الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي

UNIVERSITÉ BADJI MOKHTAR - ANNABA
BADJI MOKHTAR – ANNABA UNIVERSITY

جامعة باجي مختار-
عنابـــــــــة

Faculté :
Sciences de L'Ingéniorat
Département :
Electronique
Domaine :
Sciences et Technologie
Filière :
Electronique
Spécialité :
Instrumentation

Mémoire
Présenté en vue de l'obtention du Diplôme de Master
Thème :

Commande numérique en courant et en vitesse sans capteur d'un moteur à courant continu

Présenté par : *AMARA ROUMAISSA- AMAIDIA HAMIDA*

Encadrant : Hadj Ahmed ABBASSI PROF UBM.ANNABA

Jury de Soutenance :

Noureddine GUERSI	PROF	UBM.ANNABA	Président
Hadj Ahmed ABBASSI	PROF	UBM.ANNABA	Encadrant
Salah BENSAOULA	MCA	UBM.ANNABA	Examinateur

Année universitaire : 2019-2020

ملخص

اعتمدنا في عملنا على واحدة من اهم الطاقات المتجددة وهي الطاقة الكهروضوئية ومن اهم استخداماتها نظام الضخ الكهروضوئي الدي يعتمد أساسا على التحكم في تصحيح وتغيير سرعة المحرك ودلك باستخدام التكنولوجيات الجديدة التي تتماشى مع التطور التكنولوجي الا وهي واجهة ماتلاب-اردوينو ومن اجل ضمان أكبر كمية من الطاقة قمنا بمتابعة ومراقبة تغيراهم القيم التي تتحكم في النظام الكهروضوئي مثل الحرارة، التيار والتوتر فقمنا بإنشاء نظام الحصول على بيانات لتشخيص النظام الكهروضوئي

الكلمات الرئيسية: الطاقة الكهروضوئية. واجهة ماتلاب - اردوينو.الضخ الكهروضوئي

ABSTRACT

We have adopted in our work on one of the most important renewable energy, which is photovoltaic energy. One of its most important uses is the photovoltaic pumping system, which mainly depends on the control and the correction of the engine speed. The use of new technologies in phase with technological development; this is the Matlab-Arduino interface, in order to ensure the greatest amount of energy. We have monitored their changes of value that controlled the photovoltaic system, such as temperature, current and voltage, so we created a data acquisition system for diagnosing the photovoltaic system.

Key words: renewable energy, photovoltaic pumping system, Matlab-Arduino interface, photovoltaic energy, data acquisition system

RESUME

Nous avons adopté dans notre travail sur l'une des énergies renouvelables les plus importantes, qui est l'énergie photovoltaïque. L'une de ses utilisations les plus importantes est le système de pompage photovoltaïque, qui dépend principalement du contrôle et la correction du régime moteur. L'utilisation des nouvelles technologies en phase avec le développement technologique, c'est l'interface Matlab-Arduino, afin d'assurer la plus grande quantité d'énergie. Nous avons surveillé leurs changements de valeur qui contrôlaient le système photovoltaïque, tels que la température, le courant et la tension, nous avons donc créé un système d'acquisition de données pour le diagnostic le système photovoltaïque.

Mot clés : énergies renouvelables, l'énergie photovoltaïque, le système de pompage photovoltaïque, l'interface Matlab-Arduino, système d'acquisition de données

Dédicaces

Je dédié ce modeste travail à :

A notre cher parent que dieu les gardes et les protèges

Pour leur patience, leur amour, leur soutien et leurs encouragements.

Toute la famille AMAIDIA et AMARA

Mes chères amies : MAYSSA, NESRINE, MERIEM, NARDJES

Et toute sa famille

Sans oublier tous les professeurs que ce soit du primaire, du moyen, du secondaire ou de l'enseignement supérieur.

Remerciement

En premier lieu, nous remercions Dieu, le tout puissant, pour avoir donné, le courage, la patience, la volonté et la force nécessaires, pour affronter toutes les difficultés et les obstacles, qui se sont hissés au travers de notre chemin, durant toutes nos années d'études.

Tout d'abord, nous adressons notre gratitude et nos profonds remerciements à Monsieur Hadj Ahmed ABBASSI (Professeur à l'université Badji Mokhtar Annaba) pour nous avoir encadré dans ce mémoire de master, pour ses conseils, ses encouragements pendant les moments délicats et son travaille avec nous.

Nous tenons à exprimer nos profonds remerciements à Mr N.Guersi et Mr S.Bensaoula (Professeurs à l'université Badji Mokhtar Annaba) de nous avoir fait l'honneur d'accepte d'examiner et juger ce mémoire de master . Et je remercier ma collègue Amaidia Hamida de m'avoir aidée.

Merci à ma famille de me soutenir.

Liste des Figures

FIG	Titre	Page
II.1	Schéma bloc d'un système en boucle ouverte	13
II.2	Schéma bloc d'un système en boucle fermée avec un retour unitaire	13
II.3	Schéma bloc d'un système avec correcteur (Retour unitaire)	14
II.4	Schéma bloc d'un système avec correcteur (Boucle fermée)	15
II.5	Schéma bloc du correcteur PI en régulation de vitesse	15
III.1	Schéma d'un circuit de puissance	19
III.2	Schéma d'un circuit de consigne analogique	20
III.3	schéma d'un circuit d'asservissement de vitesse par fcem	21
III.4	schéma fonctionnel avec les actions P et I en chaine directe	23
III.5	schéma fonctionnel avec l'action P en sortie et l'action I en chaine directe	25
III.6	sélectionner de Add-Ons/Get hardware support package	27
III.7	la fenêtre de Add-Ons	28
III.8	Installation de SIMULINK SUPPORT PACKAGE FOR ARDUINO HARDWARE	28
III.9	connexion sur le compte MATLAB	29
III.10	connexion d'Arduino au PC	29
III.11	Les propriétés de la carte Arduino connecté au PC au MATLAB	30
III.12	Commun bloc d'Arduino	31
III.13	montre l'interfaçage entre la partie programmation et partie matérielle	32
III.14	la variation d'un train d'impulsion	32

III.15	une description d'un rapport cyclique	33
III.16	Schéma sur Simulink (consigne numérique)	33
III.17	Schéma électrique avec consigne numérique	34
III.18	Schéma électrique de consigne analogique	34
III.19	Schéma en Simulink (consigne analogique)	35
III.20	Schéma en Simulink (valeur min)	35
III.21	Schéma en Simulink (valeur max)	36
III.22	Schéma électrique des correcteurs p, pi	36
III.23	le bloc PID	37
III.24	paramètres de PID Controller	38
III.25	Schéma en Simulink de correcteur p	38
III.26	Schéma en Simulink de correcteur p après variation	39
III.27	Les signaux affiches sur scope de correcteur p	39
III.28	les paramètres de PID Controller	40
III.29	Schéma en Simulink de correcteur PI	41
III.30	Schéma en Simulink de correcteur PI après variation	41
III.31	Le signaux affichés sur scope de correcteur PI	42
III.32	Image de circuit final de la variation et la régulation de vitesse d'un MCC	42
IV.1	Bilan d'acquisition de donné	44
IV.2	les caractéristiques de panneau photovoltaïque	45
IV.3	Schéma de réalisation de capteur de température	46
IV.4	Schéma de capteur de tension PV	47
IV.5	Schéma de réalisation d'un capteur de courant pour PV	48
IV.6	schéma en Simulink	49
IV.7	Exemple sur les résultats de simulation de variation de la température, tension et courant de panneau	49

Liste des Symboles

Symbole	Description
PV	panneau photovoltaïque
DC/AC	courant continu/courant alternatif
DC/DC	courant continu/courant continu
PI	proportionnel-Intégral
PWM	modulation de largeur d'impulsion
Kp	gain d'action proportionnelle.
Ki	gain d'action intégrale
Kd	gain d'action dérivée.
Ti	constante de temps, dite temps d'action intégrale
Td	constante de temps, dite temps d'action dérivée.
Uc(t)	signale de commande
Ɛ(t)	signal d'erreur
MOSFET	transistor à effet de champ a structure métal-oxyde-semi-conducteur
Rds	résistance (drain source)
Vgs	voltage (gate source)
MCC	Moteur à courant continu
USB	BUS série universel
MATLAB	Matrix LABoratory

Table des matières

Résumé…….....I

Dédicaces…….....II

Remerciements……..…….....III

Liste des figures…… .....…… IV-V

Liste des symboles ……...…….....VI

Tables des matières……..……..……..1

Introduction générale……..….....4

Chapitre 1 : le système de pompage solaire

I.1. Introduction ……...5

I.2. Le système photovoltaïque…….5

I.2.1. Systèmes autonomes …….....5

I.2.2. Systèmes hybrides…….6

I.2.3. Systèmes couples au réseau……...6

I.2.4. Systèmes fonctionnant au fil du soleil (pompage photovoltaïque) ……...6

I.2.4.1. Introduction ……6

I.2.4.2. Configuration typique d'un système de pompage photovoltaïque…….. …...6

I.2.4.2.1. Système de petit puissance (50-400W)……7

I.2.4.3. Technologies des moteurs ……..7

I.2.4.3.1. Moteur à courant continu…… 7

I.2.4.3.2. Moteur à courant continu sans balais……...8

I.2.4.3.3. Moteur à courant alternatif……...9

I.2.4.4. Les types des pompes……. 9

I.2.4.4.1. La pompe volumétrique……..9

I.2.4.4.2. La pompe centrifuge ……10

I.3. Conclusion……10

Chapitre 2 : Techniques de la régulation de la vitesse d'un moteur à courant continu

II.1. Introduction ……11

II.2. Définition de la régulation /asservissement……11

II.2.1. L'asservissement ……11
II.2.2. La régulation……11
II.3. Principe générale de la régulation……12
II.4. Système en boucle ouvert ……12
II.5. Système en boucle fermé……13
II.6. Correction des systèmes asservis ……....13
II.7. Le correcteur PI (proportionnel / intégrateur) ……14
II.8. Effets du correcteur proportionnel-Intégrale……16
II.9. Conclusion ……..16
Chapitre 3 : Implémentation du régulateur de vitesse sans capteur sur carte Arduino
III.1. Introduction……....17
III.2. Régulation avec Arduino……18
III.2.1. Présentation matérielle……...…18
III.2.1.1. Circuit de puissance……18
III.2.1.2. Consigne analogique……...19
III.2.1.3. Mesure de la vitesse……20
III.2.2. Présentation logiciel……...21
III.2.2.1. Programme PWM avec consigne numérique……..21
III.2.2.2. Programme PWM avec consigne analogique……..22
III.2.2.3. Programme avec correction P……..22
III.2.2.4. Programme avec correction PI……23
III.2.2.5. Programme avec correction PI combinée (action P sur la sortie)……25
III.3. Regulation avec Matlab Arduino (Simulink)……..26
III.3.1. L'environnement Matlab/Simulink ……26
III.3.1.1. Matlab……26
III.3.1.2. Simulink ……26
III.3.2. Matlab support package for Arduino Hardware ……27
III.3.2.1. Configuration du package de support matériel pour Matlab 27
III.3.2.2. Test de Matlab……29
III.3.2.3. Examiner la bibliothèque de bloc Arduino ……..…..30

III.3.3. Fonctionnement et programmation……..……32
III.3.3.1. Consigne numérique (signal PWM) ……32
III.3.3.2. Consigne analogique (potentiomètre)……...34
III.3.3.3. Les correcteurs P, PI…….36
III.3.3.3.1. Le bloc PID en Simulink…….37
III.3.3.3.2. Correcteur proportionnel P …….37
III.3.3.3.3. Correcteur PI……...40
III.4. Conclusion ……..43
Chapitre 4 : conception et interface graphique d'un système de pompage solaire
IV.1. Introduction …… .44
IV.2 Matériels et composants utilisés ……..44
IV.2.1. Panneau photovoltaïque……44
IV.2.2. La batterie de panneau et ces caractéristiques……..45
IV.2.3. Les capteurs……45
IV.2.3.1. Capteur de température LM35……...45
IV.2.3.2 Capteur de tension de panneau PV…….46
IV.2.3.3 Capteur de courant pour (PV)…….47
IV.2.4. L'interface graphique de Matlab/arduino ……48
IV.3. Conclusion ……..50
Conclusion générale……51
Bibliographié……...52
Annexe……..53

Introduction générale

L'énergie solaire photovoltaïque (**PV)** est utilisée de plus en plus dans des applications diverses comme l'éclairage, la réfrigération et le pompage. Les systèmes **PV** peuvent être complètements autonome et ne nécessitent aucune source d'alimentation extérieure tel que les combustibles ; de plus, le système ne requiert pratiquement pas d'entretien vu qu'il ne contient aucune pièce mécanique. Par conséquent, les coûts récurrents d'opération et de maintenance sont relativement faibles. Pour ces raisons, cette source d'énergie convient particulièrement bien pour les utilisations en milieu rural où les populations sont réparties dans de petites communautés et où la demande énergétique est relativement faible [1]. L'une des applications les plus importantes de l'énergie solaire photovoltaïque(**PV**) est le pompage de l'eau soit quand l'ensoleillement est relativement fort ou bien quand le soleil est absent et tout sa grâce à l'énergie stocker par la batterie. Ceci en particulier dans les zones rurales qui n'ont pas accès aux réseaux électrique ou ils sont utilisée pour l'approvisionnement en eau potable, l'apport en eau pour les bétails et l'irrigation à petite échelle… . Les systèmes de pompage **PV** utilisent des pompes de faible puissance. Ils sont généralement alimentés par un système comprenant un générateur photovoltaïque, une batterie, un contrôleur de charge, un groupe moteur-pompe et un réservoir d'eau pour assurer la continuité de l'approvisionnement lorsque l'énergie solaire ne suffit pas. Vue la simplicité du couplage direct d'un moteur à courant continu avec la batterie cette configuration est la plus utilisée dans les systèmes de pompage photovoltaïque.

Afin d'assurer le plus d'énergie et d'éviter toute pénurie, il est nécessaire de surveiller le changement des grandeurs important dans le système PV, de plus, pour le bon fonctionnement du système de pompage, la vitesse du moteur et l'intensité du courant d'alimentation doivent être surveillées. Nous pouvons faire tout cela en utilisant la plateforme MATLAB Arduino. Le document a été divisé en quatre chapitres dont le contenu est résumé ci-dessous :

- Le premier chapitre donne une description sur le système de pompage solaire.
- Dans le second chapitre, on passe en revue de décrire les techniques de la régulation de la vitesse d'un moteur à courant continu, le principe général de la régulation et les correcteurs.
- Le troisième chapitre présente l'implémentation du régulateur de vitesse sans capteur sur carte Arduino et par Matlab-Simulink.
- Le dernier chapitre sera consacré à l'acquisition et à la présentation graphique des signaux du système de pompage solaire.

Chapitre I :

Le système de pompage solaire

I.1. Introduction :

Dans nos jours, il nous semble que personne ne peut s'en douter sur l'importance de l'eau et de l'énergie pour les besoins humains. Avec les avancées technologiques, le besoin en énergie ne cesse d'augmenter. Ce problème d'énergie est encore plus sensible dans les sites isolés où l'utilisation des ressources classiques s'avère souvent très coûteuse. Comme nous le savons, la plus grande partie de l'énergie consommée actuellement provient de l'utilisation des combustibles fossiles comme le pétrole, le charbon, le gaz naturel ou encore l'énergie nucléaire. Les études et les prévisions récentes nous alertent que l'utilisation massive de ces ressources conduira certainement à l'épuisement total de ces réserves. En plus, tout le monde est mondialement convaincu par le danger de ce processus sur l'environnement. A partir de ce constat, il été nécessaire de chercher d'autres ressources d'énergie de remplacement. Les énergies renouvelables, comme l'énergie photovoltaïque, éolienne ou hydraulique, représentent une solution de remplacement par excellence et elles sont de plus en plus utilisées dans nos jours. Ce type d'énergie n'est pas seulement gratuit et inépuisable, mais aussi très propre pour l'environnement. D'ailleurs, on parle souvent d'une énergie « verte », puisqu'elle permet d'éviter totalement la pollution produite par les sources traditionnelles. Dans ce travail de recherche, nous sommes intéressés plus particulièrement par l'énergie solaire photovoltaïque avec comme application, le pompage d'eau dans des sites isolés. Nous verrons que cette solution est particulièrement intéressante pour ce type de site. L'utilisation de l'énergie photovoltaïque pour le pompage de l'eau est bien adaptée pour la plupart des régions arides et semi-arides en raison de l'existence dans ces régions d'un potentiel hydraulique souterrain peu profond. Une autre coïncidence très importante favorise l'utilisation de ce type d'énergie pour le pompage d'eau est que la demande d'eau, surtout dans l'agriculture, atteint son maximum par temps chaud et sec où c'est justement le moment où l'on a accès au maximum d'énergie solaire.

I.2. Le système photovoltaïque

Les principales applications des systèmes photovoltaïques sont énumérées ci-après.

I.2.1. Systèmes autonomes

Une installation photovoltaïque autonome est une installation qui fonctionne indépendamment du réseau électrique ou toutes autres sources d'énergies. Dans la majorité des cas, ce système est utilisé dans les sites isolés. Une tel installation doit être capable de fournir de l'énergie, y compris lorsqu'il n'y a pas de soleil (la nuit ou en cas de mauvais temps). Il faut donc qu'une partie de la production journalière des modules photovoltaïques soit stockée dans des batteries. Cette installation se compose d'un ou plusieurs modules photovoltaïques, d'un régulateur de charge, d'une ou plusieurs

batteries et éventuellement d'un onduleur.

I.2.2. Systèmes hybrides

Les systèmes hybrides consistent en l'association de deux ou plusieurs technologies complémentaires de manière à accroître la fourniture d'énergie. Les sources d'énergie comme le soleil et le vent ne délivrent pas une puissance constante, et leur combinaison peut permettre de parvenir à une production électrique plus continue dans le temps. Les systèmes hybrides fonctionnent tels que, les batteries sont chargées par les panneaux solaires (le jour) et par le générateur éolien.

I.2.3. Systèmes couplés au réseau

Les installations raccordées au réseau électrique (ou reliées à une centrale de distribution) constituent généralement une solution optimale pour la production de l'électricité solaire, tant en termes d'énergie que de coûts. Ces installations se composent de modules photovoltaïques interconnectés, d'un (ou plusieurs) onduleur(s) raccordée au réseau électrique. L'onduleur convertit le courant continu généré par les modules photovoltaïques et produit un courant alternatif conforme au réseau électrique. Illustre le principe d'un système photovoltaïque connecté au réseau.

I.2.4. Systèmes fonctionnant au fil du soleil (Pompage photovoltaïque)

I.2.4.1 Introduction

Généralement, les systèmes de pompage photovoltaïque sont constitués d'un générateur photovoltaïque, un convertisseur du courant électrique qui peut être un convertisseur DC/AC pour un moteur à courant alternatif ou un convertisseur DC/DC pour un moteur à courant continu et d'un groupe motopompe. Ces systèmes fonctionnent au fil du soleil sans stockage électrochimique. L'eau ainsi pompée peut être utilisée directement ou stockée dans un réservoir pour des utilisations ultérieures.

Ce type de stockage de l'eau est la solution la plus adoptée par rapport au stockage électrochimique dans des batteries. Le générateur photovoltaïque est responsable de la conversion instantanée de l'énergie solaire en énergie électrique grâce à l'effet photovoltaïque.

I.2.4.2 Configuration typique d'un système de pompage photovoltaïque

La configuration d'un système de pompage PV est déterminée par la définition du type de générateur photovoltaïque, le type de pompe et de type moteur ; ainsi que le type de conditionnement de

puissance. On peut distinguer les types de systèmes de pompage photovoltaïques suivants :

I.2.4.2.1 Systèmes de petite puissance (50-400W).

Dans ce type de systèmes, il est utilisé principalement un moteur à courant continu entraînant une pompe centrifuge ou à déplacement positif [1]. Entre le générateur PV et le moteur/pompe on intercale un convertisseur DC/DC pour améliorer son adaptation [2]. Les applications de cette configuration sont généralement destinées au pompage des volumes d'eau pouvant atteindre jusqu'à 150m3/Jour.

Les principaux constituants de ces systèmes sont :

• Le Générateur photovoltaïque

• le convertisseur DC/DC (facultatif)

• Moteur à courant continu

• Pompe

• Entrepôt de stockage (facultatif).

I.2.4.3 Technologie des moteurs

On ce qui concerne la technologie des moteurs, il y a trois types de moteur actuellement utilisés pour des applications de pompage photovoltaïques.

• Moteur à courant continue

• Moteur à courant continu à aiment permanant sans balais.

• Moteur à courant alternatif.

I.2.4.3.1. Moteur à courant continu

En termes de simplicité le moteur à courant continu est une option attractive parce que le générateur photovoltaïque produit du courant continu, et moins d'équipement spécialisé sont nécessaire pour traitement de puissance.

Pour un moteur à courant continu conventionnel les pertes d'énergie qui se produisant dans les enroulements est élevé, par conséquence le rendement global est faible.

Si des aimants permanents sont utilisés pour produire le champ magnétique, aucune puissance ne

sera consommée dans les bobines d'excitation et par conséquent des rendements plus élevés seront obtenu. Cette augmentation de rendement est très appropriée aux systèmes de pompage photovoltaïques. Le problème avec le moteur à courant continu c'est qu'il a besoin des balais pour la commutation. Les balais se détériorent avec le temps et doivent être remplacées après approximativement 1000 heures de l'opération [3], ce qui est très incommode parce que ceci signifie un supplément l'entretien et des coûts.

Plusieurs travaux ont été réalisés dont on peut citer quelques-uns : Appelbaum et Sarm

[4] ont examiné le démarrage d'un moteur à courant continu couplé avec une pompe et alimenté par des modules photovoltaïques avec et sans suiveur de point de puissance maximum. Alghuwainem [5] a étudié le fonctionnement, en régime permanent, du moteur à excitation séparé avec un convertisseur élévateur fonctionnant comme un suiveur de point de puissance maximal (MPPT). Anis et al [6] ont trouvé que la charge se compose du moteur à courant continu couplé avec une pompe à volume constant considéré comme une charge non assorti pour les modules photovoltaïques. Weigner et Al [7] ont étudié l'optimisation du rendement global d'un système de pompage photovoltaïque basé sur un moteur à courant continu. Dans [8], Langridge et al ont étudié le fonctionnement d'un système de pompage photovoltaïque direct basé sur un moteur à courant continu sans balais conduisant une pompe à rotor hélicoïdal et la puissance maximale des modules est rendue disponible par une commande appropriée.

