Le contrôleur MPPT neuro-flou présenté ci dessus est programmé et simulé en utilisant la boite à outil Fuzzy sous Simulink représenté globalement par la figure III.19.

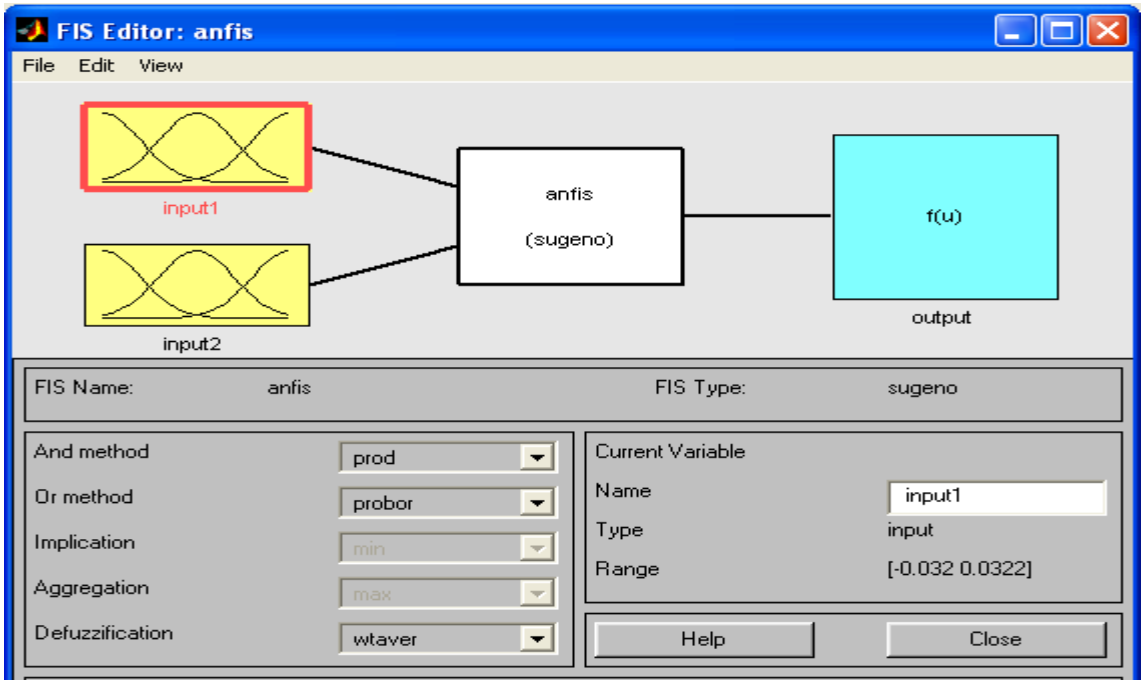


Figure III.19: Schéma général du contrôleur MPPT neuro-flou proposé sous Matlab.

Il est important de noter qu'il n'existe pas une loi pour déterminer la table des règles et le nombre de fonctions d'appartenance assigné à chaque variable d'entrée.

Généralement, ce nombre est choisi empiriquement en examinant son influence sur la réaction du système. Le résultat du processus d'apprentissage est montré sur la figure III.20.

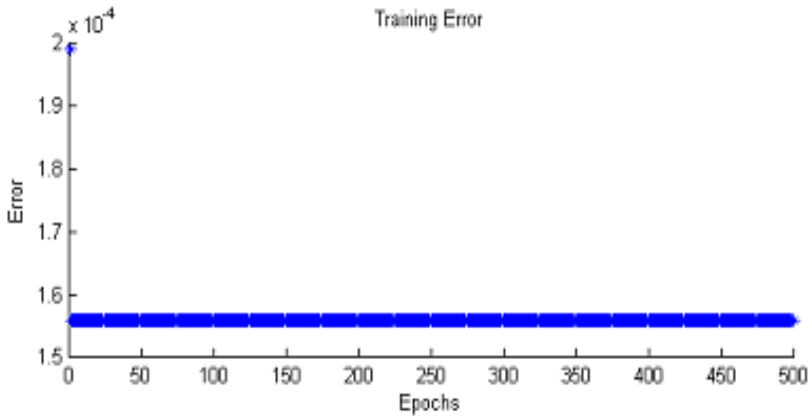


Figure III.20: Entraînement du réseau neuro-flou (avec un nombre d'itération de 500).

- **Test de la capacité de généralisation après apprentissage**

Après apprentissage du système neuro-flou adaptatif (ANFIS), nous avons testé sa généralisation, en présentant des données de validation (Checking Data).

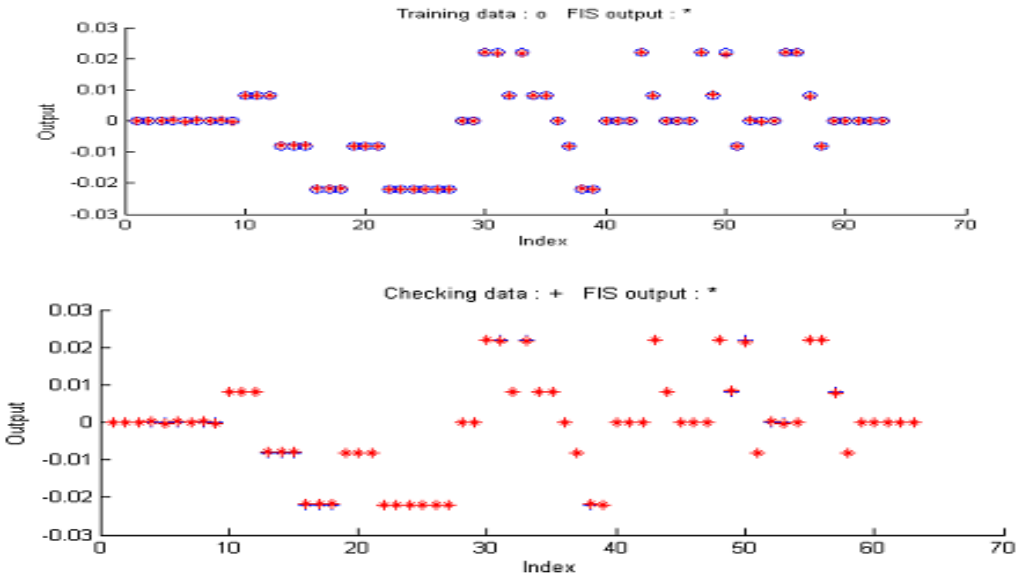


Figure III.21: Test de la capacité de généralisation après apprentissage.

Pour déterminer le pas d'apprentissage et les fonctions d'apprentissage, nous avons élaboré un programme susceptible de les déterminer.

Les résultats obtenus sont illustrés sur les figures III.22, III.23.

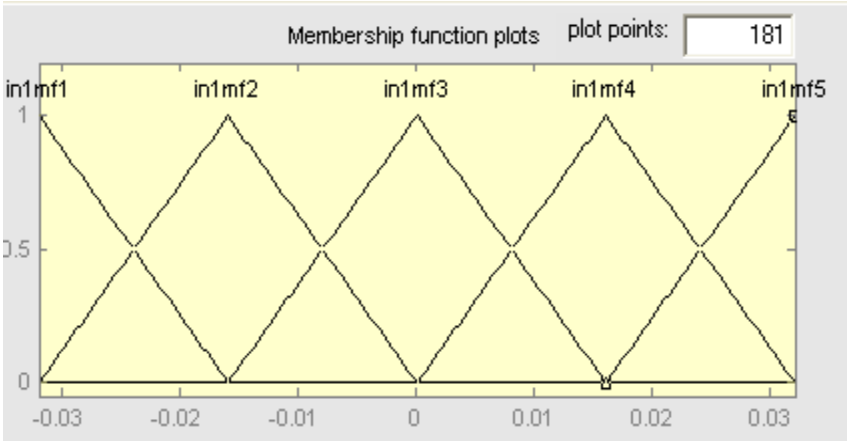


Figure III.22: Les fonctions d'appartenance de l'erreur "E" générée par ANFIS après apprentissage.

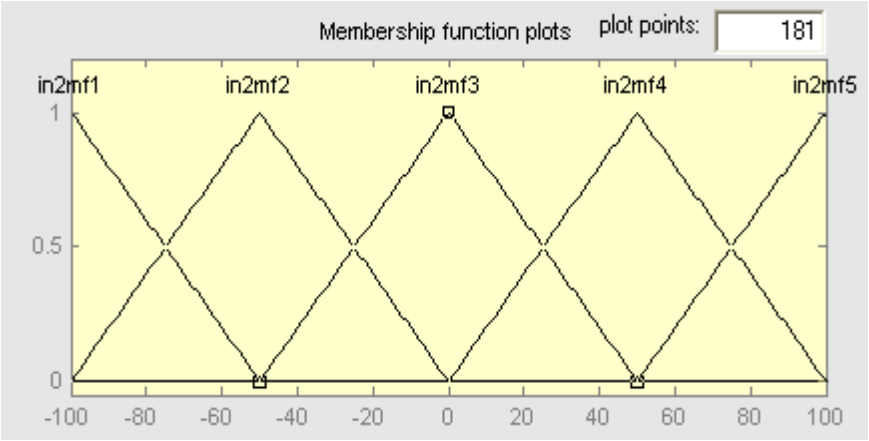


Figure III.23: Les fonctions d'appartenance du changement de l'erreur "dE" générées par ANFIS après apprentissage

Remarque : L'ANFIS génère des fonctions d'appartenance de l'erreur et le changement d'erreur, après apprentissage, selon plusieurs formes (gaussienne, triangulaire, rectangulaire…..) comme on a cité précédemment Jang a utilisé des fonctions gaussiennes, mais pour des besoins d'implémentation on a opté les fonctions d'appartenance triangulaires.

Le tableau suivant montre les règles générées par l'ANFIS :

E↓ *dE*→	B1	B2	B3	B4	B5
A1	d1	d2	d3	d4	d5
A2	d6	d7	d8	d9	d10
A3	d11	d12	d13	d14	d15
A4	d16	d17	d18	d19	d20
A5	d21	d22	d23	d24	d25

Tableau III.2. Table des règles générées par ANFIS.

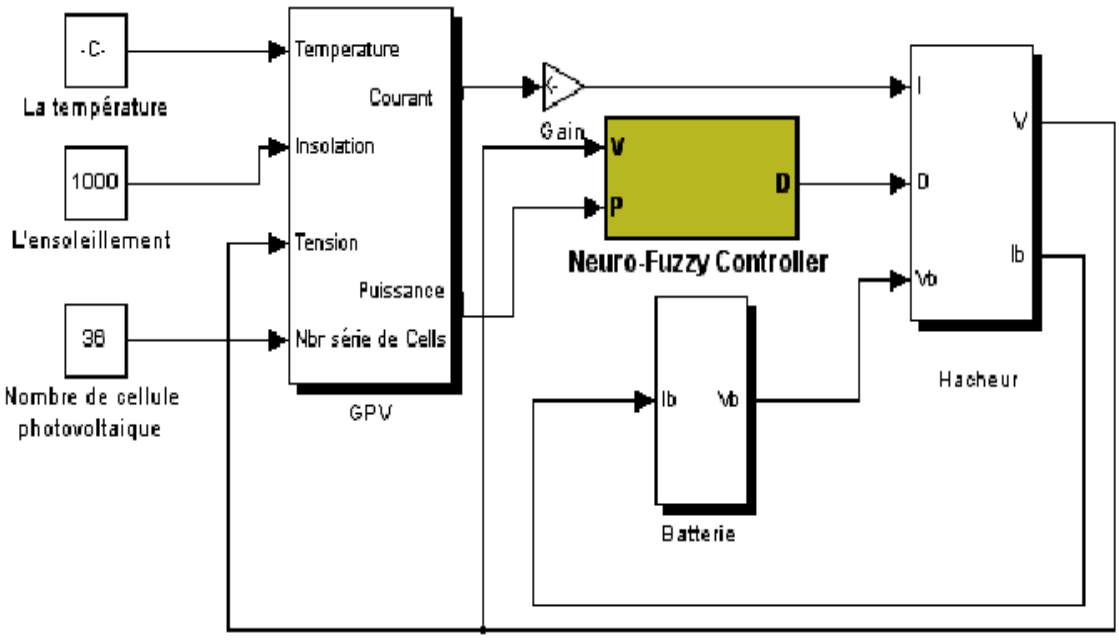


Figure III.24: Schéma synoptique d'un système photovoltaïque avec le contrôleur neuro-flou sous Simulink.

Le système photovoltaïque utilisé pour la simulation est principalement composé des blocs suivants : un générateur photovoltaïque, un hacheur, une batterie et le bloc neuro-flou Figure. III.24.

III.2.5.4.4. Simulation de l'application de l'approche neuro-floue à la poursuite du point de puissance maximale

Dans cette section, nous allons examiner les effets de cette hybridation, illustrés par les graphiques suivants qui mettent ainsi en évidence l'apport de la recherche globale dans l'amélioration des résultats d'apprentissage classique des réseaux de neurones.

Une fois le contrôleur neuro-flou conçu et testé, il est inséré dans le système photovoltaïque. Une simulation numérique est par la suite effectuée.

A. Simulation sous des conditions standards d'ensoleillement « E » et de température « T »

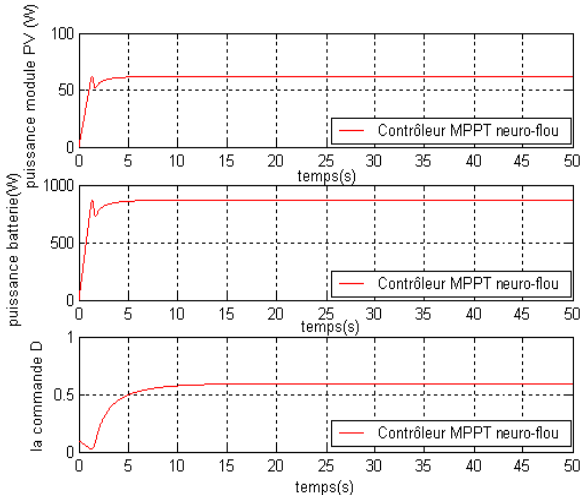


Figure III .25 : Variation des puissances de la batterie, et du module et « D » pour E = 1000W/m² et T= 25° C

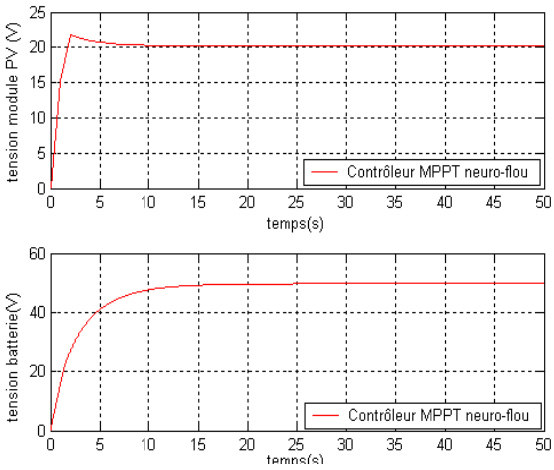


Figure III .26 : Variation des tensions de la Batterie et du module pour E =1000W/m² et une T= 25° C

B. Simulation sous des conditions variables d'ensoleillement

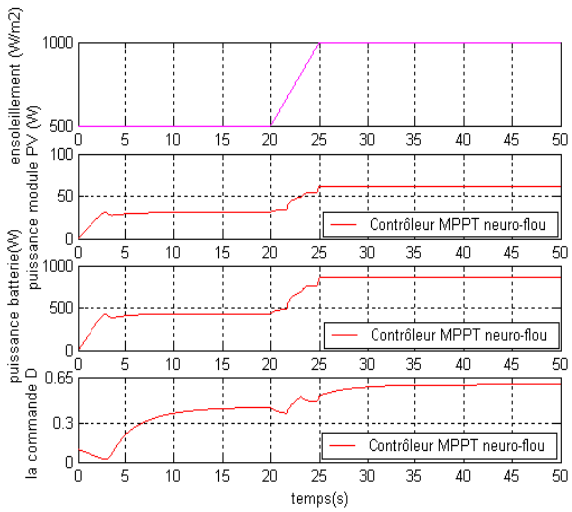


Figure III .27: Variation des puissances du module et de la batterie et la commande D pour une augmentation rapide de E de 500 à 1000 W/m² en 5 s avec une T = 25°C

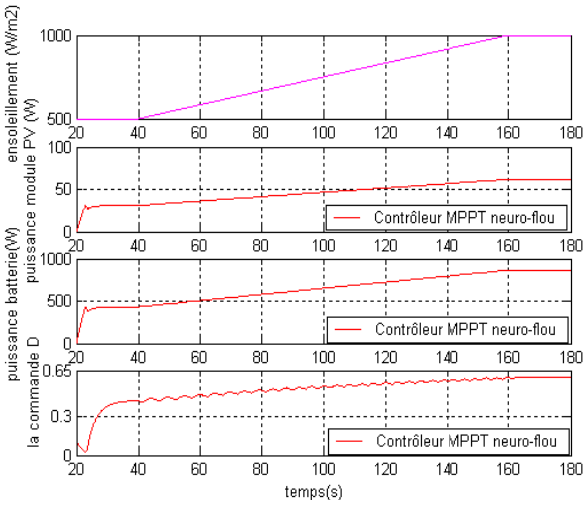


Figure III .28: Variation de la puissance du module, puissance de la batterie, la commande D pour une augmentation lente de E de 500 à 1000 W/m² en 120 s avec une T= 25°C

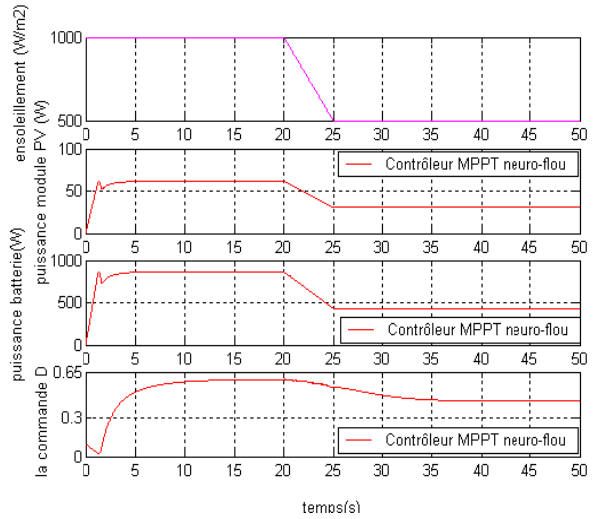


Figure III .29 : Variation des puissances du module, de batterie et la commande D pour une diminution rapide de E de **1000 à 500 W/m²** *en 5 s avec T= 25°C*

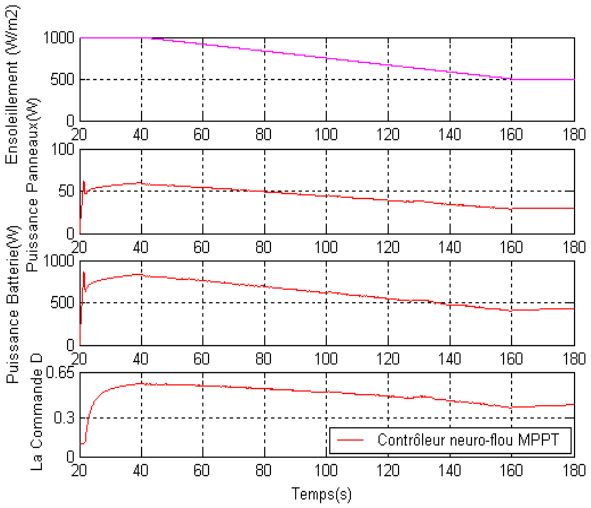


Figure III.30 : Variation de puissance du module, de la batterie et de la commande D pour une diminution lente de E de 1000 à 500 W/m² en 120 s avec une T=25°C

C. Simulation sous des conditions variables de température

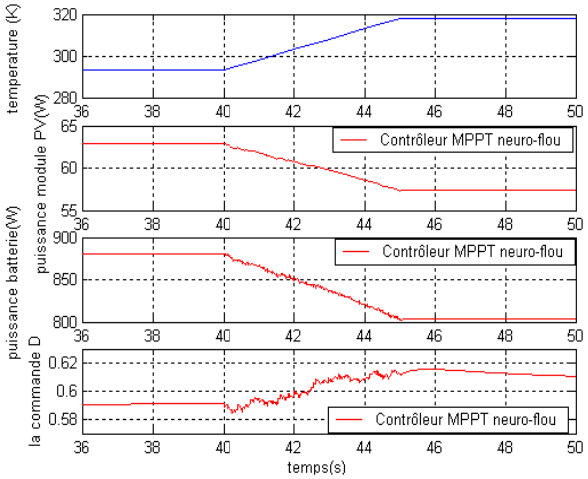


Figure III.31: *Variation de puissance du module, de la batterie et de la commande D pour une augmentation rapide de T de 20°C à 45°C en5 s avec E de 1000W/m²*

