UNIVERSITÉ DU QUÉBEC À RIMOUSKI

CONTRÔLE MPPT D'UN SYSTÈME D'ÉNERGIE RENOUVELABLE À L'AIDE DE LA LOGIQUE FLOUE

Mémoire présenté

dans le cadre du programme de maîtrise en ingénierie

en vue de l'obtention du grade de maître ès arts

PAR

© MARC LAJOIE

Mai 2016

Composition du jury :

Hussein Ibrahim, président du jury, UQAR, Technocentre Éolien

Adrian Ilinca, directeur de recherche, UQAR

Drishtysingh Ramdenee , examinateur externe, ITMI

Dépôt initial le 31 mai 2016 Dépôt final le 7 novembre 2016

REMERCIEMENTS

Par le support qu'il a apporté tout au long du projet, le professeur Adrian Ilinca (directeur de thèse) m'a permis de participer activement à l'élaboration et au développement du contrôle MPPT du système photovoltaïque.

Que soit chaleureusement remerciés tous les intervenants, professeurs, étudiants et autres personnes œuvrant à l'UQAR, qui ont contribué de près ou de loin à la réalisation de ce projet.

Université du Québec à Rimouski, le 30 avril 2016
Marc Lajoie

RÉSUMÉ

Le but du projet est de développer un contrôleur MPPT basé sur l'utilisation de la logique floue. Plusieurs recherches ont démontrées que l'utilisation d'une approche de contrôle en logique floue est plus efficace que celle utilisant un simple contrôleur P&O Le principal objectif est de développer le contrôleur sur un circuit électronique portable, peu couteux, fiable, facile à programmer et à tester. Le choix du GUI et du langage de programmation est important également. Ils doivent être bien documentés et faciles à apprendre. Le langage de programmation doit être suffisamment de bas niveau pour permettre d'optimiser le code au besoin. Le système Raspberry PI répond bien à la plupart de ces exigences. C'est un petit ordinateur qui offre toutes les possibilités d'un ordinateur et possède les périphériques de communication nécessaires pour 'parler' avec des circuits simples ou des microcontrôleurs. Les interfaces de développement, programmes et librairies du RPI sont bien documentés. Cet outil permet donc de développer rapidement une application et de passer tout aussi rapidement aux tests. L'optimisation et les corrections se font facilement. L'application MPPT développée dans ce projet est de type Mamdani avec Max-Min. Les tests unitaires ont donnés de bons résultats. La version 1 des fonctions développées est fonctionnelle.

Mots clés : MPPT, FPGA, Logique floue, RaspBerry PI II, Linux, Geany, Python, C.

ABSTRACT

The project purpose is to develop a MPPT controller based on the use of fuzzy logic. Research has shown that the use of a control approach fuzzy is more effective than using a single controller P&O's. Main objective is to develop the controller on a portable electronic circuit, inexpensive, reliable, easy to program and to test. The choice of GUI and programming language are also important. They must be well documented and easy to learn. The programming language must be sufficiently low level to help optimize the code as needed. The Raspberry PI system meets most of the requirements. It is a small computer that offers all possibilities of a computer and has the communication devices needed to 'talk' with simple circuits or microcontrollers. Interfaces, programs and libraries of the RPI are well documented. This tool makes it possible to quickly develop an application and move just as quickly to the tests. The optimization and corrections are easily made. The MPPT implementation developed in this project is Mamdani kind with Max-Min. Unit tests have given good results. Version 1 of developed functions is functional.

Keywords: MPPT, FPGA, Fuzzy Logic, Raspberry Pi II, Linux, Geany, Python, C.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS, DES SIGLES ET DES ACRONYMES

BCM	Broadcom
DPI	Display Parallel Interface
FPGA	Field Programmable Gate Array
GUI	Graphical User Interface
I	Courant
LED	Light Emitting Diode
MPPT	Maximum Power Point Tracking
P	Puissance
P&O	Pertube & Observe
RPI	Raspberry PI
R&D	Recherche et Développement
SPI	Serial Peripheral Interface
UART	Universal Asynchronous
UQAR	Université du Québec à Rimouski
USB	Universal Serial Bus
V	Volt

LISTE DES SYMBOLES

σ	Sigma – écart-type de la distribution gaussienne
&	Et

CHAPITRE 1
INTRODUCTION GÉNÉRALE

L'orientation du Laboratoire de Recherche en Énergie Éolienne de l'UQAR vers des applications en énergies renouvelables surtout en énergie éolienne et solaire est dictée par l'essor de ces filières dans le portefeuille énergétique du Québec. Le développement de 4000MW d'énergie éolienne prévu dans la stratégie énergétique du Québec, élaborée en 2005 [1] s'est traduit par des investissements majeurs, de l'ordre de 10 milliards de dollars, ainsi que par une forte demande de formation de personnel et d'aide en R&D de la part de l'industrie éolienne.

Les filières éolienne et solaire de production d'énergie sont actuellement en pleine effervescence. Beaucoup de ressources sont impliquées dans ces secteurs de recherche. Ces sources d'énergie sont non polluantes et renouvelables. Elles seront de plus en plus utilisées et étudiées, surtout dans les régions où les autres sources d'énergie sont moins performantes et plus coûteuses et pour répondre aux impératifs mondiaux de réduction des émissions de gaz à effet de serre.

Parmi les objectifs de développement des filières d'énergies renouvelables se retrouve l'optimisation des rendements de conversion en énergie électrique. Les recherches et expérimentations récentes dans le domaine de la conception et la réalisation de contrôle solaire et éolien montrent qu'il reste de la recherche et des améliorations à apporter. Les articles ont montré que certains résultats avaient été atteints à l'aide de données préenregistrées et des systèmes informatiques variés. La précision des contrôles reste à améliorer.

Différentes techniques d'optimisation sont disponibles et appliquées lors des étapes de conception des systèmes de conversion d'énergies renouvelables [2]. Une de ces techniques est le suivi du point de performance maximal, le MPPT (Maximum Power Point Tracking) qui consiste à modifier certains paramètres de manière à maximiser la quantité d'énergie renouvelable convertie en électricité [3][4][5]. Les algorithmes d'intelligence artificielle sont très populaires pour le contrôle des paramètres permettant le suivi du point de puissance maximale (MPPT) [6][7]. Parmi ces algorithmes d'intelligence artificielle, la logique floue a démontré des résultats très intéressants pour les applications dans le domaine des énergies renouvelables [8][9][10].

Ce projet porte sur l'étude, la conception et le test d'un système de contrôle des systèmes énergétiques renouvelables utilisant la programmation en logique floue.

Le présent document montre la planification et la réalisation du projet. Les différentes hypothèses sont analysées à l'aide d'une revue bibliographique de chaque domaine pertinent. Les étapes de la conception préliminaire sont décrites. La conception détaillée et l'analyse proprement dite du logiciel sont développées. Finalement, une section porte sur le développement du prototype et les tests livrés.

Une synthèse des principaux éléments du rapport et les développements recommandés sont présentés en conclusion.

CHAPITRE 2
SITUATION ACTUELLE

SITUATION ACTUELLE

Les contrôleurs MPPT récents sont disponibles actuellement sur des circuits qui utilisent des microcontrôleurs. La compagnie australienne AERL en fabriquait déjà en 1985. Récemment, les coûts des microcontrôleurs ont baissé suffisamment pour équiper également les systèmes plus petits (moins de 1 KW). Les contrôleurs MPPT sont maintenant manufacturés par plusieurs compagnies, tel Xantrex, Morningstar, Blue Sky Energy et quelques autres [17].

Les microcontrôleurs FPGA modernes sont maintenant offerts intégrés sur un circuit contenant un processeur, de la mémoire et des composants d'entrée/sortie. Ils peuvent même être accompagnés d'un 'software' dédié qui facilite la reprogrammation des FPGA. XILINK est une des compagnies qui fabrique ces produits [42][43][44].

Raspberry PI est un microcontrôleur dynamique qui offre toutes les possibilités d'un ordinateur, mais à coût moindre. Son noyau d'exploitation est basé sur Linux (Debian par défaut) et plusieurs langages de programmation sont disponibles, tels Python, C, etc. Le site 'Raspberry PI foundation' est accessible via [30][31].

RECHERCHE BIBLIOGRAPHIQUE

NUR ATHARAH KAMARZAMAN, CHEE WEI TAN, « A COMPREHENSIBLE REVIEW OF MAXIMUM POWER POINT TRACKING ALGORITHMS FOR PHOTOVOLTAIC SYSTEM », *RENEWABLE AND SUSTAINABLE ENERGY REVIEW*S, 37, 585-598, 2014.

Cet article présente les techniques existantes de MPPT, des modèles équivalents de cellules PV, une analyse et comparaison des techniques MPPT conventionnelles et stochastiques en tenant compte de la complexité du design, du coût, de la sensitivité en ce qui concerne des changements environnementaux et la vitesse de convergence.

CHEHOURI ADAM, GHANDOUR MAZEN, LIVINTI PETRU, « A REAL TIME SIMULATION OF A PHOTOVOLTAIC WITH MAXIMUM POWER POINT TRACKING », LEBANESE UNIVERSITY, FACULTY OF ENGINEERING , UNIVERSITÉ 'VASILE ALECSANDRI' DU BACAU.

Cet article présente une étude expérimentale pour ce qui est du système électronique de contrôle qui peut localiser et retracer le MPPT du PV pour assurer un transfert efficient de l'énergie (le système MPPT est développé en utilisant la méthode 'Incremental Conductance' (IC)).

ANUP ANURAG, SATARUPA BAL, SUMAN SOURAV, MRUTYUNJAYA NANDA, « A REVIEW OF MAXIMUM POWER-POINT TRACKING TECHNIQUE FOR PHOTOVOLTAIC SYSTEMS », *INTERNATIONAL JOURNAL OF SUSTAINABLE ENERGY*, 2014

Cet article, publié en avril 2014, décrit l'importance de choisir la technique MPPT la plus appropriée pour une application PV spécifique en tenant compte des facteurs comme le coût, la complexité, la précision pour obtenir une efficacité maximum du système.

MELLIT, H. RESSOUK, A. MESSAI, B. MEDJAHED, « FPGA-BASED REAL TIME IMPLEMENTATION OF MPPT-CONTROLLER FOR PHOTOVOLTAIC SYSTEMS », *RENEWABLE ENERGY*, 6, 1652-1661, 2011

Cet article décrit un algorithme MPPT basé sur la méthode perturbation et observation (P&O - Perturb and Observe) qui a été appliquée en utilisant des FPGA (Field Programmable Gate Array).

G.F. TCHOKETCH KEBIR, T.OBEIDI, A. ILINCA, C. LARBES, D. RAMDENEE, « DESIGN AND SIMULATION OF A NEW, HIGH PERFORMANCE FUZZY MPPT SOLAR ENERGY SYSTEM », SOUMIS À RENEWABLE AND SUSTAINABLE ENERGY REVIEWS, 2016

Cet article présente une stratégie de contrôle MPPT basée sur la logique floue qui offre une performance significativement plus intéressante que le mode de contrôle classique basé sur le P&O.

T. OBEIDI, G.F. TCHOKETCH KEBIR, A. ILINCA, C. LARBES, D. RAMDENEE, « MPPT FOR A SOLAR ELECTRIC VEHICLE WITH AN INTELLIGENT FUZZY CONTROLLER, 2015 », SOUMIS À ARABIAN JOURNAL OF SCIENCE AND ENGINEERING

Cet article décrit le potentiel de l'amélioration de la conversion d'énergie d'un système PV associé à un repère mobile en utilisant un algorithme de logique floue.

EFTICHIOS KOUTROULIS, FREDE BLAADJERG, « OVERVIEW OF MAXIMUM POWER POINT TRACKING TECHNIQUES FOR PHOTOVOLTAIC ENERGY PRODUCTION SYSTEMS », *ELECTRIC POWER COMPONENTS AND SYSTEMS*, 43:12, 1329-1351, D01: 10.1080/15325008.2015.1030517

Cet article publié en 2015 présente les principes d'opération des techniques d'implémentation des processus des MPPT dans un système PV.