La tension de référence des modules est basée sur une fraction de la tension à circuit- ouvert. Les performances dynamiques d'un moteur à aimant permanant sans balais alimenté par des modules photovoltaïques, ont été étudiées par Swamy et al [9]. La conception et la réalisation d'un moteur sans balais pour les applications solaires, ont été étudiées par Azoui [10].

I.2.4.3.2. Moteur à courant continu sans balais

Ces dernières années, il y a eu une croissance rapide de l'utilisation du moteur à courant continu sans balais en tant que machine rotative pour les systèmes de pompage photovoltaïques. Un moteur à courant continu sans balais est une machine électrique tournante où le stator est un stator classique triphasé comme celui d'un moteur asynchrone et le rotor a les aimants permanents montés extérieurs. À cet égard, le moteur à courant continu sans balais est équivalant à un moteur à courant continu avec le commutateur sont inversé où l'aiment tourne tandis que les conducteurs restent stationnaires.

I.2.4.3.3 Moteur a courant alternatif

Les moteurs à courant alternatif comme le moteur asynchrone ; sont des moteurs sans balais avec un rotor de construction robuste qui permet le fonctionnement fiable et sans entretien. La simplicité de la construction du rotor a également comme conséquence la base du prix du moteur et un rapport puissance /poids plus élevé. C'est l'avantage principal des moteurs asynchrone sur les moteurs à courant continu et l'un des raisons pour lesquelles ce type de moteur est le plus utilisé dans les systèmes de pompage photovoltaïques

I.2.4.4 Les type de pompes

Les pompes véhiculant des liquides se divisent en deux catégories principales : les pompes volumétriques et centrifuges. En fonction de l'emplacement physique de la pompe par rapport à l'eau pompée, nous pouvons aussi distinguer deux autres types de pompes : la pompe à aspiration et la pompe à refoulement. La hauteur d'aspiration de n'importe quelle pompe est limitée à une valeur théorique de 10 mètres. Les pompes à aspiration sont donc toujours installées à une hauteur inférieure à celle-ci. Ces pompes doivent également être amorcées, c'est-à-dire que la section en amont de la pompe doit être remplie d'eau pour amorcer l'aspiration d'eau. Les pompes à refoulement sont immergées dans l'eau et ont soit leur moteur immergé avec la pompe (pompe monobloc), soit le moteur en surface et la transmission de puissance se fait alors par un long arbre reliant la pompe au moteur. Dans les deux cas, une conduite de refoulement après la pompe permet des élévations de plusieurs dizaines de mètres, selon la puissance du moteur

I.2.4.4.1. La pompe volumétrique

Les pompes volumétriques fonctionnent en deux temps : remplissage puis vidange d'un volume de liquide, d'où leur appellation. Elles permettent des pressions importantes pour des débits relativement faibles et nécessitent un moteur qui a un couple de démarrage important. Une pompe volumétrique permet en général d'aspirer l'air contenu dans la tuyauterie, on dit alors qu'elle est auto-amorçant [9]. La multitude des différents modes de fonctionnement les classe en deux catégories : les pompes alternatives et les pompes rotatives. La plus connue des pompes alternatives est la pompe à piston. Elle comporte une soupape d'admission et une soupape d'échappement, le pompage se fait en deux temps : aspiration puis refoulement, et le débit n'est pas régulier. Dans les pompes rotatives, le débit est régulier (pompes à engrenages, à lobes, à palettes). Cette deuxième catégorie de pompes a aussi l'avantage de ne comporter, en général, ni soupape ni clapet.

I.2.4.4.2. La pompe centrifuge

La pompe centrifuge est conçue pour une hauteur manométrique totale relativement fixe. Le débit de cette pompe varie en proportion de la vitesse de rotation du moteur.

Son couple augmente très rapidement en fonction de cette vitesse et la hauteur de refoulement est fonction du carré de la vitesse du moteur. La vitesse de rotation du moteur devra donc être très rapide pour assurer un bon débit. La puissance consommée, proportionnelle à Q * HMT, variera donc dans le rapport du cube de la vitesse. On utilisera habituellement les pompes centrifuges pour les gros débits et les profondeurs moyennes ou faibles (10 à 100 mètres).

I.3.Conclusion

Dans ce chapitre On a présenté également les principales applications des systèmes photovoltaïques à savoir : les systèmes autonomes, les systèmes connectés au réseau et enfin les systèmes de pompage photovoltaïque.

Notre intérêt c'est porté sur les systèmes de pompage Une revue bibliographique sur les types des moteurs et de pompes est également reportée dans ce chapitre pour comprendre leurs fonctionnements dans un système de pompage photovoltaïque

Chapitre 2 :

Techniques de la régulation de vitesse d'un moteur à courant continu

II.1. Introduction :

Le but de l'asservissement est d'être capable de contrôler avec précision un moteur, dont la vitesse de rotation est naturellement imprécise et instable. Par exemple dans le cas d'un robot différentiel, si l'on veut faire en sortie que le robot puisse aller droit, il faut pouvoir commander les deux roues exactement à la même vitesse. Or comme un moteur à courant continu a une vitesse qui varie facilement, il faut d'une part être capable de la mesurer et, d'autre part, pouvoir faire varier la tension du moteur pour qu'il tourne correctement.

Dans ce chapitre, on va introduire les principes de base sur les systèmes asservis ainsi que la régulation en boucle ouverte et en boucle fermée.

II.2. Définition de la régulation/asservissement

Lorsqu'il y a un retour d'information de la grandeur observée sur le régulateur, on parle d'un asservissement du système ou d'une régulation du système.

II.2.1. L'asservissement :

La consigne, traduisant l'objectif désiré du procédé, n'est pas constantes et les grandeurs perturbatrices n'existent pas ou sont très peu influentes sur la grandeur à maîtriser.

Exemples :

- Asservissement de température : obtention d'un profil de température en fonction du temps dans un four de traitement thermique.
- Asservissement d'un débit d'air par rapport à un débit de gaz afin d'obtenir une combustion idéale.

II.2.2. La régulation :

La consigne, traduisant l'objectif désiré du procédé, est constante et les grandeurs perturbatrices influencent fortement sur la grandeur à maîtriser.

Exemples :

- Régulation de température dans un local subissant les variations climatiques.
- Régulation de niveau dans un réservoir dépendant de plusieurs débits d'alimentation et de soutirage

- Régulation de pH de rejets d'eau destinés à être déversés dans une rivière.

II.3. Principe général de la régulation :

Dans la plupart des appareils dans des installations industrielles et domestiques, il est nécessaire de maintenir des grandeurs physiques à des valeurs déterminées, en dépit des variations externes ou internes influant sur ces grandeurs. Par exemple, le niveau d'eau dans un réservoir, la température d'une étuve, la vitesse et la position des moteurs, étant par nature variables, doivent donc être réglés par des actions convenables sur le processus considéré.

Si les perturbations influant sur la grandeur à contrôler sont lentes ou négligeables, un simple réglage dit en boucle ouverte, permet d'obtenir et de maintenir la valeur demandée (par exemple : action sur un robinet d'eau). Dans la majorité des cas, cependant, ce type de réglage n'est pas suffisant, parce que trop grossier ou instable. Il faut alors comparer, en permanence, la valeur mesurée de la grandeur réglée à celle que l'on souhaite obtenir et agir en conséquence sur la grandeur d'action, dite grandeur réglant.

On a, dans ce cas, constitué une boucle de régulation et plus généralement une boucle d'asservissement.

Un système est dit en boucle ouverte lorsque la Cette boucle nécessite la mise en œuvre d'un ensemble de moyens de mesure, de traitement de signal ou de calcul, d'amplification et de commande d'actionneur, constituant une chaine de régulation ou d'asservissement. La consigne est maintenue constante se produit sur le procédé une modification d'une des entrées perturbatrices. L'aspect régulation est considéré comme le plus important dans le milieu industriel, car les valeurs des consignes sont souvent fixes.

Néanmoins, pour tester les performances et la qualité d'une boucle de régulation, on s'intéresse à l'aspect asservissement. [11]

II.4. Système en boucle ouverte :

La commande est élaborée sans l'aide de la connaissance des grandeurs de sortie. Au niveau des inconvénients, il n'y a aucun moyen de contrôler, à plus forte raison de compenser les erreurs, les dérives, les accidents qui peuvent intervenir à l'intérieur de la boucle, autrement dit, il n'y a pas de précision ni surtout de fidélité qui dépendent de la qualité intrinsèque des composants. Enfin, le système en boucle ouverte ne compense pas les signaux de perturbation, le schéma bloc de système en boucle ouverte est donné par la figure ci-après.

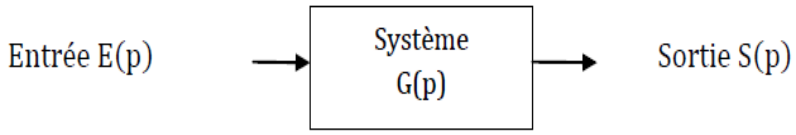


Figure II.1 : Schéma bloc d'un système en boucle ouverte

II.5. Système en boucle fermée :

La boucle fermée (contre réaction) est capable de stabiliser un système instable en boucle ouverte. Dans une régulation en boucle fermée, une bonne partie des facteurs perturbateurs externes sont automatiquement compensés par la contre-réaction à travers le procédé.

L'utilisation du retour d'information est le principe fondamental en automatique.

La commande appliquée au système est élaborée en fonction de la consigne et de la sortie.

La Figure. II.2 représente le principe de retour unitaire :

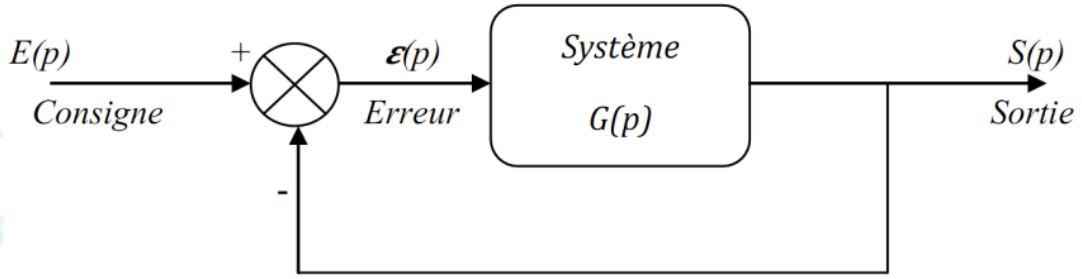


Figure II.2 : Schéma bloc d'un système en boucle fermée avec un retour unitaire

E : grandeur réglant (consigne)

S : grandeur réglée

E : erreur=E(p)–S(P)

II.6. Correction des systèmes asservis

La plupart des processus ont besoin de correcteurs pour compenser et d'améliorer la précision et la stabilité. Un correcteur est un système qui va élaborer la commande d'un système en fonction de l'erreur mesurée entre la sortie et la consigne. Si on prend le cas d'un correcteur proportionnel est un système qui donne une commande proportionnelle à l'erreur mesurée.

Beaucoup de systèmes peuvent être commandés par ce type de correcteur, qui est simple à mettre en œuvre. Le principe c'est l'ajustement du gain qui va consister à obtenir un bon compromis entre la stabilité et la précision

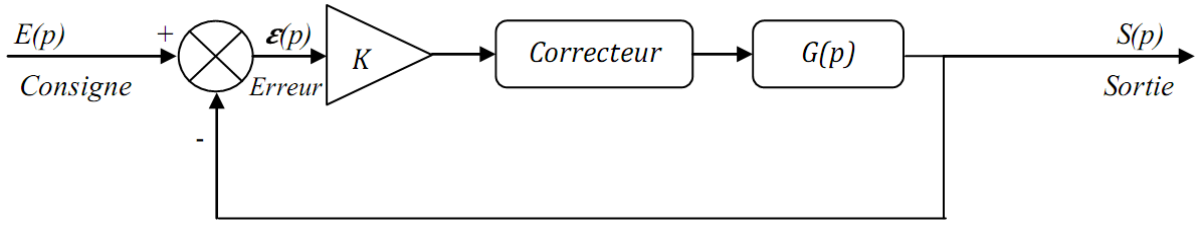


Figure. II.3 : Schéma bloc d'un système avec correcteur (Retour unitaire)

Ce genre de correcteur n'est pas toujours possible ou suffisant. Des correcteurs plus sophistiqués peuvent permettre de :

- Stabiliser un système instable.
- Augmenter le degré de la stabilité sans réduire le gain K.
- Réduire ou annuler les erreurs statiques sans toucher à la stabilité.

II.7. Le correcteur PI (proportionnel / intégrateur) :

La commande PI est dite aussi (correcteur, régulateur), se compose de deux termes P et I, d'où le 'P' correspond au terme proportionnel et 'I' pour terme intégral de la commande. Les régulateurs PI sont probablement les plus largement utilisés dans le contrôle industriel.

Le régulateur PI est une simple implémentation de retour d'information. Il a la capacité d'éliminer la compensation de l'état d'équilibre grâce à l'action intégrale.

Ce chapitre a pour but, d'implémenter la commande PI classique pour un moteur à courant continu, pour un seul objectif est d'annuler l'erreur statique, diminuer le dépassement, diminuer le temps de réponse et le temps de monté afin d'obtenir une réponse adéquate du procédé et de la régulation et d'avoir un système précis, rapide, stable et robuste.

On synthétise un correcteur appartenant à une classe de correcteurs bien connue que l'on appelle Proportionnel- Intégrateur (PI).

Ces correcteurs sont disponibles dans le commerce. Leur structure est fixée et le rôle de l'ingénieur consiste à adapter les paramètres.

On effectue la synthèse d'un correcteur spécifique au problème posé. Ils ne sont pas disponibles dans le commerce et il faut les construire.

Par l'une ou l'autre des deux approches, il existe une multitude de techniques mathématiques, graphiques et expérimentales qui permettent d'aider à synthétiser un correcteur.

Le problème de la synthèse d'un correcteur se pose comme suit : étant donné le modèle d'un système à asservir (Figure II.4)

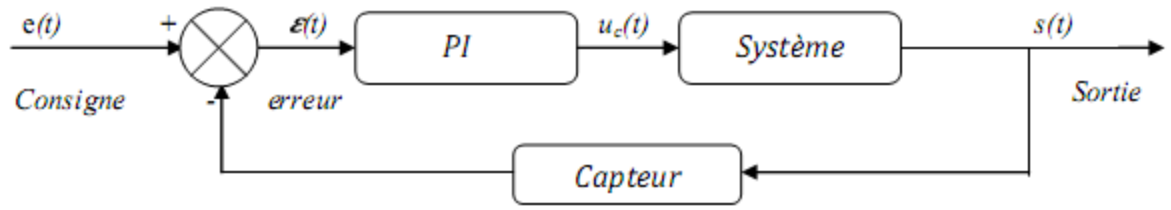


Figure. II.4 : Schéma bloc d'un système avec correcteur (Boucle fermée)

La commande proportionnel-intégral (PI) est insérée dans la chaine directe de l'asservissement, en série avec le processus. Ce régulateur élabore à partir du signal d'erreur E(t) une commande U(t) en fonction de deux actions proportionnelle et intégrale.

$$U(t) = Kp.\varepsilon(t) + \frac{1}{Ti}.\int_0^T \varepsilon(t).dt$$

$$U(t) = Kp.\varepsilon(t) + Ki.\int_0^T \varepsilon(t).dt$$

$$U(\mathrm{p}) = Kp.\varepsilon(p) + Ki.\frac{d\varepsilon(p)}{p}$$

Kp : gain d'action proportionnelle, **Ki**=1/Ti : gain d'action intégrale, **Ti** : constante de temps, dite temps d'action intégrale.

Le régulateur PI est donc conçu dans le domaine temporel comme la somme des deux actions. On obtient alors un asservissement composée d'un terme proportionnel et d'un terme intégral, mises en parallèle, on parle d'asservissement PI. [12]

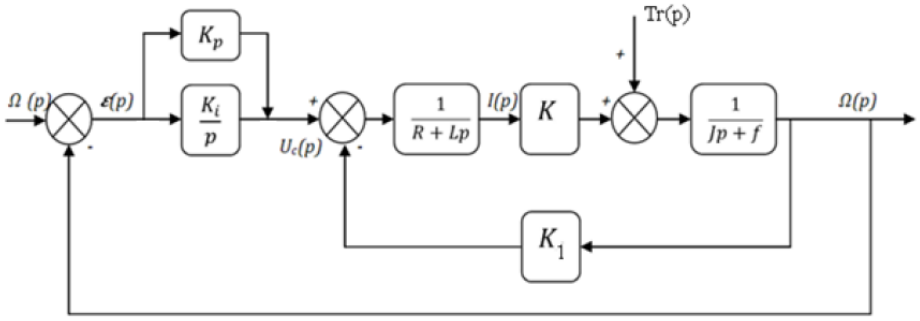


Figure II.5 : Schéma bloc du correcteur PI en régulation de vitesse.

II.8. Effets du correcteur proportionnel-Intégral :

- Diminution du temps de montée.
- Elimination de l'erreur statique.
- Augmentation du temps de stabilisation.
- Ajustement du dépassement.

II.9. Conclusion :

Dans ce chapitre nous avons présenté l'étude de la régulation d'un système asservis en générale et la régulation de la vitesse d'un moteur à courant continu en particulier

Les principes de base sur les systèmes asservis ainsi que la régulation en boucle en boucle ouvert et en boucle fermé et on étude aussi l'importance d'utiliser des correcteurs pour corriger la vitesse des moteurs à courant continu

Les principes de base sur les systèmes asservis ainsi que la régulation en boucle en boucle ouvert et en boucle fermé et on étude aussi l'importance d'utiliser des correcteurs pour corriger la vitesse des moteurs à courant continu

Chapitre 3 :

Implémentation du régulateur de vitesse sans capteur sur carte Arduino

III.1 Introduction

Arduino est une entreprise de logiciels et de matériel open source qui conçoit et fabrique microcontrôleurs monocarte et kits de microcontrôleurs pour la construction d'appareils numériques. Les produits de l'entreprise d'Arduino (cartes Arduino et logiciels) peuvent être produits et distribués par n'importe qui puisque ces produits sont concédés sous licence GNU Lesser General Public License (LGPL) ou Licence publique générale GNU (GPL).

Un programme pour le matériel Arduino peut être écrit dans n'importe quel langage de programmation avec des compilateurs qui produisent du code machine binaire pour le processeur cible.

L'environnement de développement intégré Arduino (IDE) est une application multiplateforme (pour Windows, MacOs, Linux) qui est écrit dans le langage de programmation Java.

Le 18 octobre 2019, Arduino Pro IDE (version alpha) est sorti. Le système utilise toujours Arduino CLI (interface de ligne de commande), mais les améliorations incluent une approche plus professionnelle environnement de développement, prise en charge de la saisie semi-automatique et intégration de Git.

Cependant, la programmation Arduino est censée être amusante mais peut devenir frustrante et consommer pour des tâches telles que le traçage des données des capteurs ou l'intégration de mathématiques avancées, le traitement ou contrôle les routines dans vos projets.

MATLAB® et Simulink® relèvent plusieurs défis avec la programmation Arduino traditionnelle.

Le package de support MATLAB pour Arduino permet d'écrire des programmes MATLAB qui lisent et écrivent les données vers Arduino et les périphériques connectés, tels que le blindage du moteur Adafruit, les périphériques I2C et SPI. Les principaux avantages de l'utilisation de MATLAB pour la programmation Arduino sont :

• Lire et écrire des données de capteur de manière interactive sans attendre la compilation du code.

• Analysez les données du capteur à l'aide de milliers de fonctions prédéfinies pour le traitement du signal, la machine apprentissage, modélisation mathématique, etc.

• Visualisez rapidement les données en utilisant la vaste gamme de types de tracé dans MATLAB [12]

III.2. Régulation avec Arduino

Un moteur à courant continu possède une relation directe entre sa tension d'alimentation et sa vitesse de rotation. En effet, plus la tension à ses bornes est élevée et plus son axe tournera rapidement (dans la limite de ses caractéristiques évidemment). Cependant le microcontrôleur d'Arduino n'est capable de produire que des tensions de 0 ou 5V. Nous sommes en mesure de produire à l'aide de notre microcontrôleur un signal carré dont le rapport cyclique est variable. Grâce à cela, nous obtenons une tension moyenne (comprise entre 0 et 5V) en sortie de la carte Arduino. Il faut juste bien penser à utiliser les sorties adéquates, à savoir : 3, 5, 6, 9, 10 ou 11, en utilisant la PWM, on va générer une tension par impulsions plus ou moins grandes. Ce signal va commander le transistor qui va à son tour commander le moteur. Le moteur va donc être alimenté par intermittences à cause des impulsions de la PWM. Ce qui aura pour effet de modifier la vitesse de rotation du moteur.

III.2.1. Présentation Matérielle

Un moteur à courant continu possède une relation directe entre sa tension d'alimentation et sa vitesse de rotation. En effet, plus la tension à ses bornes est élevée et plus son axe tournera rapidement (dans la limite de ses caractéristiques évidemment). Cependant le microcontrôleur d'Arduino n'est capable de produire que des tensions de 0 ou 5V. Nous sommes en mesure de produire à l'aide de notre microcontrôleur un signal carré dont le rapport cyclique est variable. Grâce à cela, nous obtenons une tension moyenne (comprise entre 0 et 5V) en sortie de la carte Arduino. Il faut juste bien penser à utiliser les sorties adéquates, à savoir : 3, 5, 6, 9, 10 ou 11, en utilisant la PWM, on va générer une tension par impulsions plus ou moins grandes. Ce signal va commander le transistor qui va à son tour commander le moteur. Le moteur va donc être alimenté par intermittences à cause des impulsions de la PWM. Ce qui aura pour effet de modifier la vitesse de rotation du moteur.

III.2.1.1. Circuit de puissance :

Pour commander un moteur avec un Arduino, il faut un dispositif capable de générer une intensité suffisante. Les broches de l'Arduino peuvent envoyer un courant de 40mA maximum ce qui est généralement trop peu pour entraîner un moteur.

L'objet de cette partie est de montrer qu'on peut aussi utiliser un type de transistor particulier, un MOSFET, qui permet de mettre en marche et de faire varier la vitesse de moteurs à courant continu.