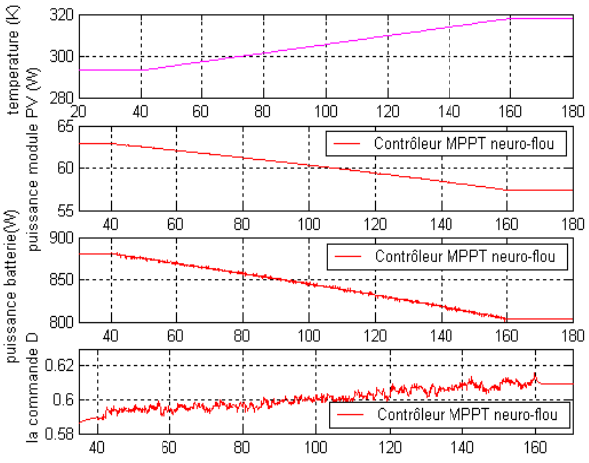


Figure III.32: *Variation de puissance du module, de la batterie et la commande D pour une augmentation lente de T de 20°C à 45°C en 120 s avec un E = 1000W/m²*

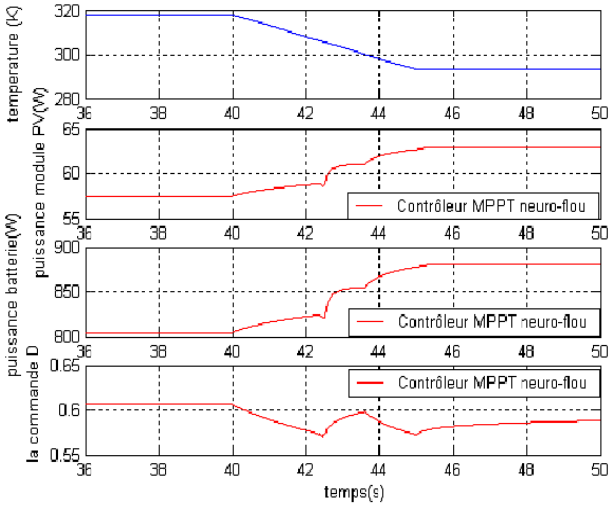


Figure III. 33 : *Variation de puissance du module, de la batterie et de la commande D pour une diminution rapide de T de 45° C à 20° C en 5 secondes avec un E de 1000W/m².*

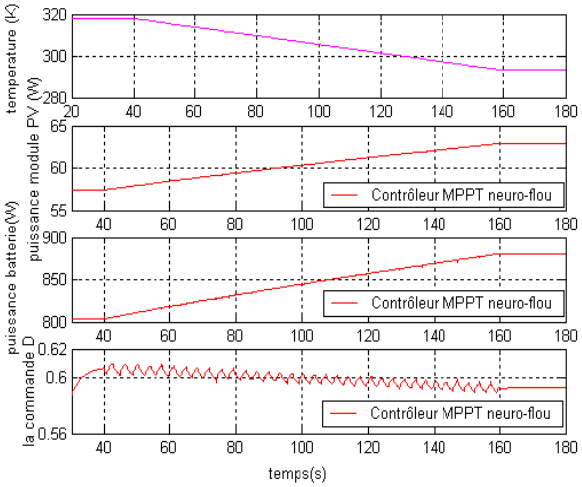


Figure III. 34 : *Variation de puissance du module, de la batterie et la commande D pour une diminution lente de la T de 45°C à 20°C en 120 s avec un E= 1000W/m²*

D'après les résultats de simulation obtenus, on remarque que la commande établie par la technique neuro-floue est performante soit du coté poursuite du point de puissance maximale du système photovoltaïque soit du côté temps de réponse. Ainsi, d'après les tests effectués, on constate que cette commande effectue une poursuite parfaite du point de vue stabilité.

III.3. Les caractéristiques majeures des techniques MPPT

Le tableau suivant représente les caractéristiques majeures des techniques MPPT abordées dans ce chapitre

TECHNIQUES MPPT	Dépendance Du panneau PV	Analogique ou digitale	Réglage de période	Vitesse de convergence	Complexité d'implémentation	Paramètres sensibles
Contre réaction De tension	Non	Les deux	Oui	Rapide	Moyenne	Tension
Méthode à tension de référence fixe	Non	Les deux	Oui	Moyenne	Moyenne	Tension
MPPT avec mesure de V_{OC} du panneau	Oui	Les deux	Oui	Moyenne	Moyenne	Tension
MPPT avec cellule pilote	Oui	Les deux	Oui	Moyenne	Moyenne	Tension
La P and O	Non	Les deux	Non	Variable	Faible	Tension / courant
Incrémentation d'inductance	Non	Digitale	Non	Variable	Moyenne	Tension/ courant
Contre réaction De courant	Non	Digitale	Non	Rapide	Moyenne	courant
Balayage du courant	Oui	Digitale	Oui	Lente	Elevée	Tension/ courant
Les algorithmes génétiques	Oui	Digitale	Oui	Rapide	Elevée	Variables
La logique floue	Oui	Digitale	Oui	Rapide	Elevée	Variables
Les réseaux de neurones	Oui	Digitale	Oui	Rapide	Elevée	Variables
Les réseaux neuro-flous	Oui	Digitale	Oui	Rapide	Elevée	Variables

Tableau III.3: Caractéristiques majeures des techniques MPPT.

III.4. Conclusion

Dans ce chapitre nous avons présenté divers principes de commandes MPPT classiques existantes dans la littérature et nous avons remarqué :

Une complexité importante de l'algorithme de recherche des commandes entraînant souvent des lenteurs de convergence, donc des pertes de rendement.

Pour contourner les problèmes liés à ces commandes, de nouvelles techniques intelligentes sont mise en œuvre telle que la commande par les algorithmes génétiques, les réseaux de neurones et la logique floue dans le but d'une tentative de prédiction d'une meilleure commande qui conduira à mieux appréhender les variations du système PV vis-à-vis les variations des conditions climatiques.

Notre travail est consacré surtout pour une autre commande intelligente basée sur l'approche neuro-floue qui est une hybridation entre les deux approches neuronique et logique floue.

Dans ce chapitre, on a abordé cette technique MPPT; définitions, principe de fonctionnement, on a présenté aussi les résultats de simulation, par MATLAB, de cette approche et on a vu son influence sur la commande MPPT soit pour des conditions climatiques standards ou variables.

L'implémentation de cette technique MPPT sur une carte FPGA sera le sujet des chapitres suivants.

Les circuits logiques programmables FPGA

IV.1. Introduction

La microélectronique est aujourd'hui l'une des bases indispensables pour la réalisation de systèmes numériques performants. Les circuits logiques programmables et les FPGA (Field Programmable Gate Array) en particulier sont devenus incontournables dans ce domaine, car ils présentent beaucoup d'avantages et répondent aux exigences de performance, de consommation, et de réduction du temps et des coûts de développement.

Les circuits logiques programmables sont des circuits composés de nombreuses cellules logiques élémentaires librement assemblables. Celles-ci sont connectées de manière définitive ou réversible par programmation, afin de réaliser la ou les fonctions numériques voulues.
L'intérêt est qu'une même puce puisse être utilisée dans de nombreux systèmes électroniques différents.

De nombreuses familles de circuits programmables et reprogrammables sont apparues depuis les années 70 avec des noms très divers suivant les constructeurs. La figure IV.1 donne une classification possible des circuits numériques en précisant où se situent les circuits FPGA dans cette classification **[33]**.

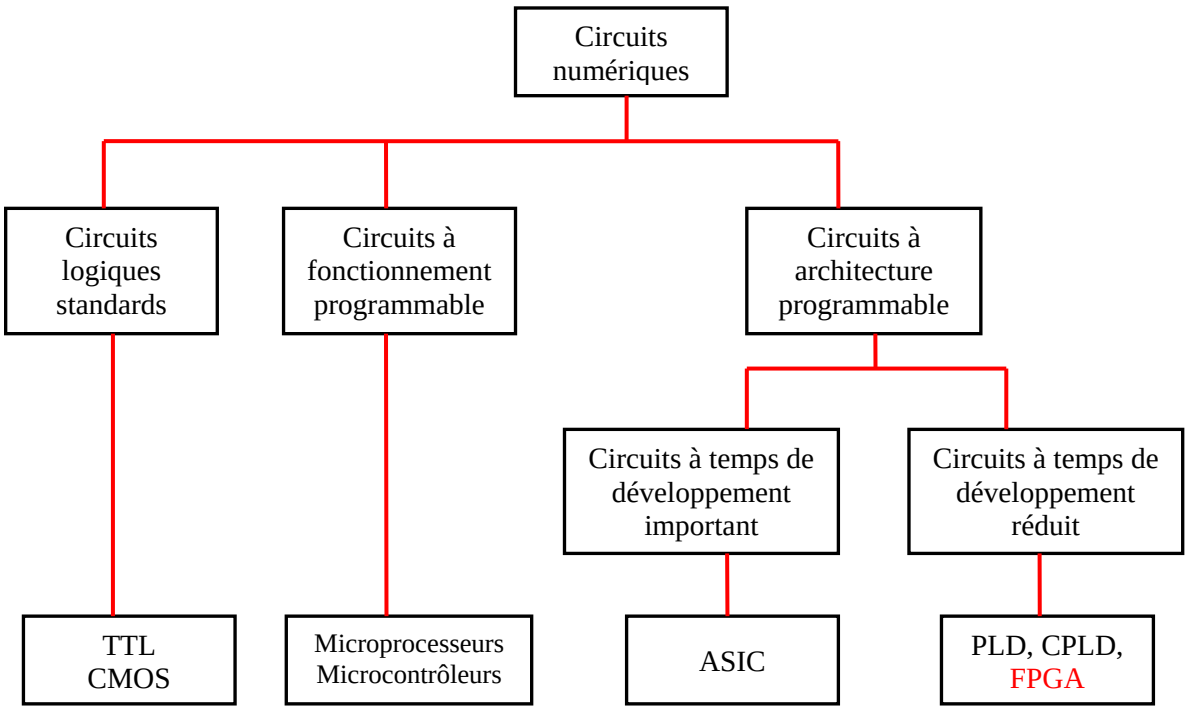


Figure IV.1: Classification des circuits numériques.

Les circuits FPGA sont certainement les circuits reprogrammables ayant le plus de succès. Ils ont été inventés par la société XILINX en 1984. Ce sont des circuits à architecture

programmable entièrement reconfigurables qui permettent d'implanter physiquement, par simple programmation, n'importe quelle fonction logique. Ils ne demandent donc pas de fabrication spéciale en usine, ni de système de développement coûteux. Un autre avantage de ces circuits est qu'ils ne sont pas limités à un mode de traitement séquentiel de l'information comme avec les microprocesseurs, et en cas d'erreur, ils sont reprogrammables électriquement sans avoir à extraire le composant de son environnement.

Beaucoup de FPGA modernes supportent une reconfiguration permettant à leurs designs d'être modifiés « à la volée » aussi bien pour la mise à niveau de systèmes (upgrade) que pour la reconfiguration dynamique comme mode de fonctionnement normal. Certains sont même capables d'être reconfigurés partiellement, c'est-à-dire qu'une partie du dispositif est reprogrammée tandis que d'autres parties continuent de fonctionner. Cette reconfiguration totale ou partielle se fait en l'espace de quelques millisecondes seulement.

Actuellement deux fabricants se disputent le marché mondial des FPGA: ***Xilinx*** et ***Altera***. De nombreux autres fabricants de moindre envergure (Lattice Semiconductor, Actel, Cypress, Atmel et QuickLogic) proposent également leurs propres produits avec des technologies et des principes organisationnels différents. Dans ce qui suit, nous allons faire une description de l'architecture utilisée par Xilinx, car c'est sur l'un de ses circuits que notre application sera implémentée.

IV.2. Architecture matérielle des circuits FPGA

Les circuits FPGA possèdent une structure matricielle de deux types de blocs (ou cellules). Il y'a les blocs d'entrées/sorties (IOB; Input Output Bloc) et les blocs logiques programmables (CLB; Configurable Logic Bloc), comme le montre la figure IV.2. Le passage d'un bloc logique à un autre se fait par un réseau d'interconnexions programmable.

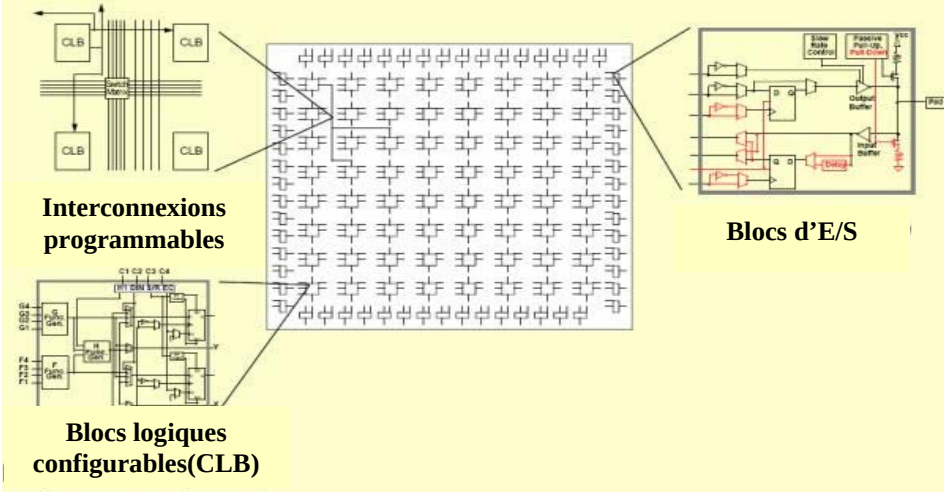


Figure IV.2: Architecture interne d'un FPGA **[33]**.

La plupart des grands FPGA modernes sont basés sur des cellules SRAM (Static Random Access Memory) aussi bien pour le routage du circuit que pour les blocs logiques à interconnecter.

Les FPGA sont bien distincts des autres familles de circuits programmables tout en offrant le plus haut niveau d'intégration logique. L'architecture, retenue par Xilinx, est constituée de deux couches:

- une couche appelée circuit configurable,
- une couche réseau mémoire SRAM (Static Random Access Memory).

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

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

IV.2.1. Les blocs logiques configurables

Les blocs logiques configurables sont les éléments déterminants des performances du circuit FPGA (figure IV.3). Chaque CLB est un bloc de logique combinatoire composé de générateurs de fonction à quatre entrées (LUT) et d'un bloc de mémorisation/synchronisation composé de bascules D. Quatre autres entrées permettent d'effectuer les connexions internes entre les différents éléments du CLB.

La LUT (Look Up Table) est un élément qui dispose de quatre entrées, il existe donc $2^4=16$ combinaisons différentes de ces entrées. L'idée consiste à mémoriser la sortie correspondant à chaque combinaison d'entrée dans une petite table de 16 bits, la LUT devient ainsi un petit bloc générateur de fonctions. La figure IV.4 montre le schéma simplifié d'un CLB de la famille XC4000 de Xilinx.

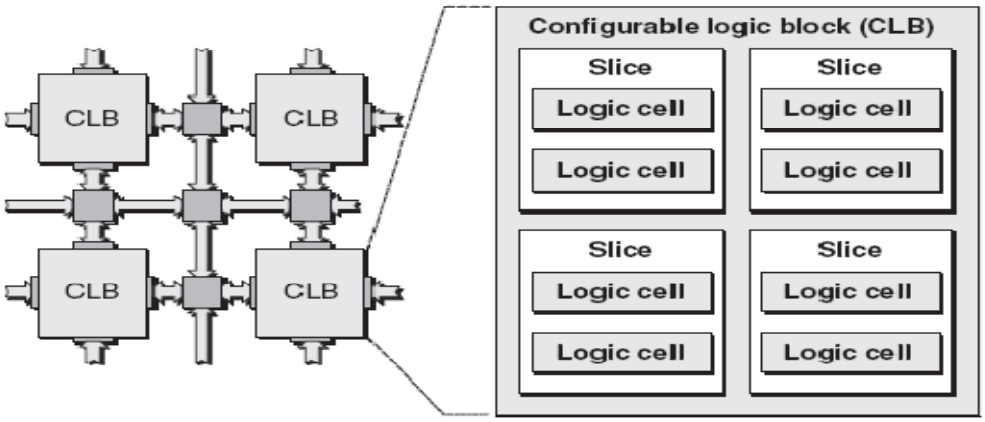


Figure IV.3: Schéma simplifié d'un bloc logique configurable.

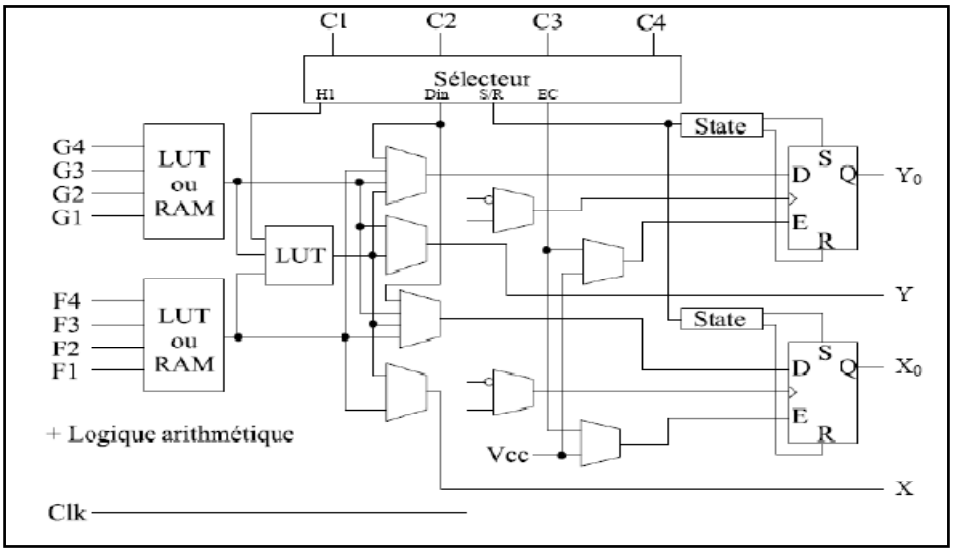


Figure IV.4: Schéma d'une cellule logique de (XC4000 de Xilinx).

IV.2.2. Les blocs d'entrée/sortie

Les blocs d'entrée/sortie ou IOB (Input Output Bloc) permettent l'interface entre les broches du composant FPGA et la logique interne développée à l'intérieur du composant. Ils sont présents sur toute la périphérie du circuit FPGA. Chaque bloc IOB contrôle une broche du composant et il peut être défini en entrée, en sortie, en signal bidirectionnel ou être inutilisé (état haute impédance).

Les IOB peuvent s'adapter à un grand nombre de standards de communication: LVTTL, LVCMOS, PCI, LVDS, LVPECL, etc. Ils sont regroupés en banques, et chaque banque a sa propre tension d'alimentation. Ceci permet de combiner des standards incompatibles entre eux sur le même circuit FPGA en utilisant des banques différentes.

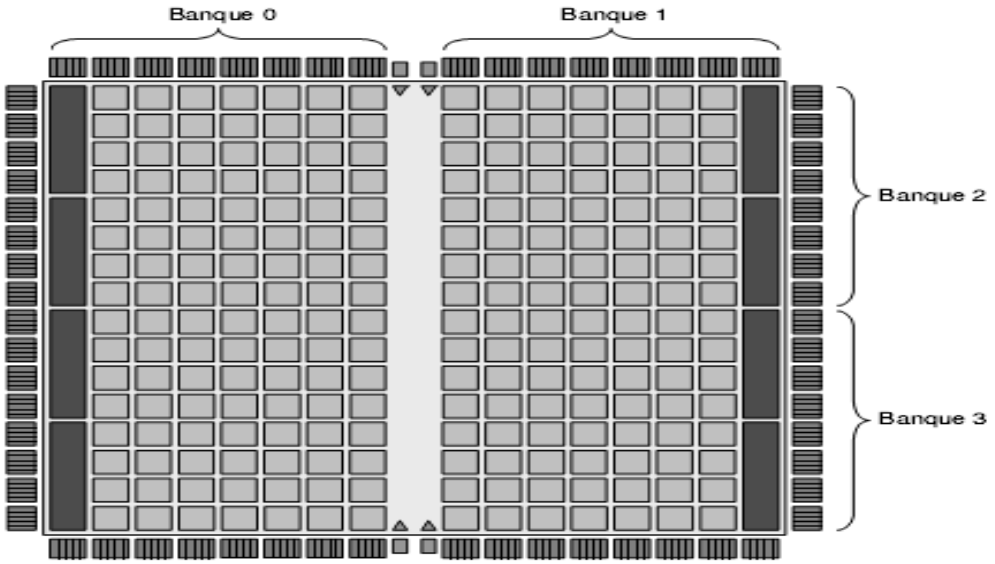


Figure IV.5: Distribution des blocs d'entrée /sortie (IOB) en banques **[34].**

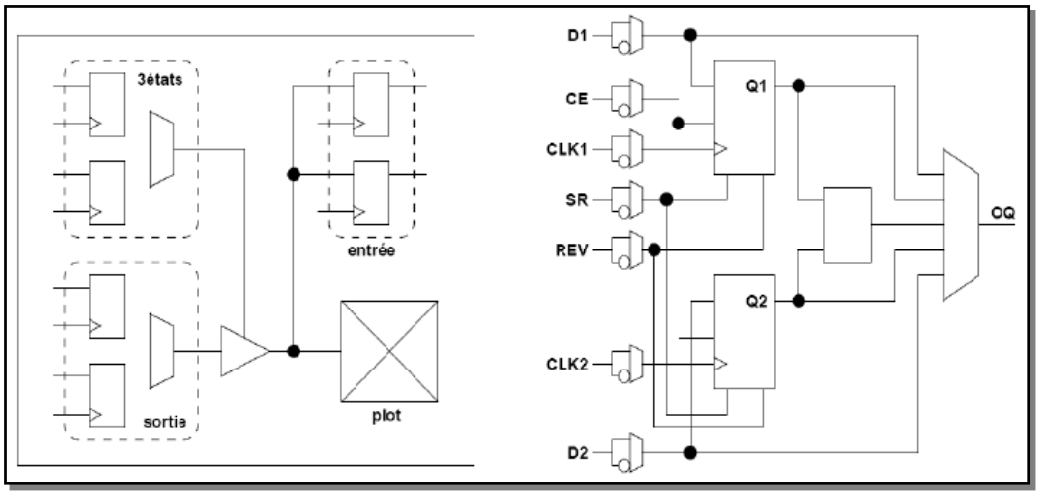


Figure IV.6: Schéma d'un bloc d'entrée/sortie (IOB) **[34].**

IV.2.3. Le réseau d'interconnexions dans un circuit FPGA

Les blocs logiques configurables (CLB) et les blocs d'entrée/sortie (IOB) sont connectés entre eux par un réseau d'interconnexions programmable. Ces connexions internes sont composées de segments métallisés distribués horizontalement et verticalement entre les divers CLB. La topologie est dite "Manhattan", en référence aux rues à angle droit de ce quartier de New York.