FREDE BLAABJERG, DAN M. IONEL, « RENEWABLE ENERGY DEVICES AND SYSTEMS – STATE-OF-THE-ART TECHNOLOGY, RESEARCH AND DEVELOPMENT, CHALLENGES AND FUTURE TRENDS », ELECTRIC POWER COMPONENTS AND SYSTEMS, 43:12, 1319-1328, DOI: 10.1080/15325008.2015.1062819.

Cet article, publié en juillet 2015, présente une révision des technologies actuelles pour les énergies renouvelables, les fondamentaux des turbines à vent et des systèmes PV, l'importance des électroniques de puissance dans l'intégration des énergies renouvelables et les systèmes d'entreposage de l'énergie ainsi que les défis à relever dans le futur dans ce domaine.

Les sections suivantes contiennent les étapes permettant de définir le modèle informatique utilisé pour l'implantation de la stratégie MPPT.

2.1.1 Objectifs

Le projet consiste à faire l'analyse et la programmation d'un contrôleur MPPT sur un outil portatif, complet, facilement configurable et utilisant les dernières technologies.

Ces outils modernes offrent de la puissance (en termes de vitesse de calcul) et des interfaces GUI et de communication matérielle. Ils réduisent le temps de développement et sont bien documentés. Ils remplissent un rôle à la fois éducatif et professionnel.

Le programme, développé à même l'outil, peut être adapté et modifié par la suite. Cette souplesse n'est pas négligeable, dans un cadre de recherche et d'optimisation futures.

2.1.2 Méthodologie

Cette section sert à proposer, selon une méthode éprouvée de développement de projet en génie, les solutions technologiques aux différents problèmes posés dans le cadre de ce projet. Une solution viable est proposée dans les paragraphes suivants :

L'outil de programmation embarqué choisi est le Raspberry Pi II (B+).

L'outil de développement choisi est le langage C. Python est un choix alternatif valide.

La méthode de développement retenue est une approche standard. Les fonctions à développer sont simples et doivent être peu gourmandes en temps machine. Les temps de

calcul des lignes de codes sont à connaître et à contrôler afin de chercher à optimiser le temps réponse du système, quoiqu'une réponse des temps de calcul inférieure à une demi-seconde soit acceptable. Ce système de contrôle MPPT fait partie d'un système électromécanique qui ne demande pas des temps réponses ultrarapides, de l'ordre des millisecondes.

CHAPITRE 3
ASPECTS THÉORIQUES

CONCEPTION DETAILLEE

3.1.1 Présentation de l'analyse

Description globale

En ce qui concerne l'utilisation du RPI II, il y a 2 volets à considérer dans ce projet : le premier est l'aspect communication entre le RPI II et le système global et le second est le choix du langage et du GUI.

Le port GPIO du RPI offre plusieurs types de communication : DPI, SPI, UART, etc. [11]. LE DPI, par exemple, permet de lire des valeurs binaires (0 ou 1) aisément.

Le langage de programmation C est un langage largement utilisé et permet de faire de la programmation de bas niveau, au niveau des registres d'entrées/sorties. Si nous désirons faire de l'optimisation au niveau de la vitesse et réduire la quantité de code, c'est un excellent choix. RPI propose un GUI tel Geany qui offre un environnement de développement adéquat.

La documentation sur l'utilisation du RPI II et plusieurs exemples sont disponibles et accessibles sur le WEB. Ceci devient important lorsqu'il faut maitriser un nouveau produit rapidement.

Un autre aspect à considérer est le contrôle des temps de calcul du programme et des temps de lecture/écriture des informations dans les registres du port de communication lors des échanges. Selon le choix du type de communication, la synchronisation avec le système global peut s'avérer importante. Idéalement, le RPI devrait simplement 'vérifier'

la présence de données en entrée, traiter les données et écrire le résultat dans les registres de sortie. Le système global aurait alors la responsabilité de gérer la synchronisation. La seule contrainte, dans ce cas, serait de mesurer les temps de calcul minimaux du programme développé pour le RPI.

Communication

Voici les solutions proposées concernant l'interface de communication entre le système général et le RPI II. Trois (3) méthodes de l'utilisation du port de communication GPIO sont présentées dans les paragraphes suivants. Le schéma du GPIO est présenté dans la figure suivante.

Le système global doit envoyer deux valeurs en entrée (Input I et II) et lire le résultat calculé par le programme du Raspberry PI II en sortie (Output).

Solution 1 : Bits de contrôle

Deux bits de contrôle (par exemple BCM 2 et BCM 3) sont utilisés pour que les systèmes se synchronisent. Les bits de valeur (par exemple BCM14, BCM15, BCM18, BCM23, BCM24, BCM25, BCM8 et BCM7) sont en lecture ou écriture selon les valeurs contenues dans les bits de contrôle). Les tableaux suivants résument la solution.

Tableau 1 : Bits de contrôle

BCM2	BCM3	Résultats
0	0	-
0	1	Écriture Input 1
1	0	Écriture Input 2
1	1	Lecture Output 0

Tableau 2 : Bits de valeur

Port	Bit de valeur
BCM14	0
BCM15	1

BCM18	2
BCM23	3
BCM24	4
BCM25	5
BCM8	6
BCM7	Signe: 0: +; 1 -

L'intervalle de valeurs lues ou écrites varie entre +127 et -127.

Voici un exemple:

Bits de contrôle: BCM2 = 1 et BCM3 = 1

Bits de valeurs: BCM14 = 1, BCM15= 1, BCM18= 1, BCM23=1, BCM24=1, BCM25=1, BCM8=1 et BCM7 = 0

Résultat: Lecture Output 0, valeur 127

Le temps machine de calcul des fonctions de lecture des registres GPIO et du 'fuzzy' (du RPI II) doit être établis afin d'assurer que la synchronisation entre le système global et le RPI II soit bonne.

Solution 2

Il y a 28 ports BCM. Donc Input1, Input2 et Output0 peuvent utiliser chacun 8 'pins' dédiés et différents. Les connexions du système global et la synchronisation sur le GPIO seront alors plus simples.

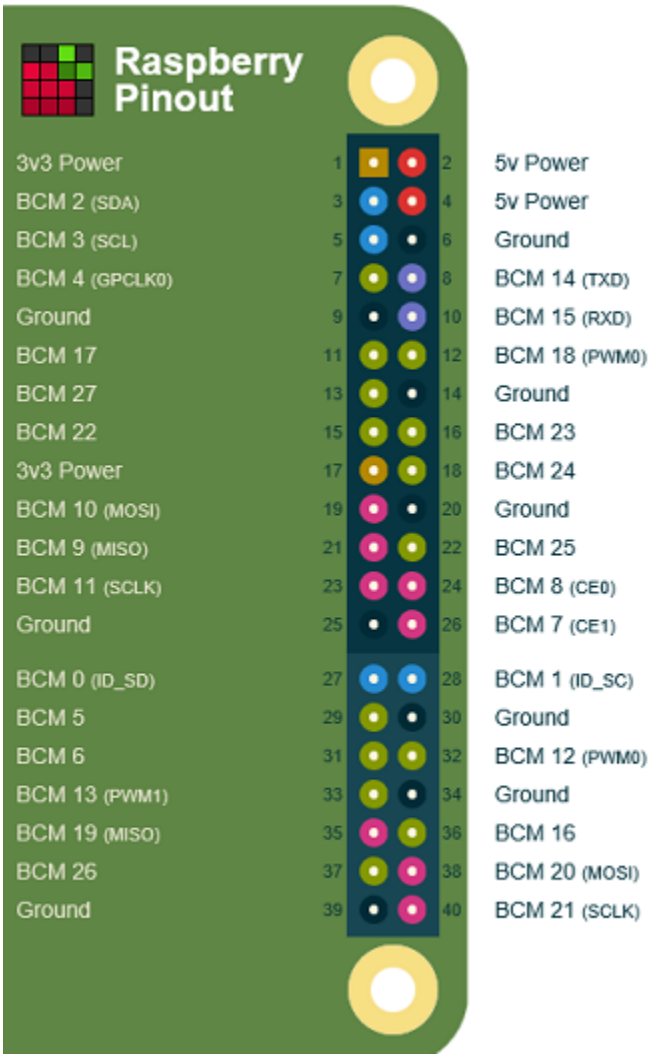

Figure 1 : GPIO du Raspberry PI II

Solution 3

L'utilisation du SPI du RPI permet à ce dernier d'envoyer et de recevoir des paquets de bits. Cela demande une programmation plus avancée, particulièrement du côté du système global. Les paquets doivent être générés et interprétés par le système global, mais cela enlève la limitation sur la grandeur des valeurs des données échangées.

P&O

Cette section présente une brève description de l'approche P&O, qui est un mode de contrôle utilisé conventionnellement pour les systèmes MPPT. La figure suivante est tirée de la référence [12]. Cet article n'a pas été publié et est en préparation.

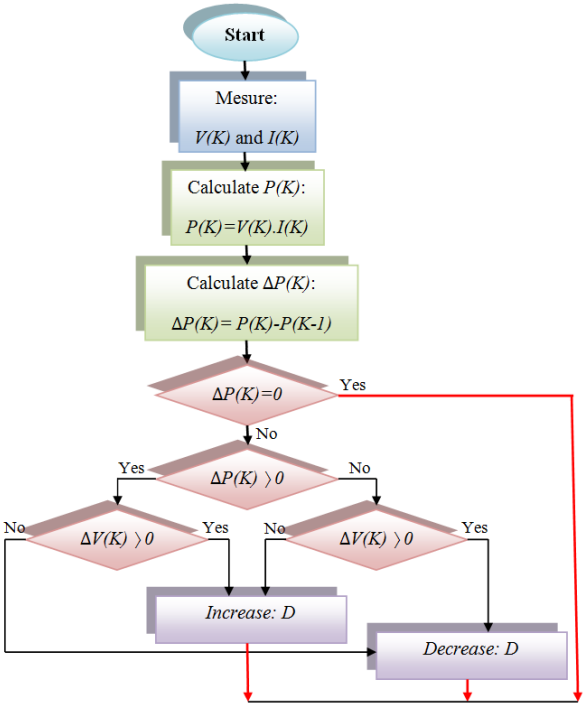

Figure 2 : Algorithme P&O

L'approche consiste à observer l'impact sur la puissance de sortie, à la suite des perturbations subies par la tension d'opération du système, tel que démontré par l'algorithme de la figure précédente. Si la puissance a augmenté depuis la dernière mesure, la perturbation sur la tension de sortie va continuer dans la même direction que le dernier cycle. Si la puissance a diminué depuis la dernière mesure, la perturbation sur la tension de sortie sera inversée, dans la direction opposée observé durant le dernier cycle. V oscillera autour de la tension idéale d'opération V_{MP}.

Comparaision P&O vs Logique floue

Selon la référence [12], un contrôleur MPPT utilisant la logique floue a un temps de réaction plus rapide, un taux de dépassement limité et une plus grande stabilité qu'un contrôleur utilisant un P&O. La figure suivante, tirée de la même référence, en est un exemple. La figure illustre le résultat d'une augmentation rapide, sur un panneau solaire, de la température de 10ºC à 50ºC à une irradiance fixe de 1000 Wm^{-2}, au temps 25 sec.