Ici on a utilisé un MOSFET de type IRF540 ; il en existe de très nombreux types, ils ne sont pas

tous compatibles avec l'Arduino. En particulier, il faut que la valeur Vgs (Gate Threshold voltage) soit en gros autour de 2 ou 4V.

Autre paramètre important, la valeur Rds(on) que l'on souhaite basse, elle conditionne la chaleur dégagée par le composant en utilisation. Ici on a Rds=0,044 ohms, ce n'est pas mal mais on trouve mieux.

Le principe du fonctionnement de ce type de transistor est que lorsque la tension de la Gate atteint une valeur suffisante, le courant passe entre le Drain et la Source. Il existe aussi de très nombreux types de transistors utilisables qui ne sont pas des MOSFET mais ils semblent moins performants pour ce type d'application (chute de tension, chaleur dégagée).

Ces Mosfets sont de braves transistors capables de supporter des intensités étonnantes, Mais, si les intensités deviennent importantes, il faut s'équiper d'un radiateur sous peine de destruction du transistor.

Les Mosfets sont costauds ne coûtent pas cher mais sont vulnérables à l'électricité statique.

La résistance de 10 kOhms est importante, elle ramène la Pin à l'état bas quand la Pin n'est pas alimentée. (Risque sinon que la Pin soit dans un état incertain …).

La diode est de type 1N4148 (elle supporte 1000v et 1A) ; placée aux bornes du moteur, elle protège le transistor des tensions importantes qui naissent quand le courant est coupé, elle est indispensable

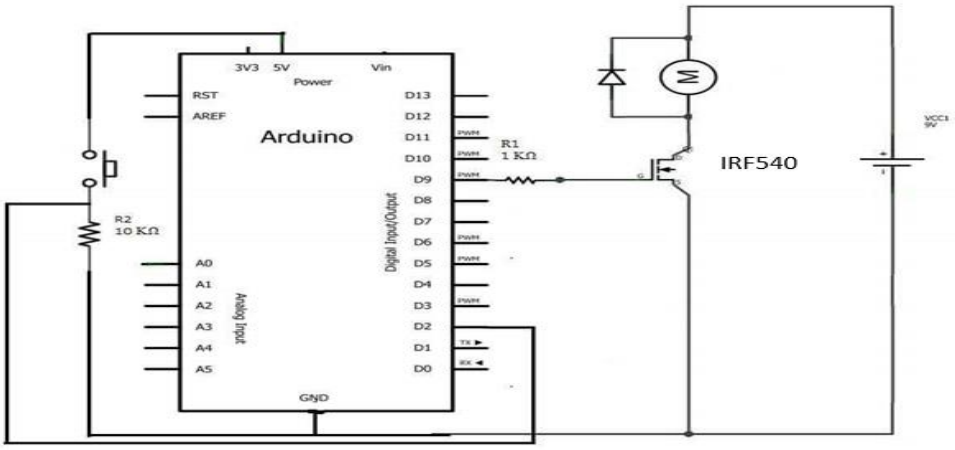


Figure III.1 : Schéma d'un circuit de puissance

III.2.1.2. Consigne analogique :

L'idée est d'introduire un potentiomètre de 100 KΩ afin de varier la vitesse du MCC. Pour que cela fonctionne on a modifié notre circuit de puissance en supprimant le bouton poussoir, la résistance

de 10 KΩ et en mettant un potentiomètre alimenté par 5 Volts depuis l'Arduino à l'entrée analogique (A0). Concernant la programmation, il a fallu trouver une instruction qui converti la tension du potentiomètre en valeur de PWM. Ceci a été possible grâce a la fonction : map() . Pendant l'expérience, le moteur a eu la réaction qu'on voulait c'est-à-dire qu'il a modifié sa vitesse a chaque fois qu'on a tourné le vice du potentiomètre

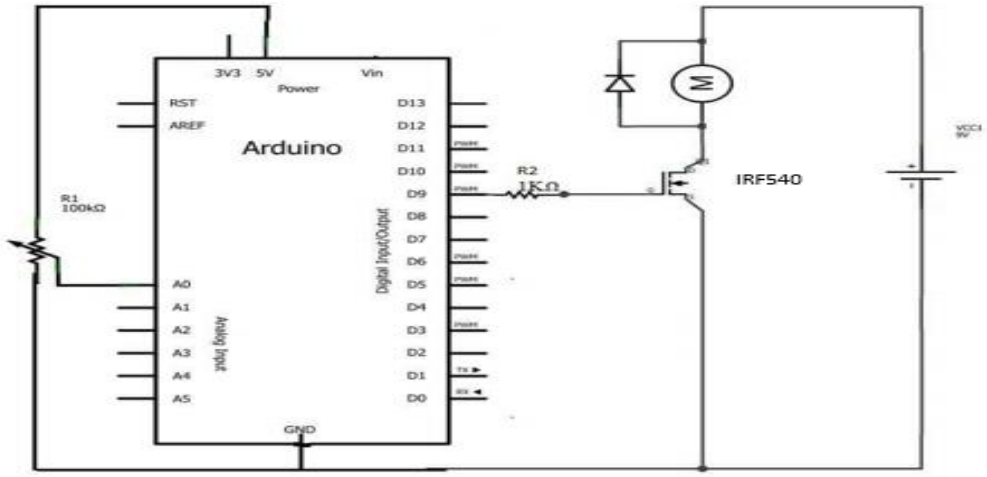


Figure III.2 : schéma d'un circuit de consigne analogique

III.2.1.3. Mesure de la vitesse :

Tous les récepteurs ont une résistance interne (la plus faible possible) qui provoque une perte par effet joule. La force contre- électromotrice (fcém) est égale à la tension aux bornes du récepteur moins la tension dû à la résistance interne du récepteur.

E = U – r * I, U = E + r * I. r : représente la résistance interne du récepteur.

Dans cette étape nous complèterons notre expérience en ajoutant une boucle de retour à l'Arduino.

On utilise la force contre électromotrice du MCC pour avoir une information sur sa vitesse (au lieu d'utiliser un capteur de vitesse), ceci est possible lorsqu'on utilise un pont à deux résistances le première : 20 KΩ et la seconde : 10 kΩ, on prenant un fil entre les deux résistance en série.

La tension obtenue évolue contrairement à la vitesse du MCC (de 4.5 V à 0 V) bien entendu elle ne dépassera pas 5 V. La figure 4.3 présente le schéma d'asservissement de vitesse d'un moteur à courant continu contrôlé par une carte Arduino.

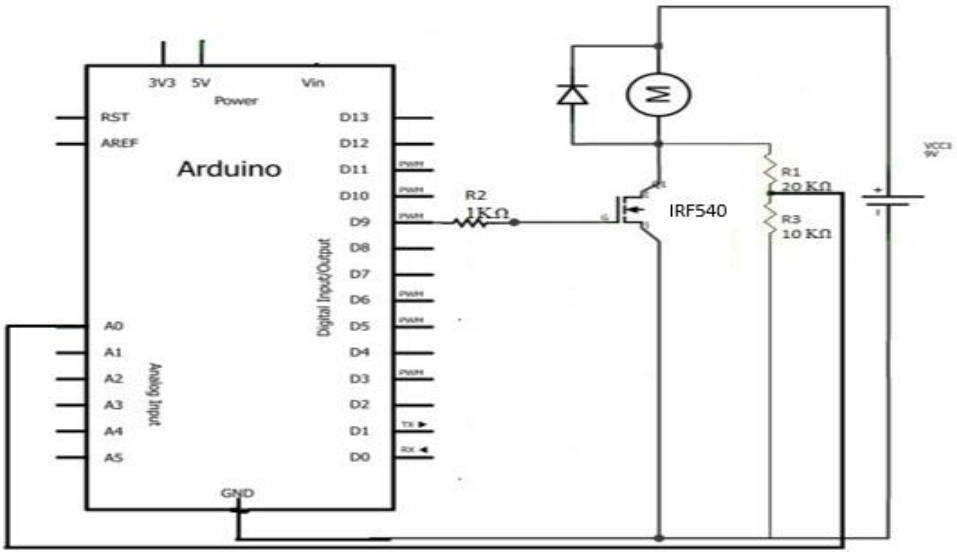


Figure III.3 : schéma d'un circuit d'asservissement de vitesse par fcem

III.2.2. Présentation logicielle :

III.2.2.1. Programme PWM avec consigne numérique :

Pour essayer de varier la vitesse d'un MCC nous allons utiliser le circuit de puissance précédent avec notre Arduino UNO, il suffit de brancher le câble qui est connecté à la base du MOSFET à la pin (9) et celui qui est connecté au bouton poussoir à la pin (2), en faisant une petite recherche on a trouvé l'instruction qui varie la vitesse du MCC, il s'agit de : analogWrite (motorPin, PWMvalue), à partir de ça on a pu composer ce programme.

L'expérience a été réalisée avec succès, le microcontrôleur a varié la vitesse du MCC comme nous lui avons indiqué sur le programme

```
const int switchPin = 2;
const int motorPin =9 ;
int switchState = 0;
void setup(){
  pinMode(motorPin, OUTPUT);
  pinMode(switchPin, INPUT);
  // PUT YOUR STEP CODES HERE, to run once:
}
void loop(){
  switchState = digitalRead(switchPin);
if (switchState==HIGH){
  analogWrite(motorPin, 20);
  delay(2000);
  analogWrite(motorPin,64);
  delay(2000);
  analogWrite(motorPin,120);
 delay(2000);
 analogWrite(motorPin,190);
   delay(2000);
    analogWrite(motorPin,255);
 delay(2000);
  analogWrite(motorPin,190);
   delay(2000);
    analogWrite(motorPin,120);
 delay(2000);
  analogWrite(motorPin,64);
   delay(2000);
    analogWrite(motorPin,20);
 delay(2000);
```

```
  analogWrite(motorPin,190);
   delay(2000);
    analogWrite(motorPin,120);
 delay(2000);
  analogWrite(motorPin,64);
   delay(2000);
    analogWrite(motorPin,20);
 delay(2000);

}
else{
  digitalWrite(motorPin, LOW);
}
}
```

III.2.2.2. Programme PWM avec consigne analogique :

```
double val;
void setup () {
pinMode(9, OUTPUT);
pinMode(A0, INPUT);
}
void loop(){
  int val= analogRead(0);
  val =map (val,0,1023,0,255);
  analogWrite(9,val);
}
```

III.2.2.3. Programme avec correction P :

Le programme est le suivant :

Soit (w1) la vitesse idéale du MCC et (W) la vitesse après la correction.

Pour tester notre programme il a fallu introduire dedans une fausse consigne sous la forme d'une instruction analogWrite(), cette dernière est supérieure ou inférieure à (w1), donc le

microcontrôleur est supposer faire l'algorithme suivant :

{Si (val ≠ w1) →U = kp(w1 - val)→W = w1 + U→analogWrite(motorPin, W)→tant que (W = w1)→analogWrite(motorPin, w1) Si (val = w1)→analogWrite(motorPin, w1)}.

Après le téléchargement de ce programme sur l'Arduino nous avons vu que le moteur a fonctionner exactement comme nous l'avons souhaité en changeant systématiquement la vitesse ver (w1) quel que soit la perturbation exercer sur lui.

```
double w1 = 125;
double W;
double kp;
double ki;
double pas= 100;
double de;
double U ;
double Ui;
double Upi;

void setup() {
  // put your setup code here, to run once:
pinMode (9, OUTPUT);
pinMode (A0, INPUT);
Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
delay(3000);
int val = analogRead(A0);
val= map(val,580,0,0,255);
Serial.print("\t val :");
Serial.print(val);
if (val<w1){
kp=1.5;
ki=0.01;
de = w1 - val;
Serial.print("\t de :");
Serial.print(de);
U= kp * de;
Ui=Ui+(ki*de);
Serial.print("\t U :");
Serial.print(U);
W= w1 + U + Ui;
Serial.print("\t W :");
Serial.print(W);
analogWrite(9,W);
}
if(val>w1){
  kp=0.3;
  ki=0.01;
  de=w1-val;
  Serial.print("\t de :");
  Serial.print(de);
  U=kp * de;
  Ui=Ui+ (ki * de);
  Serial.print("\t U :");
  Serial.print(U);
  W= w1 + U +Ui;
  Serial.print("\t W :");
  Serial.print(W);
  analogWrite(9,W);
}
}
```

III.2.2.4. Programme avec correction PI :

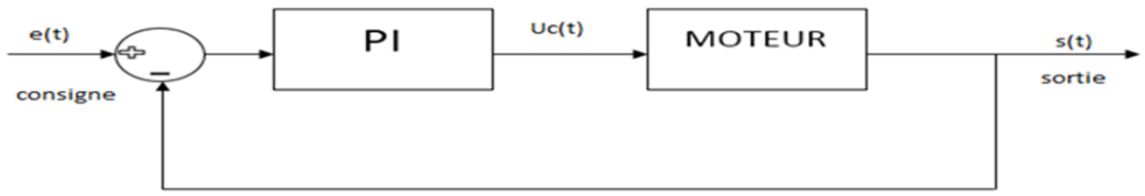


Figure III.4 : schéma fonctionnel avec les actions P et I en chaine directe

La commande proportionnel-intégral (PI) est insérée dans la chaine directe de l'asservissement, en série avec le processus, comme indiqué dans la (Figure 4.4). Ce régulateur élabore à partir du signal d'erreur E(t) une commande U(t) en fonction de deux actions ; proportionnelle et intégrale. Le programme sur Arduino et le suivant :

{Si (val ≠ w1) →U = kp(w1 - val) ,Si (val ≠ w1) →Ui = Ui+ki(w1 - val)→W = w1 + U+Ui→analogWrite(motorPin, W)→tant que (W = w1)→analogWrite(motorPin, w1) Si (val = w1)→analogWrite(motorPin, w1)}.

```
double w1=125;
double W;
double kp;
double de;
double U ;


void setup() {
  // put your setup code here, to run once:
pinMode (9, OUTPUT);
pinMode (A0, INPUT);
Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
     analogWrite(9,40);
delay(3000);
int val=analogRead (A0);
val=map(val,580,0,0,255);
Serial.print("\t val :" );
Serial.print(val);
analogWrite(9,val);
if (val<w1);
kp=1.5;
de=w1-val;
Serial.print("\t de :" );
Serial.print(de );
Serial.print("\t W :" );
Serial.print(W);
analogWrite(9,W);
while (W=w1){
  analogWrite(9,w1);
}
 if (val > w1){
  kp=0.3;
  de=w1-val;
  Serial.print("\t de:" );
  Serial.print(de);
  U=kp*de;
  Serial.print("\t U :" );
  Serial.print(U );
  W=w1+U;
  Serial.print("\t W :" );
  Serial.print(W );
  analogWrite(9,W);
  while (W=w1);
  {analogWrite (9,w1);
  }
  }
  else{
    analogWrite(9,w1);
  }


}
```

III.2.2.5. Programme avec correction PI combinée (action P sur la sortie) :

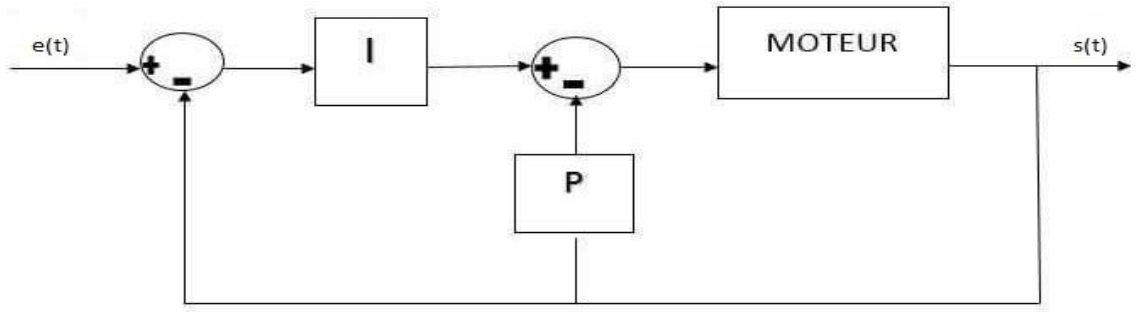


Figure III.5: schéma fonctionnel avec l'action P en sortie et l'action I en chaine directe

Lors d'un changement de consigne, le signal d'erreur entre la consigne et la sortie est très grand. La commande PI sur l'écart va engendrer une commande proportionnelle (rapide) et une commande intégrale (lente et précise) à la variation de l'erreur. L'amplitude de cette commande risque d'être inadmissible en pratique. Une solution pour limiter ce phénomène est d'appliquer l'action intégrale sur l'erreur et l'action proportionnelle seulement sur la sortie du procédé. Le programme devient le suivant :

{Si (val ≠ w1) →U = kp*val ,Si (val ≠ w1) →Ui = Ui+ki(w1 - val)→
W = w1 - U+Ui→analogWrite(motorPin, W)→tant que (W = w1)→analogWrite(motorPin, w1)
Si (val w1)→analogWrite(motorPin, w1)}.

```
double w1 = 125;
double W;
double kp;
double ki;
double pas= 100 ;   //milisecend
double deK;
double deI;
double U;
double Ui;
double Upi;

void setup() {
  pinMode (8, OUTPUT);
  pinMode (A0, INPUT);
  Serial.begin(9600);
  //put your setup code here, to run once:
}
void loop() {
delay(500);
  int val = analogRead (A0);
  val = map(val, 580, 0, 0, 255);
  Serial.print( "\t val : ");
  Serial.print(val);
 if (val < w1){
  kp = 1.5;
  ki=0.01;
  deI = w1 - val;
   Ui = Ui+ (ki * deI);
   U = kp * val;
  ki=0.01;
  deI = w1 - val;
   Ui = Ui+ (ki * deI);
   U = kp * val;
  Serial.print( "\t de : ");
  Serial.print(deI);
W = Ui - U ;
  Serial.print( "\t U : ");
  Serial.print(U);
 Serial.print( "\t W : ");
  Serial.print (W);
  analogWrite (8, W);
  }
if (val > w1){
  kp = 0.3;
    ki=0.01;
deI = w1 - val;
   Ui = Ui+ (ki * deI);
   U = kp * val;
  Serial.print( "\t de : ");
  Serial.print(deI);
W = Ui - U ;
  Serial.print( "\t U : ");
  Serial.print(U);
  Serial.print( "\t W : ");
  Serial.print (W);
  analogWrite (8, W);
}
}
```

III.3. Régulation avec Matlab ARDUINO (Simulink) :

III.3.1. L'environnement Matlab/Simulink :

C'est un logiciel de calcul mathématique pour les ingénieurs et les scientifiques créé par Mathworks.

III.3.1.1. MATLAB :

Est un environnement de programmation pour le développement d'algorithme, d'analyse de données, de visualisation, et de calcul numérique. En utilisant MATLAB pour la résolution des problèmes de calcul complexes se fait plus rapidement qu'avec des langages de programmation traditionnels, tels que C, C++, et le Fortran.

III.3.1.2. SIMULINK :

Est un environnement pour la simulation multi domaine. Il fournit un environnement graphique interactif et un ensemble de bibliothèques de bloc qui permettent de concevoir, simuler, mettre en

application, et examiner une variété de systèmes, tel que les systèmes de communications, de commandes, de traitement des signaux, de traitement visuel, et de traitement d'image.

III.3.2. MATLAB Support package for arduino Hardware:

Vous permettez d'utiliser MATLAB pour communiquer avec une carte Arduino, vous pouvez lire et écrire des données de capteur via l'arduino et voir immédiatement les résultats dans MATLAB sans avoir à compiler. Ce package de support est fonctionnel pour R2014a et au-delà [13]

III.3.2.1. Configuration du package de support matériel pour MATLAB :

Étape 1. Démarrez MATLAB (dernière version préférée).

Étape 2. Dans la section Environnement, sélectionnez Add-Ons > Get Hardware Support Packages

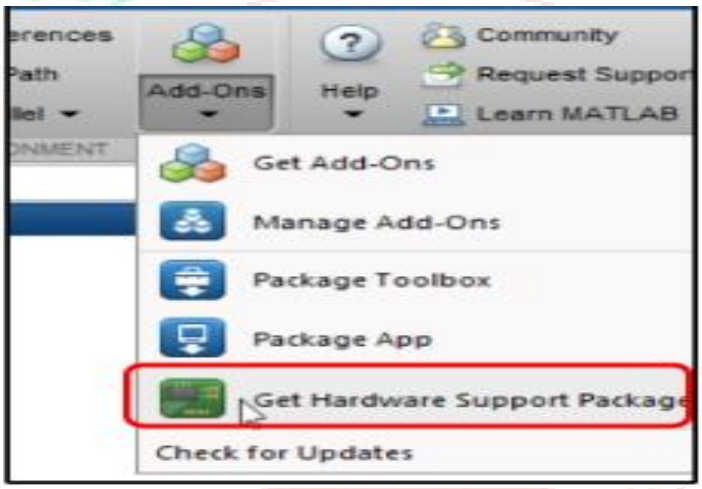


Figure III.6 : sélectionner de Add-Ons/Get hardware support package

Étape 3. Il ouvrira la fenêtre de l'explorateur de Add-On

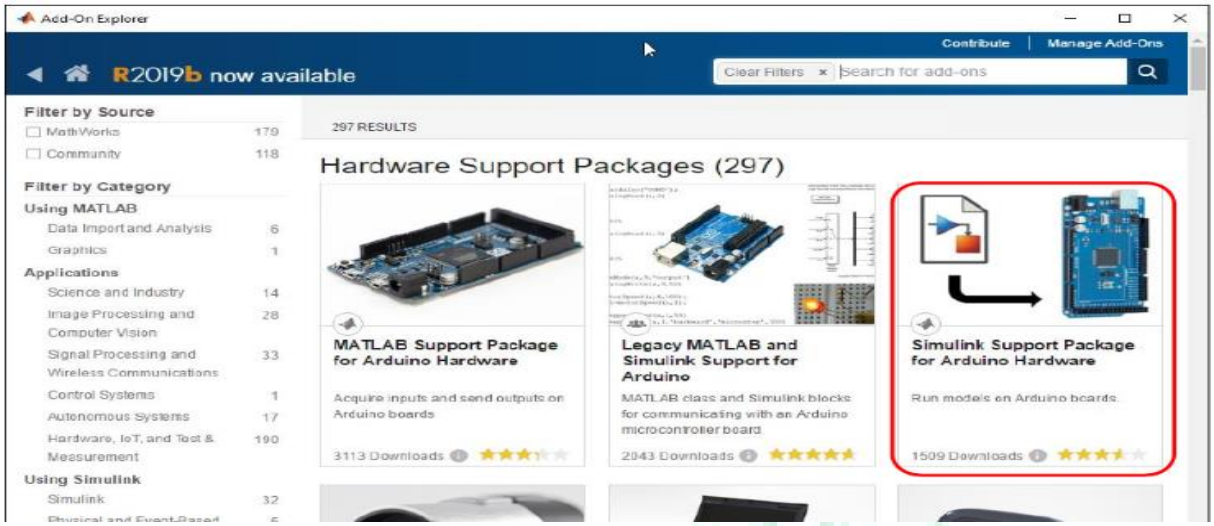


Figure III.7 : la fenêtre de Add-Ons

Étape 4. Cliquez sur MATLAB Support Package for Arduino Hardware.