Parallèlement à ces lignes, nous trouvons des matrices de routage programmables (matrices de commutation, switch matrix) réparties sur la totalité du circuit. Elles permettent d'avoir des connexions entre les diverses lignes, celles-ci sont assurées par des transistors MOS dont l'état est contrôlé par des cellules de mémoire vive ou RAM (Random Access Memory).

Le routage programmable à l'aide des matrices d'interconnexion permet la configuration à volonté du composant, mais occupe une place importante sur le silicium ce qui justifie le coût élevé des composants FPGA.

Le rôle des interconnexions est de relier avec un maximum d'efficacité les blocs logiques et les blocs d'entrées/sorties afin que le taux d'utilisation dans un circuit donné soit le plus élevé possible. Pour parvenir à cet objectif, Xilinx propose trois sortes d'interconnexions selon la longueur et la destination des liaisons:

- **Les interconnexions à usage général** : Elles servent à relier un CLB à n'importe quel autre CLB à l'aide des matrices de routage, évitant ainsi que les signaux ne traversent les grandes lignes et ne soient affaiblis.
- **Les interconnexions directes** : Elles relient les CLB voisins directement avec un maximum d'efficacité en termes de vitesse et d'occupation du circuit, sans passer par des matrices d'interconnexion.
- **Les longues lignes** : Elles traversent le FPGA dans toute sa longueur et sa largeur, et permettent de transmettre les signaux avec un minimum de retard entre les différents éléments dans le but d'assurer un synchronisme aussi parfait que possible.

De plus, ces longues lignes permettent d'éviter la multiplicité des points d'interconnexion. Les densités actuelles ne permettent plus un routage manuel, c'est donc un outil de placement-routage automatique qui fait correspondre le schéma logique voulu par le concepteur et les ressources matérielles de la puce. Comme les temps de propagation dépendent de la longueur des liaisons entre cellules logiques, et que les algorithmes d'optimisation des placeurs-routeurs ne sont pas déterministes, les performances (fréquence maximale) obtenues dans un FPGA sont variables d'un design à l'autre. L'utilisation des ressources est par contre très bonne, et des taux d'occupation des blocs logiques supérieurs à 90% sont possibles.

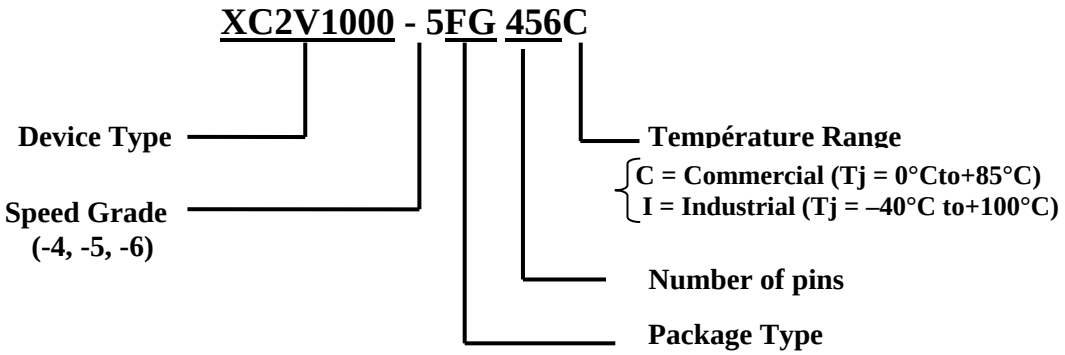
Comme la configuration (routage et LUT) est faite par des points mémoires volatiles, il est nécessaire de sauvegarder le design du FPGA dans une mémoire non volatile externe, généralement une mémoire Flash série, compatible JTAG. Certains fabricants se distinguent toutefois par l'utilisation de cellules EEPROM (Electrically-Erasable Programmable Read-Only Memory) pour la configuration, éliminant le recours à une mémoire externe, ou par une configuration par anti-fusibles (la programmation par une tension élevée fait "claquer" un diélectrique, créant un contact). Cette dernière technologie n'est toutefois pas reconfigurable.

Certains composants FPGA intègrent également des fonctionnalités particulières [35] :

- blocs de mémoire supplémentaires (hors des LUT) comme les BRAM (Block RAM), souvent double-port, parfois avec mécanisme de FIFO (First In First Out),
- multiplieurs câblés (coûteux à implémenter en LUT),
- cœur de microprocesseur enfoui (dit hard core),
- blocs PLL (Phase Locked Loop ou boucle à verrouillage de phase) pour synthétiser ou resynchroniser les horloges,
- cryptage des données de configuration,
- sérialiseurs/désérialiseurs dans les entrées-sorties, permettant des liaisons série à haut-débit,
- impédance contrôlée numériquement dans les entrées-sorties, évitant l'utilisation de nombreux composants passifs sur la carte.

IV.2.4. Nomenclature des circuits FPGA

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



- **Devise type** : le type de la famille qui est dans notre exemple VIRTEX-II.
- **Speed grade** : la vitesse du composons selon la technologie.

IV.2.5. Applications des circuits FPGA

Les circuits FPGA sont utilisés dans diverses et nombreuses applications nécessitant de l'électronique numérique, on en cite quelques exemples dans ce qui suit :

- réseaux et télécommunications,
- vidéo-numérique (effets visuels, encodage, imagerie médicale, …),
- traitement numérique du signal,
- fabrication de composants spéciaux en petite série,
- prototypage d'ASIC,
- systèmes de commande en temps réel,
- aéronautique, transports, défense,
- etc.

Les FPGA sont parfois utilisés pour le prototypage d'ASIC (Application Specific Integrated Circuit). Ils sont généralement plus lents, plus chers à l'unité et consomment davantage d'énergie que leur équivalent en ASIC. Cependant, ils ont plusieurs avantages :

- délai de mise sur le marché plus court, car ce sont des composants standard,
- temps de développement plus court, car on réutilise des fonctions de base,
- validation préalable moins stricte, car c'est possible de reconfigurer le FPGA in-situ si nécessaire,
- coût inférieur pour de petites séries (moins de 10000 unités). Avec l'évolution technologique, cette quantité tend à augmenter. En effet, le prix d'une puce est proportionnel à sa surface, et celle-ci diminue avec la finesse de gravure.

Les FPGA modernes sont assez vastes et contiennent suffisamment de mémoire pour être configurés pour héberger un cœur de processeur ou un DSP, afin d'exécuter un logiciel.
On parle dans ce cas de processeur softcore, par opposition aux microprocesseurs hard-core enfouis dans le silicium. Aujourd'hui, les fabricants de FPGA intègrent même un ou plusieurs cœurs de processeur hard-core sur un même composant afin de conserver les ressources logiques configurables du composant. On tend donc vers des « Systems On Chip SOC » avec, en plus, de la logique configurable selon l'utilisateur.
La mémoire des derniers FPGA est encore insuffisante pour exécuter des logiciels embarqués complexes et on a recours à des mémoires externes (ROM, RAM). Cependant, celles-ci devraient être intégrées dans quelques années et suffiront alors à une grande partie des applications embarquées.

IV.3. Les familles des FPGA de Xilinx

La tendance des dernières générations est de cibler certains créneaux porteurs du marché, comme la solution "bas coût" ou à l'opposé la solution "haute performance" : la taille, le type et le nombre de cellules qui varient suivant les familles de composants. Ainsi, la panoplie présentée par le fabricant montre sa volonté de couvrir tous les segments de marché. Parmi les générations des composants FPGA de XILINX on site :
XC2000, XC3000, XC4000, XC5200, XC6200, XC8100, SPARTAN, VIRTEX, VIRTEX-II, VIRTEX-II Pro, Virtex-IV, Virtex-V.

IV.4. FPGA de la famille Xilinx Virtex-II

Les FPGA du constructeur Xilinx sont divisés en 2 gammes :

1 - FPGA hautes performances : gamme « Virtex »,

2 - FPGA pour la fabrication en grande série (Low Cost) : gamme « Spartan ».

Dans la gamme Virtex, on trouve plusieurs familles dont les performances et la complexité augmentent avec chaque nouvelle génération.

On les classe par ordre chronologique dans ce qui suit [35] :

1- Virtex : sortie en Novembre 1998,

2- Virtex-II : sortie en Janvier 2001,

3- Virtex-II Pro : sortie en Mars 2002,

4- Virtex-IV : sortie en Septembre 2004,

5- Virtex-V : sortie en Mai 2006.

Dans notre application nous étudions l'architecture détaillée du circuit Vertex-II, afin d'implémenter notre commande MPPT.

La famille Virtex-II est le premier jeu de plate-forme FPGA, elle a été conçue pour réaliser des conceptions à faible ou grande densité d'intégration et exige des performances élevées (elle peut atteindre jusqu'a 10 million de portes logiques). Elle offre une densité variante de 40K à 8M portes logiques et peut atteindre une fréquence de fonctionnement de 420MHz. Son architecture est divisée en deux types de cellules de base comme le montre la figure IV.7.

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

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

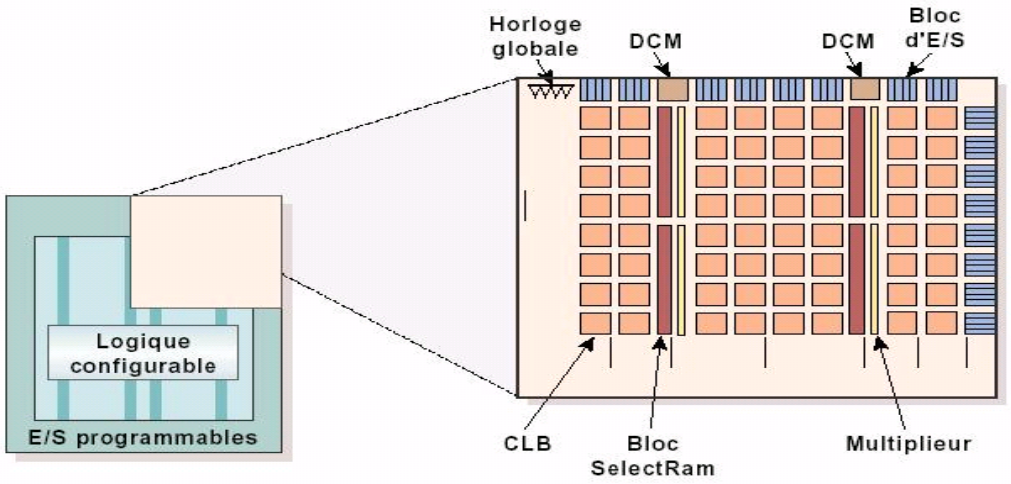


Figure IV.7 : Architecture interne de la famille VIRTEX-II

IV.4.1. Les blocs d'entrée/sortie programmable (IOB ou Input/Output Block)

Ils constituent l'interface entre les bornes du circuit et les CLB. Le dispositif de routage VersaRing offre les ressources nécessaires à l'interconnexion des CLB aux IOB. Nous pouvons ainsi modifier le système implanté sur le FPGA sans interférer avec l'attribution des bornes.

Cette caractéristique s'avère importante si nous souhaitons développer des nouvelles versions d'un produit tout en conservant la compatibilité au niveau du boîtier.

IV.4.2. Les blocs logiques (CLB ou Configurable Logic Block)

Composés de quatre cellules logiques (LC ou Logic Cell) réparties en deux tranches identiques (Figure IV.7), ils servent à construire les circuits numériques implantés sur le FPGA. Chaque LC contient essentiellement:

1. Les blocs logiques configurables (CLB)

C'est l'unité fondamentale du bloc logique qui fournit des éléments utilitaires pour la logique combinatoire et la logique synchrone, y compris les éléments du stockage de base : buffers à 3 états à associer avec chaque CLB. Les CLB incluent quatre parties identiques appelées SLICE et deux buffers à 3 états.

Chaque SILICE est équivalente et contient :

- Deux générateurs de la fonction (F & G).
- Deux éléments du stockage.
- Des portes de la logique arithmétique.
- Grands multiplexeurs.
- Une large fonctionnalité.
- Une logique de retenue rapide.
- Une chaîne de cascade Horizontale (porte OU).

2. Les blocs mémoires (Select RAM)

Ils possèdent des signaux d'initialisation (set et reset) synchrones ou asynchrones. Le bloc Select RAM produit de large élément de stockage (18 Kbit), programmable de 16K x 1 bit à 512 x 36 bits et peut être en deux blocs RAM (dual-port RAM).

3. Les blocs Multiplieurs

Les blocs multiplieurs sont des multiplieurs de18x18 bits. Le circuit VIRTEX-II incorpore une grande densité de blocs multiplieurs. Chaque multiplieur peut être associé au bloc Select RAM ou peut être utilisé indépendamment (Figure IV.7).

4. Les blocs DCM (Digital Clock Manager)

Le DCM produit le nouveau système d'horloges (soit intérieurement ou extérieurement au FPGA), il produit une large gamme de fréquences de l'horloge de multiplication et même la division, le bloc logique programmable contient plus de 12 DCM (Figure IV.7).

IV.5. Développement d'un projet sur circuit FPGA

IV.5.1. La carte de développement Xilinx Virtex-II V2MB1000 de Memec Design

Le kit de développement Virtex-II V2MB1000 de Memec Design, qu'on a utilisé pour développer notre application, fournit une solution complète de développement d'applications sur la famille Virtex-II de Xilinx. Il utilise le circuit « FPGA XC2V1000-4FG456C » qui appartient à cette famille et qui est équivalent à 1 million de portes logiques.

La haute densité d'intégration des portes ainsi que le nombre important d'entrées/sorties disponibles à l'utilisateur permettent d'implémenter des systèmes complets de solutions sur la plate forme FPGA. La carte de développement inclue aussi une mémoire 16M x 16 DDR, deux horloges, un port série RS-232 et des circuits de support additionnels. Une interface LVDS est disponible avec un port de transmission 16-bit et un port de réception 16-bit, en plus de signaux d'horloge, d'état et de contrôle pour chacun de ces ports. La carte supporte également le module d'expansion Memec Design P160, qui permet d'ajouter facilement des modules pour des applications spécifiques.

La famille FPGA Virtex-II possède les outils avancés pour répondre à la demande à des applications de haute performance. Le kit de développement Virtex-II fournit une excellente plateforme pour explorer ces outils, l'utilisateur peut alors utiliser toutes les ressources disponibles avec rapidité et efficacité.

La figure IV.8 présente une photo de la carte de développement Xilinx Virtex-II V2MB1000 de Memec Design [36].

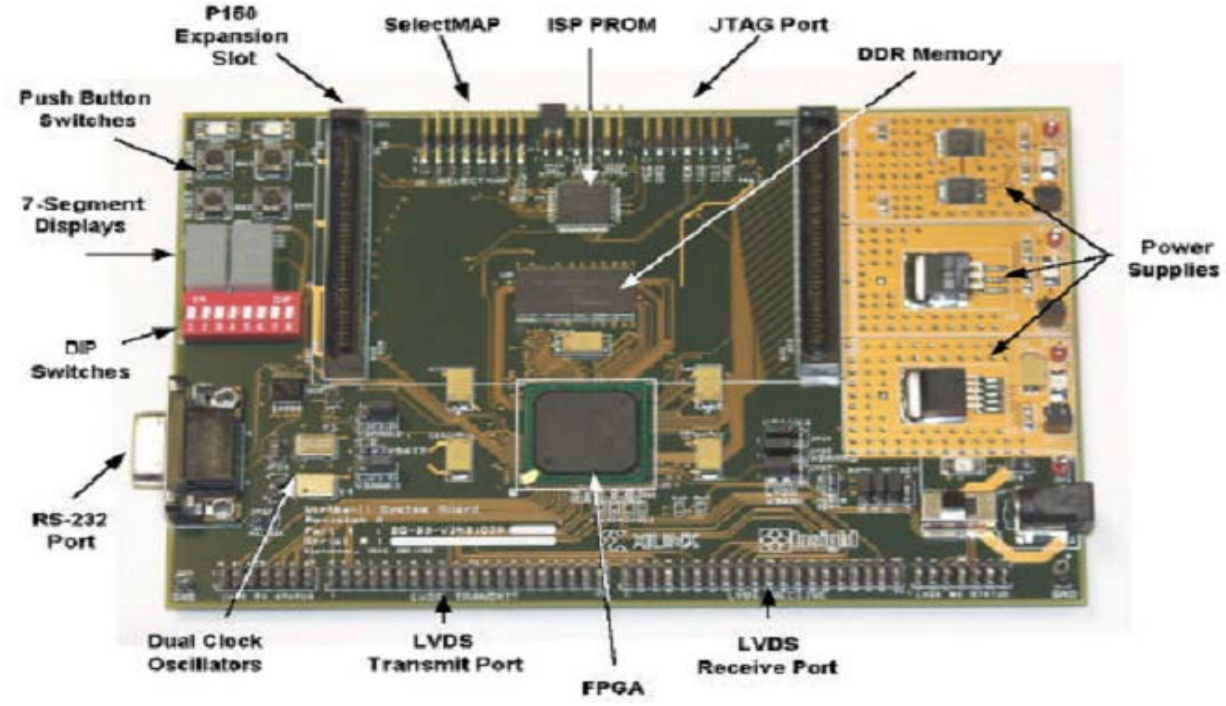


Figure IV.8: Carte de développement «Xilinx Virtex-II V2MB1000 de Memec Design».

IV.5.2. Description de la carte de développement Memec Design Virtex-II V2MB1000

Un diagramme simplifié de la carte de développement Memec Design Virtex-II V2MB1000 est illustré à la figure IV.9.