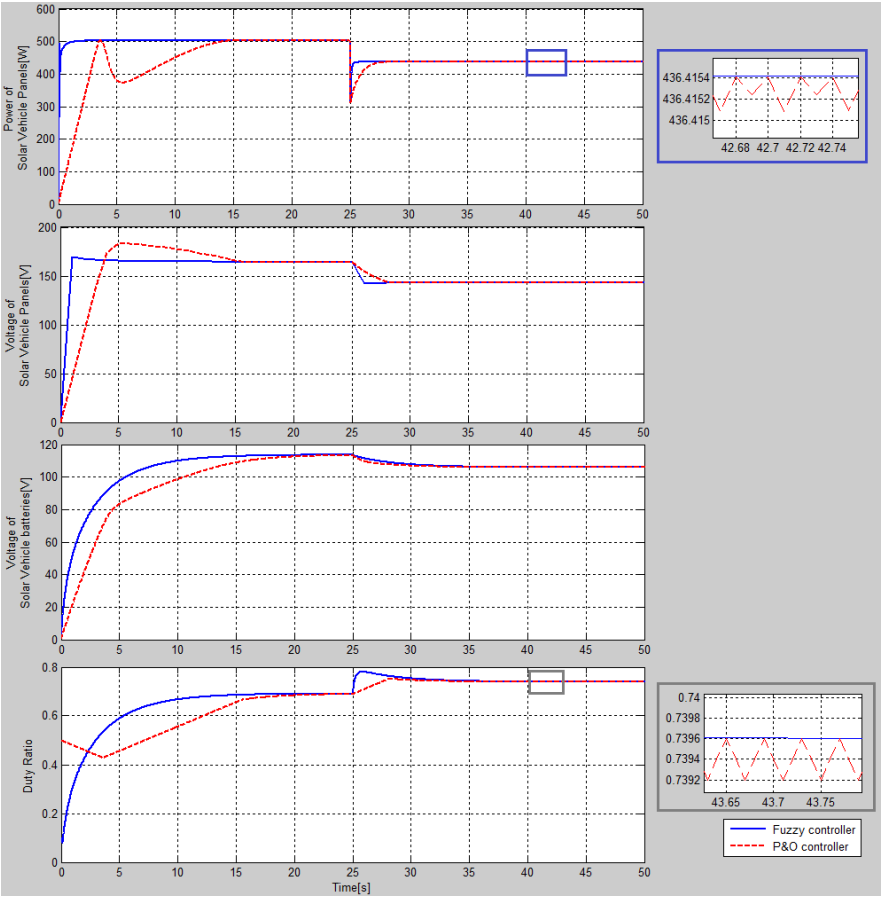

Figure 3 : Comparaison augmentation instantanée de température

Logique floue

Approche

Le module développé est programmé sur la méthodologie de logique floue expliquée dans la référence [12]. Les paragraphes suivants présentent un résumé de l'approche décrite dans cette référence :

Les variables en entrées et en sortie du contrôleur en logique floue sont exprimées en termes de fonctions d'appartenance. Les fonctions, basées sur l'expérience de spécialistes en automation et sur des analyses du système PV ont conduit aux choix des fonctions d'appartenance de la figure suivante.

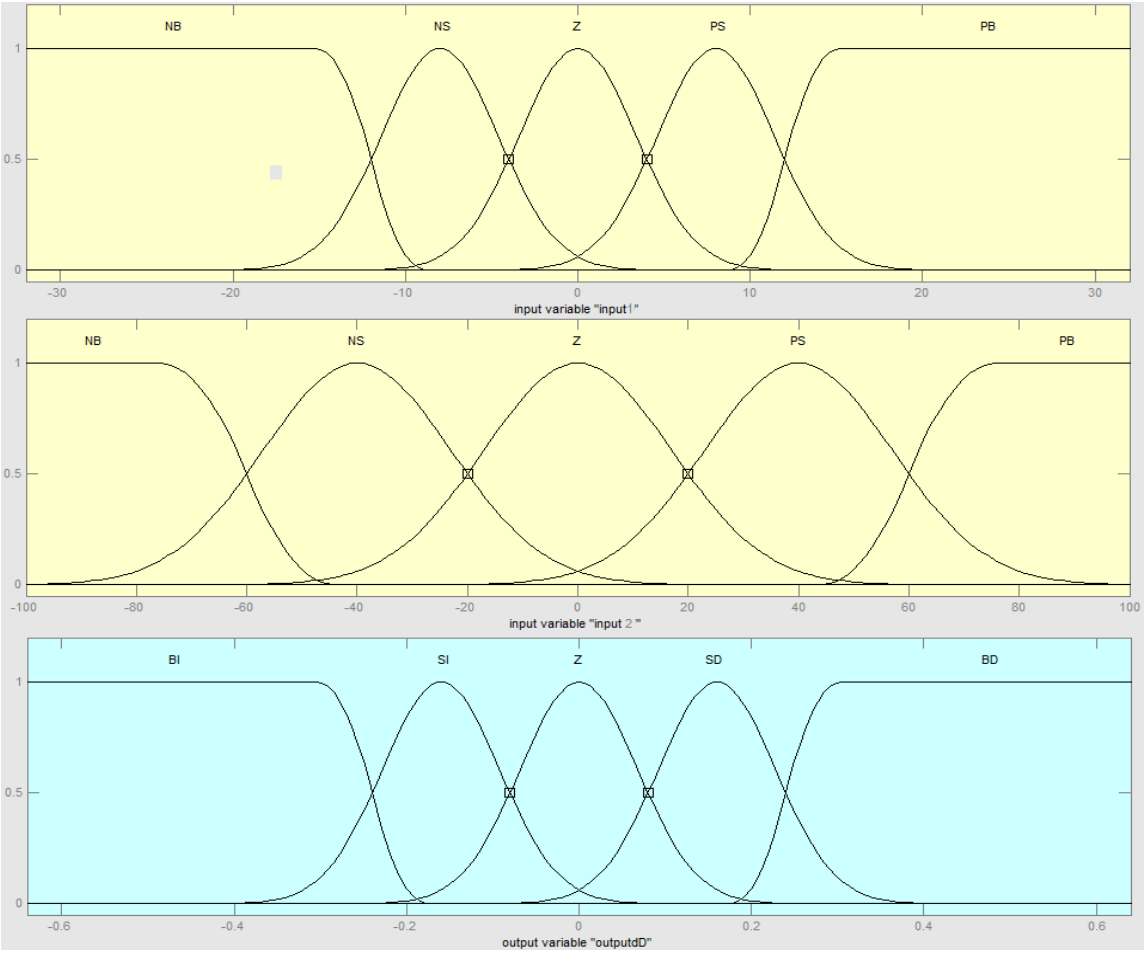

Figure 4 : Fonctions d'appartenance

Les entrées et les sorties, sous forme de variable linguistique gaussienne, sont représentées de la façon suivante :

Input1 : NB : ‘Négative Big’, NS : ‘Negative Small’, Z : ‘Zero’, PB : ‘Positive Big’, PS: ‘Positive Small’

Input2 : NB : ‘Négative Big’, NS : ‘Negative Small’, Z : ‘Zero’, PB : ‘Positive Big’, PS: ‘Positive Small’

Output : BD : ‘Big Decrease’, SD : ‘Small Decrease’, S: ‘Stabilize’, BI: ‘Big Increase’, SI: ‘Small Increase’

Les valeurs Input1, Input2 et Output ont été normalisées par un facteur d’échelle et varient de la façon suivante :

Input1 : -32 à 32

Input2 : -100 à 100

Output : -62 à 62

À la suite d’une étude exhaustive de combinaison des variables d’entrée et des sorties correspondantes, les règles d’inférence de décision suivantes ont été établies :

1. If (input1 is NB) and (input2 is NB) then (outputdD is S) (1)
2. If (input1 is NS) and (input2 is NB) then (outputdD is S) (1)
3. If (input1 is Z) and (input2 is NB) then (outputdD is SD) (1)
4. If (input1 is PS) and (input2 is NB) then (outputdD is SI) (1)
5. If (input1 is PB) and (input2 is NB) then (outputdD is BI) (1)
6. If (input1 is NB) and (input2 is NS) then (outputdD is S) (1)
7. If (input1 is NS) and (input2 is NS) then (outputdD is S) (1)
8. If (input1 is Z) and (input2 is NS) then (outputdD is SI) (1)
9. If (input1 is PS) and (input2 is NS) then (outputdD is SI) (1)
10. If (input1 is PB) and (input2 is NS) then (outputdD is BI) (1)
11. If (input1 is NB) and (input2 is Z) then (outputdD is BD) (1)
12. If (input1 is NS) and (input2 is Z) then (outputdD is SD) (1)
13. If (input1 is Z) and (input2 is Z) then (outputdD is S) (1)
14. If (input1 is PS) and (input2 is Z) then (outputdD is SI) (1)
15. If (input1 is PB) and (input2 is Z) then (outputdD is BI) (1)
16. If (input1 is NB) and (input2 is PS) then (outputdD is BD) (1)
17. If (input1 is NS) and (input2 is PS) then (outputdD is SD) (1)
18. If (input1 is Z) and (input2 is PS) then (outputdD is S) (1)
19. If (input1 is PS) and (input2 is PS) then (outputdD is S) (1)
20. If (input1 is PB) and (input2 is PS) then (outputdD is S) (1)
21. If (input1 is NB) and (input2 is PB) then (outputdD is BD) (1)
22. If (input1 is NS) and (input2 is PB) then (outputdD is SD) (1)
23. If (input1 is Z) and (input2 is PB) then (outputdD is SI) (1)
24. If (input1 is PS) and (input2 is PB) then (outputdD is S) (1)
25. If (input1 is PB) and (input2 is PB) then (outputdD is S) (1)

Figure 5 : Règles de décision

Le module développé peut être généralisé à d'autres problèmes à résoudre demandant une approche similaire en logique floue, soit celle utilisant la méthode du type Mandani avec fuzzification et défuzzification basée sur une approche Max-Min.

Analyse

Étape 1: la représentation mathématique des courbes de ‘Input1’, ‘Input2’ et ‘Output’ est à définir. Les courbes suivent une distribution gaussienne, c représente le centre et σ l'étendue de la courbe.

$$f(x;\sigma,c) = e^{\frac{-(x-c)^2}{2\sigma^2}}$$

Figure 6 : Distribution Gaussienne

Étape 2: La représentation gaussienne de chaque courbe est nécessaire afin de pouvoir identifier les courbes et réussir à projeter les points provenant des valeurs lues Input1 et Input2 sur Output0. La fuzzification est un 'ET' logique floue, suivie d'un 'OU' logique floue de Input1 et Input2 [13]. Les règles d'inférence sont appliquées sur chacun des ensembles I1 et I2 pour obtenir les pourcentages individuels.

Étape 3: Les résultats individuels sont agrégés pour former la valeur 'fuzzy' du résultat final [14].

Étape4: La défuzzification est effectuée en utilisant la formule du centre de masse sur la représentation du résultat de l'accumulation obtenue à l'étape 2. La figure suivante est tirée de la référence [15].

$$z = \frac{\sum_{j=1}^{q} Z_j u_c(Z_j)}{\sum_{j=1}^{q} u_c(Z_j)}$$

Figure 7 : Centre de masse

Pseudocode général

- Définir et identifier les courbes gaussiennes (NB, NS, etc.) de Input1, Input2 et Output0;
- Lire les valeurs précises I1 et I2 (sur le GPIO);

(Fuzification)

- Trouver les courbes interceptées par I1 sur le graphique des courbes Input1;
- Trouver les courbes interceptées par I2 sur le graphique des courbes Input2;

- Pour les courbes identifiées sur le graphique Input1, réitérer de 1 à n
    - Pour les courbes identifiées sur le graphique Input2, réitérer de 1 à n
        - Identifier, à l'aide de la règle d'inférence de décision, la courbe Output correspondante;
        - Appliquer l'opération logique floue Min (Input1, Input2);
        - Si une courbe 'Output' se répète, appliquer la fonction Max(Min(Input1a, Input2a), Min(Input1b, Input2b));

(Défuzzification, Calculer le centre de gravité [15]

- Pour chaque courbe Output0 sélectionné, réitérer de 1 à n
    - Calculer l'aire sous la courbe, en respectant le seuil trouvé plutôt;
    - Calculer la somme des U(u);
- Centre de gravité = l'aire totale sous les courbes / U(u) totale;
- Écrire la valeur discrète dans Output;

Algorithme général

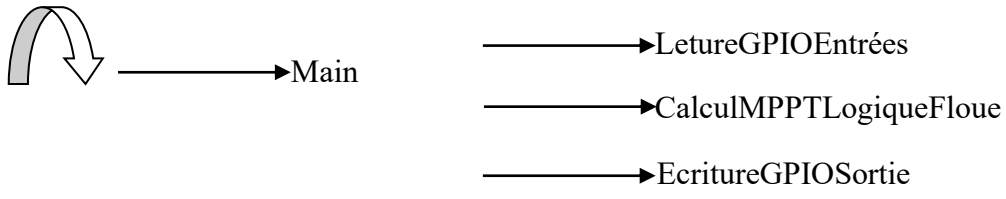

Figure 8 : Algorithme général

3.1.2 Description supplémentaire des opérations

Module principal 'Main'

Cette fonction globale vérifie s'il y a des valeurs dans les registres d'entrée. Si oui, elle appelle les fonctions de calcul logique flou et d'écriture du résultat dans le registre de sortie.