Étape 5. Cliquez sur Installer

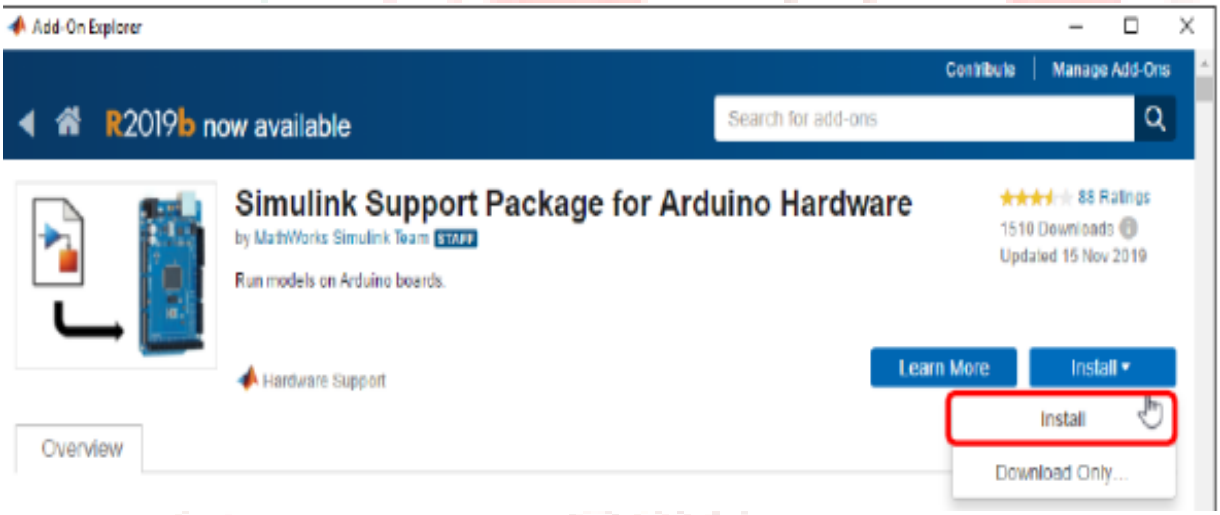


Figure III.8 : Installation de SIMULINK SUPPORT PACKAGE FOR ARDUINO HARDWARE

Maintenant le programme d'installation vous demandera de vous connecter à votre compte MathWorks. Si vous ne possédez pas de compte MathWorks, vous pouvez créer un compte lors de l'installation

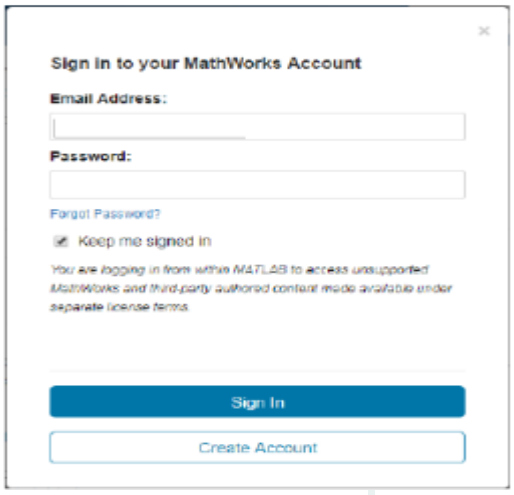


Figure III.9 : connexion sur le compte MATLAB

Étape 6. Après vous être connecté, acceptez le contrat de licence et procédez à l'installation.

Étape 7. Maintenant, attendez que le package soit téléchargé et installé.

Étape 8. Vous avez maintenant installé avec succès Arduino Support Package pour MATLAB.

III.3.2.2. Test de MATLAB :

Après avoir installé MATLAB Support Package for Arduino, nous devons vérifier s'il est installé correctement ou non.

1. Ouvrez MATLAB.
2. Connectez Arduino au PC.

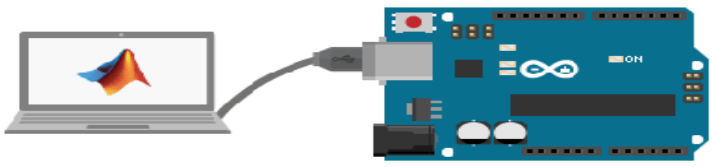


Figure III.10 : connexion d'Arduino au PC

3. Tapez la commande suivante dans la fenêtre de commande MATLAB.

a = arduino ()

4. Si nous avons plus d'un Arduino connecté à un PC, nous pouvons spécifier le type de carte et le port COM auquel il est connecté en utilisant la commande suivante.

Interface Arduino avec MATLAB - Schémas LED clignotants

a = arduino («COM3», «uno»)

5. Après avoir entré la commande ci-dessus, MATLAB essaiera de communiquer avec votre Arduino, en cas de succès, MATLAB affichera les propriétés de la carte Arduino connectée au PC.

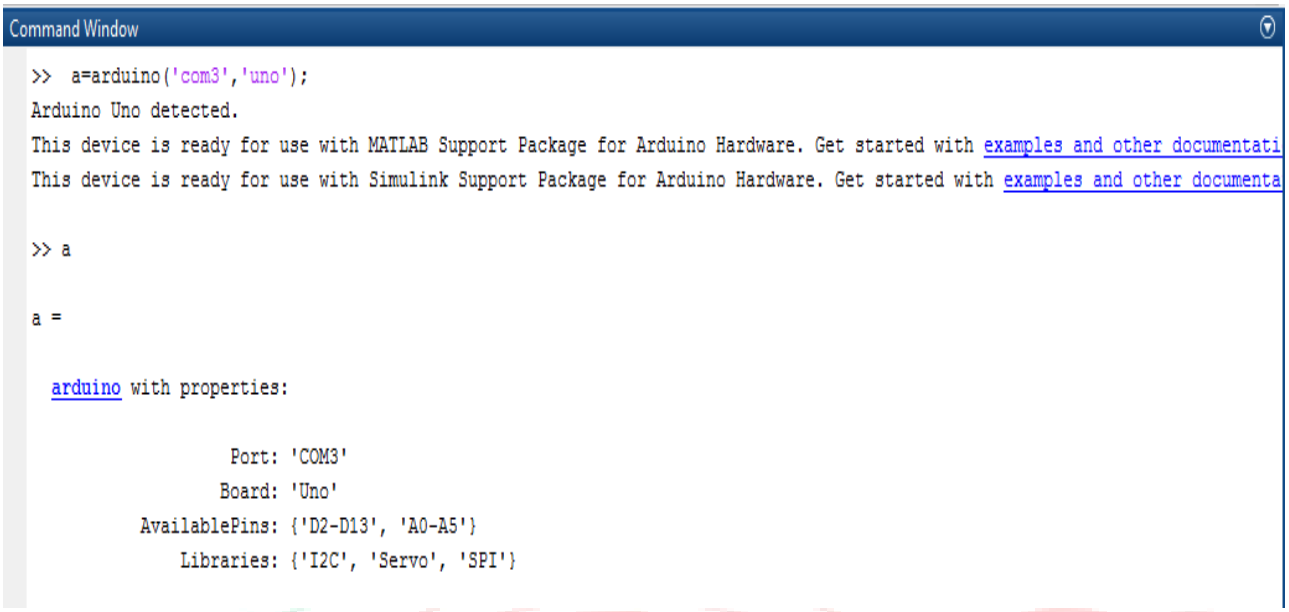


Figure III.11 : les propriétés de la carte Arduino connectée au PC au MATLAB

III.3.2.3. Examiner la bibliothèque de blocs Arduino

Le package de support Simulink pour le matériel Arduino fournit un moyen facile de créer des algorithmes qui utilisent des capteurs et des actionneurs Arduino en utilisant les blocs qui peuvent être ajoutés à votre modèle Simulink. Les blocs sont utilisés pour configurer les capteurs et actionneurs associés, ainsi que pour leur lire et écrire des données.

1. Entrez lLibraryBrowser à l'invite MATLAB. Cela ouvre le navigateur de bibliothèque Simulink.
2. Dans le navigateur de bibliothèque Simulink, accédez à Package de support Simulink pour le matériel Arduino> Commun.

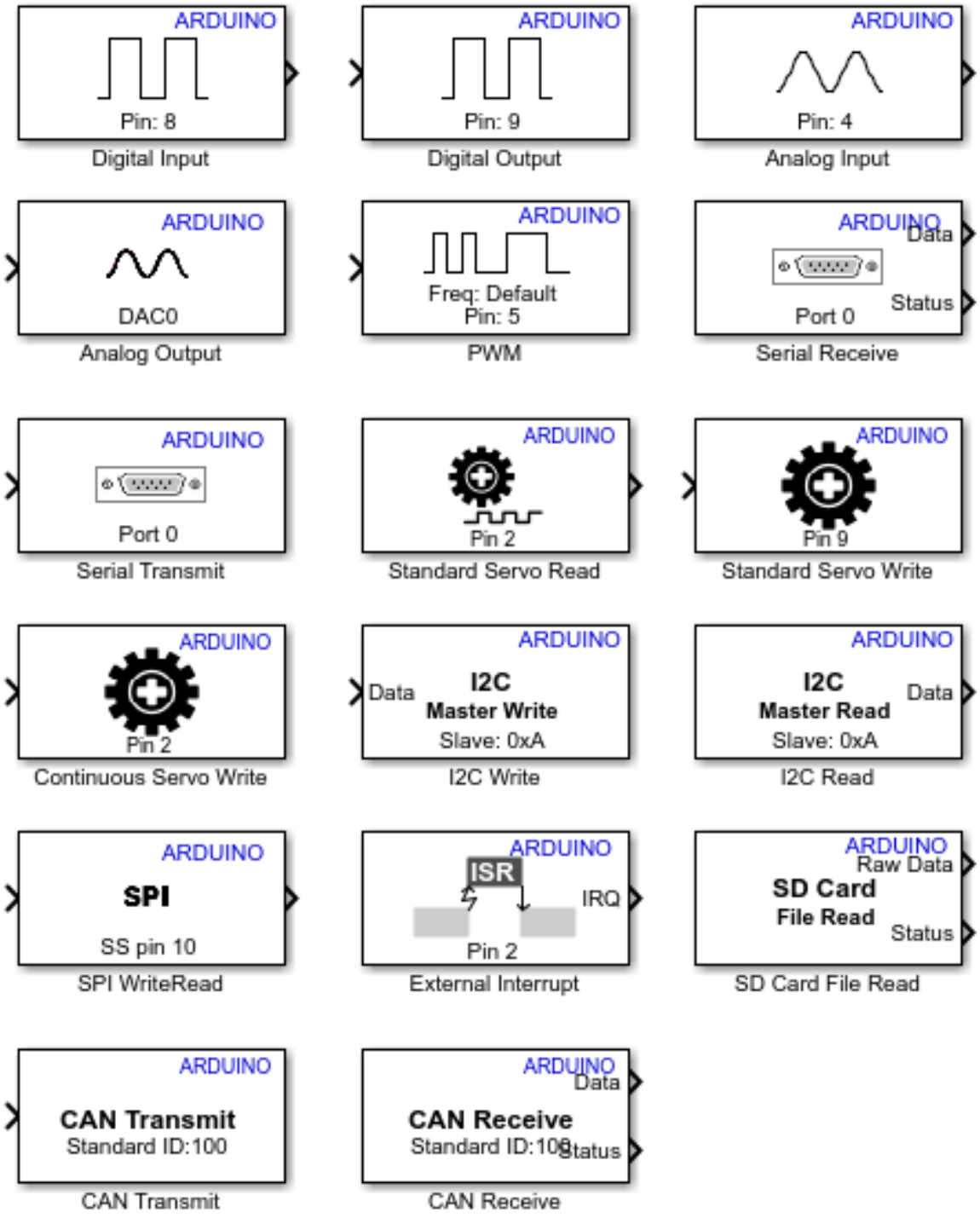
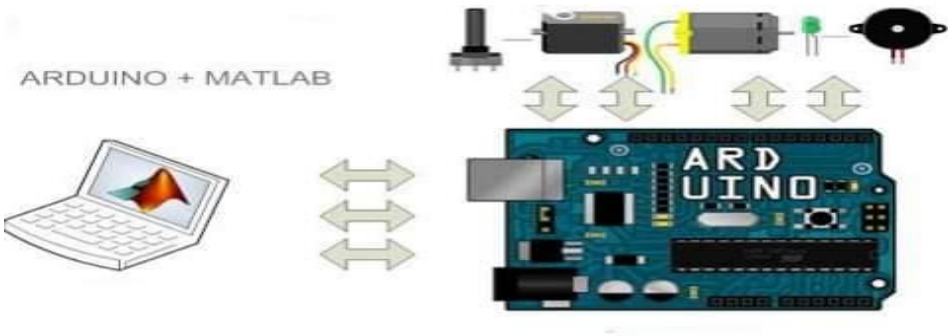


Figure III. 12 : Commun bloc d'arduino

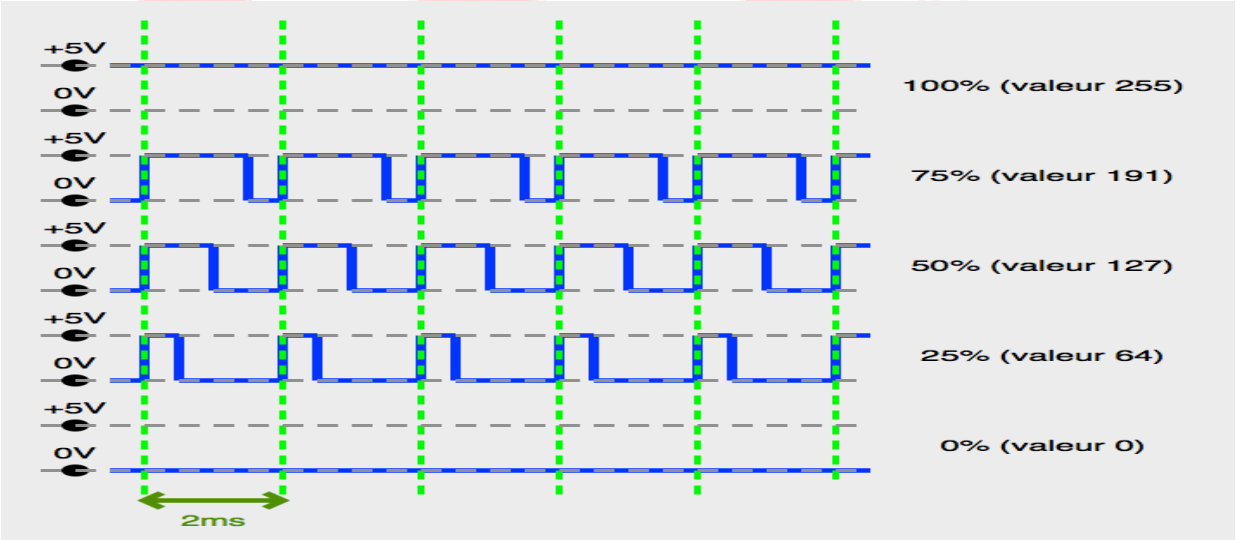
III.3.3. Fonctionnement et programmation :



FigureIII.13 : montre l'interfaçage entre la partie programmation et partie matérielle

III.3.3.1. Consigne numérique (signal PWM) :

Comme cité dans le chapitre précédent les pines (3, 5, 6, 9, 10,11) sont capables, tout comme les autres, d'envoyer soit un +5 V, soit du 0 V en mode OUTPUT. Mais ils ont un autre avantage, ils peuvent envoyer un train d'impulsions variable. Nous les commandons avec une fonction de Simulink qui va prendre une valeur entre 0 et 255.C'est cette valeur qui va définir la part de temps pendant lequel le pin sera à l'état HAUT durant chaque intervalle, (255 correspondra à 100% du temps d'intervalle à l'état HAUT). Le temps de cycle du début d'une impulsion à un autre est de 1 ms, nous appelons ce temps la période du cycle des impulsions. Toutes les 1 ms, le pin va rester à l'état HAUT pendant un pourcentage du temps et à l'état BAS pour le reste des 1 ms.



La figure III.14 :la variation d'un train d'impulsion

Cycles d'impulsions : c'est le pourcentage du temps où le pin est à l'état HAUT (sur 1 ms), et la valeur du paramètre associé.

Nous pourrions imaginer un moteur saccadé (il s'arrête, il repart, il s'arrête, il repart...) mais en fait pas du tout.

Rappelons-nous que le moteur a une inertie mécanique (il ne s'arrête pas d'un coup mais continue de tourner même sans électricité) donc le résultat est un peu comme si on le relançait par petits coups.

Le cycle des impulsions est sur 1 ms, donc on le relance plus ou moins sur ce temps très rapide.

Nous qualifions le **rapport cyclique** : le temps à l'état haut (Th) divisé par la période du cycle (T) ce qui donne : $R = \frac{Th}{T}$

Ce rapport est situé entre 0 et 1 **[14]**

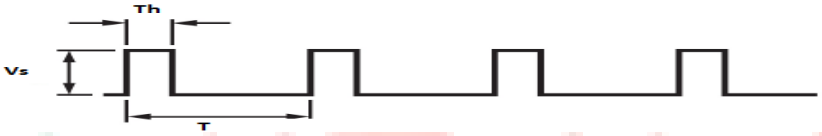


Figure III.15 : une description d'un rapport cyclique**.**

La figure III.16 suivante nous montre un schéma Simulink qui nous permet de contrôler la vitesse d'un moteur de 12v avec ce type de commande (PWM).

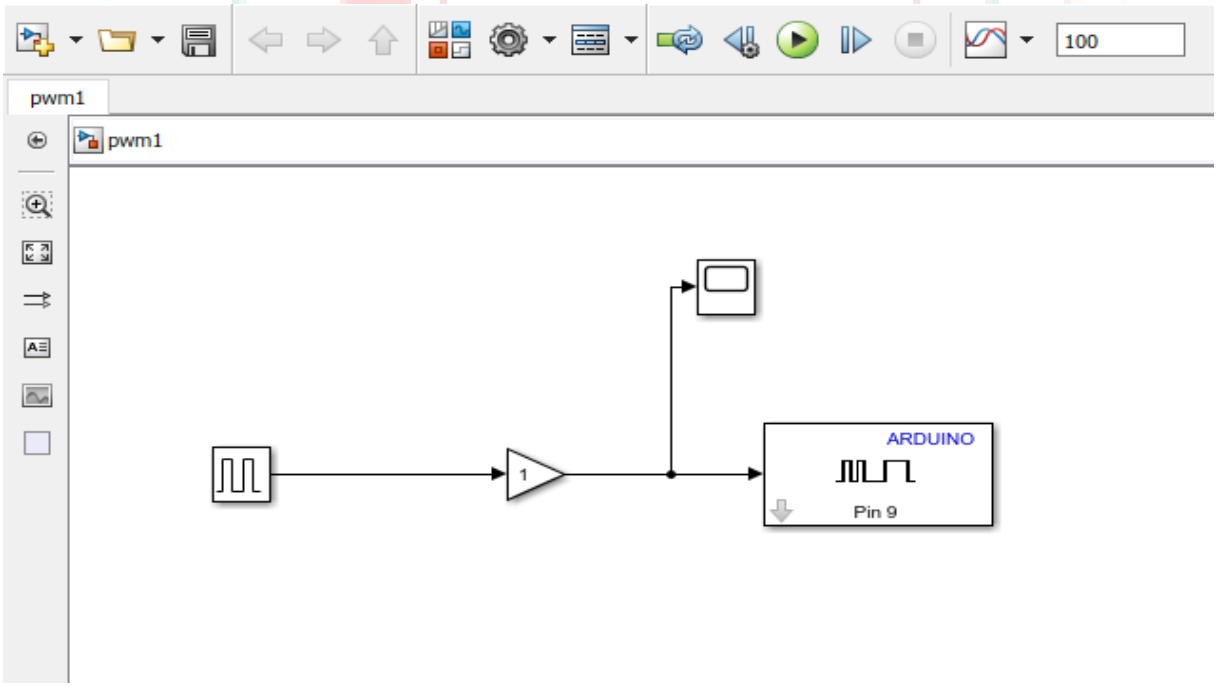


Figure III.16 : Schéma sur Simulink (consigne numérique)

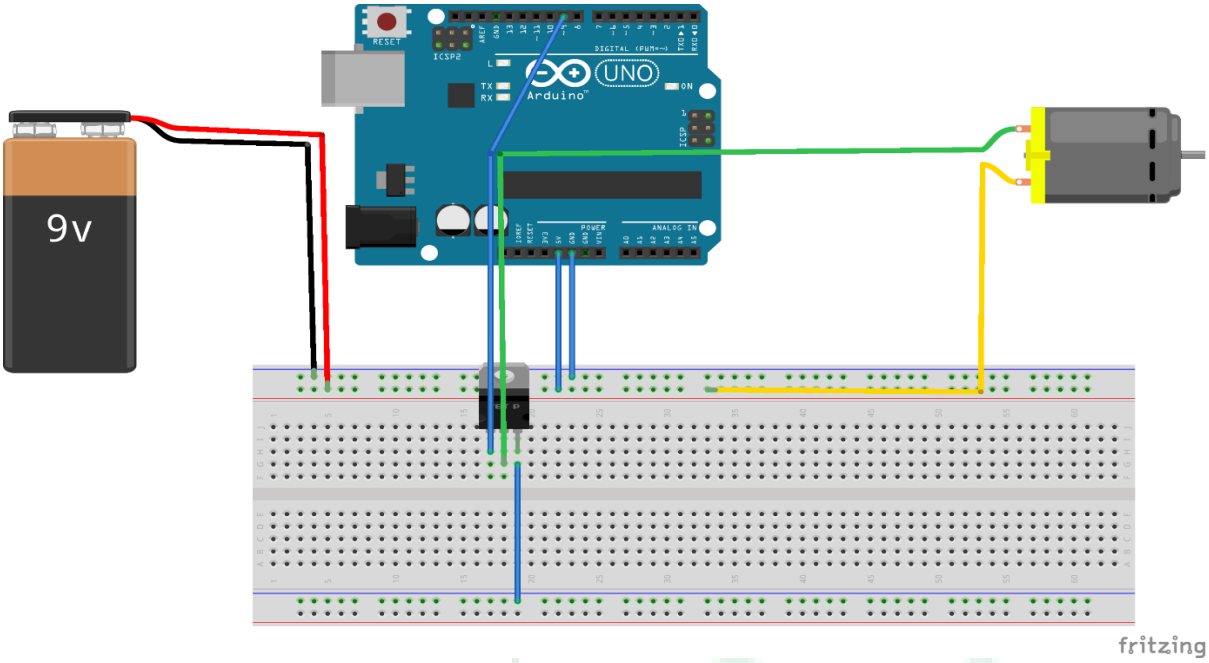


Figure III.17 : Schéma électrique avec consigne numérique

III.3.3.2. Consigne analogique (potentiomètre) :