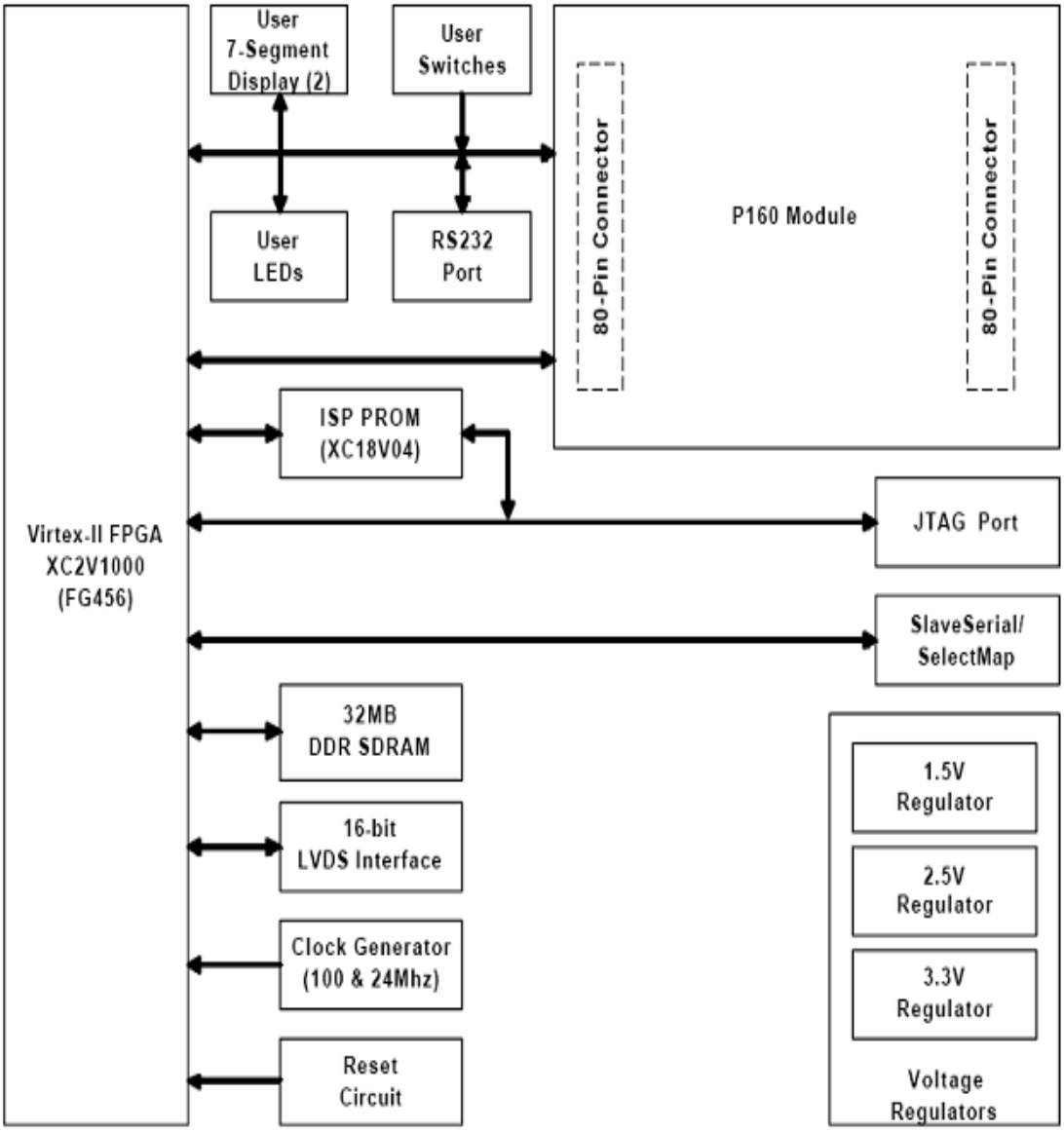


***Figure IV.9: Diagramme de la carte Memec Design Virtex-II* V2MB1000.**

La carte de développement Virtex-II contient le circuit FPGA XC2V1000-4FG456C. Ce circuit fait partie de la famille Virtex-II, qui est une famille de circuits développés pour des applications haute performance telles que les télécommunications, l'imagerie et les applications DSP. Il possède 456 broches dont 324 peuvent être utilisées en entrées/sorties. Il se compose d'une matrice de 40x32 CLB et il contient un total de 10240 LUT et 10240 bascules « flip-flop». Sa capacité maximale en Select RAM est de 163840 bits [37].

La carte contient également une mémoire DDR de 32MB. Elle présente 2 générateurs d'horloges internes, générant des signaux d'horloge à 100MHz (CLK.CAN2) et 24MHz (CLK.CAN1), un troisième signal d'horloge externe est disponible et peut être utilisé si on a besoin. Elle contient aussi circuit de remise à zéro « Reset » activé par un bouton poussoir (SW3), un bouton poussoir « PROGn » pour initialiser la configuration et charger le contenu de la PROM dans le circuit FPGA (SW2) ainsi que deux boutons poussoirs (SW5 et SW6) qui peuvent être utilisés pour générer des signaux actifs.

Deux afficheurs 7 segments à cathode commune sont présents sur la carte, et peuvent être utilisés durant la phase de test et de debugging.

Il existe aussi 8 entrées exploitables par l'utilisateur (DIP switch) qui peuvent être mis statiquement à un état haut ou bas.

La carte possède une interface RS232, une interface JTAG pour programmer l'ISP PROM et configurer le circuit FPGA ainsi qu'un connecteur de câble parallèle qui peut aussi être utilisé pour configurer le FPGA via la configuration Maître/Esclave.

Il existe également des régulateurs internes de tension qui génèrent, à partir de l'alimentation principale de 5,0V, des tensions internes d'alimentation à 1,5V, 2,5V et 3,3V.

Les entrées/sorties sont regroupées dans 8 différents groupes, et chaque groupe peut être configuré pour opérer dans le mode 2,5V ou 3,3V.

La carte de développement peut être configurée pour travailler en mode Master Serial, Slave Serial, Master SelectMap, Slave SelectMap ou JTAG selon la position des jumpers M0, M1, M2 et M3.

- **Chargement du programme sur la carte de développement**

La carte de développement Virtex-II supporte plusieurs méthodes de configuration de son circuit FPGA. Le port JTAG peut être utilisé directement pour configurer le FPGA, ou pour programmer l'ISP PROM. Une fois l'ISP PROM programmée, elle peut être utilisée pour configurer le FPGA. Le port SelectMap/Slave Serial sur cette carte peut aussi être utilisé pour configurer le FPGA. La figure suivante montre l'installation pour toutes les configurations de modes supportés par la carte de développement Virtex-II V2MB1000.

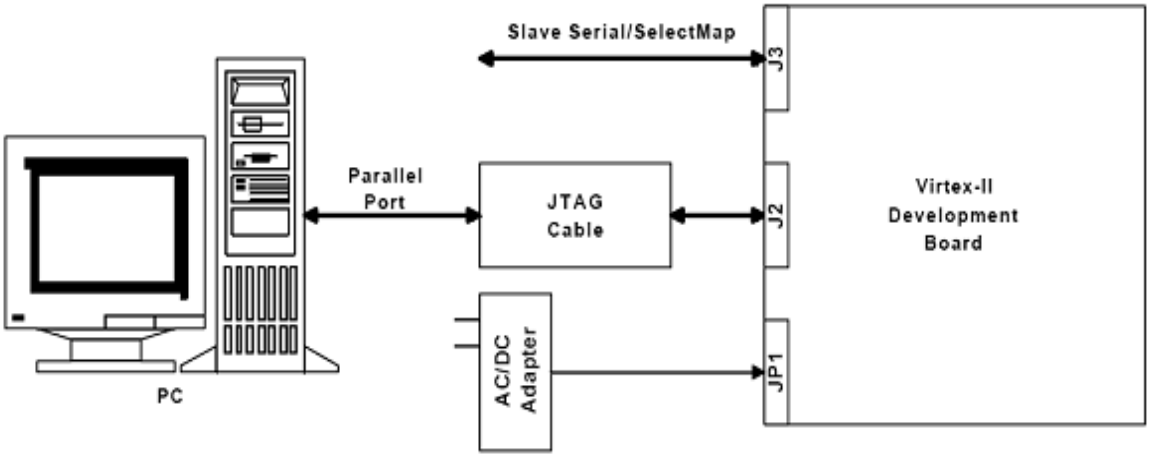


Figure IV.10 : Chargement du programme sur la carte FPGA

- **Utilisation de l'interface JTAG**

Le câble Memec Design JTAG est connecté d'un coté à la carte de développement, et de l'autre au port parallèle du PC. On utilise alors l'outil de programmation du JTAG de Xilinx (iMPACT) pour charger le programme binaire soit directement sur le circuit FPGA en mode JTAG, soit sur l'ISP PROM en mode Master Serial ou Master SelectMap, dans ce dernier cas il faut appuyer sur le bouton poussoir PROGn (SW2) pour initialiser la configuration dans le circuit FPGA.

- **Utilisation de l'interface Slave Serial**

Dans ce mode, une source externe fournit la configuration « bitstream » et la configuration clock au circuit FPGA Virtex-II. Là aussi, le programme peut être directement chargé sur le circuit FPGA, ou bien sur l'ISP PROM, et dans ce cas il faut utiliser le bouton PROGn pour initialiser le FPGA.

IV.5.3. Méthodologie de développement d'un projet sur FPGA

Notre application a été réalisée en utilisant le langage VHDL (Very high-speed integrated-circuit Hardware Description Langage). C'est est un langage de description hardware de haut-niveau, qui permet de matérialiser les structures électroniques d'un circuit.

En effet, les instructions écrites dans ce langage se traduisent par une configuration logique faite de portes et de bascules, qui est ensuite intégrée à l'intérieur du circuit cible.

La description VHDL se fait d'une manière hiérarchique, où chaque module peut soit être décrit complètement, soit faire appel à des sous-modules déjà décrits. Un code écrit en VHDL est portable, c'est-à-dire qu'une description écrite pour un circuit peut être utilisée pour un autre. Mais pour obtenir un design avec des performances optimales, il vaut mieux décrire le

comportement des algorithmes souhaités en tenant compte de la technologie et de l'architecture du circuit FPGA cible (le XC2V1000-4FG456C dans notre cas).

La réalisation d'une application sur circuit FPGA impose une méthodologie de développement structurée.

La première étape est la description des algorithmes en VHDL, une vérification des erreurs s'impose alors. Ensuite c'est l'étape de synthèse, dans laquelle les lignes de code VHDL sont traduites en schémas électroniques équivalents composés de fonctions logiques de base. L'étape suivante est le placement et routage, où les schémas électroniques sont transformés en routes et connexions sur le FPGA cible. Enfin, c'est la génération des fichiers binaires de programmation et leur chargement sur le FPGA cible à travers le port JTAG.

Il est possible de disposer de fichiers VHDL après chaque étape. Le même fichier de simulation (test-bench) est ainsi utilisable pour vérifier le fonctionnement de la description à chaque étape de la procédure de développement. Cependant, c'est la simulation comportementale qui est la plus souvent utilisée.

Après le chargement du design sur le FPGA, on doit réaliser des tests en laboratoire en appliquant des contraintes réelles, afin de valider le bon fonctionnement du dispositif dans la pratique.

IV.5.3.1. Logiciels et outils utilisés

Chaque étape du développement d'un projet sur FPGA nécessite un outil logiciel différent. Nous allons présenter brièvement les outils et logiciels utilisés dans ce projet **(**figure IV.11).

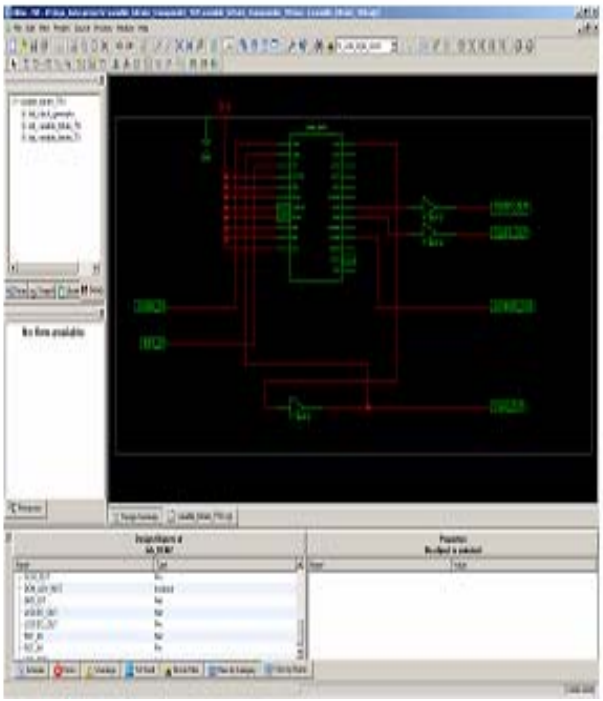
- **Xilinx ISE 7.1i**

Ce logiciel offre un environnement convivial et regroupe tous les outils nécessaires à la réalisation des différentes étapes du projet. Le code VHDL des différents modules est saisi dans un éditeur de texte. L'étape de synthèse est réalisée par l'outil XST, l'étape de placement et routage par l'outil FPGA Editor et l'étape de configuration du composant par l'outil iMPCAT. Xilinx ISE permet ainsi de suivre l'évolution du projet étape par étape.

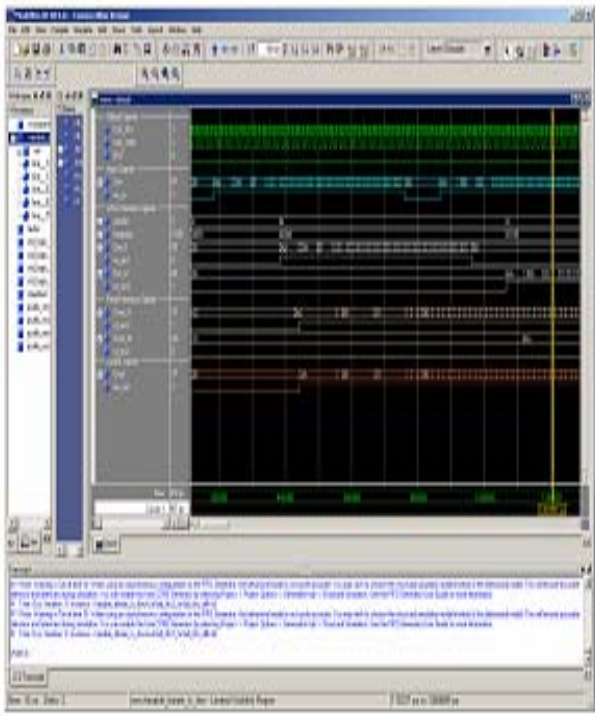
- **ModelSim Simulator XE 6.0**

Ce logiciel permet de faire la simulation du design à différentes étapes du projet. Pour faire une simulation, on doit créer des fichiers test-bench, qui décrivent la variation des entrées du design dans le temps. La simulation permet de visualiser les sorties et de les comparer par rapport aux résultats attendus, afin de faire une première validation du design.

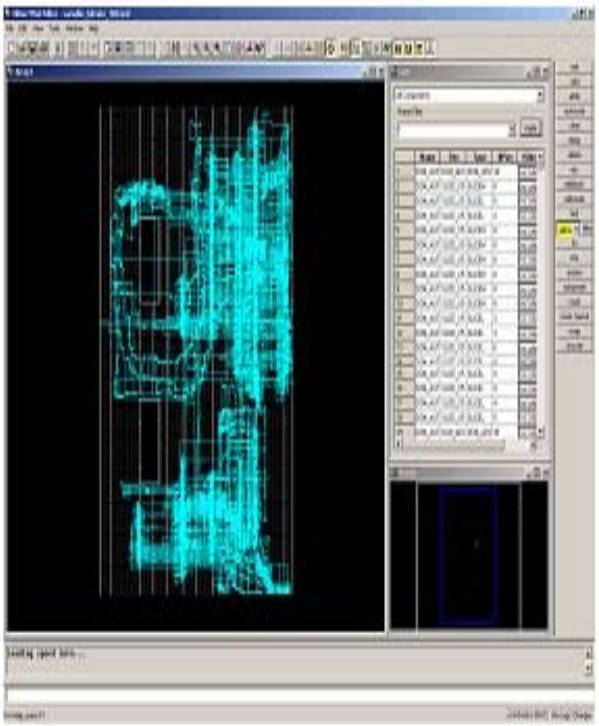
a) Xilinx ISE (gestion du projet)



b) XST (synthèse)



c) ModelSim (simulation)



d) FPGA Editor (placement routage)

Figure IV.11: Logiciels et outils utilisés pour le développement d'un projet sur FPGA.

IV.5.3.2. Implémentation d'un projet sur circuit FPGA

L'implémentation d'un projet sur le circuit FPGA suit ces étapes :

1. Description VHDL : c'est la modélisation de l'architecture et l'écriture des différents codes VHDL des différents composants constituant l'architecture ;

2. Compilation syntaxique : vérifie l'existence d'erreurs dans les instructions du code VHDL ;

3. Compilation logique : vérifie la faisabilité décrite dans les programmes ;

4. Simulation fonctionnelle : permet de tester le bon fonctionnement de l'architecture moyennant l'outil de simulation 'ModelSim' ;

5. Phase de synthèse : permet de passer d'un niveau d'abstraction élevé ' VHDL' vers un niveau d'abstraction plus bas qui est le niveau (RTL : Registre Translate Level) adapté au langage machine. Le résultat de la synthèse est un fichier dont l'extention est '**.edif** '.

6. Phase de placement routage : elle passe par 3 étapes :

- **Translation** : permet de convertir le fichier '**.edif** 'vers un fichier d'extention **'.NGD'** dont il est comprhénsible par l'outil de placement routage ;
- **Mapping** : planification de l'implémentation des différentes parties de l'architecture vis-à-vis des ressources des circuits FPGA que nous ciblons dans notre application ;
- **Placement et routage**: il s'agit d'établir les différentes interconnexions entre les composants integrés dans le circuit.

7. Génération du fichier ' . bit ' : après le placement routage, l'outil nous génére un fichier dont l'extention '**.bit**' est un fichier binaire, téléchargeable, qui permet de configurer un FPGA.

Le développement en VHDL nécessite l'utilisation de deux outils : le simulateur et le synthétiseur. Le premier va nous permettre de simuler notre description VHDL avec un fichier de simulation appelé « test-bench » ; cet outil interprète directement le langage VHDL et il comprend l'ensemble du langage. L'objectif du synthétiseur est très différent : il doit traduire le comportement décrit en VHDL en fonctions logiques de base, celles-ci dépendent de la technologie choisie ; cette étape est nommée « synthèse ». L'intégration finale dans le circuit cible est réalisée par l'outil de placement et routage. Celui-ci est fourni par le fabricant de la technologie choisie.

Avec les outils actuels, il est possible de disposer de fichiers VHDL à chaque étape. Le même fichier de simulation (test-bench) est ainsi utilisable pour vérifier le fonctionnement de la description à chaque étape de la procédure de développement. La figure IV.12 donne les différentes étapes nécessaires au développement d'un projet sur circuit FPGA.

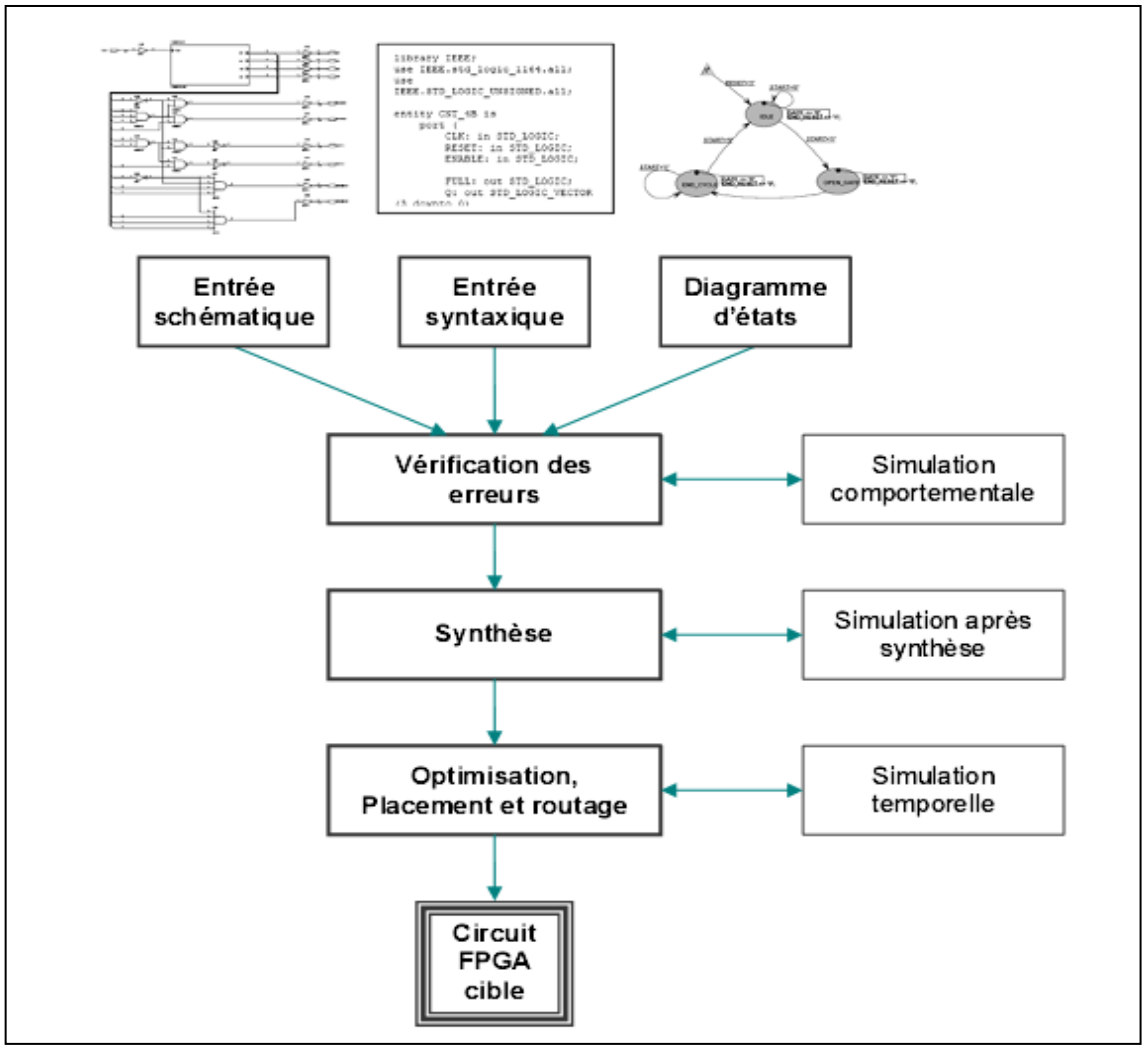


Figure IV.12 : Organisation fonctionnelle de développement d'un projet sur circuit FPGA.