Module 'fReadInput'

Cette fonction lit le registre d'entrée un (Input1) ou deux (Input2) et convertie la valeur lue en un nombre compris entre -127 et 127.

Module 'fCalculFuzzy'

Cette fonction accepte les deux paramètres ('Input1' et 'Input2') et exécute les opérations de logique floue programmées selon la méthode décrite à la section 3.1.1, sous la rubrique Logique floue, afin de générer la valeur de 'Output', qui sera retournée au système global.

Sous module 'fDecision'

Cette fonction retourne la valeur discrète 'Output', basé sur les règles de décision préétablis, correspondante aux deux (2) courbes 'Input' passées en paramètre.

Sous module 'fDefuzzification'

Cette fonction calcule le centroïde des variables linguistiques 'Output' agrégées.

Module 'fWriteOutput'

Cette fonction écrit la valeur 'Output' dans le registre de sortie, afin qu'elle puisse être récupérée par le système global.

PROCEDURE DE TEST

. La procédure de test suivra cette approche:

- Procéder aux tests unitaires, c'est-à-dire vérifier si chaque module pris individuellement se comporte de la façon souhaitée ;
- Procéder à des tests portant sur l'aspect d'interrelation entre les modules selon une approche 'top-down' ou 'bottom-up' ;
- Faire ensuite des tests de validation basés sur les critères de performance, sur les réponses attendues du logiciel et sur des cas spéciaux ;
- Procéder également à des tests de données et de volume ;
- etc.

Les tests réalisés sont décrits davantage dans le Chapitre 4.

DESCRIPTION DU PROTOTYPE

Un prototype est fonctionnel. Il est conçu dans le but de pouvoir montrer ce que pourra faire le produit fini. Le prototype constitue une excellente base du logiciel. La facilité de programmation et d'ajout de fonctionnalités avec C permettra de développer aisément les fonctionnalités futures.

Le prototype est décrit davantage dans le chapitre 4. Il consiste en un module RPI sur lequel le programme écrit en code C et détaillé au chapitre suivant, est installé et prêt à être exécuté.

Les connections et le fonctionnement des entrées et sortie sont également montrés au chapitre 4.

INSTALLATION, MISE EN MARCHE, ENTRETIEN, FIABILITE

RPI vient avec boîtier en plastique, le système est léger devra être fixé pour éviter les chocs et coups physiques.

Durant les étapes de programmation, le système s'est avéré stable, c'est-à-dire qu'il n'a eu aucune défaillance logicielle inexpliquée (plantage).

Des tests de fiabilité sur la 'résistance' physique du RPI sont à faire dans des conditions proches de la réalité, c'est-à-dire dans des conditions de chaleur et de froid prononcés.

Le fonctionnement en continu, sur de longue période, doit être vérifié.

RPI peut et devrait être connecté sur un réseau durant le mode 'Opération'. Ceci permettrait de vérifier à distance s'il y a défaillance ou vérifier d'autres informations.

Les limitations en voltages et courants des 'Pin' du GPIO sont bien décrits dans la référence [16].

IMPLEMENTATIONS

Il sera utile d'ajouter des améliorations au logiciel afin de le rendre plus intéressant, par exemple ajouter le démarrage automatique du programme pgmFuzzy (qui contient le 'Main') lorsque le RPI est branché sur une source de courant. À noter que le système d'exploitation du module RPI se charge et démarre automatiquement lorsque le composant électronique est branché sur une source d'alimentation.

CHAPITRE 4
PROGRAMMATION, TEST

Cette section présente le programme en détail et explique les particularités des fonctions. Les tests de base et la méthodologie pour les réaliser sont également décrits.

MODULE PRINCIPAL 'MAIN'

4.1.1 Code

```
#include <bcm2835.h>
#include <stdio.h>
#include <math.h>

#define MIN 0
#define DELAIS 250

char sCur_I1[5];
char sCur_I2[5];
char sCur_O[3];

int main()
{

      while(1)
      {

            int I1; int I2; int iO;

            I1 = fReadInput(1);
            I2 = fReadInput(2);
```

```
            if ((I1 != 0) && (I2 !=0))
            {
                  iO = fCalculFuzzy(I1, I2);
                  fWriteOutput(iO);
            }
      }

return 0;
}
```

4.1.2 Particularités

Au début du programme se trouve les instructions qui définissent les librairies qui sont appelées ainsi que les déclarations de constantes et de variables globales.

Voici une description des librairies appelées :

`#include <bcm2835.h>`
La librairie bcm2835.h contient les fonctions qui permettent de communiquer avec le GPIO.
`#include <stdio.h>`
Cette librairie contient les instructions d'entrées/sorties standard.
`#include <math.h>`
Cette librairie contient les instructions pour les opérations mathématiques

Voici une description des constantes utilisées afin de simplifier la programmation et les tests :

`#define MIN 0`
Cette constante permet d'enlever du calcul de fuzzification les fonctions d'appartenance touchées par les valeurs discrètes de I1 et I2 mais dont les résultats de probabilités sont très petits.
`#define DELAIS 250`
Cette constante définit le délai employé arbitrairement dans les fonctions de lecture et d'écriture sur le GPIO. C'est une valeur de synchronisation à ajuster selon les tests de vitesse entre le système global et le RPI.

Voici une description des variables globales :

```
char sCur_I1[5];
char sCur_I2[5];
```

```
char sCur_O[3];
```

Ces trois (3) variables sont utilisées abondamment dans les différentes fonctions et leur déclaration en tant que variables globales simplifie la programmation.

MODULE 'FREADINPUT'

4.1.3 Code

```
int fReadInput(int iNoInput)
{
      //
      // Fonction    : fReadInput
      // Description : Lire les pins 1 a 8 dedie a la valeur Input
      //                (fournie par le systeme global)
      //
      // Entree(s)   : iNoInput : 1 ou 2
      //
      //
      // Sortie(s)   : Nil
      //
      // Retour      : Valeur lue
      //

      uint8_t P0, P1, P2, P3, P4, P5, P6, PS; //Numero GPIO
      uint8_t R0, R1, R2, R3, R4, R5, R6, RS;
      int iResult;

      if(!bcm2835_init()) return 0;

      //Identifier les Numero de pin GPIO en entree
      switch (iNoInput)
      {
      case 1:
            {
                  PS = RPI_BPLUS_GPIO_J8_03; //2, Signe
                  P6 = RPI_BPLUS_GPIO_J8_05; //3
                  P5 = RPI_BPLUS_GPIO_J8_07; //4
                  P4 = RPI_BPLUS_GPIO_J8_08; //14
                  P3 = RPI_BPLUS_GPIO_J8_10; //15
                  P2 = RPI_BPLUS_GPIO_J8_11; //17
                  P1 = RPI_BPLUS_GPIO_J8_12; //18
```

```
                P0 = RPI_BPLUS_GPIO_J8_13; //27
                break;
        };
    case 2:
        {
                PS = RPI_BPLUS_GPIO_J8_15; //22, Signe
                P6 = RPI_BPLUS_GPIO_J8_16; //23
                P5 = RPI_BPLUS_GPIO_J8_18; //24
                P4 = RPI_BPLUS_GPIO_J8_19; //10
                P3 = RPI_BPLUS_GPIO_J8_21; //09
                P2 = RPI_BPLUS_GPIO_J8_22; //25
                P1 = RPI_BPLUS_GPIO_J8_23; //11
                P0 = RPI_BPLUS_GPIO_J8_24; //8
                break;
        };
    }

    bcm2835_delay(DELAIS);

    //Lire les valeurs binaires
    bcm2835_gpio_fsel(PS, BCM2835_GPIO_FSEL_INPT); //Mode Lecture
    RS = bcm2835_gpio_lev(PS);
    bcm2835_gpio_fsel(P6, BCM2835_GPIO_FSEL_INPT);
    R6 = bcm2835_gpio_lev(P6);
    bcm2835_gpio_fsel(P5, BCM2835_GPIO_FSEL_INPT);
    R5 = bcm2835_gpio_lev(P5);
    bcm2835_gpio_fsel(P4, BCM2835_GPIO_FSEL_INPT);
    R4 = bcm2835_gpio_lev(P4);
    bcm2835_gpio_fsel(P3, BCM2835_GPIO_FSEL_INPT);
    R3 = bcm2835_gpio_lev(P3);
    bcm2835_gpio_fsel(P2, BCM2835_GPIO_FSEL_INPT);
    R2 = bcm2835_gpio_lev(P2);
    bcm2835_gpio_fsel(P1, BCM2835_GPIO_FSEL_INPT);
    R1 = bcm2835_gpio_lev(P1);
    bcm2835_gpio_fsel(P0, BCM2835_GPIO_FSEL_INPT);
    R0 = bcm2835_gpio_lev(P0);

    //Binaire vers Decimal
    iResult = (R6 * pow(2,6)) + (R5 * pow(2,5)) + (R4 * pow(2,4))
+ (R3 * pow(2,3)) + (R2 * pow(2,2)) + (R1 * pow(2,1)) + (R0 *
pow(2,0));

    if (RS == HIGH) //Negatif
    {
        iResult = -1 * iResult;
    }

    bcm2835_close();
```

```
      return iResult;
}
```

4.1.4 Particularités

La solution 2, présentée à la section 3.1.1, sous la rubrique Communication, a été développée dans cet exemple de code. Les trois (3) entrées et sortie ont chacune leurs ‘Pins’ dédiées.

Les valeurs en entrée sont donc basées sur une lecture binaire et sont converties de binaire vers décimal pour la suite du programme.

Dans le code, si nous regardons les instructions ‘case 1’ et ‘case 2’, nous pouvons voir les numéros de ‘Pins’ alloués, par convention, aux ‘Pins’ physiques du circuit correspondants aux valeurs en entrée 1 et 2.

La fonction bcm2835_init prépare les registres pour la suite du programme. La fonction bcm2835_gpio_fsel définit l’état de la ‘Pin’ : lecture ou écriture. La fonction bcm2835_gpio_lev lit la valeur de la ‘Pin’ : 0 ou 1. La fonction bcm2835_close est appelée à la fin pour libérer la mémoire.

4.1.5 Test

Le circuit apparaissant à la figure suivante a été monté pour simuler un signal en entrée. Un ‘push button’ simule un signal en entrée et un message s’affiche lorsque la ‘Pin’ 26 est activée.