Nous voulons montrer comment varies la vitesse de moteur à l'aide du support d'entrée analogique de la prise en prise en charge Simulink pour le matériel arduino. Pour notre signal analogique, nous utiliserons simplement un potentiomètre de 10 kOhm la sortie centrale du potentiomètre est connecté à la broche A0 de la carte arduino

Si nous modifions la valeur de potentiomètre, la vitesse du moteur changera

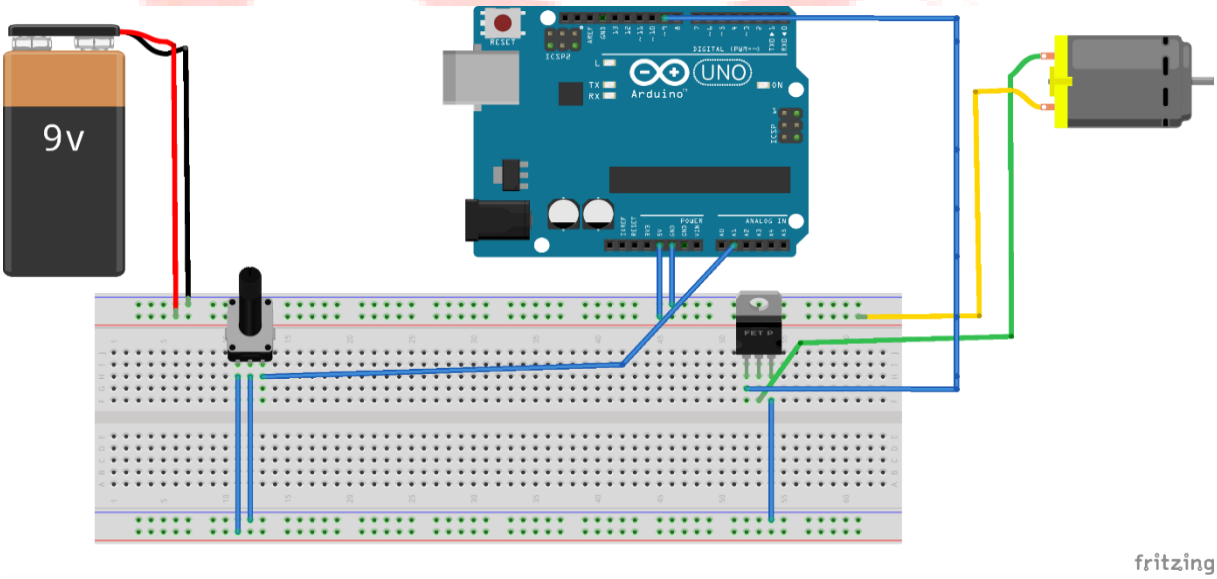


Figure III.18 : Schéma électrique de consigne analogique

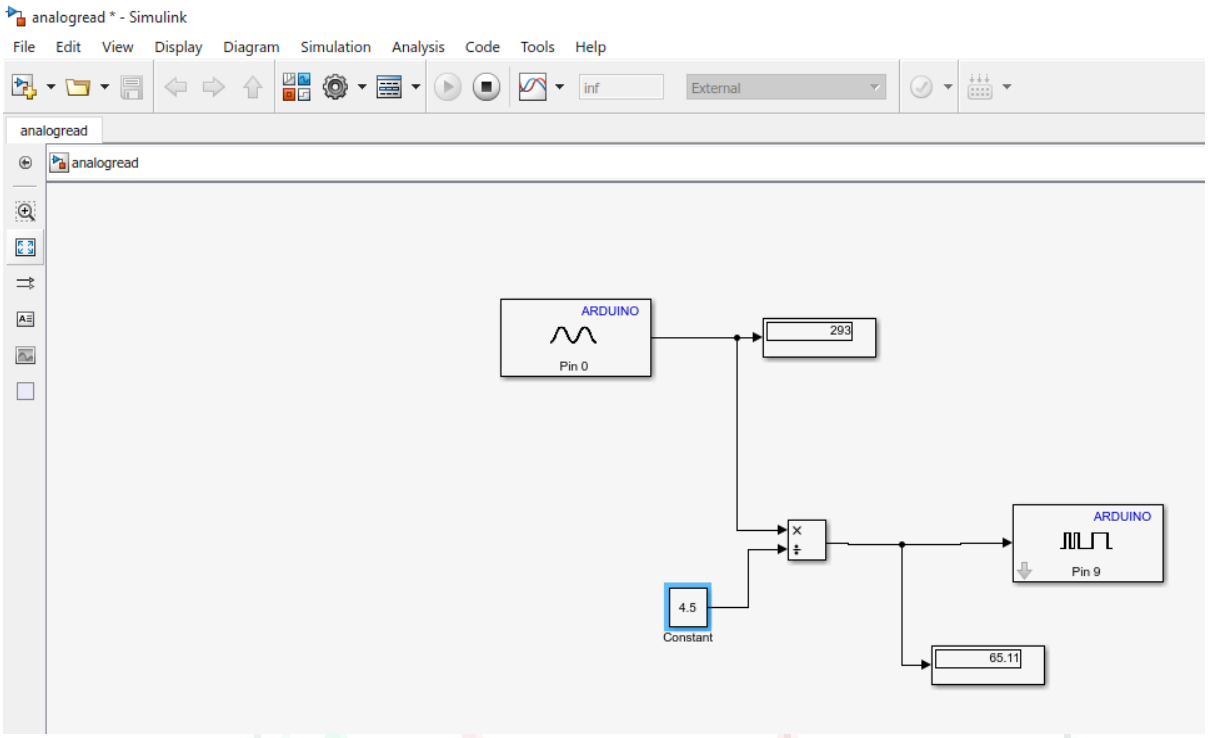


Figure III.19 : schéma en Simulink (consigne analogique)

-Après les modifications sur le potentiomètre

-Au minimum : Lorsque le potentiomètre est à la valeur la plus basse le moteur tourne à plein régime

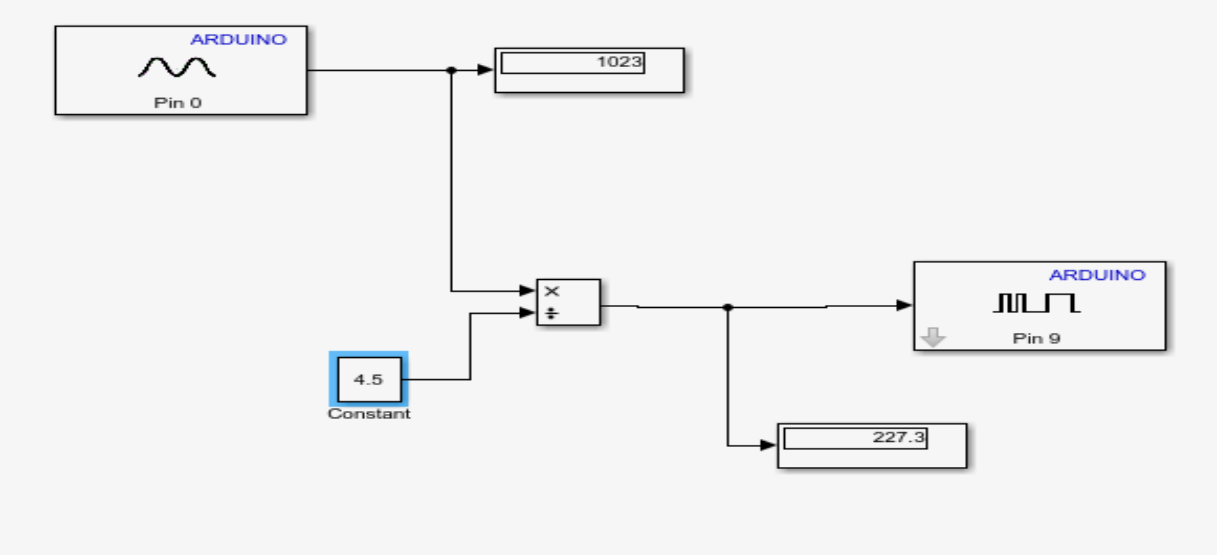


Figure III.20 : Schéma en Simulink (valeur min)

-Au maximum : Lorsque le potentiomètre est à son valeur maximum le moteur est à bas régime

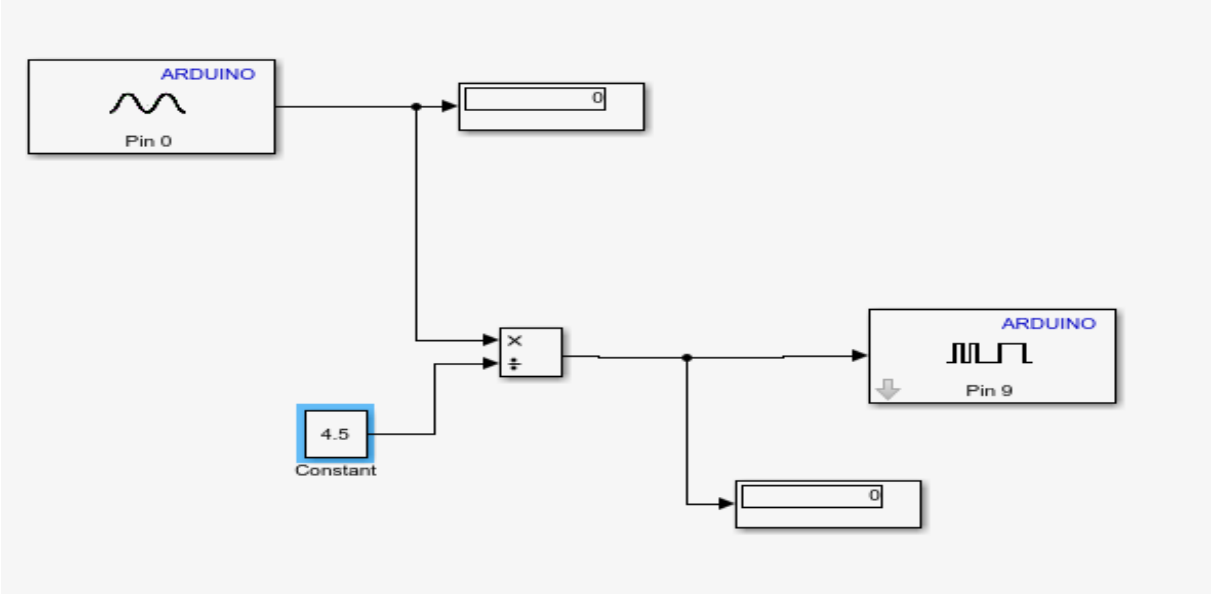


Figure III. 21 : Schéma en Simulink (valeur max)

III.3.3.3. Les correcteurs p, pi :

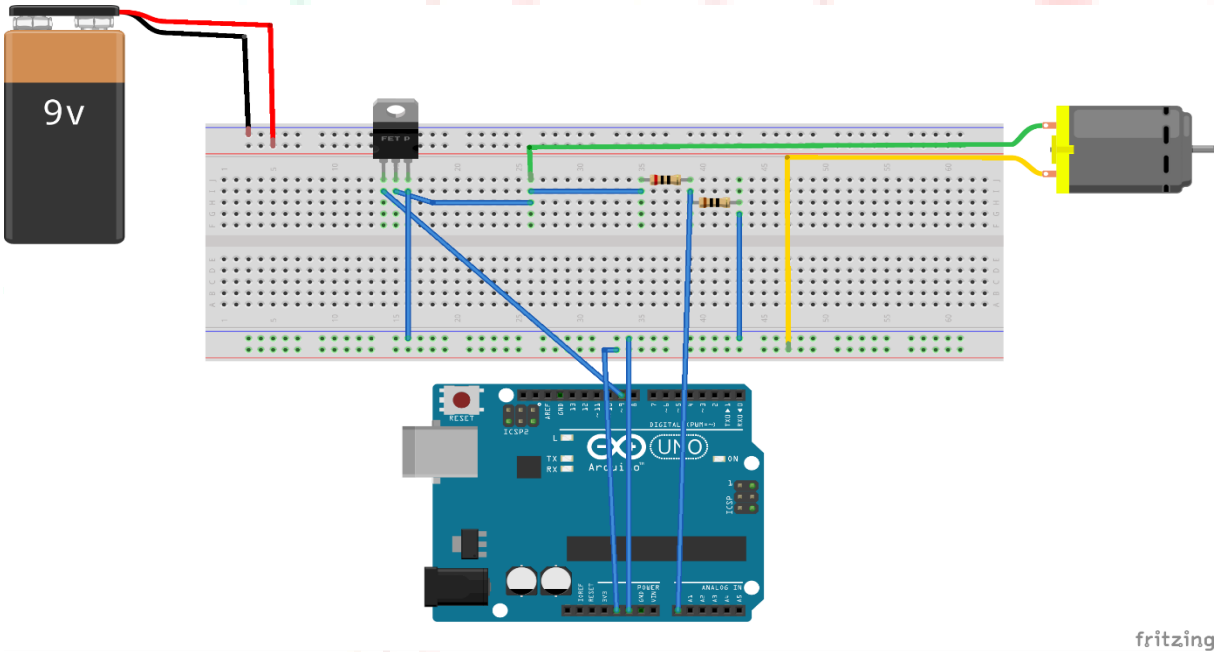


Figure III.22 : Schéma électrique des correcteurs p, pi

III.3.3.3.1. Le bloc PID en Simulink :

Figure III.23 : le bloc PID

La description Le bloc Contrôleur PID implémente un contrôleur PID (PID, PI, PD, P uniquement ou I uniquement). Le bloc est identique au bloc Contrôleur PID discret avec le paramètre Domaine temporel défini sur Temps continu.

La sortie de bloc est une somme pondérée du signal d'entrée, l'intégrale du signal d'entrée et la dérivée du signal d'entrée. Les poids sont les paramètres de gain proportionnel, intégral et dérivé. Un pôle de premier ordre filtre l'action dérivée

Le bloc prend en charge plusieurs types et structures de contrôleur. Les options configurables dans le bloc incluent :

-Type de contrôleur (PID, PI, PD, P uniquement ou I uniquement)

- Formulaire contrôleur (parallèle ou idéal)

- Domaine temporel (continu ou discret)

- Conditions initiales et déclencheur de réinitialisation

- Limites de saturation de sortie et mécanisme anti-enroulement intégré

- Suivi du signal pour le transfert de contrôle sans à-coups et le contrôle multiloop

Réglage du gain PID : Les coefficients du contrôleur PID sont réglables manuellement ou automatiquement. Le réglage automatique nécessite le logiciel Simulink Control Design [12]

III.3.3.3.2. Correcteur proportionnel P :

Le correcteur proportionnel est le plus simple des correcteurs. Il ne permet pas toujours d'obtenir des performances très élevées mais il peut suffire dans certains cas si le cahier des charges n'est pas trop contraignant ou si le système a un comportement assez simple. Expérimentalement, son réglage peut se faire directement en partant d'un gain faible et en augmentant petit à petit jusqu'à atteindre un comportement satisfaisant. Pour un gain trop élevé, le système deviendra instable ce qui se manifestera d'abord par des oscillations de plus en plus importantes

On règle les paramètres de PID Controller

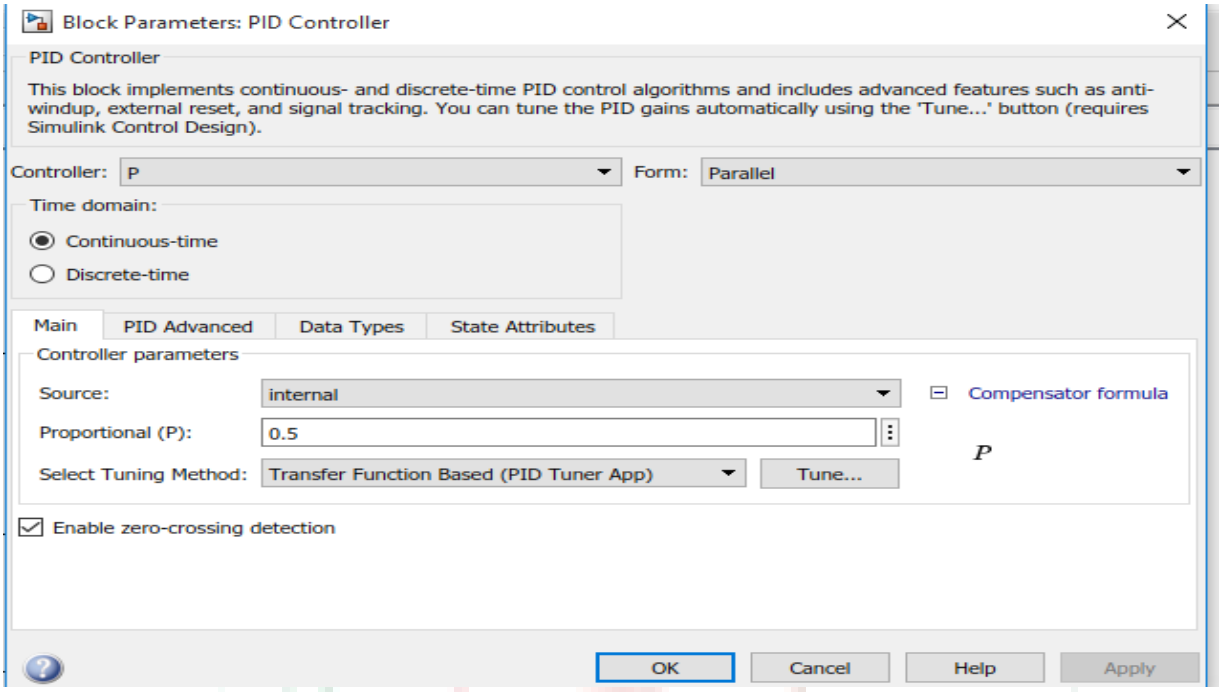


Figure III.24 : paramètres de PID Controller

Schéma en Simulink :

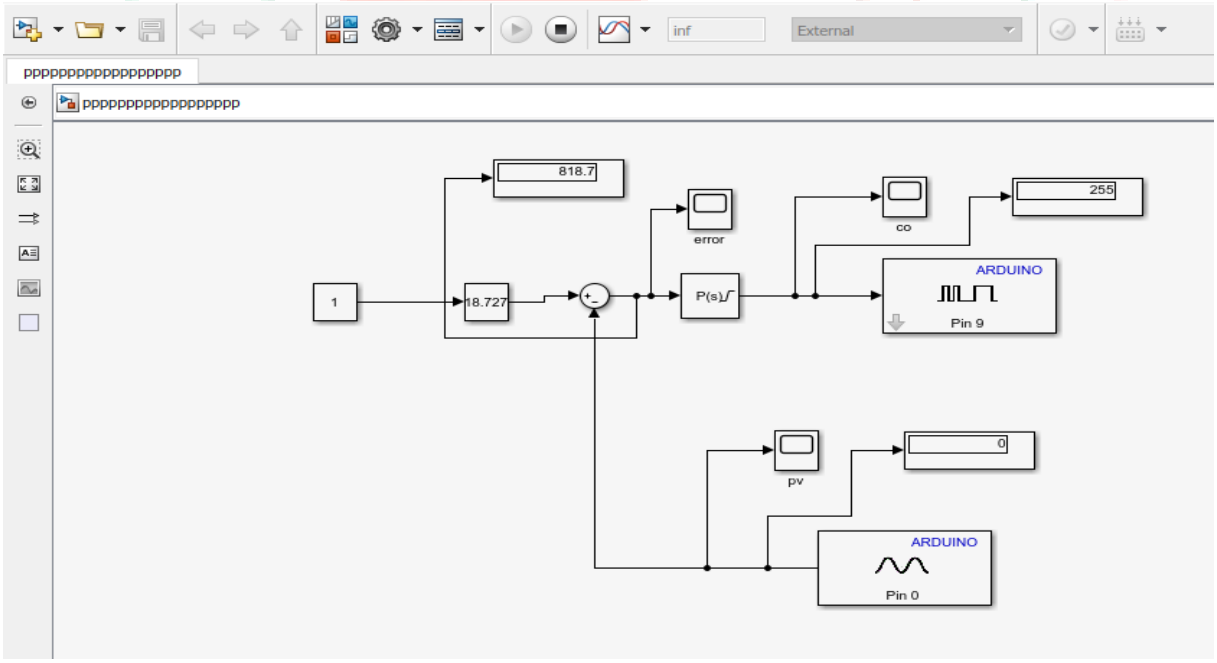


Figure III.25 : Schéma en Simulink de correcteur p

Après avoir appuyé sur le moteur :