- **Saisie du texte VHDL**

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

La figure (IV.11.a) montre comment se présente le logiciel «ISE Xilinx Project Navigator».

La saisie du texte se fait sur la partie droite de l'écran, on voit en haut à gauche la hiérarchie du projet, et en bas à gauche les nombreux outils nécessaires tout au long du développement du projet.

Il faut commencer par créer un projet, ensuite inclure des fichiers sources dans lesquels il faut saisir le texte VHDL désiré. On peut inclure autant de sources qu'on veut dans un projet.

- **Vérification des erreurs**

Cette étape est effectuée en appuyant sur le bouton « check syntax ». Elle permet de vérifier les erreurs (errors) de syntaxe du texte VHDL et d'afficher les différentes alarmes (warnings) liées au programme, par exemple des signaux déclarés mais non utilisés dans le programme. S'il y'a des erreurs dans le programme, il ne peut pas être synthétisé, mais la présence d'alarmes n'empêche pas de poursuivre normalement les autres étapes du développement.

Cette étape permet donc de valider la syntaxe du programme et de générer la «netlist», qui est un fichier contenant la description de l'application sous forme d'équations logiques.

- **Synthèse**

La synthèse permet de réaliser l'implémentation physique d'un projet. Le synthétiseur a pour rôle de convertir le projet, en fonction du type du circuit FPGA cible utilisé, en portes logiques et bascules de base. L'outil « View RTL Schematic » permet de visualiser les schémas électroniques équivalents générés par le synthétiseur (figure IV.11.b).

De plus, le synthétiseur permet à l'utilisateur d'imposer des contraintes de technologie (User constraints) : par exemple fixer la vitesse de fonctionnement (Create Timing Constraints) ; délimiter la zone du circuit FPGA dans laquelle le routage doit se faire (Create Area constraints) ou affecter les broches d'entrées/sorties (Assign Package Pins). La figure IV.13 montre un aperçu de l'outil d'assignation des broches d'entrées/sorties.

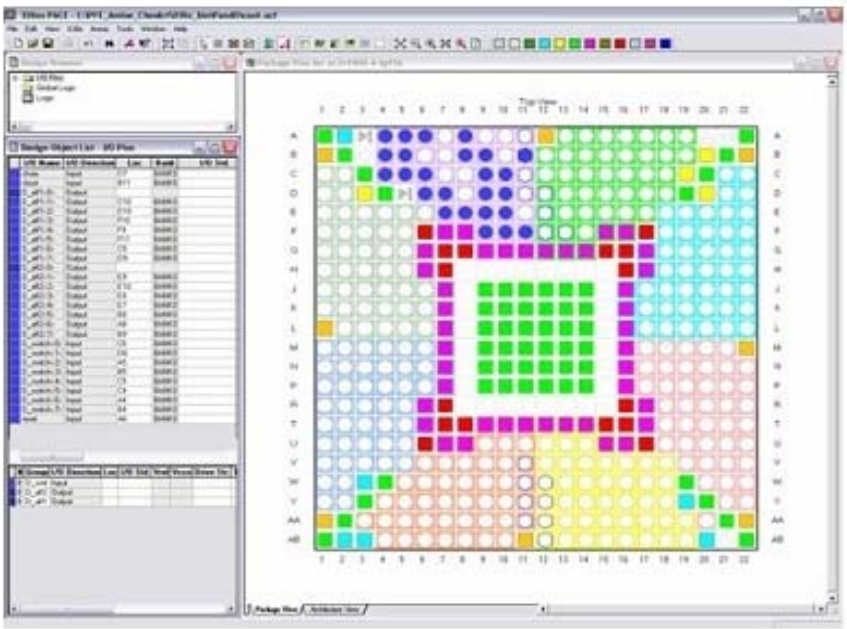


Figure IV.13: Aperçu de l'outil d'affectation des broches d'entrées/sorties.

- **Simulation**

Le simulateur utilisé est le **« ModelSim Simulator »** (figure IV.11.c). La simulation permet de vérifier le comportement d'un design avant ou après implémentation dans le composant cible. Elle représente une étape essentielle qui nous fera gagner du temps lors de la mise au point sur la carte. Il faut juste noter qu'un projet peut être simulé même s'il n'est pas synthétisable.

Lors de l'étape de simulation comportementale, on valide l'application indépendamment de l'architecture et des temps de propagation du futur circuit cible. La phase de simulation après synthèse valide l'application sur l'architecture du circuit cible, et enfin la simulation temporelle prend en compte les temps de propagation des signaux à l'intérieur du circuit FPGA.

- **Optimisation, placement et routage**

Pendant l'étape d'optimisation, l'outil cherche à minimiser les temps de propagation et à occuper le moins d'espace possible sur le circuit FPGA cible. Le placement et routage permet de tracer les routes à suivre sur le circuit afin de réaliser le fonctionnement attendu. La figure (IV.11.d) donne un aperçu de l'outil de placement et routage « FPGA Editor » qui permet de visualiser et d'éditer le circuit routé.

- **Programmation du composant et test**

Dans cette dernière étape (Generate Programming Files), on génère le fichier à charger sur le circuit FPGA à travers l'interface JTAG. Une fois le programme chargé sur le circuit, on peut tester et visualiser les résultats directement sur la carte de développement Virtex-II à travers les nombreuses interfaces qu'elle offre, soit directement sur les deux afficheurs 7 segments, soit à travers l'interface RS232 ou bien à travers l'interface LVDS 16 bits.

IV.6. Conclusion

Ce chapitre a été consacré aux circuits programmables FPGA, plus particulièrement le circuit FPGA Xilinx Virtex-II. On a aussi expliqué les déférentes étapes de développement d'une implémentation d'un projet sur circuit FPGA.

Le dernier chapitre sera consacré à l'implémentation de la commande MPPT par l'approche neuro-floue sur circuit FPGA en commençant par le saisi du texte VHDL jusqu'à le chargement du programme sur le circuit cible. Les résultats de simulation, d'implémentation et leurs discussions seront détaillés.

Application, résultats et discussion

V.1. Introduction

L'objet de notre étude est l'implémentation sur circuit FPGA de la commande MPPT par l'approche « neuro-floue». Dans ce dernier chapitre, on présente les différentes étapes de cette implémentation. Nous commençons par donner la structure générale des programmes écrits en VHDL, puis les résultats de simulation obtenus pour des conditions différentes de température et d'ensoleillement et enfin une interprétation des résultats obtenus. Pour montrer les performances d'un tel contrôleur, on a implémenté la même commande MPPT par l'approche floue afin de faire une comparaison entre les deux approches MPPT.

Enfin, on termine par donner les observations et les conclusions qu'on peut tirer à partir de ces résultats.

La simulation nous permet de valider théoriquement les programmes écrits en VHDL avant de les implémenter sur la carte de développement « Virtex II V2MB1000 de Memec design ». Une fois les programmes chargés sur la carte, il faut effectuer des tests réels pour les valider pratiquement.

V.2. Description du contrôleur MPPT flou

La structure de base du contrôleur MPPT flou a été expliquée dans le chapitre II. On a écrit le programme en VHDL conformément à cette structure en le divisant en trois blocs principaux comme suit :

- Le premier bloc (Fuzzification) contient les formes des fonctions d'appartenance qui sont mémorisées sous forme de tableaux ; il permet de calculer les valeurs des degrés de vérité des différentes fonctions d'appartenance (NGE, NPE, ZEE, PPE, PGE pour l'entrée E, et NGdE, NPdE, ZEdE, PPdE, PGdE pour l'entrée dE) à partir des valeurs des entrées E et dE.
- Le second bloc (Inférences) contient les règles d'inférences qui permettent de calculer, à partir des degrés de vérité de E et dE, le degré d'activation de chacune des fonctions d'appartenance (NGD, NPD, ZED, PPD, PGD) de la sortie dD.
- Enfin, le troisième bloc (Défuzzification) construit par agrégation l'ensemble flou de sortie et calcule la valeur réelle de la sortie dD par la méthode du centre de gravité.

Le contrôleur flou se compose en réalité des trois blocs suscités, on a ajouté un bloc en entrée et un autre en sortie. Le bloc d'entrée permet de calculer les valeurs de E et dE à partir de la puissance P et la tension V, et le bloc de sortie calcule la valeur du rapport cyclique D à partir de la sortie dD.

Chacun de ces blocs est écrit sous forme de plusieurs process qui se déclenchent à chaque changement d'état de leurs entrées, et fournissent les résultats en sortie instantanément. Seul le premier bloc (entrée) est commandé par un signal d'horloge (CLK), donc toute la chaîne ne commence à travailler qu'à partir d'un front montant du signal d'horloge.

La figure V.1 donne le schéma détaillé du contrôleur flou.

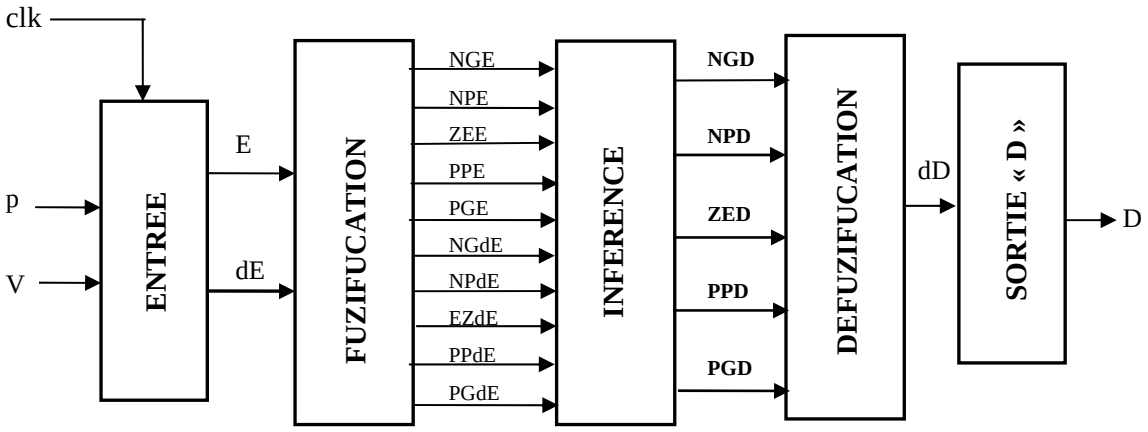


Figure V.1: Schéma synoptique détaillé du bloc "MPPT flou "

Le programme peut être représenté par une boite noire commandée par un signal d'horloge (CLK), ayant comme entrée la valeur de la puissance instantanée P et la tension V, et comme sortie la valeur du rapport cyclique (D) qui sera appliquée au convertisseur continu-continu

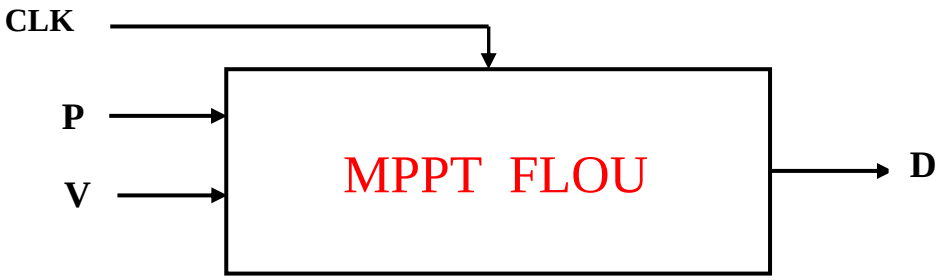


Figure V.2 : Schéma synoptique simplifié du bloc "MPPT flou "

V.3. Description du contrôleur MPPT neuro-flou

La structure de base du contrôleur MPPT neuro-flou a été expliquée dans le chapitre II. On a écrit le programme en VHDL conformément à cette structure en le divisant en blocs (entités) ainsi qu'un bloc d'entrée qui permet de calculer la valeur de E et dE à partir de la puissance P et la tension V comme suit :

- **L'entité fuzzification (couche fuzzification)**

Cette entité a pour rôle de fuzzifier les deux variables E et dE d'entrées. Grâce aux intervalles uniformes des fonctions d'appartenance, nous avons considéré chacun comme une zone ayant un code sur trois bits (trois bits pour pouvoir coder les six zones).

Connaissant le code de la zone d'appartenance, par une simple comparaison de la valeur d'entrée avec les valeurs limites des zones, il suffit de calculer une valeur pour déduire la seconde, car nous savons que chaque valeur d'entrée engendre deux règles.

La fuzzification d'une variable est facile vu l'allure des fonctions d'appartenance dans chaque zone qui sont des segments de droites, donc, il suffit d'avoir la pente des segments et la largeur des intervalles pour calculer tout simplement la valeur fuzzifiée puis déduire la deuxième par la complémentation à 1. La figure V.3 explique l'algorithme de fuzzification des 'E' et 'dE'.

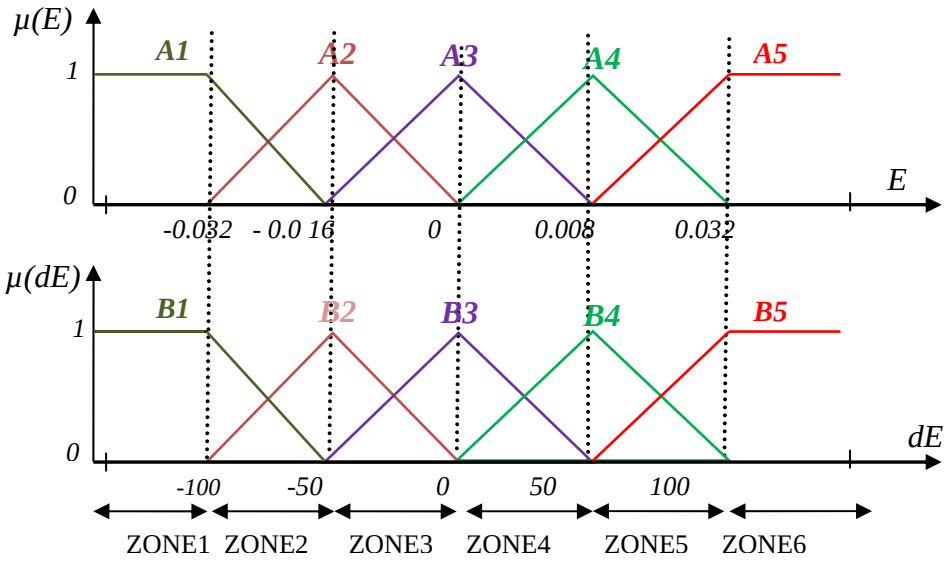


Figure V.3: Illustration de l'algorithme de fuzzification

- **Entité inférence « la couche de règles floues »**

Cette couche calcule les valeurs des poids ***w1,w2 …….w25*** à partir des valeurs des fonctions d'appartenance ***A1, A2 , A3, A4 ,A5 ,B1, B2, B3, B4, B5.***

Le rôle de ce composant est de déduire les règles correspondantes aux valeurs E et dE à partir de leurs valeurs fuzzifiées en effectuant un produit direct entre les valeurs de A_i et B_i, les sorties déduites sont les quatre combinaisons entre les degrés d'appartenance selon les zones de travail choisies (pour faire combiner deux zones de E et dE on a deux valeurs pour la première et deux autre valeurs pour la deuxième).

Les valeurs obtenues sont les entrées de la phase de défuzzification.

- **L'entité défuzzification « couche de défuzzification »**

Cette couche à pour rôle de calculer les sorties D_i à partir des sorties de la couche de normalisation et les valeurs des règles générées par ANFIS (d1, d2, d3…...d25) (voir tableau III.2).

Remarque : par programmation, on a inclut la couche de « normalisation », qui permet de calculer les poids normalisés de *Wi*, dans la couche de défuzzification.

Dans la phase de défuzzification, il s'agit de faire combiner les règles d'inférence en se basant sur les valeurs des éléments de l'entité précédente(inférence) et la connaissance des zones de travail pour chaque intervalle des fonctions d'appartenance .

- **L'entité Division « couche de sommation ; « sortie »**

La dernière phase de défuzzification est la sortie qui est le dernier composant du contrôleur, cette entité effectue la division des deux membres de l'équation (III.16).

Elle permet finalement de donner la valeur de la sortie du rapport cyclique D qui sera appliquée au convertisseur continu-continu.

La figure V.4 donne le schéma détaillé du contrôleur MPPT neuro-flou.

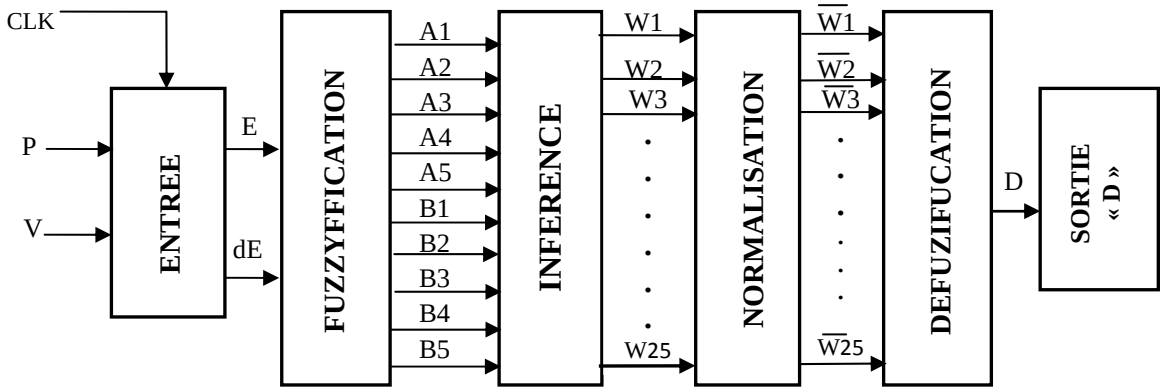


Figure V.4 : Schéma synoptique détaillé du bloc "MPPT neuro-flou ".

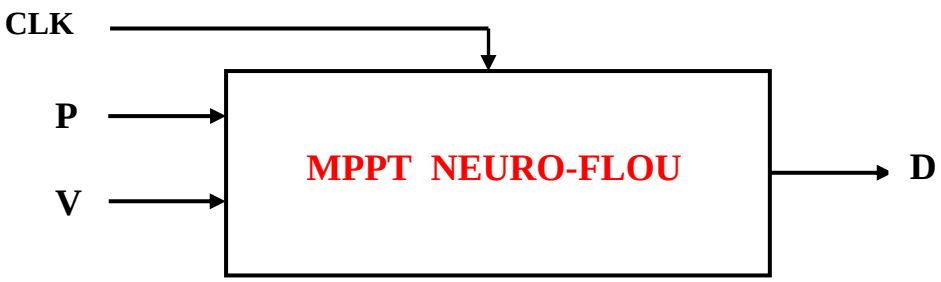


Figure V.5 : Schéma synoptique simplifié du bloc "MPPT neuro-flou ".

V.4. Environnement de test en vue d'une implémentation sur la carte FPGA

Afin de tester le bon fonctionnement des programmes écrits en VHDL sur la carte de développement Virtex-II V2MB1000, on a conçu un environnement de test qui exploite les ressources offertes par la carte de développement et permet de visualiser les résultats en temps réel.

Cet environnement de test est constitué de plusieurs blocs comme le montre la figure V.6. Cette dernière est suivie par une brève explication du rôle de chaque bloc dans l'environnement de test.

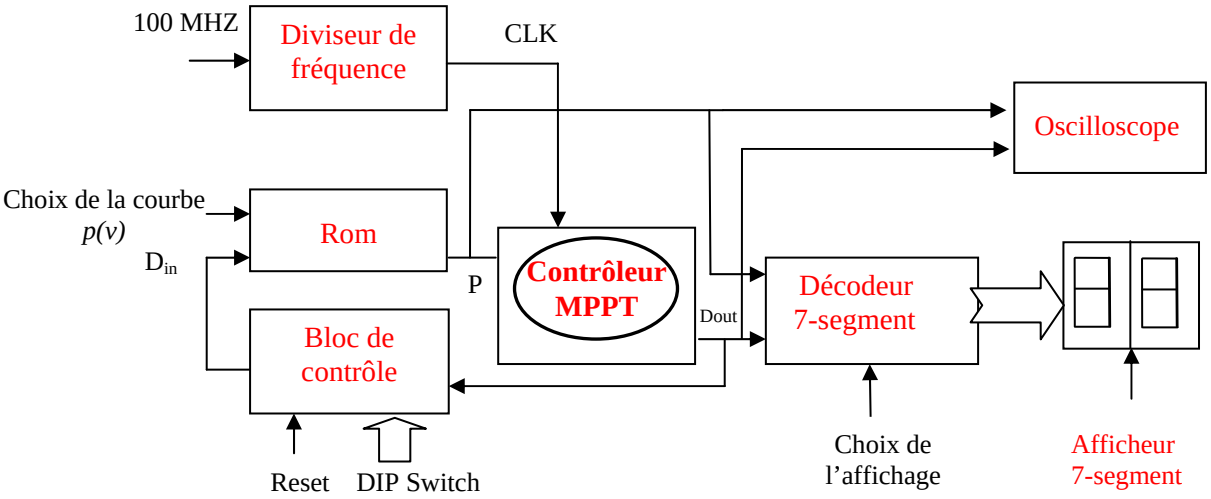


Figure V.6: Schéma synoptique détaillé de l'environnement de test.