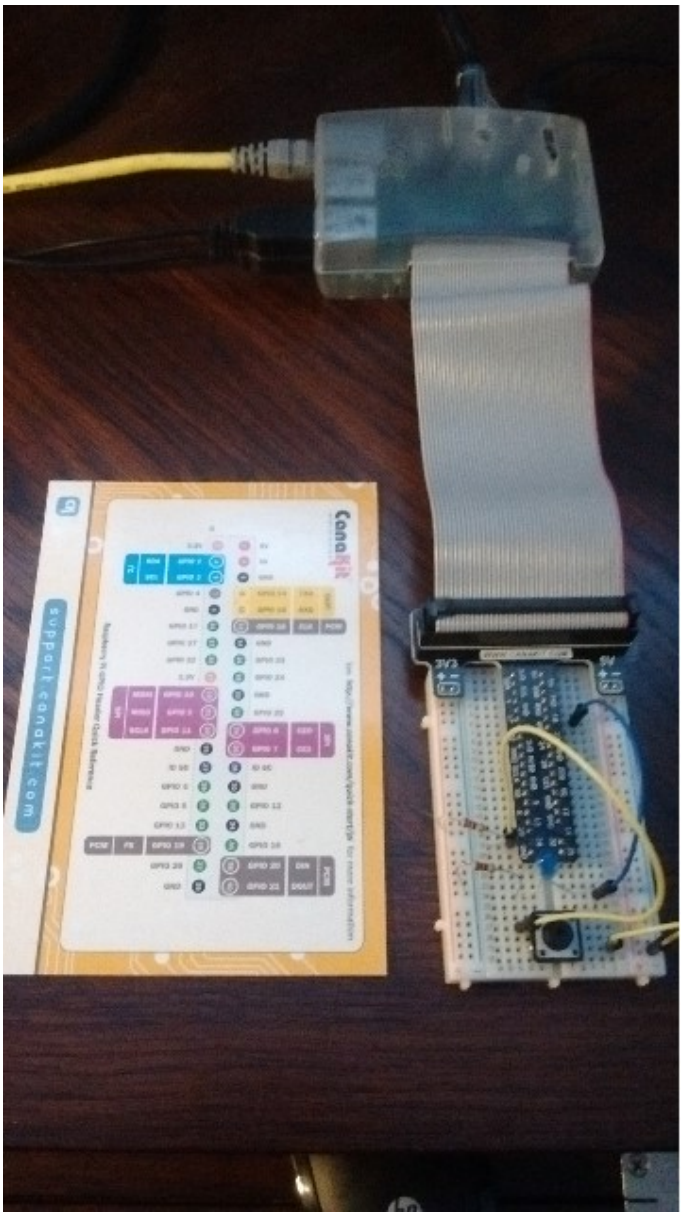

Figure 9 : Circuit de test du GPIO

MODULE 'FCALCULFUZZY'

4.1.6 Code

```
float fCalculFuzzy(int iInput_1, int iInput_2)
{
      //
      // Fonction    : fCalculFuzzy
      // Description : Calcul base sur la methode Mamdani Max-Min
      //
      // Entree(s)   : iInput_1 :
      //               iInput_2 :
      //
      // Sortie(s)   : Nil
      //
      // Retour      : Valeur calcule par la methode
      //

      int i; int j;
      float fNB_1; float fNS_1; float fZ_1; float fPS_1; float
fPB_1; //Input 1
      float fNB_2; float fNS_2; float fZ_2, fPS_2; float fPB_2;
//Input 2
      float fBI; float fSI; float fZ; float fSD; float fBD;
//Output
      float fCur_I1; float fCur_I2;
      int bCur_1; int bCur_2;
      float fCur_O;

      //Identifier les menbership functions touchees de Input_1
      fNB_1 = 0; fNS_1 = 0; fZ_1 = 0; fPS_1 = 0; fPB_1 = 0;
      fBI = 0; fSI = 0; fZ = 0; fSD = 0; fBD = 0;
      for(i = 1; i <= 5; i++)
      {
            switch (i)
            {
                  case 1:
                  {     if (iInput_1 <= -15)
                        {
                              fNB_1 = 1;
                        }
                        else
                        {
                              fNB_1 = exp((-
1*pow((iInput_1+15),2))/(2*pow(3.25,2)));
                        }
```

```
                        break;
                  };
                  case 2:
                  {
                        fNS_1 = exp((-
1*pow((iInput_1+8),2))/(2*pow(3.25,2)));
                        break;
                  };
                  case 3:
                  {
                        fZ_1 = exp((-
1*pow((iInput_1),2))/(2*pow(3.25,2)));
                        break;
                  };
                  case 4:
                  {
                        fPS_1 = exp((-1*pow((iInput_1-
8),2))/(2*pow(3.25,2)));
                        break;
                  };
                  case 5:
                  {
                        if (iInput_1 >= 15)
                        {
                              fPB_1 = 1;
                        }
                        else
                        {
                              fPB_1 = exp((-1*pow((iInput_1-
15),2))/(2*pow(3.25,2)));
                        }
                        break;
                  };

            }
      }

      //Identifier les menbership functions touchees de Input_2
      fNB_2 = 0; fNS_2 = 0; fZ_2 = 0; fPS_2 = 0; fPB_2 = 0;
      for(i = 1; i <= 5; i++)
      {
            switch (i)
            {
                  case 1:
                  {     if (iInput_2 <= -74)
                        {
                              fNB_2 = 1;
                        }
```

```
                        else
                        {
                              fNB_2 = exp((-
1*pow((iInput_2+74),2))/(2*pow(16.5,2)));
                        }
                        break;
                  };
                  case 2:
                  {
                        fNS_2 = exp((-
1*pow((iInput_2+40),2))/(2*pow(16.5,2)));
                        break;
                  };
                  case 3:
                  {
                        fZ_2 = exp((-
1*pow((iInput_2),2))/(2*pow(16.5,2)));
                        break;
                  };
                  case 4:
                  {
                        fPS_2 = exp((-1*pow((iInput_2-
40),2))/(2*pow(16.5,2)));
                        break;
                  };
                  case 5:
                  {
                        if (iInput_2 >= 74)
                        {
                              fPB_2 = 1;
                        }
                        else
                        {
                              fPB_2 = exp((-1*pow((iInput_2-
74),2))/(2*pow(16.5,2)));
                        }
                        break;
                  };

            }
      }

      //Trouver les minimums des membership functions de Output
      for (i = 1; i <= 5; i++)
      {
            bCur_1 = 0;
            strncpy(sCur_I1, " ", 1);
            switch (i)
```

```
{
      case 1:
      {
            if (fNB_1 > MIN)
            {
                  bCur_1 = 1;
                  fCur_I1 = fNB_1;
                  strcpy(sCur_I1, "NB_1");
            }
            break;
      };
      case 2:
      {
            if (fNS_1 > MIN)
            {
                  bCur_1 = 1;
                  fCur_I1 = fNS_1;
                  strcpy(sCur_I1, "NS_1");
            }
            break;
      };
      case 3:
      {
            if (fZ_1 > MIN)
            {
                  bCur_1 = 1;
                  fCur_I1 = fZ_1;
                  strcpy(sCur_I1, "Z_1");
            }
            break;
      };
      case 4:
      {
            if (fPS_1 > MIN)
            {
                  bCur_1 = 1;
                  fCur_I1 = fPS_1;
                  strcpy(sCur_I1, "PS_1");
            }
            break;
      };
      case 5:
      {
            if (fPB_1 > MIN)
            {
                  bCur_1 = 1;
                  fCur_I1 = fPB_1;
                  strcpy(sCur_I1, "PB_1");
```

```
            }
            break;
      };
} //switch(i)
if (bCur_1 == 1)
{

      for (j = 1; j <= 5; j++)
      {
            bCur_2 = 0;
            strncpy(sCur_I2, " ", 1);
            switch (j)
            {
                  case 1:
                  {
                        if (fNB_2 > MIN)
                        {
                              bCur_2 = 1;
                              fCur_I2 = fNB_2;
                              strcpy(sCur_I2, "NB_2");
                        }
                        break;
                  };
                  case 2:
                  {
                        if (fNS_2 > MIN)
                        {
                              bCur_2 = 1;
                              fCur_I2 = fNS_2;
                              strcpy(sCur_I2, "NS_2");
                        }
                        break;
                  };
                  case 3:
                  {
                        if (fZ_2 > MIN)
                        {
                              bCur_2 = 1;
                              fCur_I2 = fZ_2;
                              strcpy(sCur_I2, "Z_2");
                        }
                        break;
                  };
                  case 4:
                  {
                        if (fPS_2 > MIN)
                        {
                              bCur_2 = 1;
```

```
                                    fCur_I2 = fPS_2;
                                    strcpy(sCur_I2, "PS_2");
                              }

                              break;
                        };
                        case 5:
                        {
                              if (fPB_2 > MIN)
                              {
                                    bCur_2 = 1;
                                    fCur_I2 = fPB_2;
                                    strcpy(sCur_I2, "PB_2");
                              }
                              break;
                        };
                  } //switch(j)
                  if (bCur_2 == 1)
                  {

                        strncpy(sCur_O, " ", 1);
                        //Appeler la Regle de decision
                        fDecision(sCur_I1, sCur_I2, sCur_O);

                        //Trouver le minimum de Output
membership Function (I1 U I2)
                        if (fCur_I1 < fCur_I2)
                        {
                              fCur_O = fCur_I1;
                        }
                        else
                        {
                              fCur_O = fCur_I2;
                        }

                        if (strncmp(sCur_O, "BI", 4) == 0)
                        {
                              if (fCur_O > fBI)
                              {
                                    fBI = fCur_O;
                              }
                        }
                        if (strncmp(sCur_O, "SI", 4) == 0)
                        {
                              if (fCur_O > fSI)
                              {
                                    fSI = fCur_O;
                              }
```

```
                                    }
                                    if (strncmp(sCur_O, "Z", 4) == 0)
                                    {
                                          if (fCur_O > fZ)
                                          {
                                                fZ = fCur_O;
                                          }
                                    }
                                    if (strncmp(sCur_O, "SD", 4) == 0)
                                    {
                                          if (fCur_O > fSD)
                                          {
                                                fSD = fCur_O;
                                          }
                                    }
                                    if (strncmp(sCur_O, "BD", 4) == 0)
                                    {
                                          if (fCur_O > fBD)
                                          {
                                                fBD = fCur_O;
                                          }
                                    }

                              }//if (bCur_2 == 1)

                        } //for (j = 1; i <= 5; i++)
                  } //if (bCur_1 == 1)
            }//for (i = 1; i <= 5; i++)

            //Defuzzification
            float fTmp = fDefuzzification(fBI, fSI, fZ, fSD, fBD);
            return (fTmp);
}
```

4.1.7 Particularités

Le programme suit exactement le pseudocode de la section 3.1.1, sous la rubrique pseudocode général.

Afin d'alléger le code de cette fonction, deux (2) sous-fonctions sont appelées dans ce module : fDecision et fDefuzzication (voir la section 3.1.1, sous la rubrique Algorithme général, pour une description sommaire).

Cette fonction est bâtie sur la méthode expliquée au chapitre trois (3) dans la section logique floue. La méthode provient de la référence [12]. Elle pourrait être adaptée afin de satisfaire un autre système également basé sur une approche Mamdani avec opérations Max-Min.

4.1.8 Test

La justesse des calculs de cette fonction ont été testés dans un tabulateur Excel. La démonstration se trouve à l'Annexe 1.