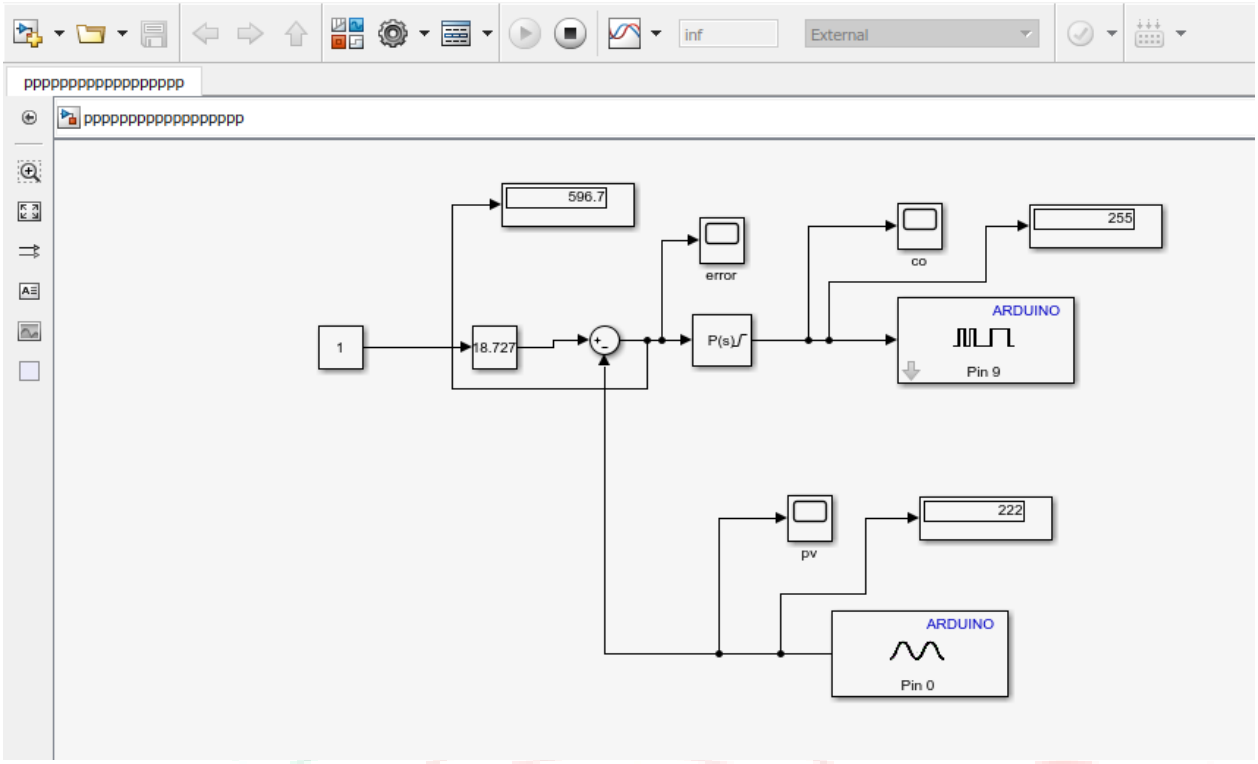


Figure III.26 : schéma en Simulink de correcteur p après variation

Les signaux obtenus :

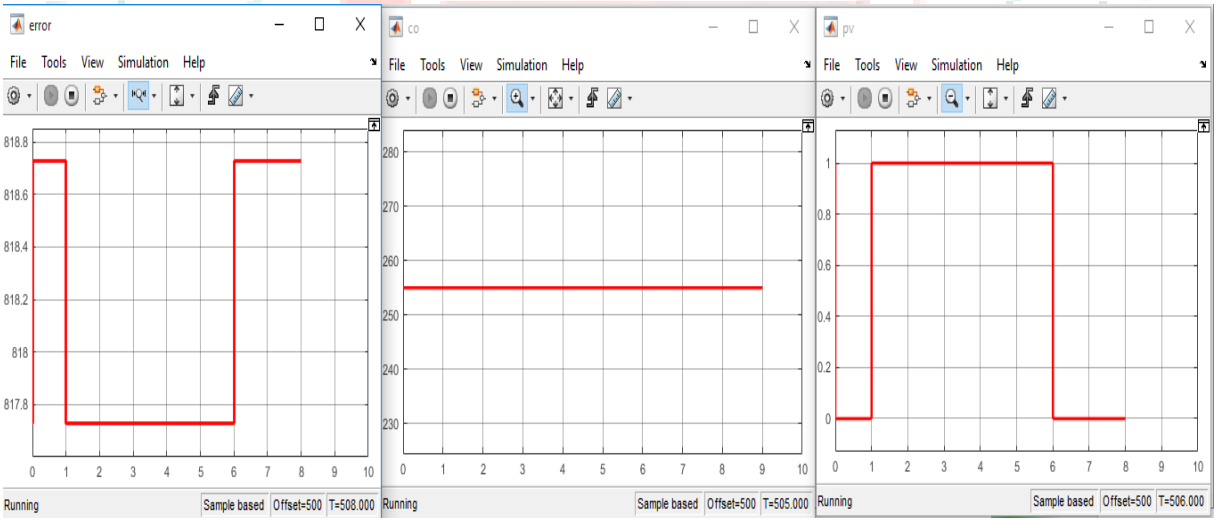


Figure III.27 : Les signaux affiches sur scope de correcteur p

Commentaires :

Error : c'est le signal avant la correction

CO : c'est le signal après la correction

PV : c'est le signal utilisé ne change pas (lorsque on appuie sur le moteur) et ce signal est le résultat que nous recherchons

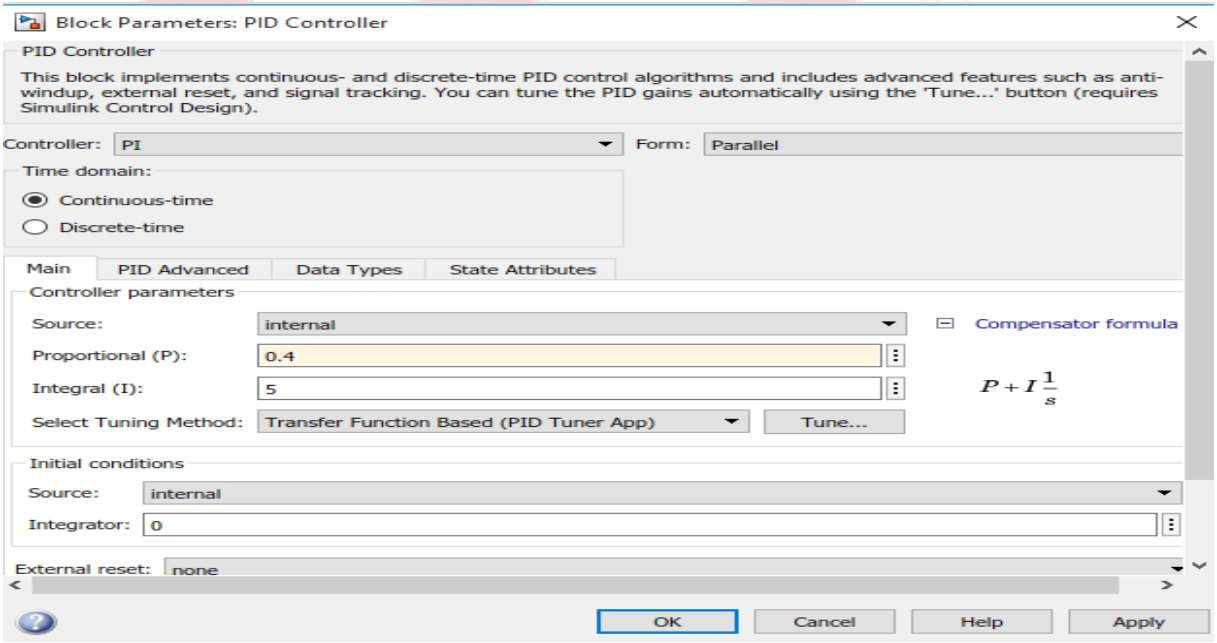
III.3.3.3.3. Correcteur pi :

L'intérêt de la correction de type intégral est de permettre une erreur statique nulle. En effet, si le système comporte un intégrateur et se stabilise à un point d'équilibre, tous les signaux sont constants, pour que la sortie d'un intégrateur soit constante, il est nécessaire que son entrée soit nulle. Si l'entrée de l'intégrateur est reliée à l'erreur de régulation, l'effet du terme intégral sera bien d'annuler cette erreur. Notons que l'erreur est annulée même en présence d'une perturbation. Annuler l'erreur en régime permanent est une chose, mais le faire rapidement en est une autre. Il convient donc d'être capable de régler le correcteur de manière adéquate.

Quand utiliser un régulateur PI ?

- Dérivée peu fréquemment utilisée pour la régulation des procédés industriels.
- Approprié pour les systèmes réglés dont le comportement transitoire est proche de celui d'un système du premier ordre.
- Approprié pour les systèmes d'ordre supérieur s'il n'est pas nécessaire d'atteindre des performances élevées.

On règle les paramètres de PID Controller



FigureIII.28 : les paramètres de PID Controller

Schéma en Simulink

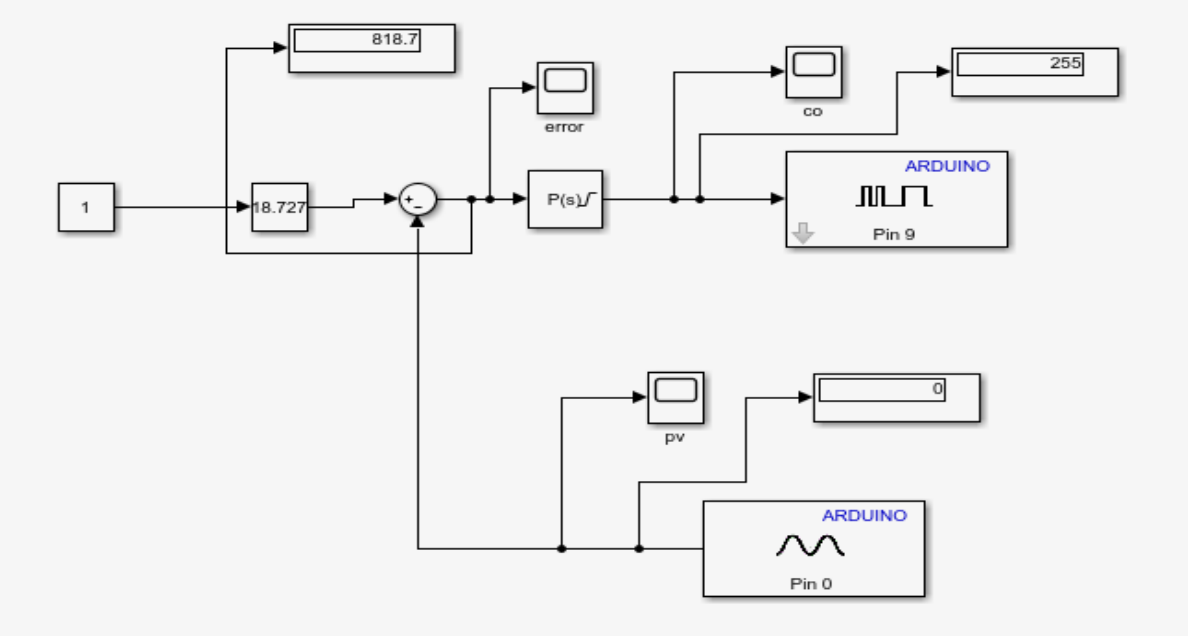


Figure III.29 : schéma en Simulink de correcteur pi

Après avoir appuyé sur le moteur :

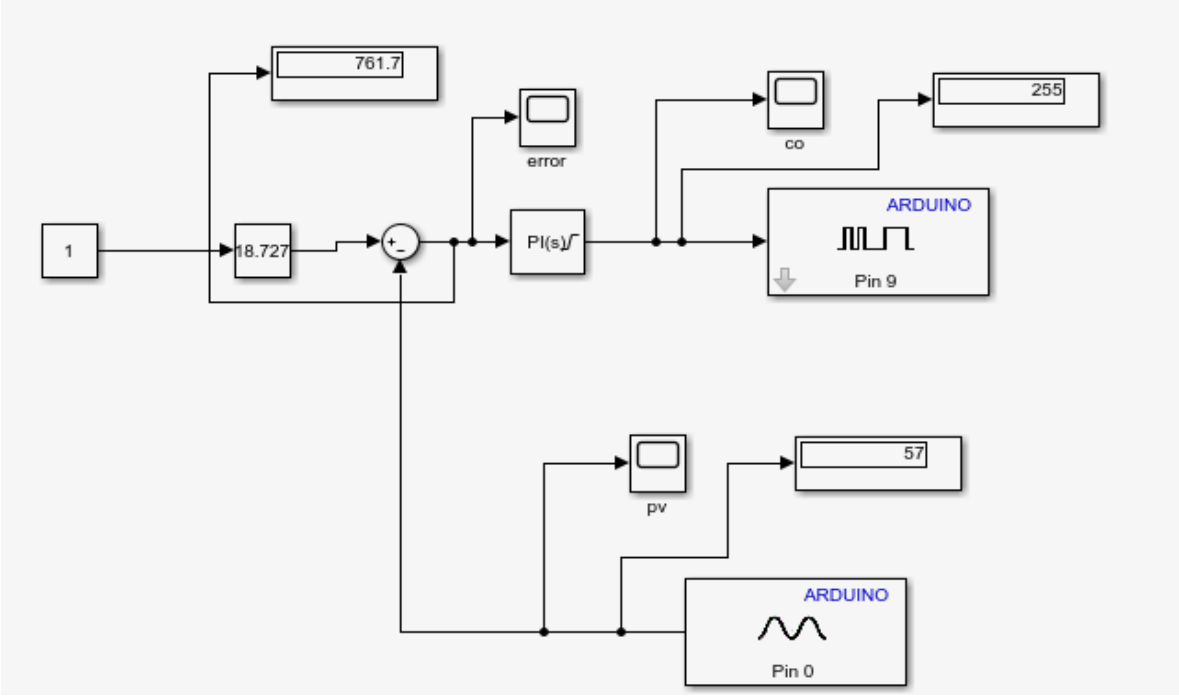


Figure III. 30 : Schéma en Simulink de correcteur pi après variation

Les signaux obtenus

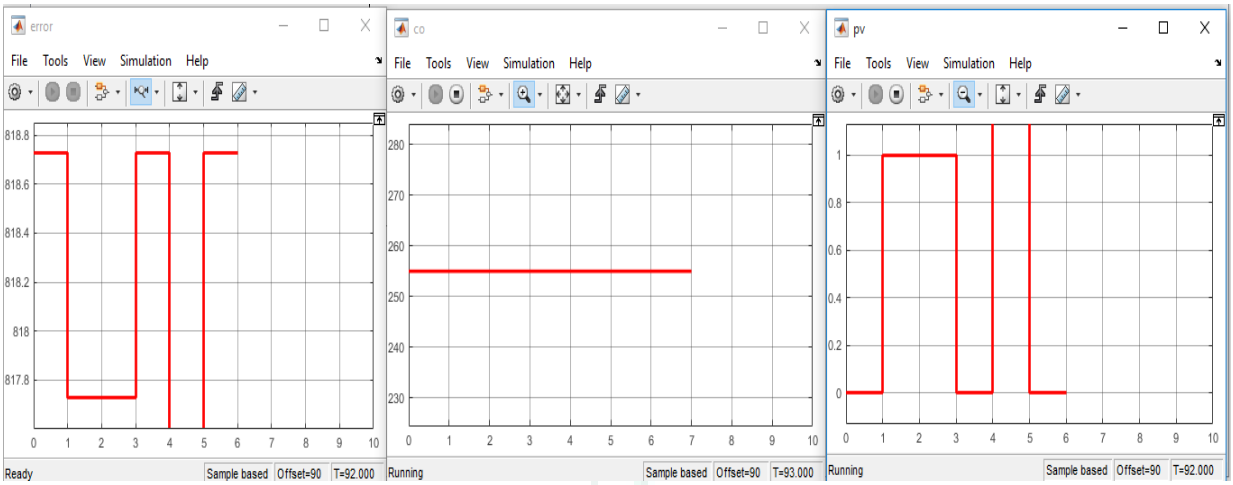


Figure III.31 : Les signaux affichés sur scope de correcteur pi

Commentaires

Les mêmes commentaires précédents mais on a ici une meilleure correction parce que nous avons utilisé le correcteur PI

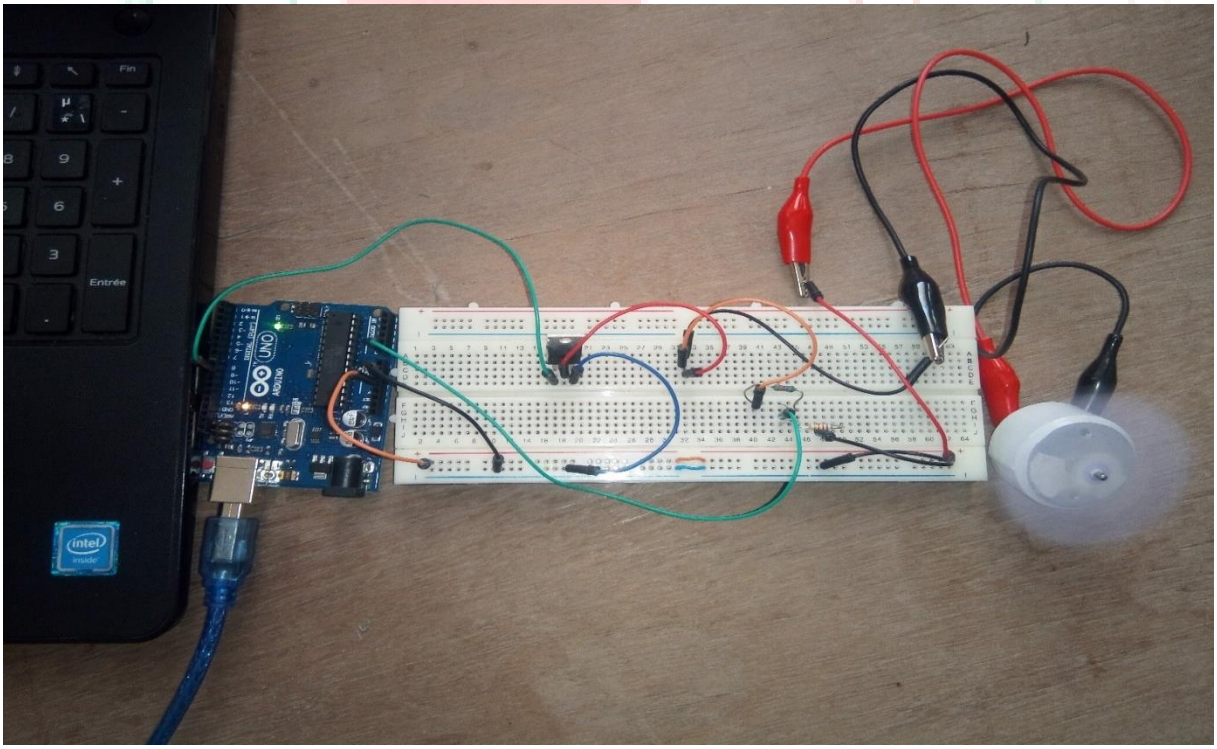


Figure III.32 : Image de circuit final de la variation et la régulation de vitesse d'un MCC

III.4. Conclusion

Ce travail a été consacré à la régulation de vitesse d'un moteur à courant continu par une carte Arduino on utilisons arduino IDE une plateforme Matériel très répandue et une nouvelle méthode s'est porté sur la combinaison des deux environnements MATLAB Simulink Un programme de modélisation /simulation Professionnel bien connu et arduino. Malgré le développement des moteurs à vitesse variable, les moteurs à courant continu sont très utilisés de nos jours, notamment dans les applications automobiles, dans des applications de faible puissance utilisant des batteries ou encore pour la traction électrique.

Durant notre projet, nous avons vu que la régulation de vitesse des moteurs électriques à courant continu était une solution qui pouvait offrir de bonnes performances, que ce soit au niveau de la précision, du temps de réponse, du dépassement, ou de la stabilité

Diverses méthodes permettent la régulation, mais le régulateur idéal n'existe pas, chaque optimisation d'un paramètre (précision, stabilité, rapidité) se fait aux dépends d'un autre. Il convient alors de trouver le meilleur compromis en fonction des exigences initiales du cahier de charge.

Dans ce chapitre, nous avons présenté l'étude, la conception et l'implémentation d'un régulateur PI de vitesse d'un MCC, le rôle de la commande PWM dans la variation de la vitesse du MCC ainsi que la capacité de la force contre électromotrice à donner une information sur la vitesse du MCC.

Chapitre 4 :

Conception et interface graphique d'un système de pompage solaire

IV.1. Introduction :

Notre travail consiste à études et mesures les trois grandeurs d'un système photovoltaïque simple constitué d'un panneau solaire et leurs transmissions à un ordinateur à base d'Arduino

Et l'interface MATLAB.

Les grandeurs à mesurer sont : la température le courant et la tension.

On utilise la puissance de PV pour alimenter notre charge Il se compose du moteur, MOSFET (notre circuit de puissance) et de la pompe à eau.