- **Le "Diviseur de fréquence"**

Ce bloc a pour rôle de générer un signal d'horloge à 1Hz à partir du signal d'horloge interne de 100 MHz. En réalité le système travaille habituellement avec une fréquence de 100 Hz, mais pour les besoins du test et pour pouvoir visualiser les résultats à l'œil nu, on travaille avec une période d'horloge de 1 seconde.

- **Le "Décodeur 7-segments"**

Dans ce bloc, la valeur de P ou D_{out} (selon le choix de l'utilisateur) est traduite en format 7-segments et adaptée pour pouvoir être affichée sur les deux afficheurs7-segments disponibles sur la carte de développement.

- **Les "Deux afficheurs 7-segments"**

Ces deux afficheurs permettent de visualiser la valeur de la puissance P ou celle du rapport cyclique D en format hexadécimal. On peut ainsi suivre la variation de l'une de ces deux grandeurs (selon le choix de l'utilisateur) en temps réel et ainsi vérifier le bon fonctionnement du dispositif.

- **L'oscilloscope**

L'oscilloscope est le dispositif qui nous permet de visualiser le rapport cyclique « D » généré sous forme d'un signal PWM qui sert à commander l'hacheur MPPT.

- **Le bloc "ROM"**

Ce bloc contient la caractéristique puissance-tension d'un générateur photovoltaïque sous forme de tableau, il remplace donc dans notre test un système photovoltaïque complet. Il contient deux courbes différentes de la caractéristique puissance-tension correspondant à des conditions atmosphériques différentes d'ensoleillement et de température. Les données de ces courbes sont tirées des caractéristiques puissance-tension illustrées sur les figures I.5 et I.6 et le choix de la courbe désirée est effectué en appuyant sur le bouton poussoir "choix".

On a introduit deux courbes différentes de la caractéristique puissance-tension dans le bloc ROM pour disposer de deux points MPP différents. On peut ainsi estimer et comparer la rapidité de la réponse des deux méthodes de poursuite MPPT en cas de changement brusque des conditions météorologiques.

- **Le "Bloc de contrôle"**

Ce bloc permet de réinitialiser le système avec une nouvelle valeur du rapport cyclique en appuyant sur le bouton poussoir "Reset". L'utilisateur peut introduire la valeur initiale désirée du rapport cyclique D en agissant sur l'entrée à 8 bits "DIP Switch". Ces Switch peuvent être mis statiquement à 1 ou à 0 par l'utilisateur pour former n'importe quel mot binaire de 8 bits et ainsi balayer tout l'intervalle de variation du rapport cyclique D.

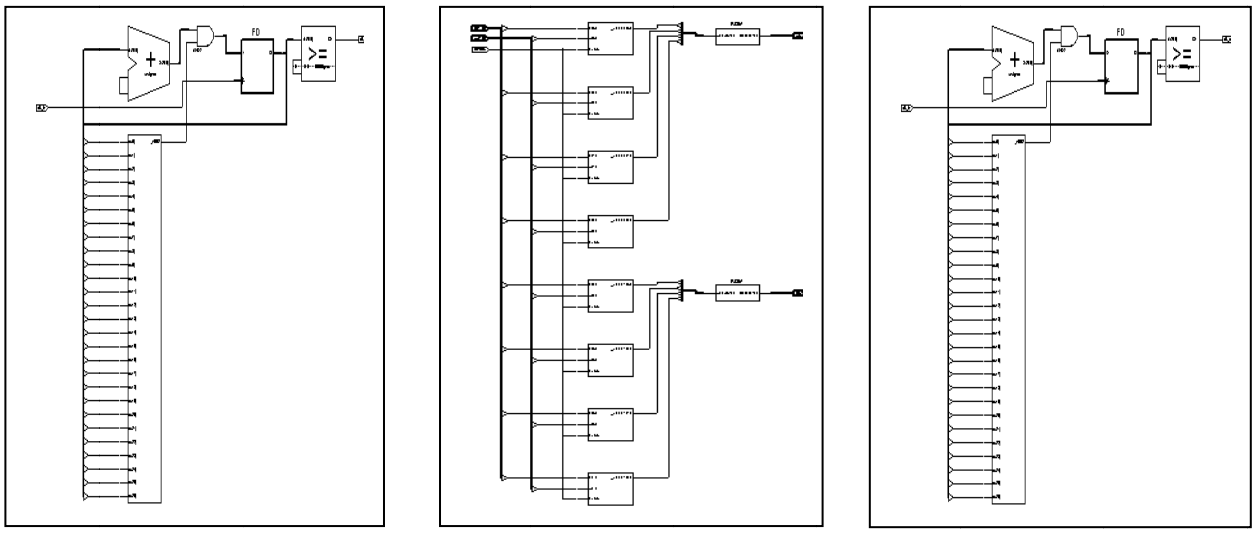
- **Le "Contrôleur MPPT"**

Ce bloc représente le bloc VHDL à tester, c'est-à-dire le cœur du programme. Pour la première méthode, on l'a remplacé par le bloc *"MPPT flou"*, et pour la seconde méthode par le bloc *"MPPT neuro-flou"*. Ces deux blocs ont été détaillés avec des schémas simplifiés et leur principe de fonctionnement a été expliqué plus haut.

V.5. Synthèse des programmes de l'environnement de test

Pendant l'étape de synthèse, le synthétiseur convertit le programme VHDL en portes logiques et bascules de base, c'est-à-dire en structure électronique. L'outil **« View RTL Schematic »** permet de visualiser les schémas équivalents générés par le synthétiseur pour chaque bloc du programme ainsi que la relation entre les différents blocs dans le programme principal.

Les schémas suivants représentent les différents blocs de l'environnement de test.



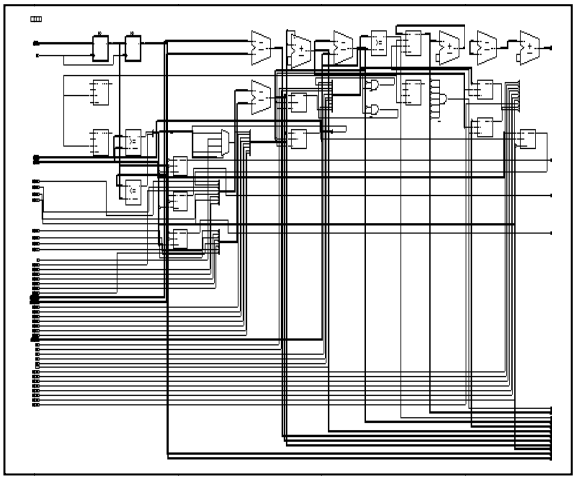
Bloc "Diviseur de fréquence" **Bloc "Décodeur 7- segments"** **Bloc Rom**

Figure V.7: Schémas équivalents des différents blocs de l'environnement de test

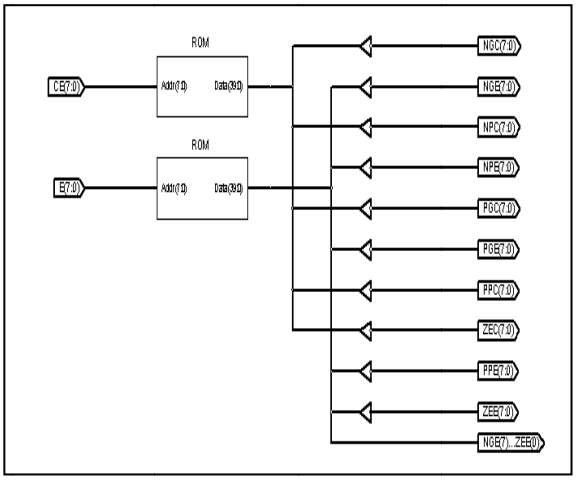
V.6. Implémentation du contrôleur MPPT flou sur la carte FPGA

V.6.1. Synthèse des programmes du contrôleur MPPT flou

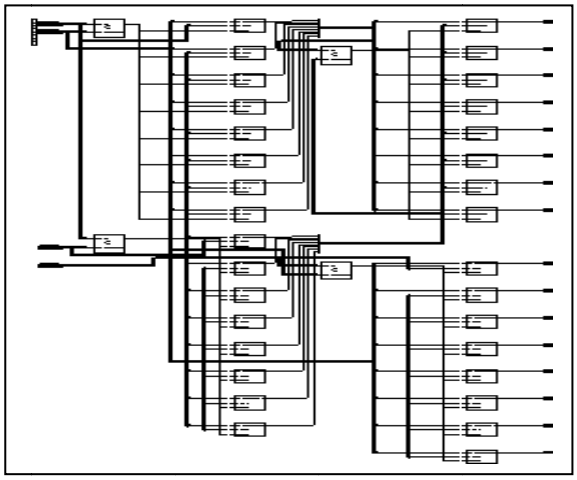
Les figures suivantes représentent les blocs constituants le contrôleur MPPT flou



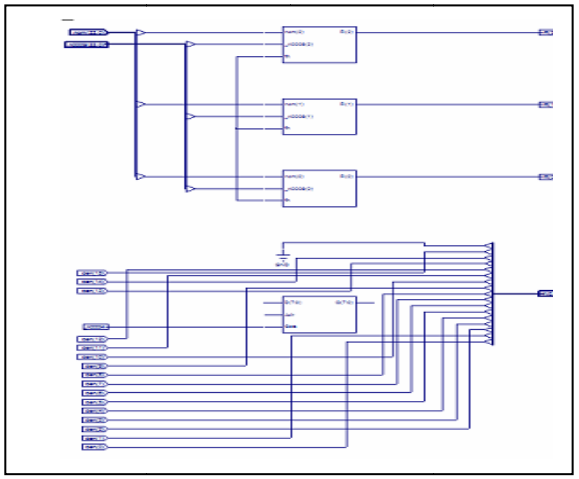
Bloc d'entrée



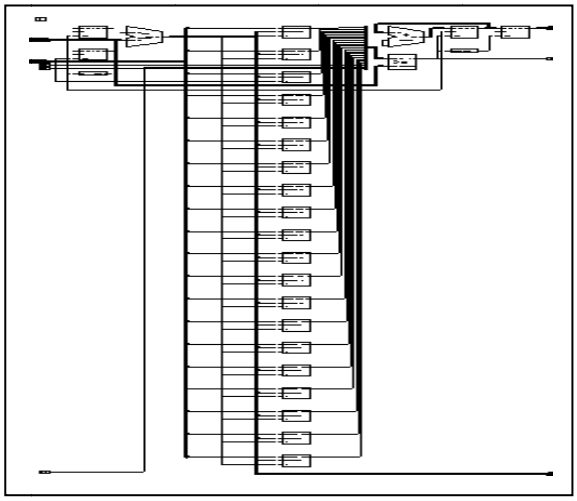
Bloc de fuzzification



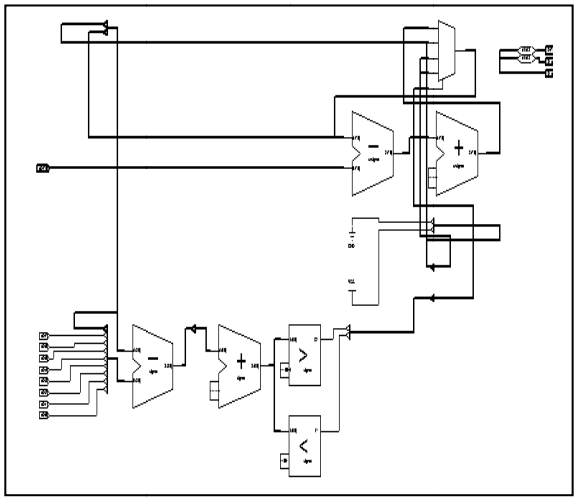
Bloc des inférences



Bloc de défuzzification



Bloc de division2



Bloc de sortie

Figure V.8 : Schémas équivalents des blocs constituants le " Contrôleur MPPT flou "

La figure V.9 montre le schéma équivalent du bloc VHDL "MPPT flou" dans son environnement de test.

On voit au milieu de la figure les trois blocs principaux du contrôleur flou ("Fuzzification", "Inférences" et "Défuzzification") ainsi que les blocs secondaires ("Entrée" et "Sortie").

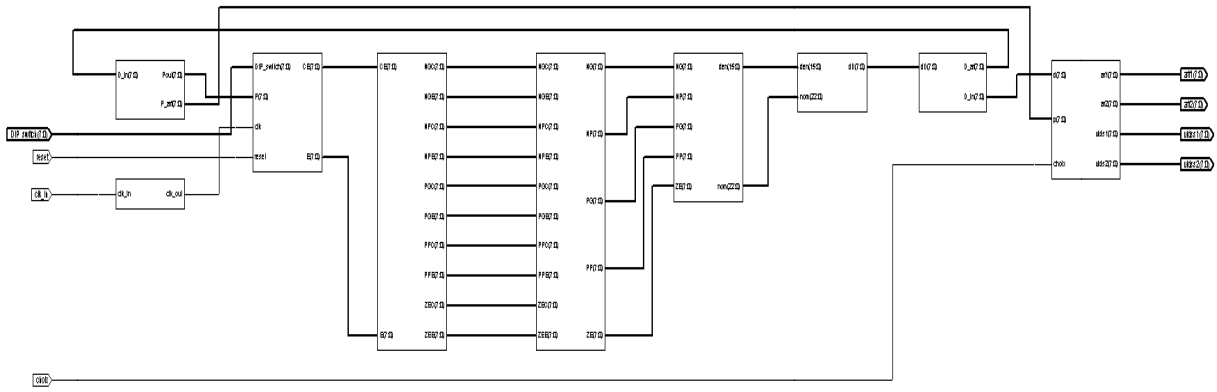


Figure V.9 : Contrôleur MPPT flou.

V.6.2. Simulation des programmes du contrôleur MPPT flou

Comme on l'a déjà cité dans le chapitre IV, cette opération nous permet de valider le projet à chaque étape du développement. C'est donc dans cette partie qu'on va pouvoir visualiser les résultats obtenus par la méthode floue et obtenir ses performances. Dans la première étape, on teste les performances du contrôleur flou dans la recherche du point MPP; On le force à commencer la recherche à partir d'un point donné et on calcule le temps nécessaire pour qu'il atteigne le point de puissance maximale MPP et se stabilise autour de lui. Dans la seconde étape, on agit sur la caractéristique puissance-tension en basculant d'une courbe à l'autre (changement brusque des conditions atmosphériques) afin que le contrôleur soit stabilisé autour du point MPP.

V.6.2.1. Recherche du point de puissance maximale

Dans cette étape, on initialise le contrôleur pour qu'il commence à travailler avec la valeur minimale du rapport cyclique (D=10%), on observe le comportement du contrôleur et on calcule le temps nécessaire pour qu'il atteigne le point MPP. La figure V.10 montre la réponse du contrôleur "MPPT flou" pour des conditions constantes d'ensoleillement et de température.

D'après les résultats obtenus, on remarque que le contrôleur converge rapidement vers le point MPP. La réponse du contrôleur MPPT flou est estimée par 12 cycles d'horloge seulement (ce qui est équivalent à 0,12s à 100 HZ).

Pour le contrôleur flou on peut constater que le pas d'incrémentation n'est pas constant, et que la recherche du point MPP se divise en deux phases, une phase rude où le pas dD est grand

pour atteindre rapidement la zone du point MPP, et une phase fine pour affiner la recherche et se stabiliser autour du point MPP en minimisant les ondulations.

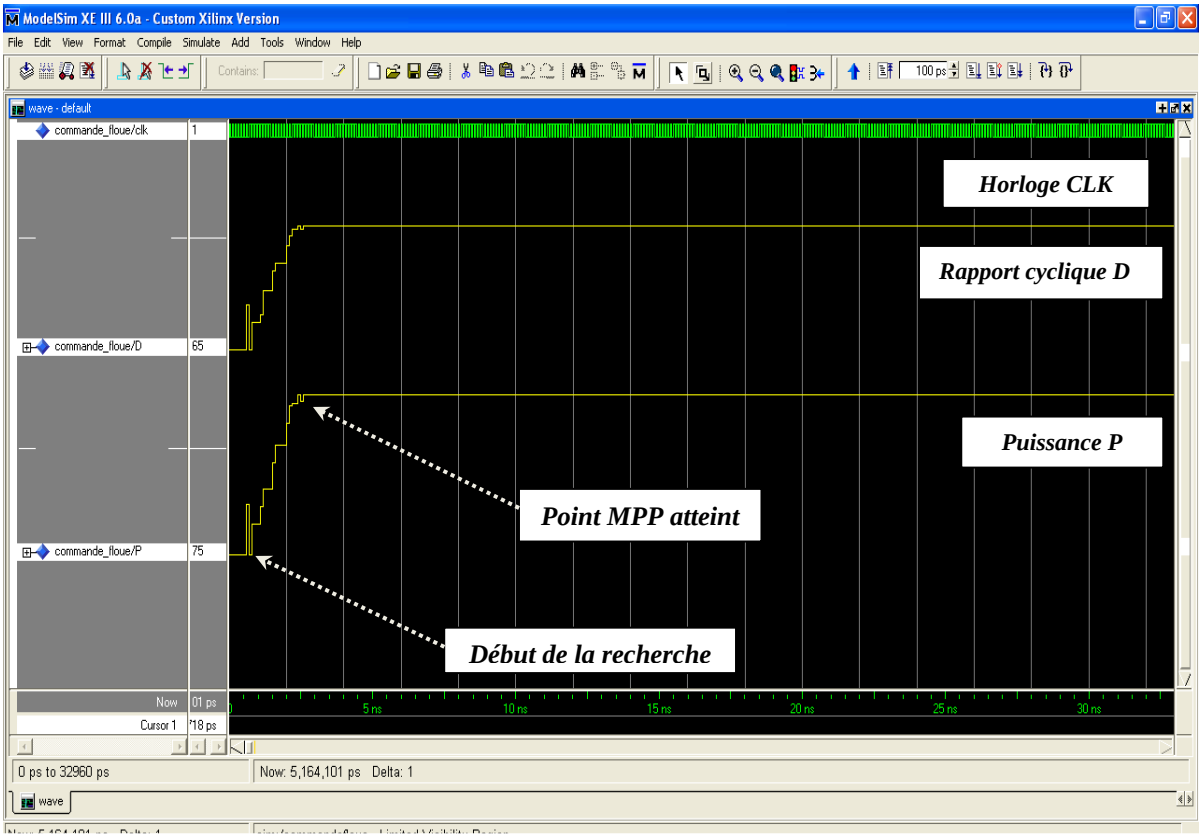


Figure V.10: Simulation de la recherche du point MPP par le contrôleur "MPPT flou".

V.6.2.2. Poursuite du point de puissance maximale (changement des conditions météorologiques)

Dans cette seconde étape, on observe la réponse du contrôleur lors de changements brusques des conditions météorologiques. On agit sur la caractéristique puissance-tension en basculant d'une courbe *p(v)* à l'autre par le bouton choix (on met deux courbes différentes *p(v)* selon les variations de température et d'ensoleillement) ; afin que le contrôleur se stabilise autour du point MPP, et on calcule le temps nécessaire pour qu'il atteigne le nouveau point MPP.

Pour le contrôleur MPPT flou (figure V.11), on remarque que la poursuite du point MPP est très rapide. En effet, lors d'une diminution brusque de l'ensoleillement, il trouve le nouveau point MPP après 3 cycles d'horloges seulement (0,03s à 100 Hz), et lors d'une augmentation brutale de l'ensoleillement, il ne met que 4 cycles d'horloge avant de se stabiliser à la valeur de Pmax.