MODULE 'FWRITEOUTPUT'

4.1.9 Code

```
int fWriteOutput(int iValue)
{

      //
      // Fonction    : fWriteOutput
      // Description : Ecrire sur les pins 1 a 8 dedie la valeur a
      //               retourner au systeme global
      //
      // Entree(s)   : iValue :
      //
      //
      // Sortie(s)   : Nil
      //
      // Retour      : 0 : OK
      //               1 : Erreur
      //

      uint8_t P0, P1, P2, P3, P4, P5, P6, PS; //No GPIO
      uint8_t R0, R1, R2, R3, R4, R5, R6, RS;

      if(!bcm2835_init()) return 1;
```

```
      //Decomposer la valeur en binaire sur 7 chiffres, le 8e est
le signe, 0 = positif, 1 = negatif
      if (iValue < 0)
      {
            RS = 1;
      }
      else
      {
            RS = 0;
      }

      if (iValue == 0) //test division par 0
      {
            R0 = 0;
      }
      else
      {
            R0 = (iValue % 2);
            iValue = (iValue / 2);
      }
      if (iValue == 0)
      {
            R1 = 0;
      }
      else
      {
            R1 = (iValue % 2);
            iValue = (iValue / 2);
      }
      if (iValue == 0)
      {
            R2 = 0;
      }
      else
      {
            R2 = (iValue % 2);
            iValue = (iValue / 2);
      }
      if (iValue == 0)
      {
            R3 = 0;
      }
      else
      {
            R3 = (iValue % 2);
            iValue = (iValue / 2);
      }
      if (iValue == 0)
```

```
{
      R4 = 0;
}
else
{
      R4 = (iValue % 2);
      iValue = (iValue / 2);
}
if (iValue == 0)
{
      R5 = 0;
}
else
{
      R5 = (iValue % 2);
      iValue = (iValue / 2);
}
if (iValue == 0)
{
      R6 = 0;
}
else
{
      R6 = (iValue % 2);
      iValue = (iValue / 2);
}

//Identifier les Numero de pin GPIO de sortie
PS = RPI_BPLUS_GPIO_J8_26; //7, Signe
P6 = RPI_BPLUS_GPIO_J8_29; //5
P5 = RPI_BPLUS_GPIO_J8_31; //6
P4 = RPI_BPLUS_GPIO_J8_32; //12
P3 = RPI_BPLUS_GPIO_J8_33; //13
P2 = RPI_BPLUS_GPIO_J8_35; //19
P1 = RPI_BPLUS_GPIO_J8_36; //16
P0 = RPI_BPLUS_GPIO_J8_37; //26

bcm2835_delay(DELAIS);

//Ecrire les valeurs binaires
bcm2835_gpio_fsel(PS, BCM2835_GPIO_FSEL_OUTP);
if (RS == 1)
{
      bcm2835_gpio_set(PS); //1, negatif
}
else
{
      bcm2835_gpio_clr(PS); //0, positif
```

```
}
bcm2835_gpio_fsel(P6, BCM2835_GPIO_FSEL_OUTP);
if (R6 == 1)
{
     bcm2835_gpio_set(P6);
}
else
{
     bcm2835_gpio_clr(P6);
}
bcm2835_gpio_fsel(P5, BCM2835_GPIO_FSEL_OUTP);
if (R5 == 1)
{
     bcm2835_gpio_set(P5);
}
else
{
     bcm2835_gpio_clr(P5);
}
bcm2835_gpio_fsel(P4, BCM2835_GPIO_FSEL_OUTP);
if (R4 == 1)
{
     bcm2835_gpio_set(P4);
}
else
{
     bcm2835_gpio_clr(P4);
}
bcm2835_gpio_fsel(P3, BCM2835_GPIO_FSEL_OUTP);
if (R3 == 1)
{
     bcm2835_gpio_set(P3);
}
else
{
     bcm2835_gpio_clr(P3);
}
bcm2835_gpio_fsel(P2, BCM2835_GPIO_FSEL_OUTP);
if (R2 == 1)
{
     bcm2835_gpio_set(P2);
}
else
{
     bcm2835_gpio_clr(P2);
}
bcm2835_gpio_fsel(P1, BCM2835_GPIO_FSEL_OUTP);
if (R1 == 1)
```

```
	{
		bcm2835_gpio_set(P1);
	}
	else
	{
		bcm2835_gpio_clr(P1);
	}
	bcm2835_gpio_fsel(P0, BCM2835_GPIO_FSEL_OUTP);
	if (R0 == 1)
	{
		bcm2835_gpio_set(P0);
	}
	else
	{
		bcm2835_gpio_clr(P0);
	}

	bcm2835_close();

	return 0;
}
```

4.1.10 Particularités

A l'opposé de la fonction 'fReadInput', la valeur à retourner au système global doit être convertie de la notation décimale vers la notation binaire.

Les 'Pins' de sortie, arbitrairement choisis, pour contenir la représentation binaire sont indiqués dans le tableau suivant.

Tableau 3 : No de Pins de sortie

No 'Pin'du GPIO	Poids binaire
7	Signe (+ / -)
5	6
6	5
12	4
13	3
19	2
16	1
26	0

4.1.11 Test

Le circuit de la figure de la section 4.2.3 a également servi pour vérifier l'envoi d'un signal vers la 'Pin' six (6) du GPIO. Un LED fût connecté sur cette 'Pin' et le test a été concluant.

CHAPITRE 5
RÉSULTATS, ANALYSE

RESULTATS DE TEST REEL

Afin de valider si les valeurs à la sortie de la défuzzication sont bonnes, il convient d'utiliser des valeurs en entrée qui sont basées sur un graphique de données de référence. Le graphique suivant provient de la référence [12].

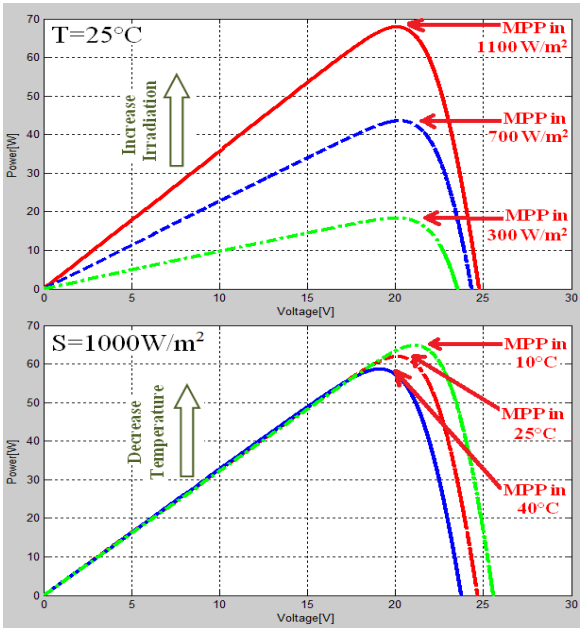

Figure 10 : Variation du MPP selon l'irradiation et la température

La courbe de la section du bas, identifié par 'MPP in 40º', a été recréée dans Excel (figure suivante), afin d'en extraire les données. Le tableau 4 présente les valeurs.

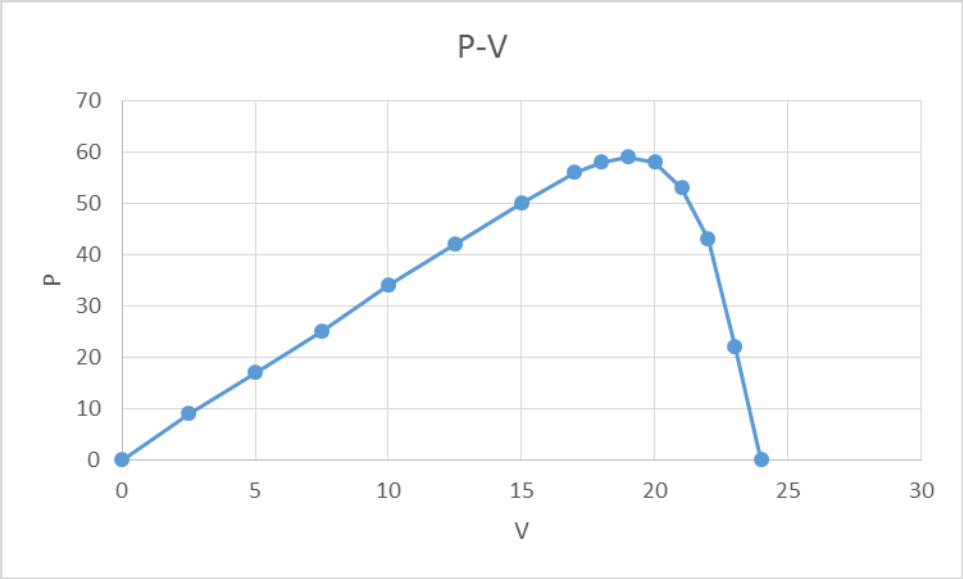

Figure 11 : ‘MPP in 40º’

Tableau 4 : Données de test réel

I (Courant A)	V (Tension V)	P (Puissance W)
0	0	0
3,60	2,50	9,00
3,40	5,00	17,00
3,33	7,50	25,00
3,40	10,00	34,00
3,36	12,50	42,00
3,33	15,00	50,00
3,29	17,00	56,00
3,22	18,00	58,00
3,11	19,00	59,00
2,90	20,00	58,00
2,52	21,00	53,00
1,95	22,00	43,00
0,96	23,00	22,00
0,00	24,00	0,00

Les valeurs, prises du tableau 4, ont été ensuite incorporées dans le programme fCalculFuzzy. Une nouvelle fonction, décrite à la section 5.1.1 a été écrite afin de transformer les données de Tension V et de courant I en valeurs 'fuzzifiables' : 'Input1' et 'Input2'.

5.1.1 Code de 'fFuzzInput'

```
float fVpr = 0;
float fPpr = 0;

int fFuzzInput(float V, float I, float *iImp1, float *iImp2)
{
      //
      // Fonction    : fFuzzInput
      // Description : A partir de V et I, calculer Input1 et
Input2,
      //               qui serviront d'entree dans la fonction
Fuzzy
      //
      // Entree(s)   : V : Voltage
      //               I : Courant
      //
      // Sortie(s)   : iImp1 : Input1
      //               iImp2 : Input2
      //
      // Retour      : 0: OK, 1: Erreur
      //

      if (V == fVpr) //division par zero, pente indéfini
      {
            *iImp1 = 0;
            *iImp2 = 0;
            return (1);
      }

      float fP = V * I;
      //Pente de la courbe P-V
      *iImp1 = (fP - fPpr) / (V - fVpr);
      //Rate of change de P
      *iImp2 = fP - fPpr;
```

```
        fPpr = fP;
        fVpr = V;

        return (0);
}
```

5.1.2 Description de 'fFuzzInput'

Basé sur la documentation trouvée dans la référence [12], la pente de la courbe P-V donne 'Input 1' et le taux de changement ('rate of change') de P donne 'Input 2'. Les valeurs de 'Input 1' et 'input 2' sont ensuite récupérées par la fonction 'fCalculFuzzy'.

5.1.3 Résultat obtenu par le programme 'fCalculFuzzy'

Les valeurs suivantes ('Output') sont les résultats obtenus :

```
V: 2.500000, I : 3.600000
 Input 1: 3.600000d, Input 2: 9.000000 Output:  -8.08
V: 5.000000, I : 3.400000
 Input 1: 3.200000d, Input 2: 8.000000 Output:  -7.17
V: 7.500000, I : 3.330000
 Input 1: 3.189999d, Input 2: 7.974998 Output:  -7.15
V: 10.000000, I : 3.400000
 Input 1: 3.610001, Input 2: 9.025002 Output:  -8.10
V: 12.500000, I : 3.360000
 Input 1: 3.200000, Input 2: 8.000000 Output:  -7.17
V: 15.000000, I : 3.330000
 Input 1: 3.179999, Input 2: 7.949997 Output:  -7.13
V: 17.000000, I : 3.290000
 Input 1: 2.990002, Input 2: 5.980003 Output:  -6.71
V: 18.000000, I : 3.220000
 Input 1: 2.029999, Input 2: 2.029999 Output:  -4.75
V: 19.000000, I : 3.110000
 Input 1: 1.129997, Input 2: 1.129997 Output:  -3.22
V: 20.000000, I : 2.900000
 Input 1: -1.089996, Input 2: -1.089996 Output:   1.03
V: 21.000000, I : 2.520000
 Input 1: -5.080002, Input 2: -5.080002 Output:   7.37
```

```
V: 22.000000, I : 1.950000
 Input 1: -10.019997, Input 2: -10.019997 Output:  30.93
V: 23.000000, I : 0.960000
 Input 1: -20.820002, Input 2: -20.820002 Output:   0.00
V: 24.000000, I : 0.000000
 Input 1: -22.080000, Input 2: -22.080000 Output:   0.00


------------------
(program exited with code: 0)
Press return to continue
```

5.1.4 Discussion sur les résultats obtenus

Ce que nous observons en regardant les valeurs de 'Output', c'est que le 'duty ratio' se comporte de façon logique, c'est-à-dire que lorsque la puissance augmente, 'output' ('duty ratio') envoie le signal de réduire la tension de sortie puisque la puissance fournie par les panneaux solaire est en augmentation et lorsque la puissance diminue, 'output' ('duty ratio') envoie le signal (vers la batterie de secours) d'augmenter la tension d'opération des panneaux solaires.