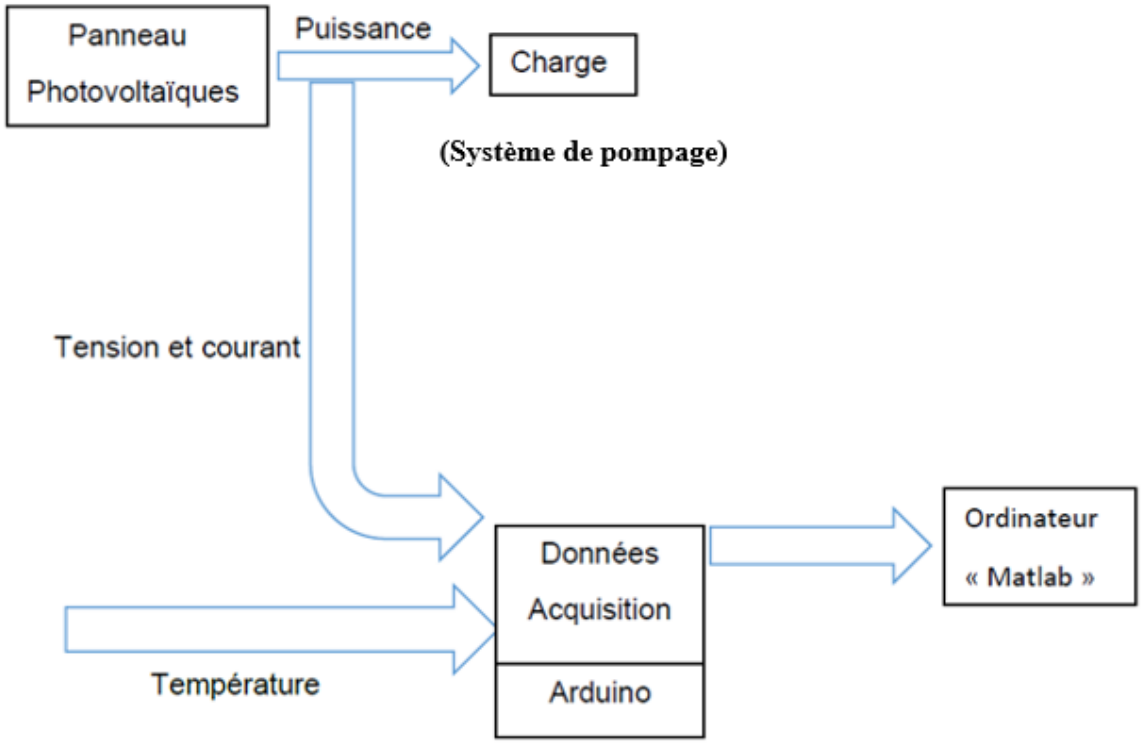


Figure IV.1 : Bilan d'acquisition de donné

IV.2 Matériels et composants utilisés :

IV.2.1 Panneau photovoltaïque

Le panneau photovoltaïque utilisé est fabriqué par Condor Algérie, il présente les caractéristiques suivantes :

Puissance nominale (Pmp)	W	100	
Voltage Circuit Ouvert (Voc)	V	21.64	
Courant de Court Circuit Isc	A	5.8	
Tension de mpp Vmpp	V	18	
Courant de mpp Impp	A	5.48	
Voltage max du Systéme de VDC	V	1000	
Charge maximale De fusibles (A)	A	10	
Facteur de température (cellule)			
Température de fonctionnement normale		NOCT	45±2° C
Coefficient de température de puissance		%/°C	-0,41
Coefficient de température du Courant		%/°C	+0,03
Coefficient de température du Voltage		%/°C	-0,32

Figure IV.2 : les caractéristiques de panneau photovoltaïque

IV.2.2 La batterie de panneau et ces caractéristiques

Toute batterie est un ensemble d'éléments ou de cellules électrochimiques capable de stocker de l'énergie électrique sous forme chimique et Les paramètres électriques suivants sont généralement employés pour caractériser une batterie :

- Tension nominale : 12V
- Capacité nominale : 4,5Ah

IV.2.3 Les capteurs

Nous avons utilisé les capteurs suivants :

IV.2.3.1 Capteur de température LM35

Pour mesurer la température de milieu extérieur on a utilisé le capteur de température LM35 et voici ses caractéristiques :

- Tension d'alimentation : 35V
- Tension de sortie : 6V
- Courant de sortie : 10mA

La figure IV.3 montre le schéma de réalisation de ce capteur

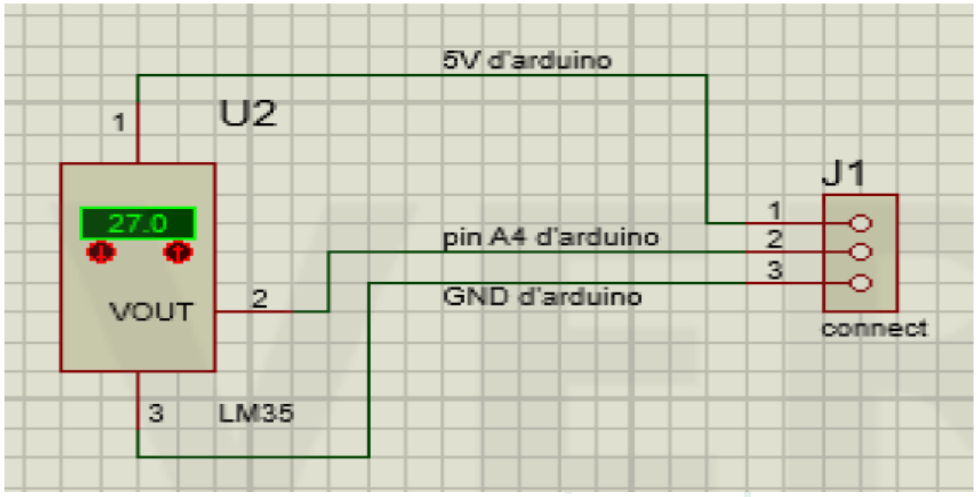


Figure IV.3 : Schéma de réalisation de capteur de température

IV.2.3.2 Capteur de tension de panneau PV :

Pour ce capteur nous avons utilisé le diviseur de tension, telle que :

$$Vs = \frac{R2}{R1 + R2} * Ve$$

- Ve : tension d'entrée (PV)
- Vs : tension de sortie (PV)

Tel que :

Ve = 18V et Vs = 5V

Nous avons fixé R2 à 10KΩ et on détermine R1 comme suite :

Vs*(R1+R2) = R2*Ve ⇒ R1*Vs+R2*Vs = R2*Ve

⇒ R1 = R2*(Ve/Vs –1)

AN : $R1 = 10 * \left(\frac{18}{5} - 1\right) = 26\ K\Omega$

La figure IV.4 nous montre le schéma de réalisation du capteur de tension

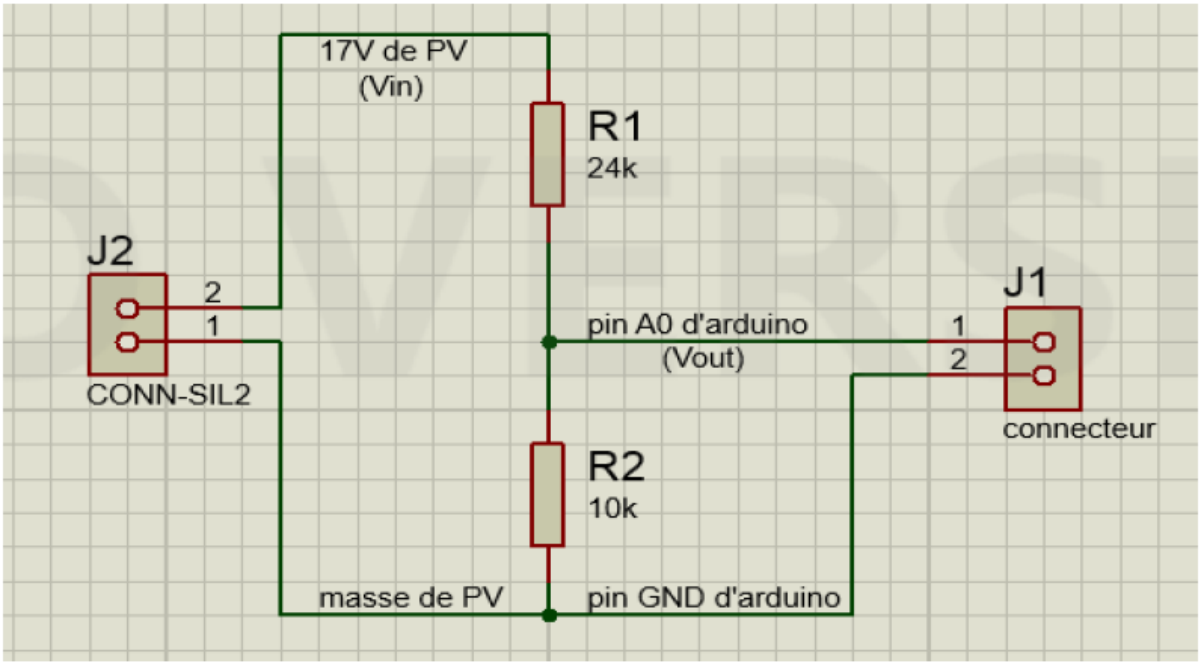


Figure IV.4 : Schéma de capteur de tension PV

IV.2.3.3 Capteur de courant pour (PV)

Pour mesurer le courant de PV nous avons réalisé un capteur de courant par la méthode shunt, le problème rencontré c'est que le principe de cette méthode consiste à choisir une faible résistance, de quelque Ω. Et la tension au borne de cette résistance est très faible, de quelque mv, pour régler ce problème nous avons utilisé un amplificateur LM358 pour amplifier la tension de sortie.

Déterminant quelques paramètres pour la réalisation de capture : [15]

Nous avons :

$$U_{shunt} = R_{shunt} * I_e \Rightarrow U_{shunt} = 0.01 * 0.6$$

$$\Rightarrow U_{shunt} = 0.006V$$

$$U = U_{shunt} * \left(1 + \frac{R2}{R1}\right) \Rightarrow U = R_{shunt} * I_{shunt} * \left(1 + \frac{R2}{R1}\right)$$

Donc : U=18V, $U_{shunt} = 0.06V$. Reste à déterminer R_1 et R_2.

Pour déterminer R_1 et R_2, il se fit de fixé une résistance et déterminer la deuxième. Dans notre cas nous avons fixé R_1 à 0.001KΩ et nous avons déterminé R_2 comme suite :

$$U = U_{shunt} * \left(1 + \frac{R2}{R1}\right) \Rightarrow U - U_{shunt} = U_{shunt} * \frac{R2}{R1}$$

$\Rightarrow R_2 = (U - U_{shunt}) * R_1/U_{shunt}$

$\Rightarrow R_2 = (18\text{-}0.006) * 0.001 / 0.006$

$\Rightarrow R_2 = 2.99\ K\Omega \approx 3\ K\Omega$

Après la détermination de ces paramètres nous avons réalisé le capteur de courant pour PV. La figure IV.5 illustre ce capteur.

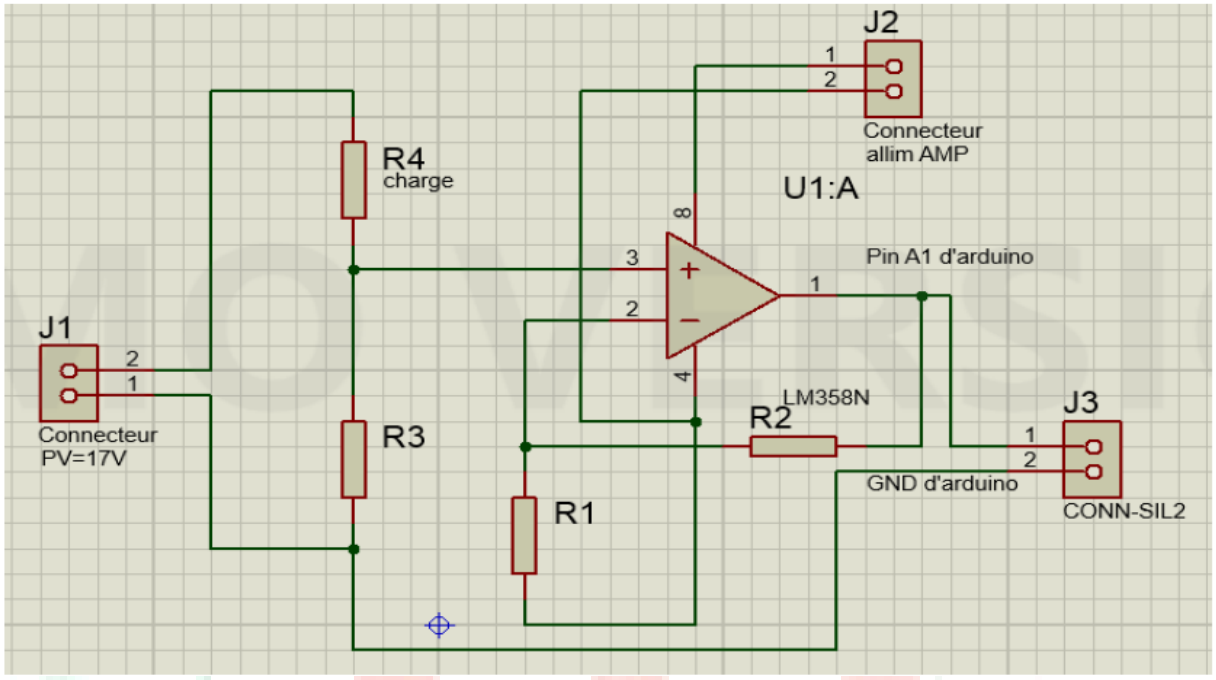


Figure IV.5 : Schéma de réalisation d'un capteur de courant pour PV

IV.2.4. L'interface graphique de Matlab/arduino

Nous voulons montrer et mesurer la variation de température, tension et de courant de panneaux PV à l'aide de l'interface MATLAB arduino et du support d'entrée analogique en Simulink

Mesure de température :

D'après le schéma de capteur de température on branche la 2ème pate de capteur à la broche A4 de l'arduino et d'après le bloc analogique en Simulink et le scope nous pouvons étudier et voir les changements de température.

Mesure de tension :

D'après le schéma de capteur de tension On branche la sortie de ce capteur à la broche A0 de l'arduino (la même méthode que nous avons utilisée dans capteur de température)

Mesure de courant :

D'après le schéma de capteur de courant on branche la sortie (vout) de ce capteur à la broche A1 de l'arduino (identique à la manière précédente

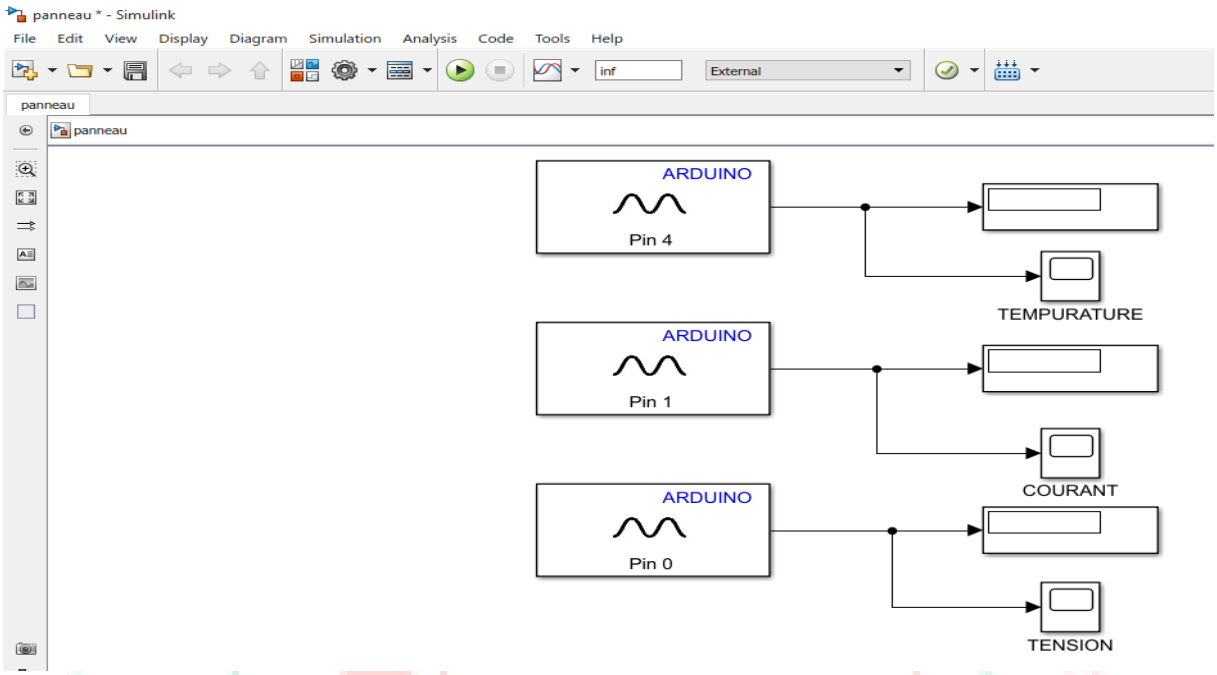


Figure IV.6 : schéma en Simulink

Les signaux obtenus :

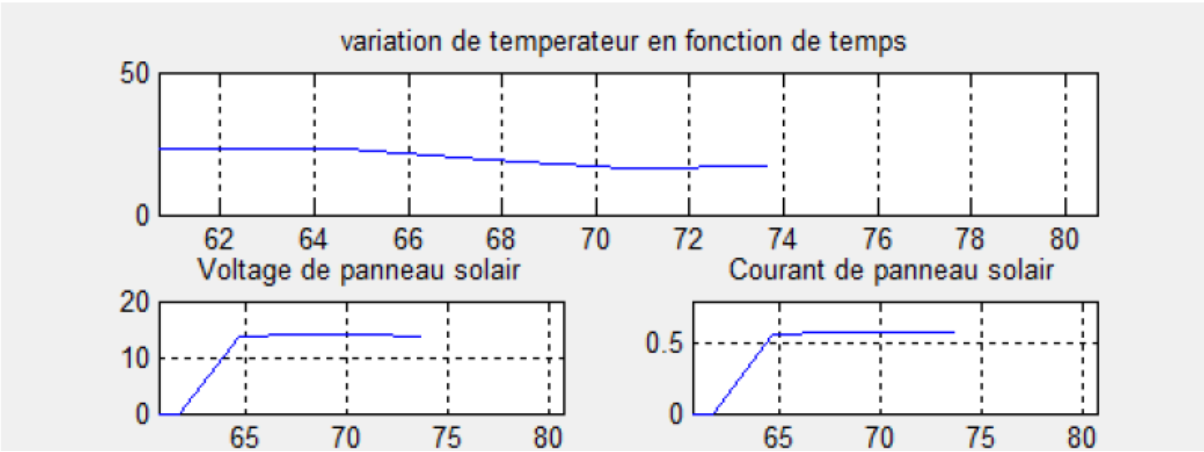


Figure IV.7 : Exemple sur les résultats de simulation de variation de la température, tension et courant de panneau

IV.3. Conclusion :

Ce chapitre est consacré à la présentation de la réalisation matérielle de notre système d'acquisition. Dans le premier temps c'est la présentation de matériels utilisés comme le panneau solaire et après les différents capteurs utilisés avec leurs caractéristiques et leurs schémas de réalisation.

La phase finale de ce travail c'est la réalisation avec l'utilisation de matériels et logiciels (Arduino UNO, Matlab) et la présentation des valeurs mesurées affichées par l'interface graphique de Matlab.

Conclusion générale :

Le système de pompage autonome PV régulée est une solution idéale pour l'alimentation en eau pour les régions peu peuplées ou trop ensoleiller. Les moteurs à courant continu avec une régulation PI (proportionnel intégrale) apportent une solution intéressante pour la réalisation des systèmes de pompage photovoltaïques autonome, ces moteurs présentent plusieurs avantages par rapport aux moteurs à courant alternatif ou à courant continu sont balais.

Dans ce mémoire ; nous avons tout d'abord étudié d'une façon générale l'énergie photovoltaïque ; ceci s'est avéré nécessaire pour acquérir une connaissance du comportement du GPV. Ensuite on a réalisé un prototype de pompage solaire autonome avec une régulation PI programmée sur Arduino de deux façons avec les deux environnements arduino IDE et Matlab/arduino Simulink.

On plus nous avons fait une étude sur la réalisation d'un système d'acquisition de signaux pour bien analyser le système de pompage solaire.

Finalement les perspectives que nous pouvons proposer pour l'amélioration et l'évaluation des performances de système, une commande simple et efficace peut agir et apporter des améliorations. Le rôle de la commande MPPT est d'amener la puissance obtenue et la stabiliser à un point de puissance maximale, plusieurs solutions peuvent être proposées. Aussi on propose d'utiliser un module Bluetooth dans le système d'acquisition pour transmettre les données au PC (un système embarquer d'acquisition de données)

Bibliographies

[1] M. Akbaba, and al: « Matching Of Separately Excited Dc Motors To Photovoltaic Generators For Maximum Power Output», Solar Energy, Vol. 63, N0. 6, 1998, pp.: 375- 385.

[2] V. C. Mummadi: « Steady State And Dynamic Performances Analysis Of PV Supplied DC Motors Fed From Intermediate Power Converter», Solar Energy Materials & Solar Cells, Vol-61, 2000, pp.: 365-381.

[3] R. Chenni: « Etude technico-économique d'un système de pompage photovoltaïque dans un village solaire». Thèse de Doctorat, Université Mentouri de Constantine, 2007.

[4] J. Appelbaum, M.S. Sarme: « The operation of a permanent magnet DC motors powered by a common source solar cells», IEEE Trans. on energy. conv, Vol 4, pp635- 642,1989.

[5] A.A. Ghoneim: « Design optimization of photovoltaic powered water pumping systems». Energy Conversion and Management 47 (2006) 1449–1463.

[6] W.R Anis et al, «Coupling of a volumetric pump to a photovoltaic array», Solar cells,Vol.14, pp27-42, 1985.

[7] D. Weigner and at Levinson: «Watt pumping optimal operation », Electrical Machines and power system, Electric machines and components, Vol 24, N°3, pp277-288, 1996.

[8] D. Langrigge et Al:« Development of a photovoltaic pumping system using brushless DC motor and helical rotor pump »,, Solar energy, Vol 56,N°2, pp151-160,1996

[9] Swamy, C.L.P: «Dynamic performance of permanent magnet brushless DC motor powered by a PV array for water pumping». Solar. Energy. Mater. Sol. Cells 36, 187–200.1995

[10] B. Azoui.: «Concept and realization of a three-phase brushless DC motor for solar application», Doctor d'états Es'-Science, University of Batna 2002.

[11] http://www.elearn.univ-ouargla.dz

[12] http://www.fr.m.wikipedia.org

[13] http://www.Mathwork.com 14/04/2020

[14] http://www. playground.arduino.cc

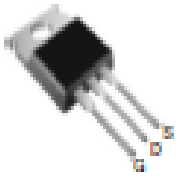
[15] F. Abdoune, B. Saadi « Réalisation d'un système embarqué d'acquisition de donné pour l'analyse et la diagnostique d'un système photovoltaïque », thèse de master, Université Abderrahmane Mira – Bejaia,2016.

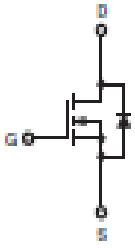
Annexes

Power MOSFET

PRODUCT SUMMARY		
V_{DS} (V)	100	
$R_{DS(on)}$ (Ω)	V_{GS} = 10 V	0.077
Q_g (Max.) (nC)	72	
Q_{gs} (nC)	11	
Q_{gd} (nC)	32	
Configuration	Single	

TO-220AB





N-Channel MOSFET

FEATURES

- Dynamic dV/dt Rating
- Repetitive Avalanche Rated
- 175 °C Operating Temperature
- Fast Switching
- Ease of Paralleling
- Simple Drive Requirements
- Compliant to RoHS Directive 2002/95/EC

Available
RoHS* COMPLIANT

DESCRIPTION

Third generation Power MOSFETs from Vishay provide the designer with the best combination of fast switching, ruggedized device design, low on-resistance and cost-effectiveness.