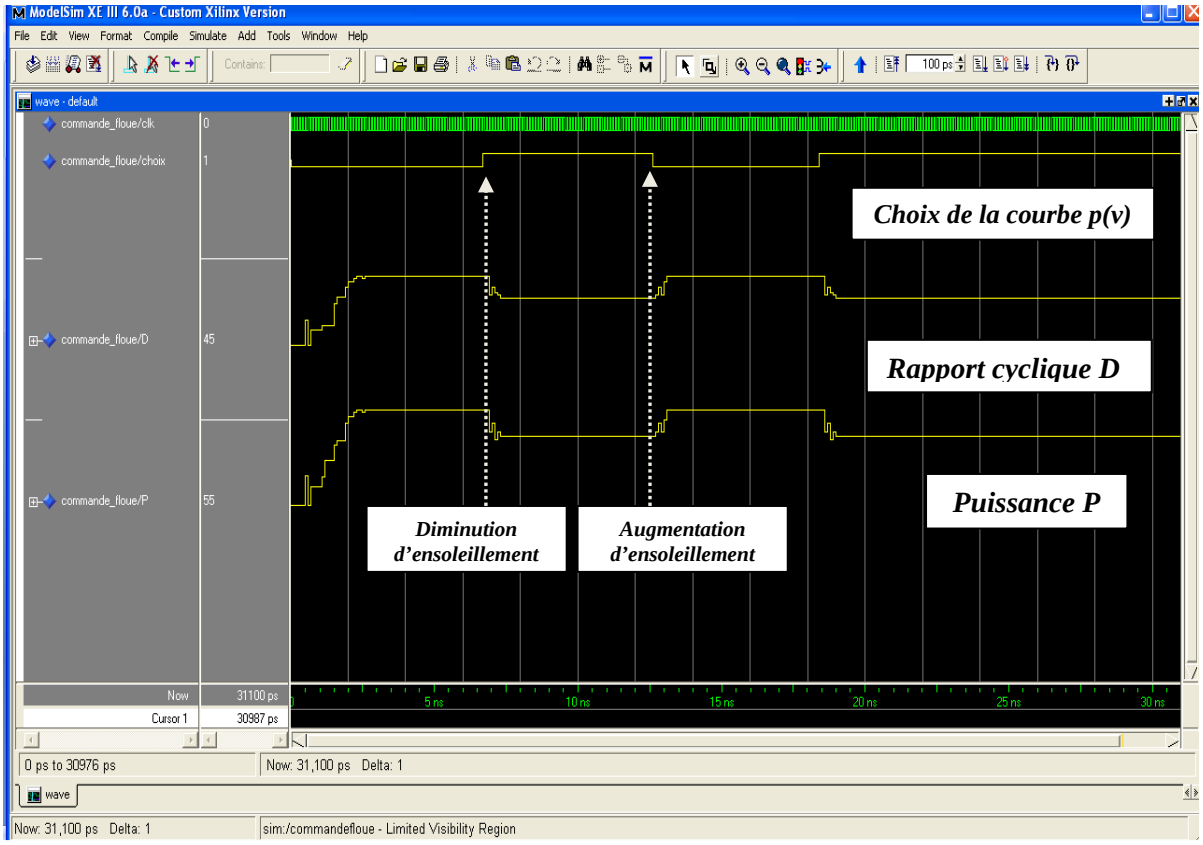


Figure V.11: Simulation de la poursuite du point MPP par le contrôleur "MPPT flou" pour des changements brusques des conditions météorologiques

V.6.3. Placement et routage des programmes du contrôleur MPPT flou sur le circuit FPGA

Dans cette étape, l'outil de placement et routage trace les routes sur le circuit FPGA afin qu'il puisse réaliser le fonctionnement attendu. L'outil « **FPGA Editor** » nous permet de visualiser le routage réalisé sur le circuit FPGA.

La figure V.12 montre le routage du contrôleur "MPPT flou" sur le circuit FPGA et la figure V.13 donne un aperçu sur l'affectation des broches E/S.

Le tableau V.1 contient les informations sur les ressources utilisées par les programmes du contrôleur MPPT flou. Ces informations sont fournies à la fin de l'étape de synthèse.

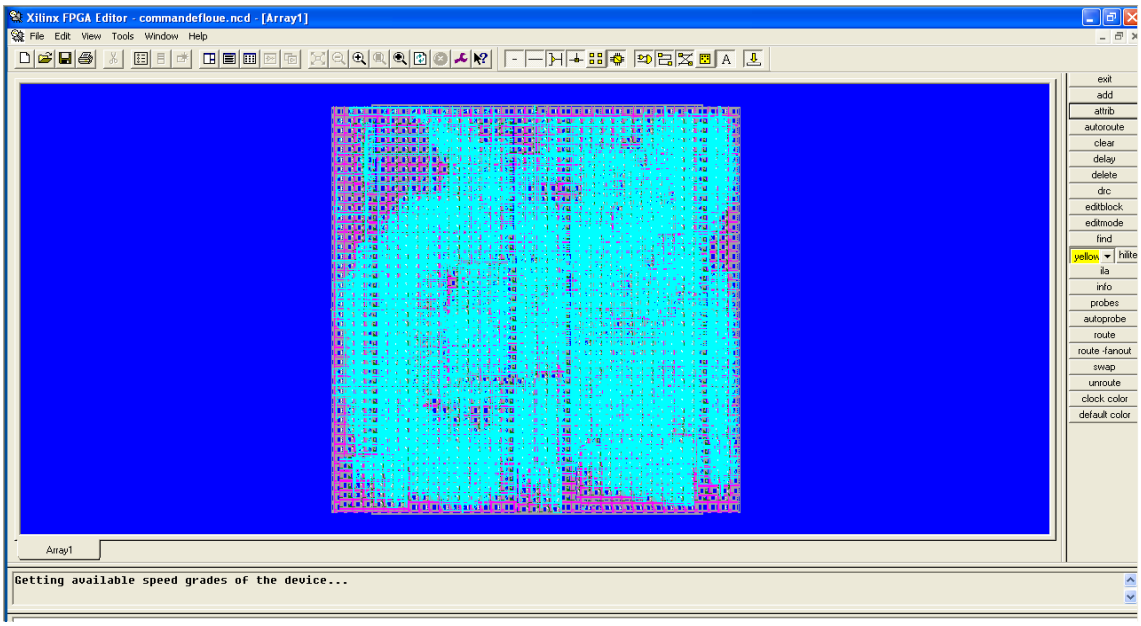


Figure V.12 : Routage du circuit FPGA pour le programme "MPPT flou ".

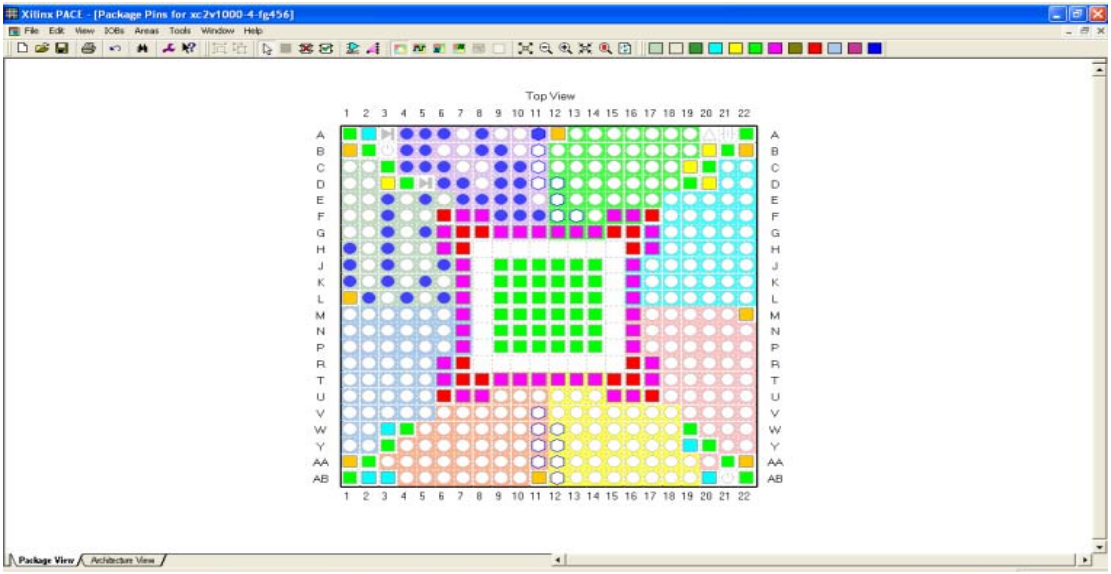


Figure V.13 : Aperçu de l'outil d'affectation des broches d'entrées/sorties pour le contrôleur MPPT flou.

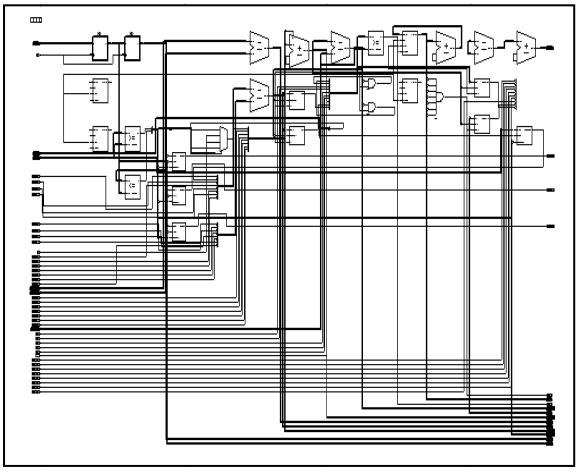
Ressources utilisées	**MPPT flou**		**Nombre total disponible**
Bascules	3536	68%	5120
Bascules «Flip-Flop»	255	2%	10240
Lut à 4 entrées	6706	67%	10240
IOB	27	8%	324
Multiplieurs 18X18	1	2%	40
Horloges	8	50%	16

Tableau V.1 : Ressources du circuit FPGA utilisées par la méthode floue.

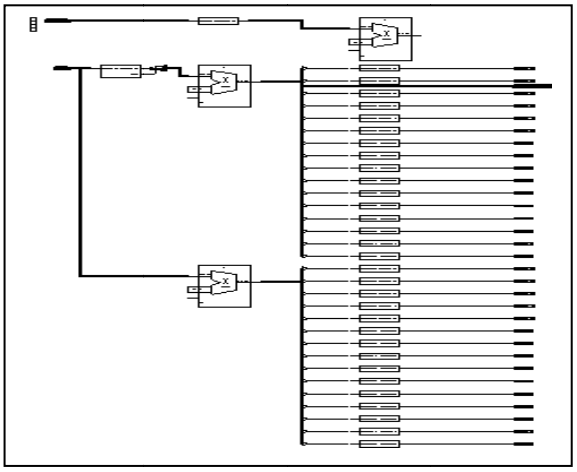
V.7. Implémentation du contrôleur MPPT neuro-flou sur la carte FPGA

V.7.1. Synthèse des programmes du contrôleur MPPT neuro-flou

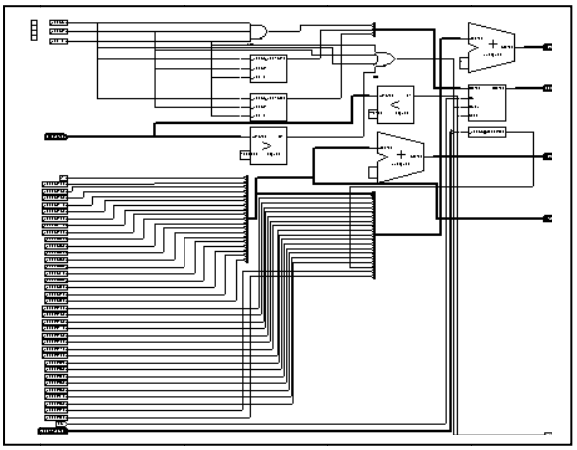
Les schémas suivants représentent les blocs (couches) constituants le MPPT neuro-flou.



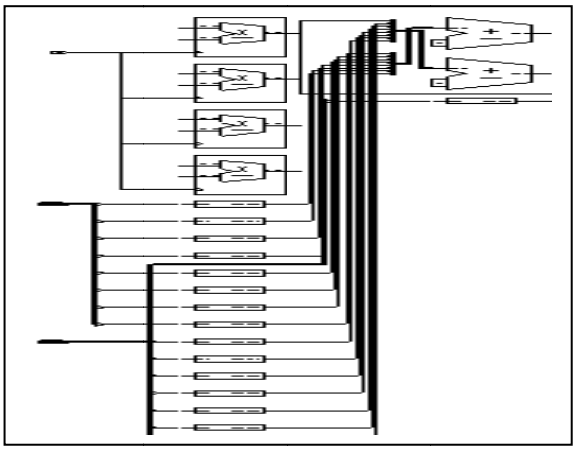
Bloc d'entrée



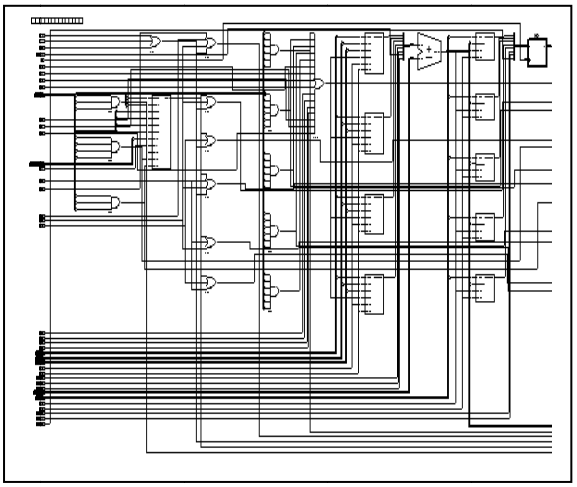
Bloc de fuzzification E



Bloc de fuzzification dE



Bloc d'inférence



Bloc de défuzzification

Bloc de sortie

Figure V.14 : Schémas équivalents des blocs constituants le "Contrôleur MPPT neuro-flou".

La figure V.15 montre le schéma équivalent du bloc VHDL "MPPT neuro-flou " dans son environnement de test.

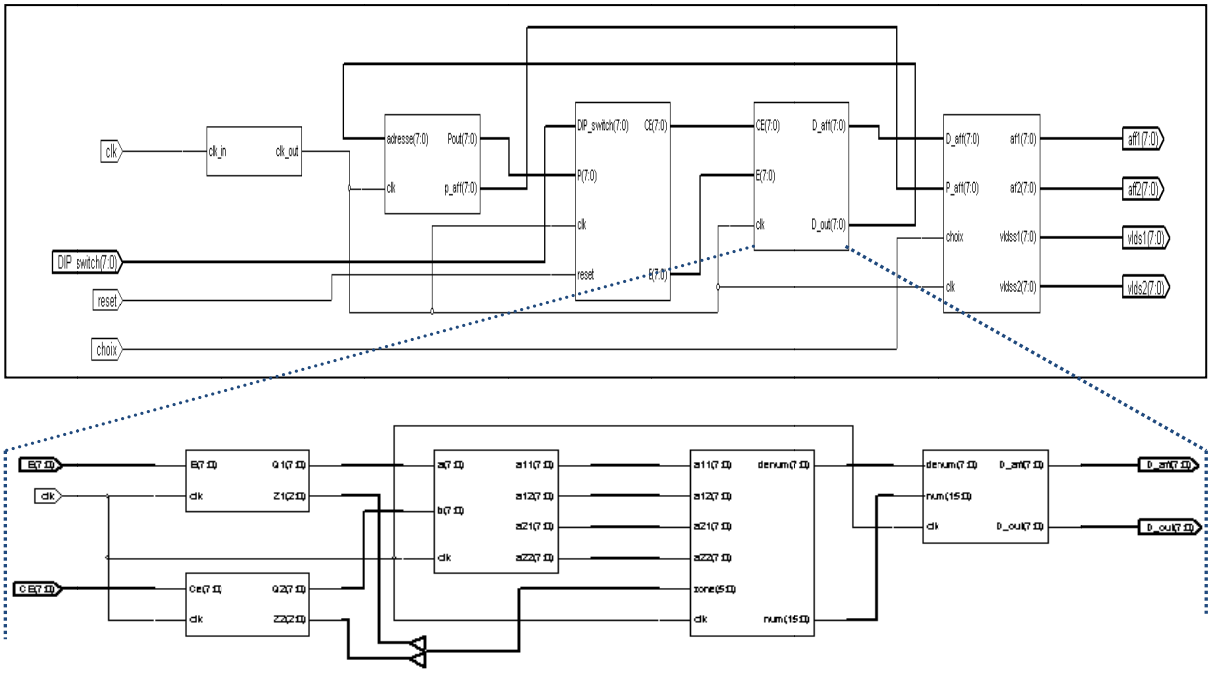


Figure V.15 : Contrôleur MPPT neuro-flou.

V.7.2. Simulation des programmes du contrôleur MPPT neuro-flou

Pour le contrôleur MPPT neuro-flou, on suit les mêmes étapes que celles du contrôleur flou, premièrement on force le contrôleur à commencer la recherche à partir du même point que celui du contrôleur flou pour qu'on puisse comparer après les performances des deux méthodes floue et neuro-floue dans la recherche du point MPP. On calcule le temps nécessaire pour qu'il atteigne le point de puissance maximale MPP et se stabilise autour de lui.

Dans la seconde étape, on agit aussi sur la caractéristique puissance-tension en basculant d'une courbe à l'autre (changement brusque des conditions atmosphériques) afin que le contrôleur neuro-flou se stabilise autour du point MPP.

V.7.2.1. Recherche du point de puissance maximale

On initialise le contrôleur pour qu'il commence à travailler avec la valeur minimale du rapport cyclique (D=10%), on observe le comportement du contrôleur et on calcule le temps nécessaire pour qu'il atteigne le point MPP.

La figure V.16 montre la réponse du contrôleur "MPPT neuro-flou" pour des conditions constantes d'ensoleillement et de température.

D'après les résultats obtenus, on remarque sur la figure V.16 que le contrôleur converge rapidement vers le point MPP. La réponse du contrôleur est estimée par 13 cycles d'horloge seulement (ce qui est équivalent à 0,13s à 100 HZ).

Pour le contrôleur neuro-flou, nous pouvons remarquer que les performances restent pratiquement similaires à celles de la technique floue, soit coté stabilité, soit coté temps de réponse; pour des conditions constantes d'ensoleillement et de température.

Figure V.16: Simulation de la recherche du point MPP par le contrôleur "MPPT neuro-flou".

V.7.2.2. Poursuite du point de puissance maximale (changement des conditions météorologiques)

Le contrôleur neuro-flou met 2 cycles d'horloges seulement lors d'une diminution brusque de l'ensoleillement et 3cycles lors de l'augmentation brusque d'ensoleillement avant de se stabiliser à la valeur maximale de puissance (figure V.17).

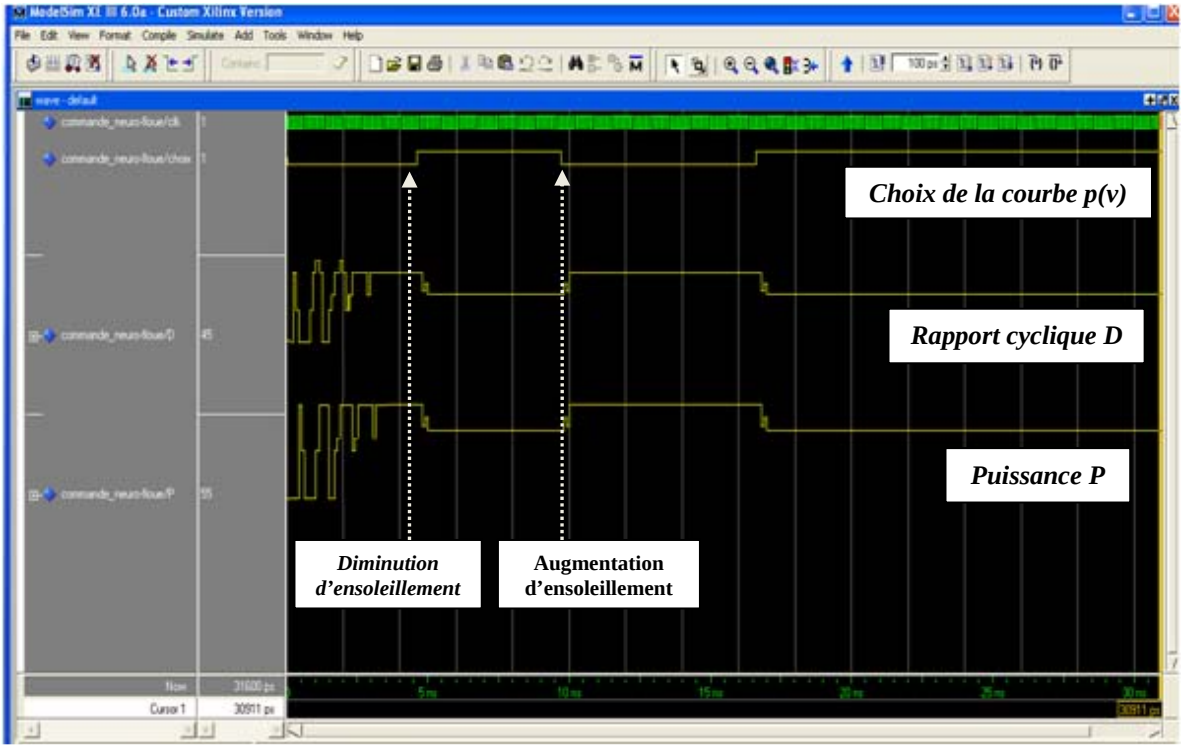


Figure V.17: Simulation de la poursuite du point MPP par le contrôleur "MPPT neuro-flou" pour des changements brusques des conditions météorologiques.