ANALYSE GÉNÉRALE

L'annexe A contient une vérification de l'algorithme (et du code C) qui a été développé sur le prototype. La vérification a été faite grâce à une simulation écrite dans le tabulateur Excel. Les valeurs discrètes Input1 = 6 et Input2 = -25 ont donc été incorporées dans le programme C et dans le tabulateur Excel. Les tableaux du test Excel présenté à l'annexe A montrent uniquement les résultats des calculs obtenus et non pas les formules. Voici un exemple de formules pour un calcul intermédiaire :

=EXP(-1*POWER((L5+18);2)/(2*POWER(6,5;2))) ; Gaussien

=MIN(F17;M5) ; Minimum

Le résultat Output obtenu dans les deux (2) cas a été comparé :

Output Code C : -13.42

Output Excel : -12.89

L'explication de la différence obtenue dans le résultat du test unitaire de l'annexe A tient du fait que dans le programme du prototype, dès qu'une valeur discrète 'Input 1' ou 'Input 2' passe par une des courbe représentant les 'membership functions', la courbe est immédiatement ajouté dans le calcul des Max-Min, même si la valeur de probabilité (entre 0 et 1) est extrêmement petite. Pour éliminer cet inconvénient, la constante (`#define MIN 0.05`) a été ajouté dans le programme. Cela empêche les courbes touchant à peine aux valeurs discrètes d'être sélectionnées.

Une analyse plus poussée des erreurs d'arrondis mérite d'être faite pour deux (2) raisons. Premièrement, il y a de nombreux calculs qui sont exécutés pendant l'étape de fuzzification et durant l'agrégation et une accumulation des erreurs d'arrondis pourrait se produire. Deuxièmement, pour accélérer la vitesse des opérations arithmétiques, il serait pertinent d'utiliser des types de variables ayant moins de précision (versus le nombre de décimale) et cela pourrait également ajouter une erreur d'arrondis.

Dans la mesure ou devrait se poursuivre ce projet, c'est-à-dire dans l'éventualité probable ou la programmation serait revue, améliorée et optimisée, une fonction, entre autre, de contrôle des vitesses d'exécution des temps machine des différents modules ou partie de code devrait être écrite. Cette fonction doit, en simplifiant, noter dans un log le module testé, l'heure d'entrée et l'heure de sortie du module au millième ou millionième de seconde près. Elle permettra, de plus, d'aider à la synchronisation des échanges avec le système global.

Le code est évidemment adaptable pour résoudre, par l'application de la méthode Mamdani Max-Min, un problème similaire. L'ajout ou le retrait d'une 'membership fonction' ou l'utilisation d'une distribution autre que gaussienne est relativement simple à

faire. Dans le processus de résolution d'un problème par la logique floue, le défi est davantage de définir les règles d'inférence.

CHAPITRE 6
CONCLUSION GÉNÉRALE

Le projet porte sur le développement d'un module de contrôle MPPT avec une approche en logique floue sur une plateforme 'légère' afin d'être intégré dans un système général et mobile de conversion d'énergie renouvelable en électricité. Des études ont démontré que la logique floue donnait de meilleur résultat, moins d'oscillation dans les ajustements dues aux fluctuations, que la méthode P&O sur ce type de contrôle. MPPT (Maximum Power Point Tracker) est le point ou la puissance générée par le système est optimale, en dépit des changements de rayonnement solaire ou de température.

Une partie importante du projet est de choisir l'outil sur lequel sera implanté le contrôleur MPPT. De nos jours, les microcontrôleurs, tel le FPGA, sont offerts en 'package' c'est-à-dire qu'ils font partie d'un circuit avec chips de mémoire, convertisseurs, ports d'entrées/sorties et logiciel dédié pour la configuration. Un meilleur choix est d'utiliser le Raspberry PI II. RPI III est maintenant disponible sur le marché, mais le développement de ce projet a été fait sur RPI II. Cet outil remplit les fonctions d'un ordinateur, il possède les ports USB pour un clavier, une souris ou un connecteur réseau sans-fil. Il est doté, de plus, d'un port GPIO par lequel plusieurs protocoles de communication sont accessibles.

RPI possède un noyau d'exploitation Linux (Debian par défaut) et supporte plusieurs outils de développement et langages de programmation. La librairie BCM2835 simplifie la communication avec le port GPIO par l'entremise de plusieurs fonctions bien documentées, écrites dans le langage C.

Trois (3) principales fonctions ont donc été écrites dans le langage C et avec l'aide du GUI Geany, elles ont été 'débuggés' et testés. Deux (2) de ces fonctions sont exclusivement spécialisées pour la communication avec le GPIO, soit une (1) pour la lecture et une (1) pour l'écriture. La fonction restante accepte les 'Input1' et 'Input2' et fait le calcul, par la méthode de logique floue Mamdani Max-Min, de la valeur 'Output' ('duty ratio') qui a pour office de signaler au système l'ajustement à faire sur la tension d'opération. Les tests unitaires ont été réalisés ainsi qu'un test à partir de valeurs réelles et les résultats sont concluants.

La fiabilité physique de RPI doit cependant être testé et validé dans des conditions normales d'utilisation et à des températures basses et élevées. Il peut être connecté sur un réseau global et donc être auditer régulièrement afin de déceler des anomalies de fonctionnement ou des pannes.

Le code est ouvert à l'amélioration, par exemple l'utilisation d'une distribution autre que gaussienne pourrait être programmée à l'étape de défuzzication et à l'optimisation, par exemple programmer une version deux (2) plus rapide en temps machine et plus courte en ligne de code. Du fait de la souplesse de son port de communication GPIO, le RPI pourrait effectuer des tâches qui sont actuellement remplies par le système global et réduire le nombre de microcontrôleurs sur les circuits du système global. L'exemple le plus frappant, à ce stade, est la fonction qui a été écrite 'fFuzzInput' pour réaliser le test avec des données réelles à la section 5.1. Autrement dit, les valeurs numériques de tension V et de courant I pourrait être envoyées directement dans les registres de lecture du GPIO, à la place des valeurs 'Input1' et 'Input2'.

RÉFÉRENCES BIBLIOGRAPHIQUES

[1] GOUVERNEMENT DU QUÉBEC. « LA STRATÉGIE ÉNERGÉTIQUE DU QUÉBEC 2006-2015 ». <http://mern.gouv.qc.ca/publications/energie/strategie/strategie-energetique-2006-2015.pdf>.

[2] CHEHOURI A., YOUNES R., ILINCA A., PERRON J., "REVIEW OF PERFORMANCE OPTIMIZATION TECHNIQUES APPLIED TO WIND TURBINES", 2015, APPLIED ENERGY, VOL. 142, PP. 361-388)

[3] A. ANURAG, S. BAL, S. SOURAV, M. NANDA – A REVIEW OF MAXIMUM POWER-POINT TRACKING TECHNIQUES FOR PHOTOVOLTAIC SYSTEMS », INTERNATIONAL JOURNAL OF SUSTAINABLE ENERGY, 2014, DOI : 10.1080/14786451.2014.918979;

[4] A. K. NUR, W. T. CHEE – « A COMPREHENSIVE REVIEW OF MAXIMUM POWER POINT TRACKING ALGORITHMS FOR PHOTOVOLTAIC SYSTEMS », RENEWABLE AND SUSTAINABLE ENERGY REVIEWS 37 (2014) 585–598;

[5] T. ESRAM, P. L. CHAPMAN, « COMPARISON OF PHOTOVOLTAIC ARRAY MAXIMUM POWER POINT TRACKING TECHNIQUES », IEEE TRANSACTIONS ON ENERGY CONVERSION, VOL. 22, NO. 2, JUNE 2007.

[6] C. LARBES, , S.M. AIT CHEIKH, T. OBEIDI, , A. ZERGUERRAS, 2009. GENETIC ALGORITHMS OPTIMIZED FUZZY LOGIC CONTROL FOR THE MAXIMUM POWER POINT TRACKING IN PHOTOVOLTAIC SYSTEM. RENEWABLE ENERGY 34, (2009), 2093–2100;

[7] M.A. FARAHAT, H.M.B. METWALLY, AHMED ABD-ELFATAH MOHAMED, OPTIMAL CHOICE AND DESIGN OF DIFFERENT TOPOLOGIES OF DC-DC CONVERTER USED IN PV SYSTEMS, AT DIFFERENT CLIMATIC CONDITIONS IN EGYPT, RENEWABLE ENERGY 43 (2012) 393-402)

[8] G.F. TCHOKETCH KEBIR, COMMANDE DES HACHEURS MPPT PAR LA LOGIQUE FLOUE, MÉMOIRE DE MAGISTER, ÉCOLE NATIONALE POLYTECHNIQUE (ENP); 2006;

[9] M.S. AÏT CHEIKH, C. LARBES, G.F. TCHOKETCH KEBIR, A. ZERGUERRAS. 2007. MAXIMUM POWER POINT TRACKING USING A FUZZY LOGIC CONTROL SCHEME. REVUE DES ÉNERGIES RENOUVELABLES, VOL. 10 N°3 (2007) 387-395;

[10] C.Y. WON, D.H KIM, S.C. KIM, W.S KIM, H.S. KIM, A NEW MAXIMUM POWER POINT TRACKER OF PHOTOVOLTAIC ARRAYS USING FUZZY CONTROLLER, 0-7803-1859-5/94/ 1994 IEEE).

[11] GADGETOID. « RASPBERRY PINOUT ». < http://pinout.xyz/>.

[12] T. OBEIDI, G.F. TCHOKETCH KEBIR, A. ILINCA,C. LARBES, D. RAMDENEE, « MPPT FOR A SOLAR ELECTRIC VEHICLE WITH AN INTELLIGENT FUZZY CONTROLLER, 2015 ».

[13] BINDICHEN. « Mamdani fuzzy models».

<http://www.bindichen.co.uk/post/AI/mamdani-fuzzy-model.html>.

[14] MATHWORKS. « Fuzzy inference process».

<http://www.mathworks.com/help/fuzzy/fuzzy-inference-process.html>.

[15] MATHWORKS. «GAUSMF».
<http://www.mathworks.com/help/fuzzy/gaussmf.html>.

[16] MOSAIC DOCUMENTATION WEB. «GPIO ELECTRICAL SPECIFICATIONS». <HTTP://WWW.MOSAIC-INDUSTRIES.COM/EMBEDDED-SYSTEMS/MICROCONTROLLER-PROJECTS/RASPBERRY-PI/GPIO-PIN-ELECTRICAL-SPECIFICATIONS>.

[17] WIND&SUN. «WHAT THE HECK IS AN MPPT CHARGE CONTROLLER?». <HTTP://WWW.SOLAR-ELECTRIC.COM/MPPT-SOLAR-CHARGE-CONTROLLERS.HTML>.

[18] SPAZZTECH. «C Programming on Raspberry PI - IDE».
<https://www.youtube.com/watch?v=usbwgc5xt_m>.

[19] SPAZZTECH. «C Programming on Raspberry PI – Library».
<https://www.youtube.com/watch?v=KwSq-WxNqAA>.

[20] SPAZZTECH. «C Programming on Raspberry PI – Led». <https://www.youtube.com/watch?v=jlM9KoWyPv0>.

[21] SPAZZTECH. «C Programming on Raspberry PI – GPIO». <https://www.youtube.com/watch?v=dJBHxkwH1PI>.

[22] SPAZZTECH. «C Programming on Raspberry PI – GPIO». <https://www.youtube.com/watch?v=XZMnc6Z0cco>.

[23] SPAZZTECH. «C Programming on Raspberry PI – SPI». <https://www.youtube.com/watch?v=0QcteUtzB4o>.

[24] SPAZZTECH. «C Programming on Raspberry PI – GPIO». <https://www.youtube.com/watch?v=eObSqbe9aqU>.

[25] TKJ ELECTRONICS. «Raspberry PI GPIO tutorial». <https://www.youtube.com/watch?v=8mLMGlzdxrE>.

[26] GAVEN MACDONALD. «GPIO wiringPI in C». <https://www.youtube.com/watch?v=J6KsTz6hjfU>.