The TO-220AB package is universally preferred for all commercial-industrial applications at power dissipation levels to approximately 50 W. The low thermal resistance and low package cost of the TO-220AB contribute to its wide acceptance throughout the industry.

ORDERING INFORMATION	
Package	TO-220AB
Lead (Pb)-free	IRFs40PbF
	SiHFs40-E3
SnPb	IRFs40
	SiHFs40

ABSOLUTE MAXIMUM RATINGS (T_C = 25 °C, unless otherwise noted)					
PARAMETER			SYMBOL	LIMIT	UNIT
Drain-Source Voltage			V_{DS}	100	V
Gate-Source Voltage			V_{GS}	± 20	
Continuous Drain Current	V_{GS} at 10 V	T_C = 25 °C	I_D	28	A
		T_C = 100 °C		20	
Pulsed Drain Current[a]			I_{DM}	110	
Linear Derating Factor				1.0	W/°C
Single Pulse Avalanche Energy[b]			E_{AS}	230	mJ
Repetitive Avalanche Current[a]			I_{AR}	28	A
Repetitive Avalanche Energy[a]			E_{AR}	15	mJ
Maximum Power Dissipation		T_C = 25 °C	P_D	150	W
Peak Diode Recovery dV/dt[c]			dV/dt	5.5	V/ns
Operating Junction and Storage Temperature Range			T_J, T_{stg}	- 55 to + 175	°C
Soldering Recommendations (Peak Temperature)		for 10 s		300[d]	
Mounting Torque		6-32 or M3 screw		10	lbf · in
				1.1	N · m

Notes

a. Repetitive rating; pulse width limited by maximum junction temperature (see fig. 11).
b. V_{DD} = 25 V, starting T_J = 25 °C, L = 440 µH, R_g = 25 Ω, I_{AS} = 28 A (see fig. 12).
c. I_{SD} ≤ 28 A, dI/dt ≤ 170 A/µs, V_{DD} ≤ V_{DS}, T_J ≤ 175 °C.
d. 1.6 mm from case.

Caractéristique de MOSFET IRF540 :

THERMAL RESISTANCE RATINGS				
PARAMETER	SYMBOL	TYP.	MAX.	UNIT
Maximum Junction-to-Ambient	R_{thJA}	-	62	°C/W
Case-to-Sink, Flat, Greased Surface	R_{thCS}	0.50	-	
Maximum Junction-to-Case (Drain)	R_{thJC}	-	1.0	

SPECIFICATIONS (T_J = 25 °C, unless otherwise noted)							
PARAMETER	SYMBOL	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
Static							
Drain-Source Breakdown Voltage	V_{DS}	V_{GS} = 0 V, I_D = 250 µA		100	-	-	V
V_{DS} Temperature Coefficient	$\Delta V_{DS}/T_J$	Reference to 25 °C, I_D = 1 mA		-	0.13	-	V/°C
Gate-Source Threshold Voltage	$V_{GS(th)}$	V_{DS} = V_{GS}, I_D = 250 µA		2.0	-	4.0	V
Gate-Source Leakage	I_{GSS}	V_{GS} = ± 20 V		-	-	± 100	nA
Zero Gate Voltage Drain Current	I_{DSS}	V_{DS} = 100 V, V_{GS} = 0 V		-	-	25	µA
		V_{DS} = 80 V, V_{GS} = 0 V, T_J = 150 °C		-	-	250	
Drain-Source On-State Resistance	$R_{DS(on)}$	V_{GS} = 10 V	I_D = 17 A[b]	-	-	0.077	Ω
Forward Transconductance	g_{fs}	V_{DS} = 50 V, I_D = 17 A[b]		8.7	-	-	S
Dynamic							
Input Capacitance	C_{iss}	V_{GS} = 0 V, V_{DS} = 25 V, f = 1.0 MHz, see fig. 5		-	1700	-	pF
Output Capacitance	C_{oss}			-	560	-	
Reverse Transfer Capacitance	C_{rss}			-	120	-	
Total Gate Charge	Q_g	V_{GS} = 10 V	I_D = 17 A, V_{DS} = 80 V, see fig. 6 and 13[b]	-	-	72	nC
Gate-Source Charge	Q_{gs}			-	-	11	
Gate-Drain Charge	Q_{gd}			-	-	32	
Turn-On Delay Time	$t_{d(on)}$	V_{DD} = 50 V, I_D = 17 A, R_g = 9.1 Ω, R_D = 2.9 Ω, see fig. 10[b]		-	11	-	ns
Rise Time	t_r			-	44	-	
Turn-Off Delay Time	$t_{d(off)}$			-	53	-	
Fall Time	t_f			-	43	-	
Internal Drain Inductance	L_D	Between lead, 6 mm (0.25") from package and center of die contact		-	4.5	-	nH
Internal Source Inductance	L_S			-	7.5	-	
Drain-Source Body Diode Characteristics							
Continuous Source-Drain Diode Current	I_S	MOSFET symbol showing the integral reverse p - n junction diode		-	-	28	A
Pulsed Diode Forward Current[a]	I_{SM}			-	-	110	
Body Diode Voltage	V_{SD}	T_J = 25 °C, I_S = 28 A, V_{GS} = 0 V[b]		-	-	2.5	V
Body Diode Reverse Recovery Time	t_{rr}	T_J = 25 °C, I_F = 17 A, dI/dt = 100 A/µs[b]		-	180	360	ns
Body Diode Reverse Recovery Charge	Q_{rr}			-	1.3	2.8	µC
Forward Turn-On Time	t_{on}	Intrinsic turn-on time is negligible (turn-on is dominated by L_S and L_D)					

Notes

a. Repetitive rating; pulse width limited by maximum junction temperature (see fig. 11).

b. Pulse width ≤ 300 µs; duty cycle ≤ 2 %.

TYPICAL CHARACTERISTICS (25 °C, unless otherwise noted)

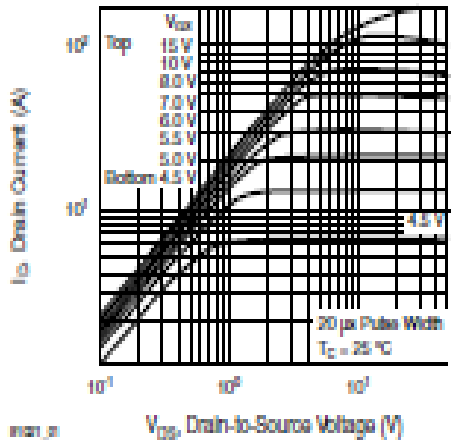


Fig. 1 - Typical Output Characteristics, T_C = 25 °C

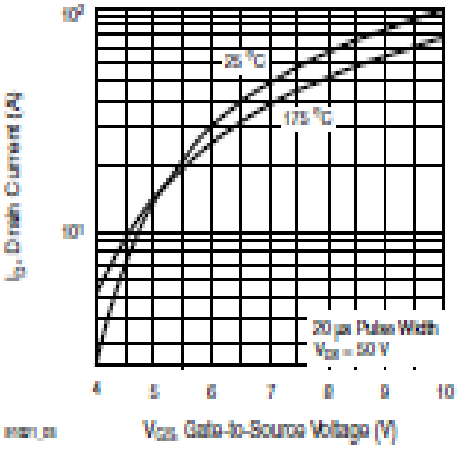


Fig. 3 - Typical Transfer Characteristics

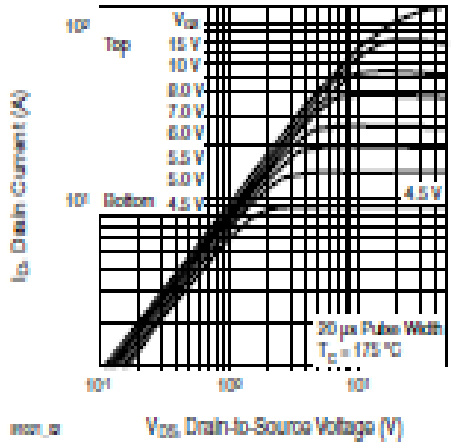


Fig. 2 - Typical Output Characteristics, T_C = 175 °C

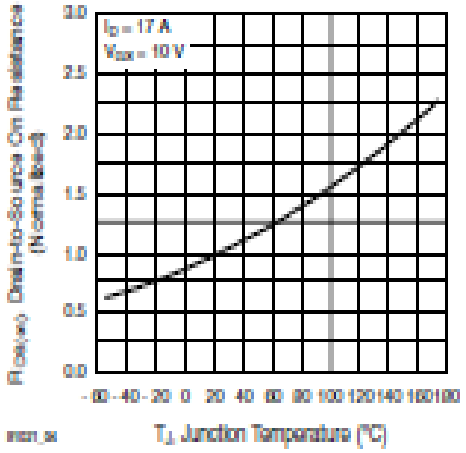


Fig. 4 - Normalized On-Resistance vs. Temperature

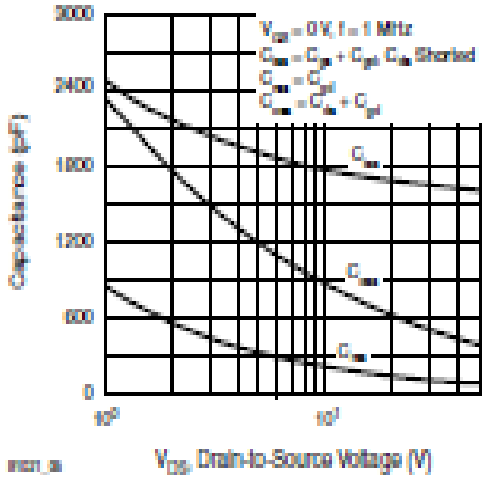


Fig. 5 - Typical Capacitance vs. Drain-to-Source Voltage

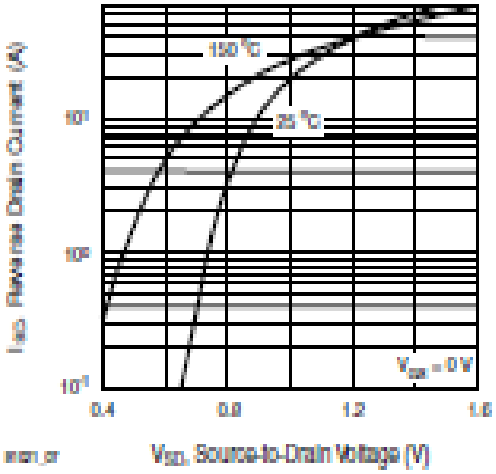


Fig. 7 - Typical Source-Drain Diode Forward Voltage

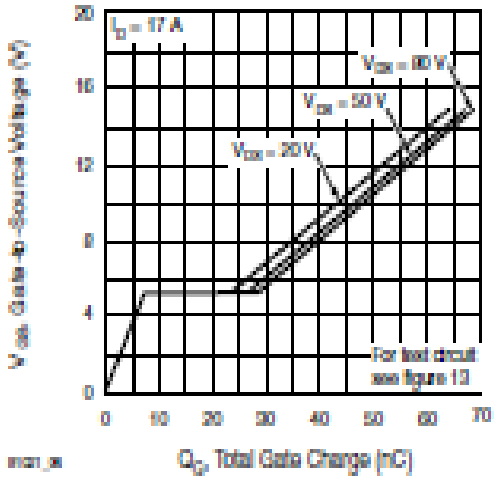


Fig. 6 - Typical Gate Charge vs. Gate-to-Source Voltage

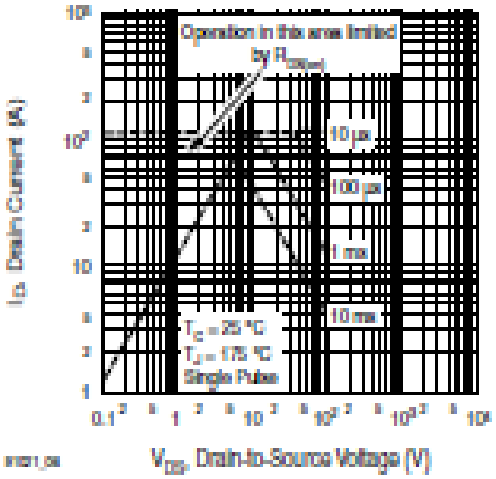


Fig. 8 - Maximum Safe Operating Area

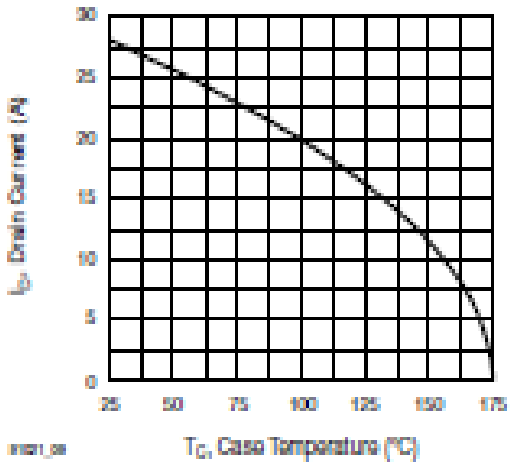


Fig. 9 - Maximum Drain Current vs. Case Temperature

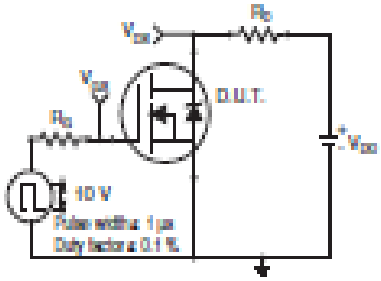


Fig. 10a - Switching Time Test Circuit

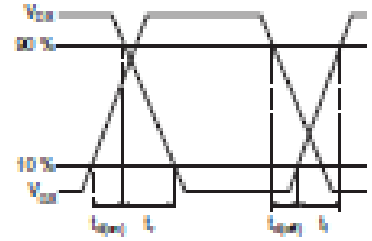


Fig. 10b - Switching Time Waveforms

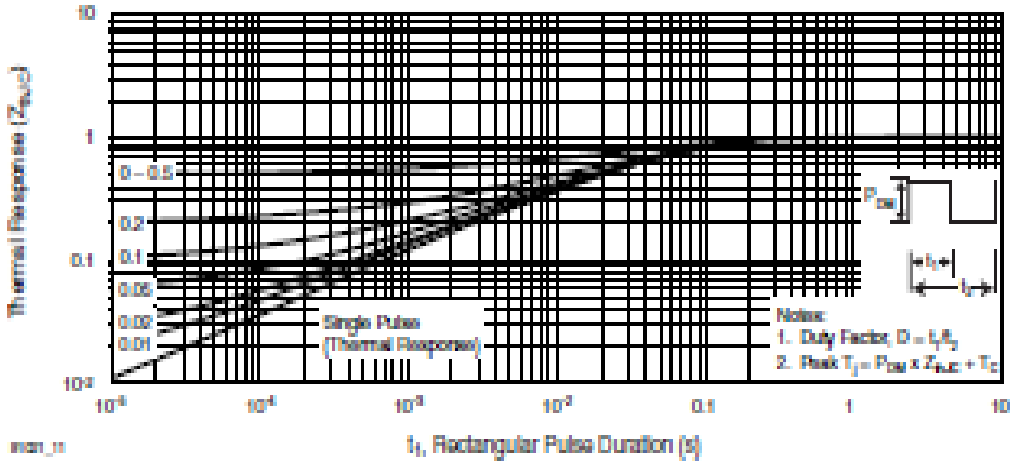
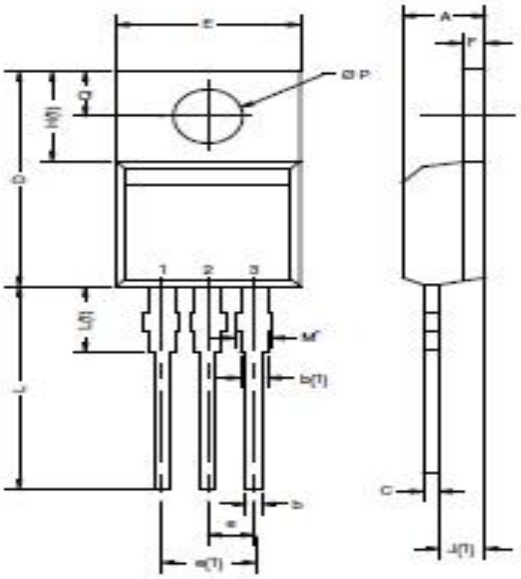


Fig. 11 - Maximum Effective Transient Thermal Impedance, Junction-to-Case

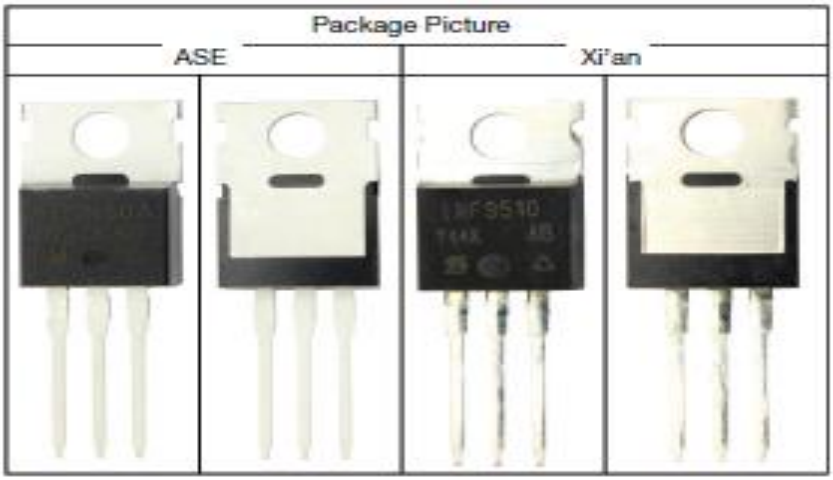
TO-220-1



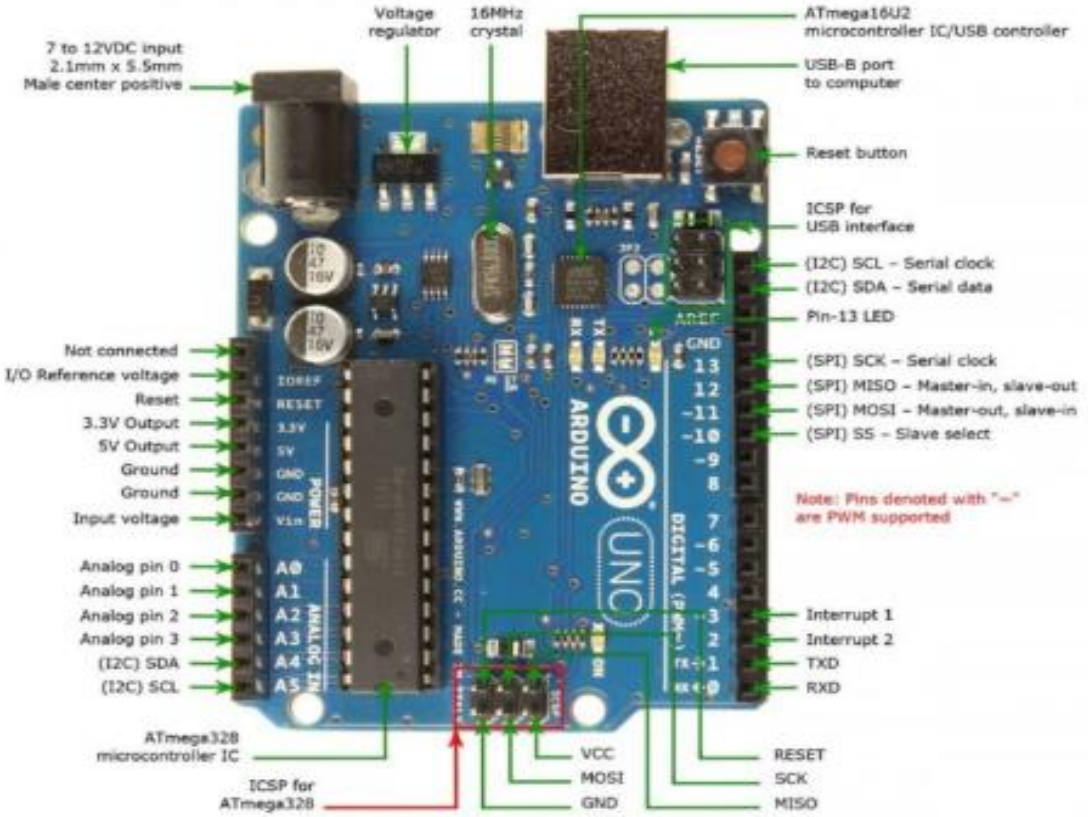
DIM.	MILLIMETERS		INCHES	
	MIN.	MAX.	MIN.	MAX.
A	4.24	4.65	0.167	0.183
b	0.69	1.02	0.027	0.040
b(1)	1.14	1.78	0.045	0.070
c	0.36	0.61	0.014	0.024
D	14.33	15.85	0.564	0.624
E	9.96	10.52	0.392	0.414
e	2.41	2.67	0.095	0.105
e(1)	4.88	5.28	0.192	0.208
F	1.14	1.40	0.045	0.055
H(1)	6.10	6.71	0.240	0.264
J(1)	2.41	2.92	0.095	0.115
L	13.36	14.40	0.526	0.567
L(1)	3.33	4.04	0.131	0.159
Ø P	3.53	3.94	0.139	0.155
Q	2.54	3.00	0.100	0.118
ECN: X15-0364-Rev. C, 14-Dec-15 DWG: 6031				

Note

- M* = 0.052 inches to 0.064 inches (dimension including protrusion), heatsink hole for HVM



Caractéristique de Arduino UNO :



- Version : R3
- alimentation:
  - via port USB ou
  - 7 à 12 V sur connecteur alim 5,5 x 2,1 mm
- Microprocesseur : ATMega328
- Mémoire flash : 32 kB
- Mémoire SRAM : 2 kB
- Mémoire EEPROM : 1 kB
- 14 broches d'E/S dont 6 PWM
- 6 entrées analogiques 10 bits
- Intensité par E/S : 40 mA
- Cadencement : 16 MHz
- Bus série, I2C et SPI
- Gestion des interruptions
- Fiche USB B
- Dimensions : 74 x 53 x 15 mm