V.7.3. Placement et routage des programmes du contrôleur MPPT neuro- flou sur le circuit FPGA

La figure V.18 montre le routage du contrôleur "MPPT neuro-flou" sur le circuit FPGA et la figure V.19 donne un aperçu sur l'affectation des broches E/S.

Le tableau V.2 contient les informations sur les ressources utilisées par les programmes du contrôleur MPPT neuro-flou. Ces informations sont fournies à la fin de l'étape de synthèse.

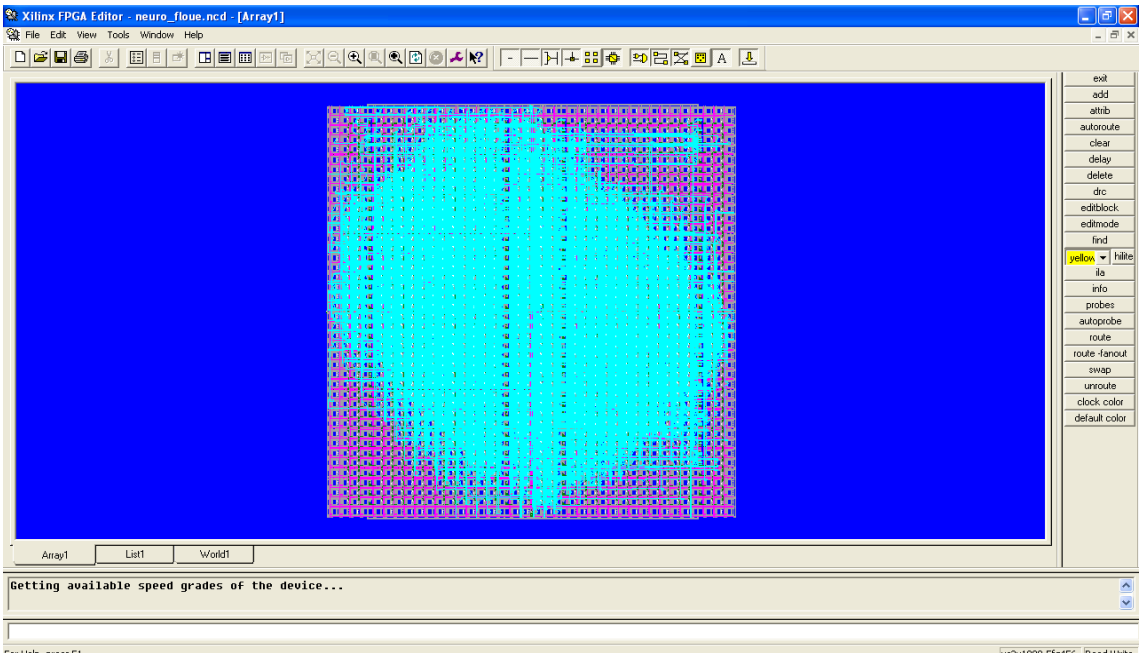


Figure V.18 : Routage du circuit FPGA pour le programme "MPPT neuro-flou ".

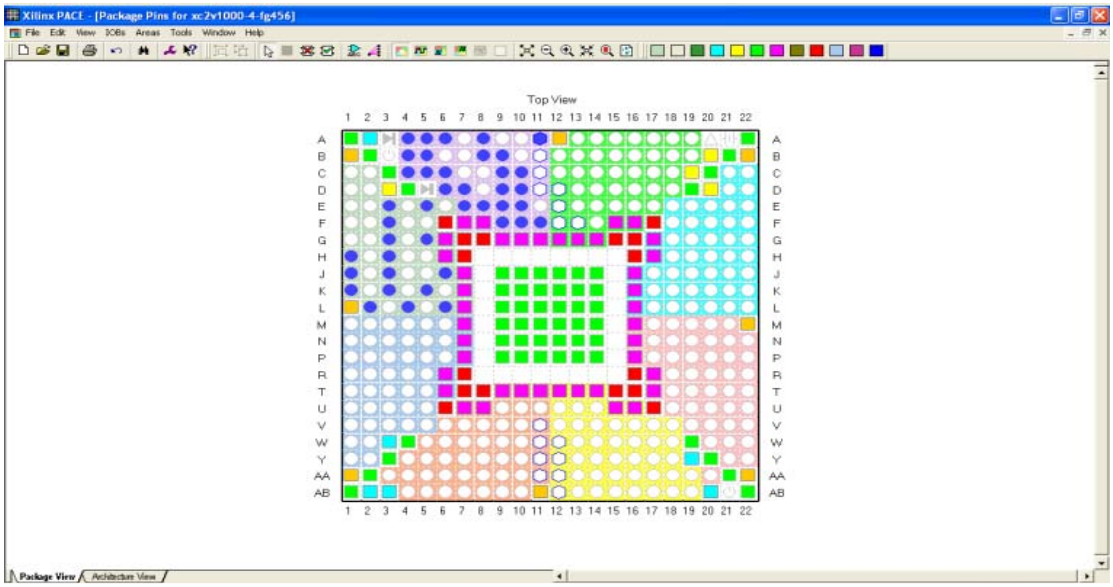


Figure V.19 : Aperçu de l'outil d'affectation des broches d'entrées/sorties pour le contrôleur MPPT neuro-flou.

Ressources utilisées	**MPPT neuro-flou**		**Nombre total disponible**
Bascules	3275	63%	5120
Bascules «Flip-Flop»	184	1%	10240
Lut à 4 entrées	6319	61%	10240
IOB	44	13%	324
Multiplieurs 18X18	14	35%	40
Horloges	2	12%	16

Tableau V.2 : Ressources du circuit FPGA utilisées par la méthode neuro-floue.

V.8. Interprétation des résultats et discussions

Les résultats de simulation obtenus, mettent clairement en évidence les capacités obtenues par l'approche neuro-floue, nous pouvons remarquer que les performances restent pratiquement similaires à ceux de la technique floue pour des conditions climatiques constantes; mais pour des conditions climatiques changeantes le contrôleur neuro-flou réagit plus rapidement que le contrôleur flou (un très court temps de réponse pour se stabiliser autour du point de puissance maximale)

Aussi, d'après les résultats obtenus de placement-routage du circuit FPGA, on peut conclure que l'avantage du contrôleur neuro-flou est qu'il occupe moins d'espace dans le circuit FPGA utilisé par rapport au contrôleur flou.

A partir des ces résultats, on peut dire que chacune des deux méthodes MPPT implémentées sur circuit FPGA a ses avantages et ses inconvénients.

L'avantage commun entre les deux méthodes est la rapidité des réponses, surtout pour des variations brusques des conditions de fonctionnement. Elles convergent rapidement vers le point de puissance maximale et se stabilisent autour de lui en annulant quasiment les ondulations autours de l'état stable.

Cependant, le plus grand inconvénient des deux méthodes réside dans la complexité de son implémentation ; en effet, elle demande une programmation lourde en VHDL.

On peut constater que le contrôle par l'approche neuro-flou, malgré sa difficulté de programmation avant de dire d'implémentation, montre son efficacité non seulement pour la poursuite du point de puissance maximale mais aussi pour son temps de réponse et sa stabilité.

La figure V.20 représente une photo prise au « Laboratoire des Dispositifs de Communication et de Conversion Photovoltaïque (LDCCP) de l'Ecole Nationale Supérieure Polytechnique. Les figures V.21et V.22 représente le signal du rapport cyclique D généré par le contrôleur neuro-flou sous forme PWM affiché par un oscilloscope numérique.

Figure V.20 : Affichage de la commande MPPT neuro-floue sur la carte VIRTEX II V2MB1000.

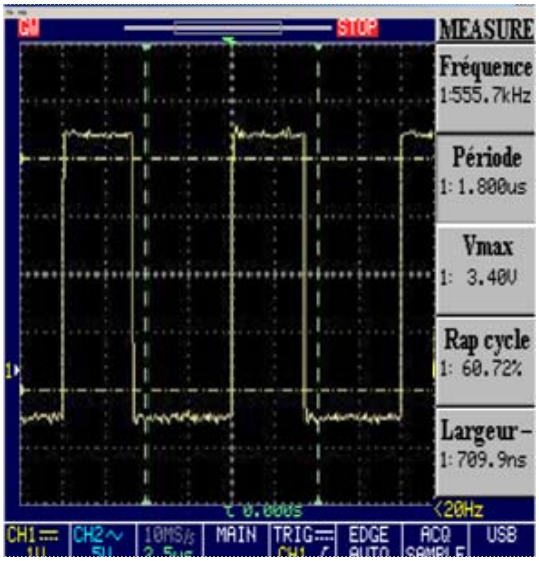


Figure V.21: Le rapport cyclique D généré par le contrôleur neuro-flou sous forme PWM affiché par l'oscilloscope sous des conditions climatiques constantes

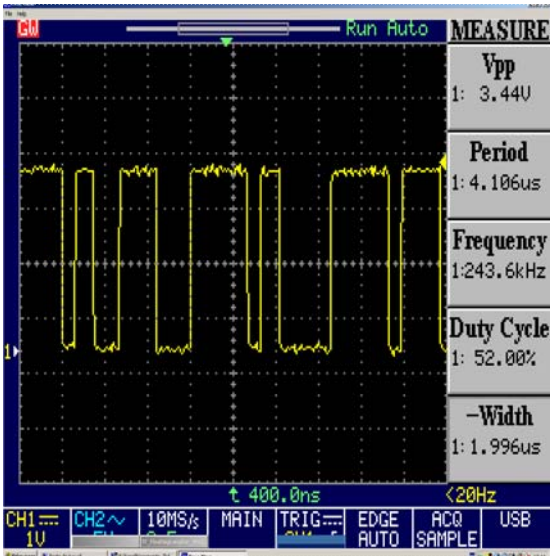


Figure V.22: Le rapport cyclique D généré par le contrôleur neuro-flou sous forme PWM affiché par l'oscilloscope sous des conditions climatiques variables

Conclusion générale et perspectives

La conversion photovoltaïque de l'énergie solaire est une des alternatives qui, depuis l'avènement des grands programmes spatiaux, a montré sa grande souplesse et son aptitude à fonctionner en milieu hostile. Il s'agit d'une solution intéressante aux moyens de production conventionnels. Cependant, l'énergie délivrée par ces générateurs est directement dépendante des conditions atmosphériques et est relativement coûteuse, en raison du prix toujours élevé de ces générateurs. Ses principaux atouts restent sa grande autonomie de fonctionnement possible, une durée de vie des générateurs garantie supérieure à vingt ans et une absence de pollution lors de la production d'énergie.

Pour assurer le fonctionnement d'un générateur photovoltaïque à son point de puissance maximale, des contrôleurs MPPT sont souvent utilisés. Ces contrôleurs sont destinés à la poursuite PPM et à minimiser ainsi l'erreur entre la puissance de fonctionnement et la puissance maximale de référence qui est variable en fonction de la charge et des conditions climatiques.

Dans le même but, plusieurs techniques de commande ont été associées dernièrement à la commande MPPT à savoir: la logique floue, les réseaux de neurones….etc.

Dans ce travail, on a tenté d'explorer et d'exploiter au maximum les ressources offertes par les circuits FPGA en implémentant deux méthodes MPPT différentes utilisées dans les systèmes photovoltaïques: la méthode du contrôleur flou, et une autre plus complexe ; celle du contrôleur neuro-flou.

On a commencé par introduire les notions théoriques sur l'énergie photovoltaïque, la logique floue, les réseaux de neurones et l'approche neuro-floue, qui sont des domaines de pointe de l'électronique moderne nécessaires pour la compréhension du but de ce travail.

On s'est ensuite intéressé aux circuits FPGA, et on a détaillé les différentes étapes de son implémentation passant par la synthèse et la simulation jusqu'au placement et routage et enfin la programmation du composant.

La simulation nous a permis de valider les programmes des deux méthodes avant de les charger sur le circuit FPGA. Elle nous a également aidé à comparer les performances des deux méthodes pour différentes conditions de travail. Grâce à elle, on a pu déterminer les avantages et les inconvénients de chacune de ces deux méthodes.

Si nous devions faire un choix entre ces deux méthodes, les deux paramètres à prendre en considération sont la complexité de l'implémentation et les performances de la méthode.
Si on privilégie la simplicité de l'implémentation, le choix, relativement, sera porté sur la méthode floue. Si on estime par contre que les performances de la méthode MPPT doivent être optimales, même pour une implémentation plus complexe, le choix, relativement aussi, sera porté sur la méthode neuro-floue mais on peut conclure que les deux contrôleurs donnent des bons résultats.

On peut constater que le contrôle MPPT par l'approche neuro-floue, malgré sa difficulté de programmation avant de dire d'implémentation, montre son efficacité non seulement pour la poursuite du point de puissance maximale mais aussi pour le temps de réponse et la stabilité. On peut conclure aussi que la sortie des signaux qui sont obtenus et visualisés sur oscilloscope démontre la faisabilité des commandes MPPT.
Ce travail se veut une initiation au développement des projets sur circuits FPGA dans le domaine des énergies renouvelables. On espère qu'il pourra contribuer à aider les personnes qui s'intéressent à ce domaine de la technologie dans l'élaboration de leurs travaux.

Comme perspective à ce travail, il serait intéressant de réaliser l'étage de puissance du hacheur et de tester ce contrôleur MPPT neuro-flou dans un système photovoltaïque.

BIBLIOGRAPHIE

[1] B. Flèche - D. Delagnes /*Energie solaire photovoltaique.doc* / juin 07.

[2] H.J. Möller, *Semiconductors for Solar Cells*, Artech House, Inc, Norwood, MA, 1993.

[3] H.KNOPF, *analysis, simulation, and evaluation of maximum power point tracking (MPPT) methods for a solar powered vehicle*, Master of Science in Electrical and Computer Engineering, Portland State University 1999.

[4] R. Gottschalg, M. Rommel, D.G. Ineld, H. Ryssel, *Comparison of different methods for the parameter determination of the solar cells double exponential equation*. In 14th European Photovoltaic Science and Engineering Conference (PVSEC), Barcelona, Spain, 1997.

[5] D-L. King, S. Igari, W. Watrta, *Solar efficiency Tables*, version 19+, Progress in photovoltaic's Research and applications, Vol. 10, pp. 55-61, 2002.

[6] C. Hua, J. Lin, C. Shen, *Implementation of DSP-controlled photovoltaic system with peak power tracking*, IEEE Transaction on industrial electronics, vol 45 No 1, February 1998.

[7] I. Etxeberria-Otadui, *"Les Systèmes de l'Electronique de Puissance dédiés à la Distribution Electrique – Application à la Qualité de l'Energie"*, Thèse de l'Institut Nationale Polytechnique de Grenoble (France), Septembre 2003.

[8] Brochure, *"Interrupteurs semi-conducteurs de puissance"*, Université de Savoie, France. Mai 1991.

[9] Hayashi, K. Koizumi, H. Ohashi, Y. Kurokawa, *"A Single-Phase Grid-Connected Inverter by Utilizing Ready-Made PWM Power IC"*, 9EEE ISIE 2006, July 9-12, 2006, Montréal, Québec, Canada.

[10] Abdelmoumene Toudeft, *Méthodes connexionnistes pour la commande des systèmes non linéaires: application a la régulation des rivières*, Thèse de doctorat de l'université Paris 6. décembre1998.

[11] F.Santos Osorio, *Un système hybride neuro-symbolique pour l'apprentissage automatique constructi*f, Thèse de doctorat, Institut National Polytechnique de Grenoble - I.N.P.G, Février 1998.

[12] N.CHIKHI, *Implémentation de la partie Feed-forward de l'algorithme de rétro-propagation du gradient sur un circuit FPGA de Xilinx*, Rapport d'activité, centre de recherche et de développement des technologies avancées d'Alger, 2004 .

[13] V. Andronova, *Utilisation de données météo et des réseaux de neurones pour la prédiction de vitesses de vent*, Projet de fin d'étude Master 2ème année, Université technique de Sofia - Université de Corse – Pasquale Paoli, juillet 2006.

[14] G.F.Tchoketch-Kebir, *Commande des hacheurs MPPT par la logique floue*, Mémoire de Magister, ENP, Algérie 2006.

[15] T. Obeidi, *Application des algorithmes génétiques dans la commande des hacheurs MPPT*, Mémoire de Magister, ENP, Algérie 2006

[16] T. Takagi, M. Sugeno, *Fuzzy identification of systems and its applications to modeling and control*, IEEE Trans. Syst, Man Cybern. 15(1985) 116–132.

[17] J.S.R. Jang, *ANFIS: Adaptive-Network-based Fuzzy Inference Systems*, IEEE Trans. Syst. Man Cybern. 23 (May/June (3)) (1993) 665–685.

[18] D.Ould Abdeslam, *Techniques neuromimétiques pour la commande dans les systèmes électriques*: *application au filtrage actif parallèle dans les réseaux électriques basse tension*, Thèse de doctorat, Université de Haute-Alsace, décembre 2005.

[19] E. Gauthier, *Gestion d'une flotte de véhicules autonomes à l'intérieur d'un parking haute densité*, Rapport d' étude, Institut National Polytechnique de Grenoble DEA Informatique Option Robotique Vision Image, juin 1995.

[20] E Gauthier, *Utilisation des réseaux de neurones artificiels pour la commande d'un Véhicule autonome*, Thèse de doctorat, Institut National Polytechnique de Grenoble, Janvier 1999.

[21] M. Geethanjali, S. Mary Raja Slochanal, M. Geethanjali, *A combined adaptive network and fuzzy inference system (ANFIS) approach for over current relay system*, Department of Electrical and Electronics Engineering, College of Engineering, Madurai-625 015, Tamilnadu, India, 2007

[22] Ying-Ming Wangn, Taha M.S, *An adaptive neuro-fuzzy inference system for bridge risk assessment*, Institute of Soft Science, Fuzhou University, Fuzhou 350002, PR China School of Mechanical, Aerospace and Civil Engineering, The University of Manchester, P.O. Box 88, Manchester M60 1QD, UK

[23] Y.Nakoula, *Apprentissage des modèles linguistiques flous, par jeu de règles pondérées*, Thèse de doctorat, Université de Savoie, 1997.

[24] K. SOBAIHI, *Etude et réalisation d'un hacheur de tracking (MPPT) à contre réaction de tension*, Mémoire de Magister, ENP, Algérie, 2003.

[25] W. Kleinkauf, F. Raptis, O. Haas ,*" Electrification with Renewable Energies, Hybrid Plant Technology for Decentrali- zed, Grid-Compatible Power Supply "*, Excerpt from Themes 96/97 Solar Ener- gy Association, Germany.

[26] SANDIA REPORT, *Status and Needs of Power Electronics for Photovoltaic Inverters*, SAND2002-1535, Unlimited Release, Printed, June 2002.

[27] B. Multon, O. Gergaud, H. Ben Ahmed, X. Roboam, S. Astier, B. Dakyo, C. Nikita, *Etat de l'art des aérogénérateurs, L'électronique de puissance, vecteur d'optimisation pour les énergies renouvelables,* Ed. Novelect-ecrin, pp. 97-154, 2002.

[28] M. S. Ait Cheikh, *Etude, Investigation et conception d'algorithmes de commande appliqués aux systèmes photovoltaïques*, Thèse de doctorat, ENP, Décembre 2007.

[29] Trishan Esram, Patrick L. Chapman, *Comparison of Photovoltaic Array Maximum Power Point Tracking Techniques*, IEEE Transactions on Energy Conversion, vol. 22, n°. 2, June 2007.

[30] S.Hadji, F.Krim, *Développement et implémentation d' un algorithme de poursuite du point de puissance maximale d'un système photovoltaïque à base d'algorithmes génétiques,* Colloque National sur l'Energie Solaire Proceeding, Bejaia, Algérie, pp 144-150, 2006.

[31] N.Patcharaprakiti, S.Premrudeepreechacharn, Y.Sriuthaisiriwong, *Maximum power point tracking using adaptive fuzzy logic control for grid-connected photovoltaic* system, Renewable Energy 30 (2005) 1771– 1788.

[32] F. Belhachat, *Commande neuro-floue d'un hacheur MPPT,* Mémoire de Magister, ENP, Algérie 2006.

[33] A. Benmosbah, *Implémentation sur FPGA d'un transpondeur à débit variable,* Rapport du stage de fin d'étude, Ecole nationale supérieure des télécommunications, Paris, Septembre 2007.

[34] MVD Training, Support de cours de formation « *Synthèse et simulation VHDL pour la conception de FPGA* », 2006.

[35] Wikipédia: Encyclopédie ouverte sur internet, Articles « Ethernet », « Gigabit Ethernet », « 8B/10B encoding » et « FPGA ».

http://en.wikipedia.org/wiki/Ethernet
http://en.wikipedia.org/wiki/Gigabit_Ethernet
http://en.wikipedia.org/wiki/8B/10B
http://en.wikipedia.org/wiki/FPGA

[36] MEMEC Design, Virtex-II V2MB1000 Development Bord user's Guide, http://legacy.memec.com/solutions/refernce/xilinx, Décembre 2002.

[37] XILINX, Virtex-II Platform FPGAs: Complete Data Sheet, http://www.xilinx.com, Mars 2005.