[27] ADVANCED DIGITAL TECHNOLOGIES - UPB BUCARAMANGA. «GPIO on Raspberry PI». <https://sites.google.com/site/semilleroadt/raspberry-pi-tutorials/gpio>.

[28] IDR SOLUTIONS. «Top 8 IDES for programmers, coders and beginners on the Raspberry PI». <https://blog.idrsolutions.com/2014/12/top-8-ides-programmers-coders-beginners-raspberry-pi/>.

[29] WIRINGPI. «GPIO interface library for the Raspberry PI». <http://wiringpi.com//>.

[30] RASPBERRY PI FOUNDATION. «BCM2835». <https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2835/>.

[31] RASPBERRY PI FOUNDATION. «GPIO and physical computing on the Raspberry PI». <https://www.raspberrypi.org/documentation/usage/gpio/>.

[32] MIKE MCCAULEY. «C library for Broadcom BCM 2835 as used in Raspberry PI». < http://www.airspayce.com/mikem/bcm2835/>.

[33] CANAKIT. «Canakit Raspberry PI GPIO breakout board bundle ». <http://www.canakit.com/raspberry-pi-cobbler-gpio-breakout-bundle.html>.

[34] RASPBERRY PI HQ. «How to: backup & restore your Raspberry PI». <http://raspberrypihq.com/how-to-backup-your-raspberry-pi/>.

[35] BENJAMIN KNAPP. «Fuzzy inference systems». <http://www.cs.princeton.edu/courses/archive/fall07/cos436/HIDDEN/Knapp/fuzzy004.htm#E7E101>.

[36] CALVIN. «Making fuzzy decisions». <http://www.calvin.edu/~pribeiro/othrlnks/Fuzzy/fuzzydecisions.htm>.

[37] ENCODER. «Fuzzy logic tutorial». <http://www.seattlerobotics.org/encoder/mar98/fuz/flindex.html>.

[38] EMATHTEACHER. «Mamdani's method». <http://www.dma.fi.upm.es/recursos/aplicaciones/logica_borrosa/web/fuzzy_inferencia/mamdani3_en.htm>.

[39] BILKENT UNIVERSITY. «A short fuzzy logic tutorial». <http://cs.bilkent.edu.tr/~zeynep/files/short_fuzzy_logic_tutorial.pdf>.

[40] MASSEY UNIVERSITY. «Fuzzy logic». <http://www.massey.ac.nz/~nhreyes/MASSEY/159741/Lectures/Lec2012-3-159741-FuzzyLogic-v.2.pdf>.

[41] GREG VIOT. «Fuzzy logic in C». <http://www.chebucto.ns.ca/science/aimet/archive/ddj/fuzzy_logic_in_c/>.

[42] WIKIPEDIA. «Field-Programmable Gate Array». <https://en.wikipedia.org/wiki/Field-programmable_gate_array>.

[43] EMBEDDED. «FPGA programming step by step». <http://www.embedded.com/design/prototyping-and-development/4006429/FPGA-programming-step-by-step>.

[44] XILINK. «Boards, kits, and modules». <http://www.xilinx.com/products/boards-and-kits.html>.

[45] HARDENT. «Academy». <http://www.hardent.com/electronic-fpga-design-consulting-services/academy/>.

ANNEXES

FONCTION 'FCALCULFUZZY'

A.1.1 Test avec le programme

Le résultat du test de la fonction sur le RPI, avec les valeurs en entrée 6 et -25, est le suivant :

Commande :

```
fCalculFuzzy (6, -25)
```

Résultat :

```
Input 1:   6, Input : -25, Output: -13.42

------------------
(program exited with code: 0)
Press return to continue
```

A.1.2 Test avec le tabulateur Excel

Les étapes et les formules ont été respectées. Les tableaux montrent uniquement les valeurs calculées et non les formules.

Les Max-Min calculés se retrouve à droite des 'membership function Output' Z et SI dans le tableau 4.

Le tableau 5 effectue l'agrégation et le résultat donne -12.89

Tableau 5 : Max-Min du test ‘fCalculFuzzy’

Input1			Input2		
z	6	0,181928412	ns	-25	0,661515
ps	6	0,827497567	z	-25	0,317321
z&ns=si		0,181928412			
z&z=z		0,181928412			
ps&ns=si		0,661514656			
ps&z=si		0,317320791			
	z	0,181928412			
	si	0,661514656			

Tableau 6 : Agrégation du test ‘fCalculFuzzy’

X	Y	Min SI = dénominateur	numérateur		Y	Min Z = dénominateur	numérateur
-62,00	1,12E-10	1,12E-10	-6,95E-09		1,75E-20	1,75E-20	-1,09E-18
-60,00	8,59E-10	8,59E-10	-5,15E-08		3,14E-19	3,14E-19	-1,89E-17
-58,00	5,98E-09	5,98E-09	-3,47E-07		5,13E-18	5,13E-18	-2,98E-16
-56,00	3,79E-08	3,79E-08	-2,12E-06		7,63E-17	7,63E-17	-4,27E-15
-54,00	2,18E-07	2,18E-07	-1,18E-05		1,03E-15	1,03E-15	-5,56E-14
-52,00	1,14E-06	1,14E-06	-5,95E-05		1,27E-14	1,27E-14	-6,59E-13
-50,00	5,46E-06	5,46E-06	-2,73E-04		1,42E-13	1,42E-13	-7,08E-12
-48,00	2,37E-05	2,37E-05	-1,14E-03		1,44E-12	1,44E-12	-6,91E-11
-46,00	9,34E-05	9,34E-05	-4,30E-03		1,33E-11	1,33E-11	-6,13E-10
-44,00	3,35E-04	3,35E-04	-1,48E-02		1,12E-10	1,12E-10	-4,93E-09
-42,00	1,10E-03	1,10E-03	-4,60E-02		8,59E-10	8,59E-10	-3,61E-08
-40,00	3,25E-03	3,25E-03	-1,30E-01		5,98E-09	5,98E-09	-2,39E-07
-38,00	8,79E-03	8,79E-03	-3,34E-01		3,79E-08	3,79E-08	-1,44E-06
-36,00	2,16E-02	2,16E-02	-7,78E-01		2,18E-07	2,18E-07	-7,86E-06
-34,00	4,83E-02	4,83E-02	-1,64E+00		1,14E-06	1,14E-06	-3,89E-05
-32,00	9,83E-02	9,83E-02	-3,15E+00		5,46E-06	5,46E-06	-1,75E-04
-30,00	1,82E-01	1,82E-01	-5,46E+00		2,37E-05	2,37E-05	-7,10E-04
-28,00	3,06E-01	3,06E-01	-8,57E+00		9,34E-05	9,34E-05	-2,62E-03

-26,00	4,69E-01	4,69E-01	-1,22E+01		3,35E-04	3,35E-04	-8,72E-03
-24,00	6,53E-01	6,53E-01	-1,57E+01		1,10E-03	1,10E-03	-2,63E-02
-22,00	8,27E-01	6,62E-01	-1,46E+01		3,25E-03	3,25E-03	-7,16E-02
-20,00	9,54E-01	6,62E-01	-1,32E+01		8,79E-03	8,79E-03	-1,76E-01
-18,00	1,00E+00	6,62E-01	-1,19E+01		2,16E-02	2,16E-02	-3,89E-01
-16,00	9,54E-01	6,62E-01	-1,06E+01		4,83E-02	4,83E-02	-7,73E-01
-14,00	8,27E-01	6,62E-01	-9,26E+00		9,83E-02	9,83E-02	-1,38E+00
-12,00	6,53E-01	6,53E-01	-7,84E+00		1,82E-01	1,82E-01	-2,18E+00
-10,00	4,69E-01	4,69E-01	-4,69E+00		3,06E-01	1,82E-01	-1,82E+00
-8,00	3,06E-01	3,06E-01	-2,45E+00		4,69E-01	1,82E-01	-1,46E+00
-6,00	1,82E-01	1,82E-01	-1,09E+00		6,53E-01	1,82E-01	-1,09E+00
-4,00	9,83E-02	9,83E-02	-3,93E-01		8,27E-01	1,82E-01	-7,28E-01
-2,00	4,83E-02	4,83E-02	-9,67E-02		9,54E-01	1,82E-01	-3,64E-01
0,00	2,16E-02	2,16E-02	0,00E+00		1,00E+00	1,82E-01	0,00E+00
2,00	8,79E-03	8,79E-03	1,76E-02		9,54E-01	1,82E-01	3,64E-01
4,00	3,25E-03	3,25E-03	1,30E-02		8,27E-01	1,82E-01	7,28E-01
6,00	1,10E-03	1,10E-03	6,57E-03		6,53E-01	1,82E-01	1,09E+00
8,00	3,35E-04	3,35E-04	2,68E-03		4,69E-01	1,82E-01	1,46E+00
10,00	9,34E-05	9,34E-05	9,34E-04		3,06E-01	1,82E-01	1,82E+00
12,00	2,37E-05	2,37E-05	2,84E-04		1,82E-01	1,82E-01	2,18E+00
14,00	5,46E-06	5,46E-06	7,64E-05		9,83E-02	9,83E-02	1,38E+00
16,00	1,14E-06	1,14E-06	1,83E-05		4,83E-02	4,83E-02	7,73E-01
18,00	2,18E-07	2,18E-07	3,93E-06		2,16E-02	2,16E-02	3,89E-01
20,00	3,79E-08	3,79E-08	7,58E-07		8,79E-03	8,79E-03	1,76E-01
22,00	5,98E-09	5,98E-09	1,32E-07		3,25E-03	3,25E-03	7,16E-02
24,00	8,59E-10	8,59E-10	2,06E-08		1,10E-03	1,10E-03	2,63E-02
26,00	1,12E-10	1,12E-10	2,92E-09		3,35E-04	3,35E-04	8,72E-03
28,00	1,33E-11	1,33E-11	3,73E-10		9,34E-05	9,34E-05	2,62E-03
30,00	1,44E-12	1,44E-12	4,32E-11		2,37E-05	2,37E-05	7,10E-04
32,00	1,42E-13	1,42E-13	4,53E-12		5,46E-06	5,46E-06	1,75E-04
34,00	1,27E-14	1,27E-14	4,31E-13		1,14E-06	1,14E-06	3,89E-05
36,00	1,03E-15	1,03E-15	3,71E-14		2,18E-07	2,18E-07	7,86E-06
38,00	7,63E-17	7,63E-17	2,90E-15		3,79E-08	3,79E-08	1,44E-06
40,00	5,13E-18	5,13E-18	2,05E-16		5,98E-09	5,98E-09	2,39E-07
42,00	3,14E-19	3,14E-19	1,32E-17		8,59E-10	8,59E-10	3,61E-08
44,00	1,75E-20	1,75E-20	7,71E-19		1,12E-10	1,12E-10	4,93E-09
46,00	8,88E-22	8,88E-22	4,08E-20		1,33E-11	1,33E-11	6,13E-10

IV

48,00	4,09E-23	4,09E-23	1,96E-21		1,44E-12	1,44E-12	6,91E-11
50,00	1,72E-24	1,72E-24	8,58E-23		1,42E-13	1,42E-13	7,08E-12
52,00	6,55E-26	6,55E-26	3,40E-24		1,27E-14	1,27E-14	6,59E-13
54,00	2,27E-27	2,27E-27	1,23E-25		1,03E-15	1,03E-15	5,56E-14
56,00	7,17E-29	7,17E-29	4,02E-27		7,63E-17	7,63E-17	4,27E-15
58,00	2,06E-30	2,06E-30	1,19E-28		5,13E-18	5,13E-18	2,98E-16
60,00	5,38E-32	5,38E-32	3,23E-30		3,14E-19	3,14E-19	1,89E-17
62,00	1,28E-33	1,28E-33	7,93E-32		1,75E-20	1,75E-20	1,09E-18
Somme		6,89E+00	-1,24E+02			2,73E+00	4,64E-16
Total						9,62E+00	-1,24E+02
Num/Den							-12,894